Learning Composite Operators for Object Detection

Bir Bhanu and **Yingqiang Lin**

Center for Research in Intelligent Systems
University of California, Riverside, CA, 92521, USA
Email: {bhanu, yqlin}@vislab.ucr.edu
Tel: 909-787-3954

Abstract

In this paper, we learn to discover composite operators and features that are evolved from combinations of primitive image processing operations to extract regions-of-interest (ROIs) in images. Our approach is based on genetic programming (GP). The motivation for using GP is that there are a great many ways of combining these primitive operations and the human expert, limited by experience, knowledge and time, can only try a very small number of conventional ways of combination. Genetic programming, on the other hand, attempts many unconventional ways of combination that may never be imagined by human experts. In some cases, these unconventional combinations yield exceptionally good results. Our experimental results show that GP can find good composite operators, that consist of primitive operators designed in this paper, to effectively extract the regions of interest in images and the learned composite operators can be applied to extract ROIs in other similar images.

1 INTRODUCTION

Object detection is an important intermediate step to object recognition. The task of object detection is to locate and extract regions from an image that may contain potential objects. These regions are called regions of interest (ROIs) or object chips. The quality of object detection is dependent on the kind and quality of features extracted from an image. There are many kinds of features that can be extracted. The question is what are the appropriate features or how to synthesize features, particularly useful for detection, from the primitive features extracted from an image. The answer to these questions is largely dependent on the intuitive instinct, knowledge, previous experience and even the bias of human image experts.

In this paper, we use genetic programming (GP) to synthesize composite features, which are the output of composite operators, to perform object detection. A composite operator consists of primitive operators and it can be viewed as a combination of primitive operations on images. The basic approach is to apply a composite operator on the original image or primitive feature images generated from the original one, then the output image of the composite operator (called composite feature) is segmented to obtain a binary image or mask to extract the region containing the object from the original image. The individuals in our GP based learning are composite operators represented by binary trees whose internal nodes represent the pre-specified primitive operators and the leaf nodes represent the original image or the primitive feature images. The primitive feature images are predetermined, and they are not the output of the prespecified primitive operators.

2 MOTIVATION AND RELATED RESEARCH

2.1 MOTIVATION

In most imaging applications, an expert designs an approach to extract ROIs from images. The approach can often be dissected into some primitive operations on the original image or a set of related feature images obtained from the original one. It is the expert who, relying on his/her rich experience, figures out a smart way to combine these primitive operations to achieve good results. The task of finding a good approach is equivalent to finding a good point in the search space of *composite operators* formed by the combination of primitive operators.

The number of ways of combining primitive operators is almost infinite. The human expert can only try a very limited number of combinations and typically only the conventional ways of combination are tried. However, a GP may try many unconventional ways of combining primitive operations that may never be imagined by human experts. In some cases, it is the unconventional ways of combination that yield exceptionally good results. The inherent parallelism of GP and the speed of computers allow the portion of the search space explored by GP to be much larger than that by human experts. Although only a very small portion of the space is tried by GP, the search

performed by GP is not a random search. It is guided by the goodness of composite operators in the population. As the search goes on, GP will gradually shift the population to the portion of the space containing good operators.

2.2 RELATED RESEARCH AND OUR CONTRIBUTION

Genetic programming, an extension of genetic algorithm, was first proposed by Koza in [1]. In GP, the individuals can be binary trees, graphs or some other complicated structures of dynamically varying size. Poli [2] used GP to develop effective image filters to enhance and detect features of interest or to build pixel-classification-based segmentation algorithms. Stanhope and Daida [3] used GP paradigms for the generation of rules for target/clutter classification and rules for the identification of objects. To perform these tasks, previously defined feature sets are generated on various images and GP is used to select relevant features and methods for analyzing these features. Howard et al. [4] applied GP to automatic detection of ships in low-resolution SAR imagery using an approach that evolves detectors. Roberts and Howard [5] used GP to develop automatic object detectors in infrared images.

Unlike the work of Stanhope and Daida [3], Howard et al. [4] and Roberts and Howard [5], the input and output of each node of the tree in our system are images, not real numbers. Also, the primitive features defined in this paper are more general and easier to compute than those used in [5]. In summary, the primitive operators and primitive features designed by us are very basic and domain-independent, not specific to a kind of imagery. Thus, our system can be applied to a wide variety of images.

3 TECHNICAL APPROACH

In our GP based approach, individuals are composite operators, which are represented by binary trees. The search space of GP is the space of all possible composite operators. The space is very large. In order to illustrate this, consider only a special kind of binary tree, where each tree has exactly 30 internal nodes and one leaf node and each internal node has only one child. For 17 primitive operators and only one primitive feature image, the total number of such trees is 17^{30}. It is extremely difficult to find good operators from this vast space unless one has a smart search strategy.

3.1 DESIGN CONSIDERATIONS

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.

- **The Set of Terminals:** The set of terminals used in this paper are seven primitive feature images generated from the original image: the first one is the original image; the others are mean and standard deviation images obtained by applying templates of sizes 3×3, 5×5 and 7×7. These images are the input to the composite operators. GP determines which operations are applied on them and how to combine the results. To get the mean image, we translate the template across the original image and use the average pixel value of the pixels covered by the template to replace the pixel value of the pixel covered by the central cell of the template. To get the standard deviation image, we just compute the square root of the pixel value difference between the pixel in the original image and its corresponding pixel in the mean image.

- **The Set of Primitive Operators:** 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.

In the following, A and B are images of the same size and c is a constant. For operators such as ADD_OP, SUB_OP, MUL_OP, etc that take two images as input, the operations are performed on the pixel-by-pixel basis.

1. ADD_OP: A + B. Add two images pixel by pixel.
2. SUB_OP: A – B. Subtract image B from image A.
3. MUL_OP: A * B. Multiply images A and B.
4. DIV_OP: 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).
5. MAX2_OP: A max B. The pixel in the resultant image takes the larger pixel value of images A and B.
6. MIN2_OP: A min B. The pixel in the resultant image takes the smaller value of pixels in images A and B.
7. ADD_CONST_OP: A + c. Increase pixel value by c.
8. SUB_CONST_OP: A - c. Decrease pixel value by c.
9. MUL_CONST_OP: A * c. Multiply pixel value by c.
10. DIV_CONST_OP: A / c. Divide pixel value by c.
11. SQRT_OP: sqrt(A). For each pixel p with value v, if $v \geq 0$, change its value to $\sqrt{v}$. Otherwise, to $-\sqrt{-v}$.
12. LOG_OP: log(A). For each pixel p with value v, if $v \geq 0$, change its value to log(v). Otherwise, to –log(-v).
13. MAX_OP: max(A). Replace the pixel value by the maximum pixel value in a 3×3, 5×5 or 7×7 neighborhood.
14. MIN_OP: min(A). Replace the pixel value by the minimum pixel value in a 3×3, 5×5 or 7×7 neighborhood.
15. MED_OP: med(A). Replace the pixel value by the median pixel value in a 3×3, 5×5 or 7×7 neighborhood.
16. REVERSE_OP: rev(A). Reverse the pixel value. Suppose the maximum and minimum pixel values of

image A are Vmax and Vmin respectively. If a pixel has value v, change its value to Vmax – v + Vmin.

17. STDV_OP: stdv(A). Obtain standard deviation image of image A by applying a template of size 3×3, 5×5 or 7×7.

- **The Fitness Measure:** The fitness value of a composite operator is computed in the following way. Suppose G and G' are foregrounds in the ground truth image and the resultant image of the composite operator respectively. Let $n(X)$ denote the number of pixels within region X, then $Fitness = n(G \cap G') / n(G \cup G')$. The fitness value is between 0 and 1. If G and G' are completely separated, the value is 0; if G and G' are completely overlapped, the value is 1.

- **Parameters and Termination:** The key parameters are the population size M, the number of generations N, the crossover rate and the mutation rate.

The GP stops whenever it finishes the pre-specified number of generations or whenever the best operator in the population has fitness value greater than the fitness threshold.

3.2 REPRODUCTION, CROSSOVER AND MUTATION

The GP searches through the space of composite operators to generate new operators, which may be better than the previous ones. By searching through the composite operator space, GP gradually adapts the population of composite operators from generation to generation and improves the overall fitness of the whole population. More importantly, GP may find an exceptionally good operator during the search. The search is done by performing reproduction, crossover and mutation operations. The initial population is randomly generated and the fitness of each individual is evaluated.

The reproduction operation involves selecting a composite operator from the current population. In this research, we use tournament selection, where a number of individuals are randomly selected from the current population and the one with the highest fitness value is copied into the new population.

To perform crossover, two composite operators are selected on the basis of their fitness values. These two composite operators are called parents. One internal node in each of these two parents is randomly selected, and the two subtrees with these two nodes as root are exchanged between the parents. In this way, two new composite operators, called offspring, are created.

In order to avoid premature convergence, mutation is introduced to randomly change the structure of some of the individuals to help maintain the diversity of the population. Once a composite operator is selected to perform mutation operation; an internal node of the binary tree representing this operator is randomly selected, then the subtree rooted at this node is deleted, including the node selected. Another binary tree is randomly generated and this tree replaces the previously deleted subtree. The resulting new binary tree represents a new composite operator. This new composite operator replaces the old one in the population.

3.3 STEADY_STATE AND GENERATIONAL GENETIC PROGRAMMING

In *steady-state GP*, two parental composite operators are selected on the basis of their fitness for crossover. The children of this crossover, perhaps mutated, replace a pair of composite operators with the smallest fitness values. The two children are executed immediately and their fitness values are recorded. Then another two parental composite operators are selected for crossover. This process is repeated until crossover rate is satisfied. In *generational GP*, two composite operators are selected on the basis of their fitness values for crossover. Then, two composite operators with the smallest fitness values, among those that have not been selected for replacement, are selected. They will be replaced by the children of the crossover. At this time, the replacement has not occurred. The above process is repeated until crossover rate is satisfied. A composite operator may be repeatedly selected for crossover, but it cannot be repeatedly selected for replacement. After crossover operations are finished, all the children resulted from the crossover operations replace all the composite operators selected for replacement at once. In addition, we adopt an elitism replacement method that copies the best composite operator from generation to generation. The steady state and generational genetic programming algorithms are given in the following.

- **Steady-state Genetic Programming:**

```
0.  randomly generate population P and evaluate each
    composite operator in P.
1.  for gen = 1 to generation_num do
2.      keep the best composite operator in P.
3.      perform reproduction to generate population P'
        from P.
4.      number_of_crossover = population_size * cross
        over_rate / 2.
5.      for i = 1 to number_of_crossover do
6.          select 2 composite operators from P' based on
            their fitness values for crossover.
7.          select 2 composite operators with the lowest
            fitness values in P' for replacement.
8.          perform crossover operation and let the 2
            offspring composite operators replace the 2
            composite operators selected for replacement.
9.          if  mutation is performed on the composite
                operators from the crossover then
10.             perform mutation on the 2 offspring
                operators with probability mutation_rate.
            end.
```

11. execute the 2 offspring composite operators and evaluate their fitness values.
end // loop 5
12. if mutation is performed on the composite operators from the whole population P' then
13. perform mutation on each composite operator with probability mutation_rate.
14. execute and evaluate mutated composite operators.
end
15. let the best composite operator from population P replace the worst composite operator in P'.
16. let P = P'
17. if the fitness value of the best composite operator in P is above fitness threshold value then
18. stop.
end
end // loop 1

- **Generational Genetic Programming:**

0. randomly generate population P and evaluate each composite operator in P.
1. for gen = 1 to generation_num do
2. keep the best composite operator in P.
3. perform reproduction to generate population P' from P. (crossover and mutation are performed on population P')
*4. number_of_crossover = population_size * crossover_rate / 2.*
*5. perform crossover number_of_crossover times and record 2 * number_of_crossover composite operators to be replaced.*
6. perform mutation on the composite operators generated from crossover or on the composite operators from the whole population. If a composite operator is mutated, recorded it for later execution.
7. execute offspring composite operators from crossover and the mutated composite operators and evaluate their fitness values.
8. put offspring composite operators from crossover in P' and remove the composite operators selected for replacement from P'.
9 let the best composite operator from population replace the worst composite operator in P'.
10. let P = P'
11. if the fitness value of the best composite operator in P is above fitness threshold value then
12. stop.
end
end // loop 1

4 EXPERIMENTS

Various experiments were performed to test the efficacy of genetic programming in extracting regions of interest from real SAR (synthetic aperture radar) images and color images. In this paper, we show some selected examples. It is to be noted that the training and testing images are different and the ground truth is used only during training. In all the experiments, the maximum size of composite operator is 30 and the threshold value used in segmentation is 0.

4.1 REAL SAR IMAGES

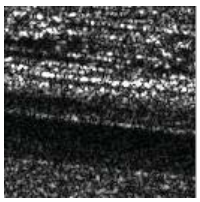
In the four experiments with real SAR images, the population size is 100, the number of generations is 100, the crossover rate is 0.6, the mutation rate is 0.1 and the selection type is tournament selection. In each experiment, GP is invoked ten times with the same parameters and the experimental results from one run and the average performance of ten runs are reported in Table 1. We select the run in which GP finds the best composite operator among the composite operators found in all ten runs to report. The first two rows show the fitness value of the best composite operator and population fitness value (average fitness value of all composite operators in the population) in the initial and final generations in the selected run. The numbers in the parenthesis in the "Best fitness" columns are the fitness values of the best composite operators on the testing SAR images. The last two rows show the average values of the above fitness values over all ten runs. The regions extracted during the training and testing by the best composite operator from the selected run are shown in the following examples.

- **Example 1. Road Extraction:** Three images contain road. The first one contains horizontal paved road and field; the second one contains vertical paved road and grass; the third one contains unpaved road and field. Training is done using the image shown in Figure 1(a) and testing is performed on images shown in Figure 3(a) and 3(c). Figure 1(b) show the ground truth provided by the user, and the white region corresponds to the road.

The generational GP was used to generate a composite operator to extract the road. The fitness threshold value is 0.90. Figure 1(c) shows the output image (corresponding to training image 1(a)) of the best composite operator in the initial population, and Figure 1(d) shows the binary image after segmentation. The output image has both positive pixels in lighter shade and negative pixels in darker shade. Positive pixels belong to the region to be extracted. The fitness value of the best composite operator in the initial population is 0.47 and the population fitness value is 0.19. Figure 1(e) shows the output image of the best composite operator after 100 generations and Figure 1(f) shows the binary image after segmentation. The fitness value of the best composite operator in the final population is 0.92 and the population fitness value is 0.89. The best composite operator has 30 internal nodes and its depth is 21. It has eight leaf nodes, two contains the original image and the other six contain 5×5 mean images,

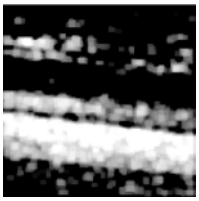
Table 1. The Performance of Genetic Programming on Various Examples of SAR Images.

	Road		Lake		River		Field	
	Best fitness	Population fitness	Best fitness	Population fitness	Best fitness	Population fitness	Best fitness	Population fitness
Initial fitness	0.47	0.19	0.65	0.42	0.43	0.21	0.62	0.44
Final fitness	0.92 (0.92, 0.89)	0.89	0.93 (0.92)	0.92	0.74 (0.84)	0.68	0.87 (0.68)	0.86
Ave. Inital fitness	0.47	0.18	0.73	0.39	0.37	0.11	0.65	0.41
Ave. Final fitness	0.81	0.76	0.92	0.87	0.68	0.58	0.84	0.77



(a) paved road vs. field

(b) ground truth



(c) best initial image

(d) segmented image

(e) best final image

(f) segmented image

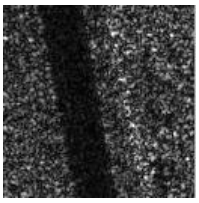
Figure 1. Training real SAR image containing road.

```
(LOG_OP (MIN2_OP (MED_OP (MAX_OP (MAX_OP
(MAX_OP (MAX_OP (MUL_CONST_OP
(DIV_CONST_OP (MAX_OP (MAX_OP
(DIV_CONST_OP (MAX_OP (MAX_OP
(MUL_CONST_OP (MAX_OP (MUL_CONST_OP
(ADD_OP (SUB_CONST_OP (MAX2_OP PF_IM3
PF_IM3)) (LOG_OP (MIN2_OP PF_IM3 (STDV_OP
PF_IM0))))))))))))))))) (DIV_CONST_OP (ADD_OP
(SUB_CONST_OP (MAX2_OP PF_IM3 PF_IM3))
(LOG_OP (MIN2_OP PF_IM3 (STDV_OP
PF_IM0)))))))
```

Figure 2. Learned composite operator tree in LISP notation.

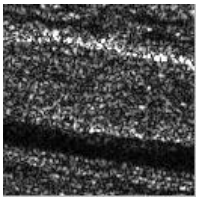
which are very useful in the noise reduction. It is shown in Figure 2, where PM_IM0 is original image and PF_IM3 is 5×5 mean image. It is possible to have a more compact tree representation of this composite operator.

We applied the composite operator obtained in the above training to the other two real SAR images shown in Figure 3(a) and 3(c). Figure 3(b) shows the region extracted by the composite operator from Figure 3(a) and the fitness value of the region, which is 0.92. Figure 3(d) shows the region extracted by the composite operator from Figure 3(c) and the fitness value of the region, which is 0.89.



(a) paved road vs. grass

(b) road detection (fitness 0.92)



(c) unpaved road vs. field

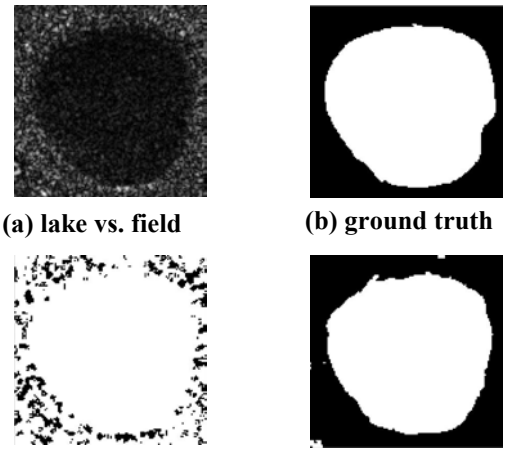
(d) road detection (fitness 0.89)

Figure 3. Testing real SAR images and corresponding road detection results.

- **Example 2. Lake Extraction:** Two SAR images contain lake. The first one contains a lake and field, and the second one contains a lake and grass. Figure 4(a) shows the original image containing lake and field. Figure 4(b) shows the ground truth provided by the user, and the white region corresponds to the lake to be extracted. Figure 5(a) shows the image containing lake and grass.

We used the SAR image containing the lake and field as the training image and applied the composite operator generated by GP to the SAR image containing the lake and grass. The steady-state GP was used to generate the composite operator and the fitness threshold value is 0.95. Figure 4(c) shows the region extracted by the best composite operator in the initial population after segmentation. The fitness value of the best composite operator in the initial population is 0.65 and the population fitness value is 0.42. Figure 4(d) shows the region extracted by the best composite operator in the final population (it is found after 65 generations) after segmentation. The fitness value of the best composite operator in the final population is 0.93 and the population fitness value is 0.92.

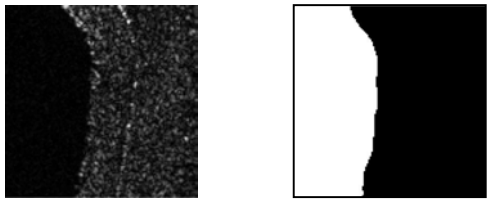
We then applied the composite operator to the image containing a lake and grass. Figure 5(b) shows region extracted and its fitness value 0.92.



(a) lake vs. field **(b) ground truth**

(c) initial segmented image **(d) final segmented image**

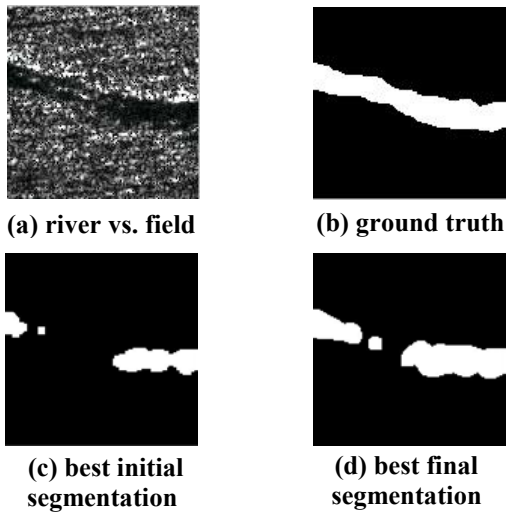
Figure 4. Real SAR image containing lake and field.



(a) lake vs. grass **(b) segmented image (fitness 0.92)**

Figure 5. Testing Real SAR image containing lake and grass.

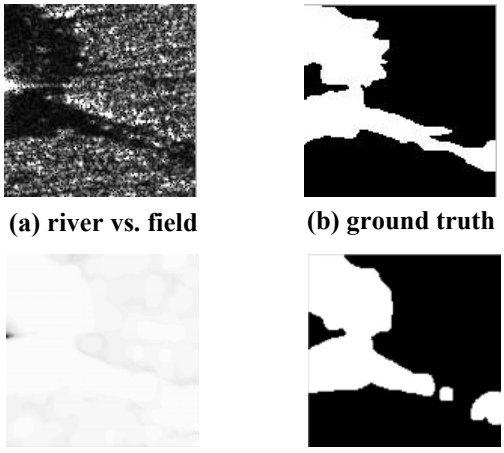
- **Example 3. River Extraction:** We have two SAR images containing river and field. Figure 6(a) and 7(a) show the original images and Figure 6(b) and 7(b) show the ground truth provided by the user. The white region in Figure 6(b) and 7(b) corresponds to the river to be extracted. The SAR image shown in Figure 6(a) was used as the training image by GP. GP generated a composite operator to extract the river in the image. Then the composite operator was applied to the SAR image shown in Figure 7(a) to test its efficacy in extracting the river.



(a) river vs. field **(b) ground truth**

(c) best initial segmentation **(d) best final segmentation**

Figure 6. Training real SAR images containing river.



(a) river vs. field **(b) ground truth**

(c) composite operator output **(d) segmented image (fitnesss 0.84)**

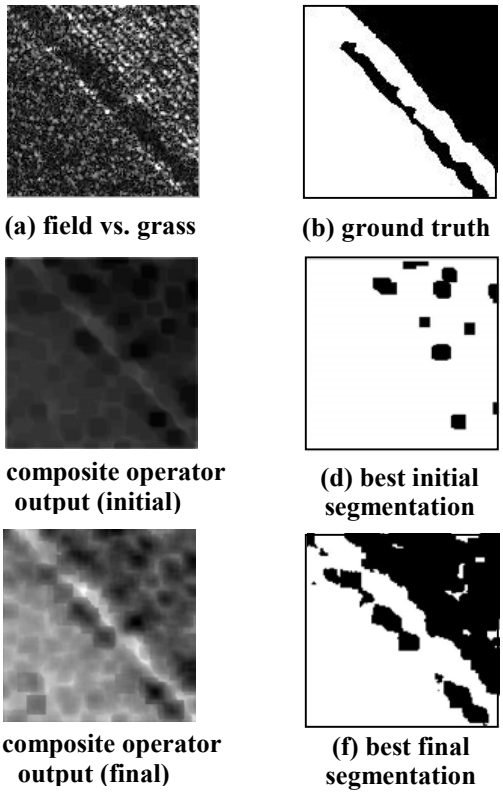
Figure 7. Testing real SAR images containing river.

The steady-state GP was used to generate the composite operator and the fitness threshold value is 0.85. Figure 6(c) shows the region extracted by the best composite operator in the initial population after segmentation. The fitness value of the best composite operator in the initial population is 0.43 and the population fitness value is 0.21. Figure 6(d) shows the region extracted by the best composite operator in the final population (it was found after 40 generations) after segmentation. The fitness value of the best composite operator in the final population is 0.74 and the population fitness value is 0.68. The fitness value of the best composite operator in the final population is not very good. Two reasons account for this. First, the river in Figure 6(a) accounts for only a small percentage of the total area in the image. Second, there are some islands in the river. These islands are similar to the field,

i.e., pixels belong to the islands have similar pixel values to those belong to the field, but they are not excluded from the ground truth.

We applied the composite operator to the image shown in Figure 7(a). Figure 7(c) shows the output image of the composite operator. Figure 7(d) shows the region extracted after segmentation and its fitness value 0.84. This number is larger than the fitness value in the training. The main reason is that the river in Figure 7(a) accounts for a much larger percentage of the total area of the image than that in Figure 6(a).

- **Example 4. Field Extraction:** Two SAR images contain field and grass. Figure 8(a) and 9(a) show the original images and Figure 8(b) and 9(b) show the ground truth. The white region in Figure 8(b) and 9(b) corresponds to the field to be extracted. We consider extracting field from a SAR image containing field and grass as the most difficult task among the four experiments with the SAR images, since the grass and field are similar to each other. We used the SAR image in Figure 8(a) as the training image and applied the composite operator generated by GP to the SAR image in Figure 9(a).



(a) field vs. grass **(b) ground truth**

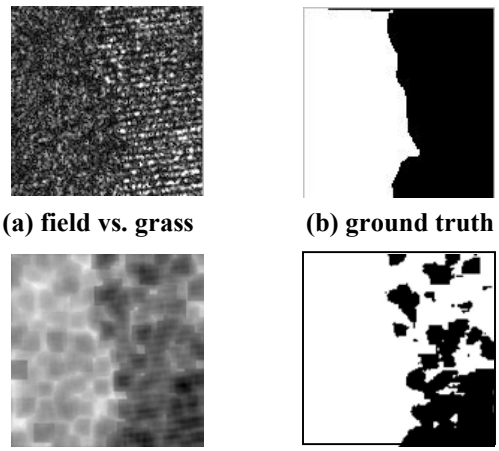
(c) composite operator output (initial) **(d) best initial segmentation**

(e) composite operator output (final) **(f) best final segmentation**

Figure 8. Training real SAR image containing field and grass.

The generational genetic programming was used to generate the composite operator and the fitness threshold value is 0.85. Figure 8(c) shows the output image of the best composite operator in the initial population. The fitness value of the best composite operator in the initial population is 0.62 and the population fitness value is 0.44. Figure 8(d) shows the region extracted after segmentation. Figure 8(e) shows the output image of the best composite operator in the final population and Figure 8(f) shows the region extracted after segmentation. The fitness value of the best composite operator in the final population is 0.87 and the population fitness value is 0.86. Figure 8(c) is very dark. One may not see anything meaningful in this image. The reason is that almost all the pixels in this image have very low pixel values. Some pixels have positive pixel values, but the pixel values are close to 0.

We applied the composite operator to the image in Figure 9(a). Figure 9(c) shows the output image of the composite operator. Figure 9(d) shows the region extracted after segmentation and its fitness value 0.68.



(a) field vs. grass **(b) ground truth**

(c) composite operator output **(d) final segmentation (fitness 0.68)**

Figure 9. Testing real SAR image containing field and grass.

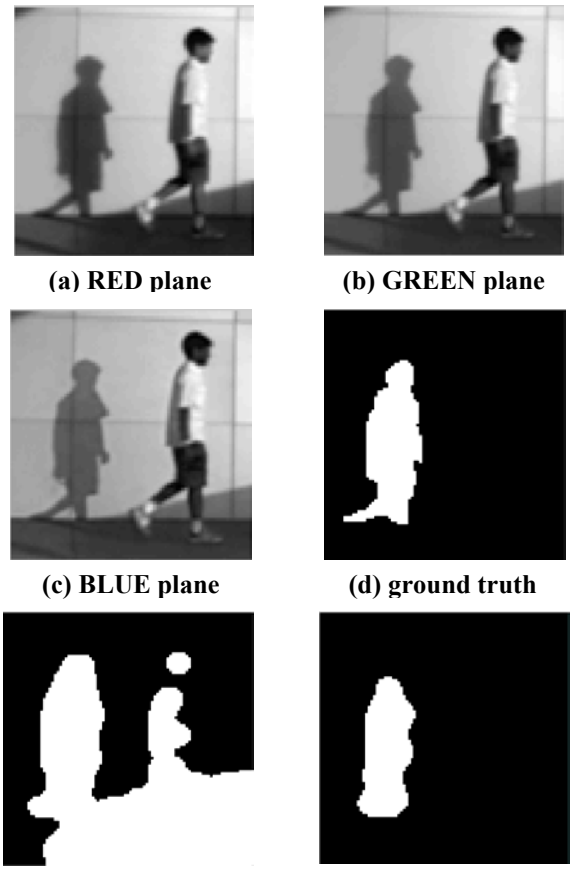
4.2 COLOR IMAGES

In this subsection, we attempt to generate a composite operator to extract the shadow of a person from an RGB color image. The generated composite operator was then tested on two other similar images.

Figure 10 shows the image used for training and the ground truth provided by the user. We don't show a color image, rather the RED, GREEN and BLUE planes of the color image in Figure 10(a), 10(b), 10(c) respectively. The RED, GREEN and BLUE planes of the color image are gray scale intensity images and they are used as primitive feature images in this experiment.

The generational genetic programming was used to generate the composite operator. The population size is 200, the number of generation is 200, the fitness threshold value is 0.80, the crossover rate is 0.1 and the mutation rate is 0.05.

Figure 10(e) shows the region extracted by the best

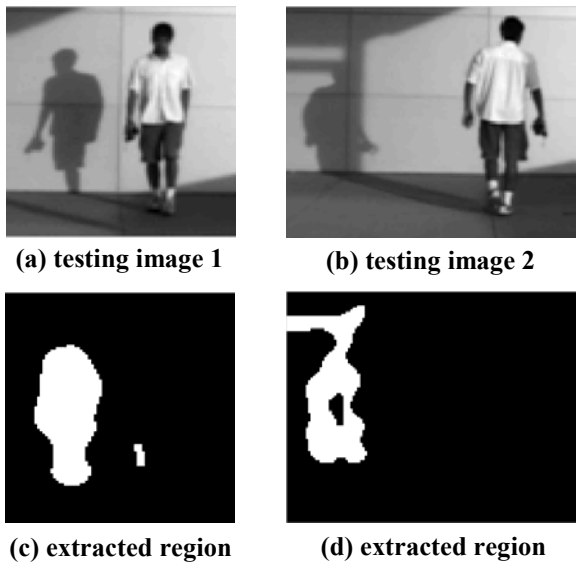


(a) RED plane (b) GREEN plane

(c) BLUE plane (d) ground truth

(e) initial segmentation (f) final segmentation

Figure 10. RED, GREEN and BLUE planes of RGB color image used in training.



(a) testing image 1 (b) testing image 2

(c) extracted region (d) extracted region

Figure 11. GREEN planes of testing RGB color images.

composite operator in the initial population after segmentation. The fitness value of the best composite operator in the initial population is 0.28 and the population fitness value is 0.16. Figure 10(f) shows the region extracted by the best composite operator in the final population after segmentation. The fitness value of the best composite operator in the final population is 0.80 and the population fitness value is 0.76. GP found a good composite operator to extract the shadow.

The composite operator generated by GP was then applied to another two similar color images to test its efficacy in extracting the shadow. The GREEN planes of these two color images are shown in Figures 11(a) and 11(b). When the composite operator is applied to extract shadow regions in these two color images, the RED, GREEN and BLUE planes of the color images are the primitive feature images used by the composite operator. The testing results are shown in Figure 11(c) and Figure 11(d). The fitness values for these two results were 0.76 and 0.54 respectively. It can be seen from these images that the composite operator generated by GP is capable of extracting shadows in the color images similar to the color image used in training.

5 CONCLUSIONS

Our experimental results show that the primitive operators selected by us are effective. GP can find good composite operators to extract the regions of interest in an image and the composite operators can be applied to extract ROIs in other similar images. In our experiments, we did not find any significant difference between the steady-state and generational genetic programming algorithms. In the future, we plan to extend this work by designing smart crossover and mutation operators and discovering new features within the regions of interest for automated object recognition.

Acknowledgement: 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] J. R. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*, MIT Press, 1994.

[2] R. Poli, "Genetic programming for feature detection and image segmentation," in *Evolutionary Computation*, T. C. Forgarty Ed., pp. 110-125, 1996.

[3] S. A. Stanhope and J. M. Daida, "Genetic programming for automatic target classification and recognition in synthetic aperture radar imagery," *Proceeding Conference. Evolutionary Programming VII*, pp. 735-744, 1998.

[4] D. Howard, S. C. Roberts, and R. Brankin, "Target detection in SAR imagery by genetic programming," *Advances in Engg. Software*, 30(5), pp. 303-311, May 1999.

[5] S. C. Roberts and D. Howard, "Evolution of vehicle detectors for infrared line scan imagery," *Proceeding Workshop, Evolutionary Image Analysis, Signal Processing and Telecommunications*, pp. 110-125, 1999.