Multi-objective Genetic Programming for Figure-Ground Image Segmentation

Yuyu Liang(✉), Mengjie Zhang, and Will N. Browne

School of Engineering and Computer Science, Victoria University of Wellington, P.O. Box 600, Wellington 6140, New Zealand
{yuyu.liang,mengjie.zhang,will.browne}@ecs.vuw.ac.nz

Abstract. Figure-ground segmentation is a crucial preprocessing step in areas of computer vision and image processing. As an evolutionary computation technique, genetic programming (GP) can evolve algorithms automatically for complex problems and has been introduced for image segmentation. However, GP-based methods face a challenge to control the complexity of evolved solutions. In this paper, we develop a novel exponential function to measure the solution complexity. This complexity measure is utilized as a fitness evaluation measure in GP in two ways: one method is to combine it with the classification accuracy linearly to form a weighted sum fitness function; the other is to treat them separately as two objectives. Based on this, we propose a weighted sum GP method and a multi-objective GP (MOGP) method for segmentation tasks. We select four types of test images from bitmap, Brodatz texture, Weizmann and PASCAL databases. The proposed methods are compared with a reference GP method, which is single-objective (the classification accuracy) without considering the solution complexity. The results show that the new approaches, especially MOGP, can significantly reduce the solution complexity and the training time without decreasing the segmentation performance.

Keywords: Figure-ground segmentation · Genetic programming · Solution complexity · Multi-objective optimisation

1 Introduction

Figure-ground image segmentation is the process of separating foreground objects or regions of interest from their background. It is considered a crucial preprocessing step, as the results can be input to many higher-level computer vision and image processing tasks, such as object recognition, object tracking and image editing [26]. There are several problems in the existing approaches that include bottom-up [10] and top-down methods [8,9]. Bottom-up methods rely on continuity principles, which are sensitive to factors, such as illumination variations, noise and image contrast [10]. In contrast, top-down methods can learn from prior knowledge, so they can adapt to different image domains. As the figure-ground segmentation often needs certain rules for specific images to achieve accurate

T. Ray et al. (Eds.): ACALCI 2016, LNAI 9592, pp. 134–146, 2016.
DOI: 10.1007/978-3-319-28270-1_12

performance, top-down methods are more preferable. However, top-down methods often require heavy human guidance/work [8,9], making them difficult to be applied in diverse image domains. Moreover, the more human work required, the higher probability of introducing human bias.

Genetic programming (GP) is an evolutionary computation technique inspired by biological evolution [12]. It can handle user-defined tasks automatically by evolving computer programs, and does not require users to specify the form or structure of solutions [19]. Therefore, GP has the potential to evolve good performing segmentors without requiring much human work. Actually, GP has been introduced in the area of image segmentation by several works [15,20,23,24], which show that GP-evolved segmentors can deal with a wide range of images and achieve accurate segmentation results in certain domains [15,20,23,24]. For evolutionary algorithms [14], particularly GP, it is difficult to control the complexity of evolved solutions [22]. Program sizes can grow without (significant) corresponding increases in fitness, which is known as bloat [19].

Parsimony pressure is a simple and widely-used way to control bloat [5]. One kind of parsimony pressure methods is to penalize the fitness of programs based on the program size, and combine multiple objectives to form a single fitness function (known as weighted sum methods). For example, $f = \sum_i w_i * f_i$, where w_i represent the weight of the ith fitness function f_i. Zhang et al. [25] propose a fitness function in GP for object detection problems. This fitness function is a linear combination of detection rate, false alarm rate and false alarm area. The false alarm area is defined as the number of false alarm pixels which are incorrectly detected as object centres without clustering. Results show that this fitness function can reflect the smoothness of evolved programs, and this GP based method performs well on small and regular objects with uncluttered backgrounds. Alternative parsimony pressure methods modify the selection process to lean towards individuals with smaller sizes among individuals with equal fitnesses or rank (known as lexicographic parsimony pressure methods) [16]. In this paper, we propose a weighted sum method with a new fitness function.

The solution complexity can also be treated as a separate objective to tackle the bloat problem. They are multi-objective optimisation algorithms, which aim to evolve a Pareto front of tradeoff solutions based on all the objectives. Shao et al. [22] utilize multi-objective GP (MOGP) to develop global feature descriptors for image classification tasks. There are two objectives in this paper, which are the classification error and the tree complexity. The proposed method achieves better classification accuracies than many state-of-the-art feature extraction techniques, including local binary patterns (LBP) and Gabor filters. One problem is that the MOGP method requires a long time for evolution (e.g. 7.6 h on Caltech-101 dataset). Sarro et al. [21] formulate the effort estimation problem as an optimisation problem. They compare single-objective GP with five different fitness functions (e.g. mean magnitude of relative error) and the multi-objective GP considering the five functions simultaneously. It is concluded that single-objective GP with certain fitness functions can achieve comparable results with those produced by MOGP.

This paper aims to develop two new figure-ground image segmentation methods based on weighted sum GP and MOGP respectively. Both of them take the classfication accuracy and the solution complexity into account. One conducts a weighted sum of them to form a single fitness function, while the other keeps them separately and optimises the two objectives simultaneously using the Pareto front approach. To investigate whether the new approaches can perfrom well, we will test them on a sequence of figure-ground segmentation problems with increasing difficulties, and compare them with a reference GP-based approach, which takes the classfication accuracy as the single objective. Specifically, we investigate the following objectives:

1. whether the new complexity control weighted sum and multi-objective methods can outperform the reference GP method that does not control the complexity, and
2. which of the two new approaches can better perform the image segmentation tasks.

The rest of this paper is organized as follows. Section 2 introduces the reference GP-based method, which includes the algorithm framework, GP settings and the fitness function. Section 3 introduces the two new approaches: the weighted sum and multi-objective GP methods. Section 4 describes experiment preparations, including image datasets and three evaluation measures. In Sect. 5, results on four datasets produced by two new methods are analyzed and compared with that of the reference GP. Conclusions are drawn in Sect. 6.

2 Reference GP Method

As shown in Fig. 1, the segmentation system of the reference method has three major phases. Firstly, a binary classifier is evolved by GP. In this step, an equal number of sub-images, labeled as class A (objects) or B (background), are captured from objects and background in the training images. Features are extracted from these sub-images, which are raw pixel values in this paper. Based on the training samples, GP evolves binary classifiers that can classify sub-images as objects or background. Secondly, a shifting window is utilized to sweep across test images, capturing sub-images that have the same size as those in the first step. Next, the feature extraction is conducted to form the test samples, which can be categorized as class A or B by the evolved classifier. Finally, since the shifting window has overlaps, pixels in test images may have more than one assigned label. We apply a majority voting scheme to determine the estimated label of each pixel and produce the segmentation results.

2.1 GP Settings

In this paper, raw pixel values are directly used as input to the system, which makes the feature extraction phase simpler and time-saving. Table 1 displays the function set. It consists of four standard arithmetic operators and five conditional

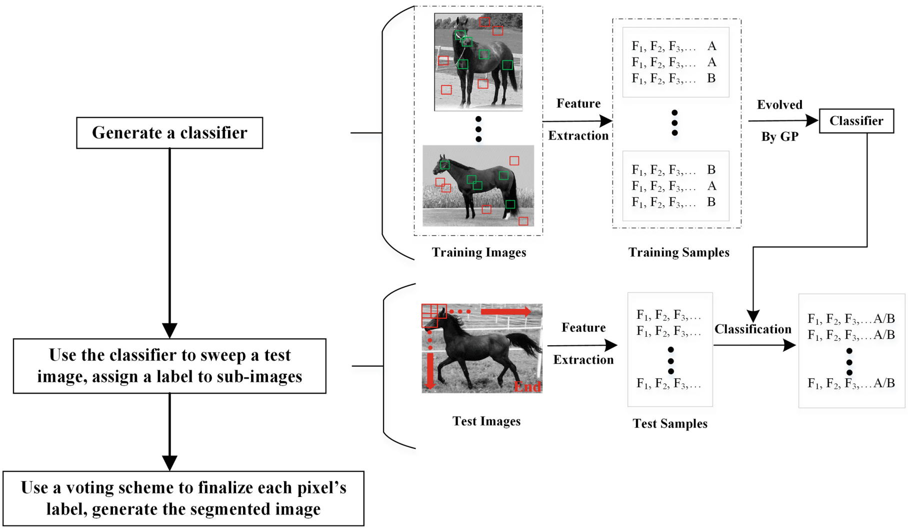


Fig. 1. The framework of GP based figure-ground segmentation method.

Table 1. Function set.

Function Name	Definition
$\mathrm{Add}(a_1, a_2)$	$a_1 + a_2$
$\mathrm{Sub}(a_1, a_2)$	$a_1 - a_2$
$\mathrm{Mul}(a_1, a_2)$	$a_1 * a_2$
$\mathrm{Div}(a_1, a_2)$	$\begin{cases} a_1/a_2 \text{ if } a_2! = 0 \\ 0 \quad \text{ if } a_2 == 0 \end{cases}$
$\mathrm{IF}(a_1, a_2, a_3)$	$\begin{cases} a_2 \text{ if } a_1 \text{ is true} \\ a_3 \text{ if } a_1 \text{ is false} \end{cases}$
$<= (a_1, a_2)$	$\begin{cases} true \text{ if } a_1 <= a_2 \\ false \text{ if otherwise} \end{cases}$
$>= (a_1, a_2)$	$\begin{cases} true \text{ if } a_1 >= a_2 \\ false \text{ if otherwise} \end{cases}$
$== (a_1, a_2)$	$\begin{cases} true \text{ if } a_1 == a_2 \\ false \text{ if otherwise} \end{cases}$
$\mathrm{Between}(a_1, a_2, a_3)$	$\begin{cases} true \text{ if } a_2 <= a_1 <= a_3 \\ false \text{ if otherwise} \end{cases}$

operators, all of which are simple and easy to be calculated. We set the population size to 500, and use crossover and mutation as reproduction operators, whose rates are 90 % and 10 % respectively. The other GP set-up parameters follow the settings used by Koza [12].

2.2 Fitness Function

As a simple and effective evaluation measure, the classification accuracy (shown in Eq. 1) is commonly used as the fitness function for evolutionary algorithms based classification problems [4]. It is employed in our reference method as the single objective.

$$f_1 = \frac{Number.of.correctly.classified.samples}{Number.of.total.training.samples} * 100\,\%. \tag{1}$$

3 New Methods with Solution Complexity Control

Based on the reference GP method, two new approaches are introduced to control the solution complexity. The solution complexity is measured by a novel exponential function. The difference between these two approaches is how to use this complexity measure.

3.1 Weighted Sum Method

To control the complexity/size of evolved solutions, this problem is considered along with the classification accuracy (f_1). The function selected to measure the solution size can be added to f_1, which means both of them should be monotonic and have the same value range. In this paper, we apply an expontional function $p(x) = exp(-\beta * x)$, where β is a scaling factor. For GP, the solution size ($size$) can be calculated by adding the number of terminals and functions in a program. $p(size)$ belongs to the range $(0, 1]$ and is monotonically decreasing, the same as f_1. Therefore, they meet the above requirements.

For the weighted sum GP method, $p(size)$ is combined with the accuracy linearly as a size penalty part to adjust the original individual's fitness evaluation (shown in Eq. 2). Based on our observation, the size of solutions evolved by the reference GP method can be dozens or hundreds. Due to the fact that $exp(-x)$ is close to 0 with the increase of x, which makes complexity control meaningless. Therefore, we introduce a weighting factor (β) and set it to 0.01.

$$f_2 = \alpha * Accuracy + (1 - \alpha) * exp(-\beta * size). \tag{2}$$

where α is a weight factor between the classification accuracy and the size penalty part ($\alpha \epsilon (0, 1]$). $size$ represents the size of a program or solution.

3.2 Multi-objective Method

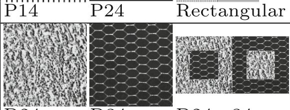
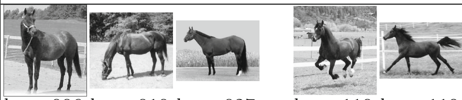
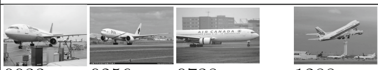
For the multi-objective method, $p(size) = exp(-\beta * size)$ is utilized as an independent objective from the classification accuracy. It aims to find a set of the best trade-off solutions, called Pareto front, between the two objectives. Nondominated sorting genetic algorithm II (NSGA-II) [11] is a well-known technique for multi-objective optimisation. It employs a genetic algorithm (GA) as the search algorithm to evolve Pareto fronts using multiple objectives. Based on NSGA-II, we develop nondominated sorting genetic programming (NSGP), in which GP replaces GA as the evolutionary algorithm to search for Pareto fronts.

4 Experiment Preparation

4.1 Datasets

Four types of images are selected to test the evolved segmentors in this paper. These images have different difficulty levels for the segmentation task. As described in Table 2, they include bitmap, texture and object images. Specifically, the bitmap image, named as "Rectangular", is sythesized from two bitmap patterns (P14 and P24) [3], and is a binary image. The texture image, D24vs34, is a grayscale image that is synthesized from two Brodatz textures (D24 and D34) [1]. In addition, the horse images from the Weizmann dataset [7] and the passenger air-plane images from the PASCAL dataset [2] are images containing certain objects and complex backgrounds. The Weizmann images contain one horse object per image and the objects vary in the position (standing, running and eating). The passenger air-plane images have the largest sizes and vary in object shapes and sizes. Moreover, some air-plane images contain multiple objects. Empirically, object images with larger sizes and complex variations are more difficult to be segmented accurately than binary or texture images [18]. Therefore, PASCAL images are considered as the most difficult images for segmentation tasks; while bitmap images are the simplest ones.

Table 2. Four types of images used in this paper.

Database	Images	Descriptions
Bitmap	 P14 P24 Rectangular	Size: 256*256 Synthetic, binary images
Brodatz	 D24 D34 D24vs34	Size: 320*160 Grayscale images
Weizmann	 horse006 horse010 horse027 horse110 horse119  horse121 horse122 horse159 horse165 horse317	Average Size:248*211 Real images Varing horse positions One object per image
PASCAL (Name prefix: 2007_00)	 0033 0256 0738 1288  1761 2099 2266 2376	Average Size:500*350 Real images Varing object sizes Multiple objects per image

4.2 Evaluation Measures

The segmentation accuracy (Eq. 3) is applied as an evaluation measure in this paper, as it is simple and commonly-used. However, it may not be sufficient to reflect the real segmentation performance in all cases. For example, when the background takes up a large proportion of an image, even though all the objects are incorrectly segmented as background, the segmentation accuracy would still be quite high. Therefore, another two evaluation methods, F_1 measure (Eq. 4) and NRM (negative rate metric, Eq. 5), are considered here to compensate it. F_1 combines precision and recall together; while NRM takes mismatches between a prediction and the ground truth into account [6].

The segmentation accuracy and F_1 reach its worst at 0 and best at 1; while NRM is worst at 1 and best at 0. To make it easy to analyze the result values obtained from the evaluation measures, we derive CNRM (complementary negative rate metric, shown in Eq. 5) from the NRM. In this way, all the values are the higher the better.

$$SegmentationAccuracy = \frac{TP + TN}{Total.Pixel.Number.In.An.Image} \tag{3}$$

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4}$$

$$NRM = (FNR + FPR)/2;\ CNRM = 1 - NRM \tag{5}$$

where $FNR = \frac{FN}{TP+FN}$, $FPR = \frac{FP}{FP+TN}$, $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$. TP, TN, FP and FN stand for true positives, true negatives, false positives and false negatives respectively. In the context of segmentation, TP represents the pixel number of desired objects that are correctly segmented as objects; TN means the pixel number of non-objects are correctly segmented as background; FP and FN represent the number of non-objects and objects that are incorrectly segmented respectively. FPR and FNR mean false positive rate and false negative rate respectively.

5 Results

In this section, we analyse the results of segmentors evolved by the proposed two approaches. In addition, results are also compared with the reference GP method. We set the size of sliding window to 4 for bitmap images and 16 for other test images; therefore, the feature dimensions are 16 for bitmap images and 256 for other images. Both the shifting step in horizontal and vertical direction are set to 2.

For bitmap and texture images, the training set has 1000 samples, in which there are 500 samples for each bitmap (or texture) patterns. The test images are shown in Table 2, named as Rectangular (the bitmap image) and D23vs34 (the texture image). Due to the limited number of the Weizmann and PASCAL images, the leave-one-out (LOO) cross-validation [13] is employed. Two hundred samples are extracted from each Weizmann image (1800 in total). Considering PASCAL images are much larger, 500 sampels are extracted from each PASCAL image (3500 in total). Each experiment has 30 independent runs, and final results are the average of those from 30 runs.

5.1 The Weighted Sum GP Method

As shown in Fig. 2, evaluation measures reach the highest when the weights are between 0.95 to 1.0 on all the datasets, except the CNRM measure on texture images. Since we attach more importance to the segmentation performance than the solution size, even though weights less than 0.95 may lead to lower solution sizes, we will focus on the weight range [0.95,1.0] in the following experiments.

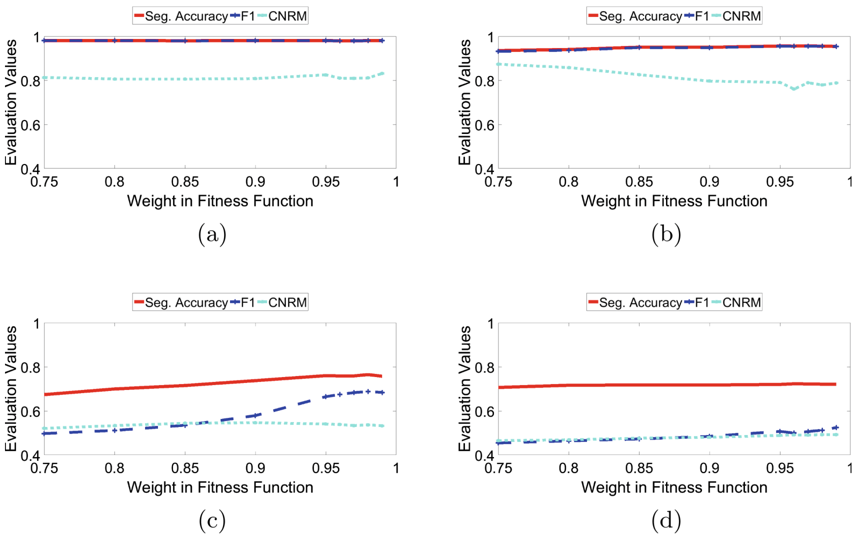


Fig. 2. Performance with different weights in The Fitness Function. (a) on a bitmap image; (b) on a texture image; (c) on Weizmann images; (d) on PASCAL images.

Results on different datasets are displayed in Table 3. In the tables, $TrainTime(s)$ means the average training time (second) for one GP run, which is determined by the number of training samples, the dimension of features and specific learning algorithms; $TestTime(s)$ represents the average time cost to segment one test image determined by the size of test images and specific algorithms used.

When testing on the bitmap image, the proposed weighted sum methods reduce the program size from 76 to around 12, which costs less than one-third training time per run of that of the reference method. For texture images, the fitness function with weight 0.95 performs best among the weighted sum functions. Compared with the reference method, both the size of solutions and training

Table 3. Statistical results of weighted sum method (Acc. represents accuracy).

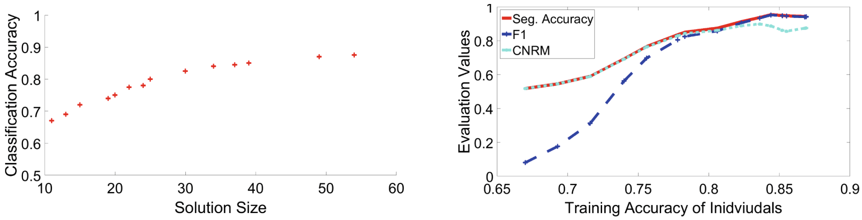
Dataset	Fitness Measure		Segmentation Acc. (%)	F_1	CNRM	Size	Train Time (s)	Test Time (s)
Bitmap	Reference: f_1		98.12 ± 0.22	0.98	0.82	76	15.24	0.012
	Weighted Sum: f_2	$\alpha = 0.99$	98.07 ± 0.28	0.98	0.83	13	4.46	0.009
		$\alpha = 0.98$	98.05 ± 0.24	0.98	0.81	12	4.14	0.009
		$\alpha = 0.97$	97.94 ± 0.20	0.98	0.81	12	4.18	0.009
		$\alpha = 0.96$	97.96 ± 0.18	0.98	0.81	12	4.04	0.009
		$\alpha = 0.95$	98.06 ± 0.22	0.98	0.83	12	3.93	0.009
Texture	Reference: f_1		94.98 ± 2.75	0.95	0.81	343	23.86	0.042
	Weighted Sum: f_2	$\alpha = 0.99$	95.44 ± 0.93	0.95	0.79	260	21.40	0.036
		$\alpha = 0.98$	95.55 ± 0.95	0.96	0.78	202	18.48	0.033
		$\alpha = 0.97$	95.39 ± 1.70	0.95	0.78	163	15.85	0.030
		$\alpha = 0.96$	95.54 ± 1.19	0.95	0.78	132	14.48	0.029
		$\alpha = 0.95$	95.61 ± 0.87	0.96	0.79	127	13.38	0.030
Weizmann	Reference: f_1		75.76 ± 8.37	0.67	0.54	303	49.74	0.036
	Weighted Sum: f_2	$\alpha = 0.99$	75.75 ± 7.48	0.68	0.53	225	37.80	0.032
		$\alpha = 0.98$	76.46 ± 7.10	0.69	0.54	175	30.29	0.029
		$\alpha = 0.97$	75.89 ± 6.94	0.68	0.53	133	25.51	0.026
		$\alpha = 0.96$	75.87 ± 7.11	0.67	0.54	103	21.18	0.025
		$\alpha = 0.95$	76.01 ± 6.91	0.66	0.54	80	18.07	0.022
PASCAL	Reference: f_1		71.84 ± 10.63	0.51	0.49	321	129.82	0.129
	Weighted Sum: f_2	$\alpha = 0.99$	72.10 ± 8.26	0.52	0.49	176	93.45	0.103
		$\alpha = 0.98$	72.14 ± 8.59	0.51	0.49	100	80.29	0.090
		$\alpha = 0.97$	72.24 ± 8.35	0.51	0.49	77	55.19	0.072
		$\alpha = 0.96$	72.10 ± 9.16	0.50	0.49	59	50.32	0.073
		$\alpha = 0.95$	72.01 ± 8.68	0.51	0.49	43	44.24	0.063

time almost halved. On Weizmann images, the weighted sum methods with the weight factor of 0.98 and 0.96 perform better than the reference function in all the three measures, especially the one with 0.98. They also spend less training and test time. On PASCAL images, they achieve generally better performance under all the three evaluation measures than the reference method. For example, the function with the weight of 0.95 achieves a little higer accuracy and the same F_1 and CNRM values. However, the training time per GP run only takes up one third of that for the reference method.

Compared with the reference GP method, the proposed weighted sum approach, which adds penalty to the program complexity in the fitness function, produces similar or even better results. More importantly, as the sizes of evolved programs dereases significantly, both the training time and the test time are reduced. Since the test time is quite low already, the changes of the training time are much more obvious.

5.2 The NSGP Method

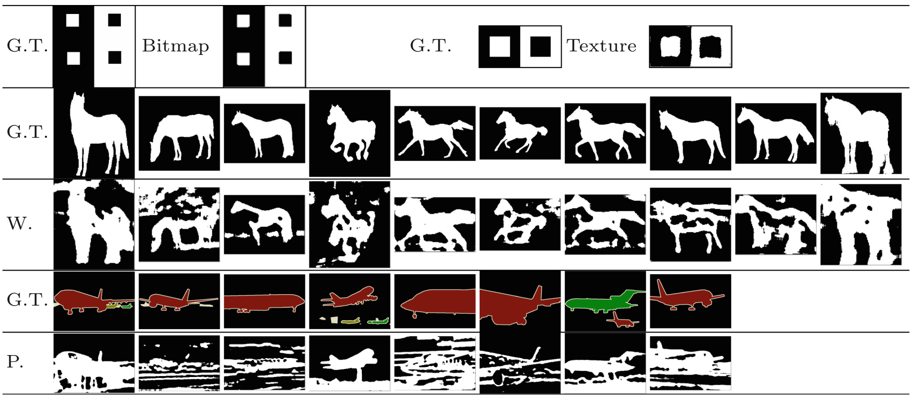
In this section, we conduct segmentation experiments using the proposed multi-objective approach – NSGP. Considering if the size of an evolved program is too small (e.g. 5), this solution is normally too simple and not effective to solve the complex segmentation task, so we set an restriction to the objective of the complexity measure ($p(size)$). Specifically, if an individual's size is less than 10, this objective will be given the lowest value 0. Fig. 3a shows the Pareto front produced in one GP run using texture samples. The Pareto front provides insights into the tradeoff of the two objectives (the classfication accuracy and the solution complexity) for the segmentation problem. As there is normally a large number of solutions on the Pareto front generated in the training process, solutions need to be selected based on our preference to segment the test images.



(a) (b)

Fig. 3. (a) NSGP Pareto front evolved using texture training samples; (b) Performance of solutions from this pareto front on texture image.

Table 4. Result examples of NSGP method (G.T. means ground truth; Bitmap, Texture, W. and P. represent result examples of Bitmap, Texture, Weizmann and PASCAL images respectively).



Between the classification accuracy and the solution complexity, we lean towards the former one. We assume that front solutions, generated on the training set, with higher classfication accuracies produce better results on the test dataset. This assumption can be testified by Fig. 3b, which shows that the values of three evaluation measures on the test dataset grow as the increase of front solutions' accuracies. Therefore, we select 20 solutions with the highest accuracies along the front in each GP run of the training process, and use them to segment test images. The best one of the 20 results is used to represent the algorithm's performance of this GP run. After 30 runs, we calculate the average performance.

Table 4 displays one example result of each test image using the NSGP approach. It can be seen that for bitmap and texture images, different patterns/textures have been accurately separated. For Weizmann and PASCAL images, even though some examples do not have clear object boundaries, objects are located accurately and the results are promising. Table 5 compares NSGP with the weighted sum method and the reference method. The weighted sum methods with certain weighting factors are selected due to their better performance. Based on the Mann-Whitney U-Test [17] with the significance level 5 %, NSGP achieves similar results to those of the reference method and the weighted sum method. However, compared with the weighted sum mentod, the solution sizes are further reduced for segmentation tasks. It leads to a further decrease in the training time. For example, the training time cost per GP run reduces two thirds on the texture image and around half on the Weizmann images.

Table 5. Statistical results of NSGP method (Seg. Acc. represents segmentation accuracy).

Dataset	Fitness Measure	Seg. Acc. (%)	F_1	CNRM	Size	Train Time (s)	Test Time (s)
Bitmap	Reference: f_1	98.12 ± 0.22	0.98	0.82	76	15.24	0.012
	WeightedSum:$\alpha = 0.95$	98.06 ± 0.22	0.98	0.83	12	3.93	0.009
	NSGP	98.12 ± 0.23	0.98	0.82	12	2.47	0.029
Texture	Reference: f_1	94.98 ± 2.75	0.95	0.81	343	23.86	0.042
	WeightedSum:$\alpha = 0.95$	95.61 ± 0.87	0.96	0.79	127	13.38	0.030
	NSGP	94.57 ± 2.67	0.94	0.82	38	4.72	0.033
Weizmann	Reference: f_1	75.76 ± 8.37	0.67	0.54	303	49.74	0.036
	WeightedSum:$\alpha = 0.98$	76.46 ± 7.10	0.69	0.54	175	30.29	0.029
	NSGP	76.83 ± 5.92	0.65	0.56	39	16.36	0.039
PASCAL	Reference: f_1	71.84 ± 10.63	0.51	0.49	321	129.82	0.129
	WeightedSum:$\alpha = 0.97$	72.24 ± 8.35	0.51	0.49	77	55.19	0.072
	NSGP	73.31 ± 8.09	0.51	0.49	49	28.78	0.140

6 Conclusions and Future Work

This paper developed a weighted sum GP method and a multi-objective GP method (NSGP) for the figure-ground image segmentation. To control solution complexity, an exponential function was designed as an additional objective to the function of the classification accuracy. The two functions were combined linearly in the weighted sum method, and treated separately as two objectives in NSGP.

A GP method, which took the classfication accuracy as the single objective, was employed as a reference method. Compared with it, the two proposed approaches with the complexity control achieved similar results in terms of three evaluation measures (the segmentation accuracy, F_1 and CNRM). However, both of them reduced the size of evolved solutions, leading to a significant decrease in the training time. In particular, the NSGP produced even smaller solutions than that of weighted sum method with similar segmentation performance. This indicates that considering the solution complexity the two new approaches outperformed the reference GP. Moreover, NSGP is more powerful in reducing solution complexity than the weighted sum method without reducing the segmentation performance.

In this paper, we used raw pixel values as input of GP directly. Since certain image features, such as Gabor filters, are powerful image descriptors, we will consider more powerful image features in the future, from which GP may evolve segmentors with better performance.

References

1. Brodatz texture database. http://multibandtexture.recherche.usherbrooke.ca/original_brodatz.html
2. The pascal visual object classes homepage. http://pascallin.ecs.soton.ac.uk/challenges/VOC/
3. Song, V.C.A.: Texture segmentation by genetic programming. Evol. Comput. **16**(4), 416–481 (2008)
4. Al-Sahaf, H., Song, A., Neshatian, K., Zhang, M.: Extracting image features for classification by two-tier genetic programming. In: 2012 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. IEEE (2012)
5. Alex, A.: Summary of parsimony pressure made easy. https://wiki.umn.edu/pub/UmmCSci4553s09/ResearchPaperGroupsAndTopics/ppMadeEzSummary.pdf
6. Ashburner, J., Friston, K.J.: Unified segmentation. Neuroimage **26**(3), 839–851 (2005)
7. Borenstein, E.: Weizmann horse database. http://www.msri.org/people/members/eranb/
8. Borenstein, E., Sharon, E., Ullman, S.: Combining top-down and bottom-up segmentation. In: Proceedings IEEE Workshop on Perceptual Organization in Computer Vision, pp. 1–8 (2004)
9. Borenstein, E., Ullman, S.: Class-specific, top-down segmentation. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002, Part II. LNCS, vol. 2351, pp. 109–122. Springer, Heidelberg (2002)
10. Cote, M., Saeedi, P., et al.: Hierarchical image segmentation using a combined geometrical and feature based approach. J. Data Anal. Inf. Process. **2**(04), 117 (2014)
11. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)

12. Koza, J.R.: Genetic Programming: On the Programming of Computers by Natural Selection. MIT Press, Cambridge (1992)
13. Kuhn, M.: Futility analysis in the crossvalidation of machine learning models, pp. 1–22 (2014). arXiv:1405.6974
14. Liang, Y., Zhang, M., Browne, W.N.: Image segmentation: a survey of methods based on evolutionary computation. In: Dick, G., Browne, W.N., Whigham, P., Zhang, M., Bui, L.T., Ishibuchi, H., Jin, Y., Li, X., Shi, Y., Singh, P., Tan, K.C., Tang, K. (eds.) SEAL 2014. LNCS, vol. 8886, pp. 847–859. Springer, Heidelberg (2014)
15. Liang, Y., Zhang, M., Browne, W.N.: A supervised figure-ground segmentation method using genetic programming. In: Mora, A.M., Squillero, G. (eds.) Applications of Evolutionary Computation. Lecture Notes in Computer Science, vol. 9028, pp. 491–503. Springer, Heidelberg (2015)
16. Luke, S., Panait, L.: Lexicographic parsimony pressure. In: Proceedings of GECCO-2002, pp. 829–836. Morgan Kaufmann Publishers (2002)
17. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. **18**, 50–60 (1947)
18. Poli, R.: Genetic programming for feature detection and image segmentation. Evol. Comput. **1143**, 110–125 (1996)
19. Poli, R., Langdon, W.B., McPhee, N.F.: A Field Guide to Genetic Programming. Published via http://lulu.com and freely available at http://www.gp-field-guide.org.uk, UK (2008)
20. Poli, R.: Genetic Programming for feature detection and image segmentation. In: Fogarty, T.C. (ed.) AISB-WS 1996. LNCS, vol. 1143, pp. 110–125. Springer, Heidelberg (1996)
21. Sarro, F., Ferrucci, F., Gravino, C.: Single and multi objective genetic programming for software development effort estimation. In: Proceedings of the 27th Annual ACM Symposium on Applied Computing, pp. 1221–1226. ACM (2012)
22. Shao, L., Liu, L., Li, X.: Feature learning for image classification via multiobjective genetic programming. IEEE Trans. Neural Netw. Learn. Syst. **25**(7), 1359–1371 (2014)
23. Singh, T., Kharma, N., Daoud, M., Ward, R.: Genetic programming based image segmentation with applications to biomedical object detection. In: Proceedings of the 11th Annual conference on Genetic and evolutionary computation, pp. 1123–1130. ACM (2009)
24. Song, A., Ciesielski, V.: Texture segmentation by genetic programming. Evol. Comput. **16**(4), 461–481 (2008)
25. Zhang, M., Andreae, P., Pritchard, M.: Pixel statistics and false alarm area in genetic programming for object detection. In: Raidl, G.R., Cagnoni, S., Cardalda, J.J.R., Corne, D.W., Gottlieb, J., Guillot, A., Hart, E., Johnson, C.G., Marchiori, E., Meyer, J.-A., Middendorf, M. (eds.) EvoIASP 2003, EvoWorkshops 2003, EvoSTIM 2003, EvoROB/EvoRobot 2003, EvoCOP 2003, EvoBIO 2003, and EvoMUSART 2003. LNCS, vol. 2611, pp. 455–466. Springer, Heidelberg (2003)
26. Zou, W., Bai, C., Kpalma, K., Ronsin, J.: Online glocal transfer for automatic figure-ground segmentation. IEEE Trans. Image Process. **23**(5), 2109–2121 (2014)