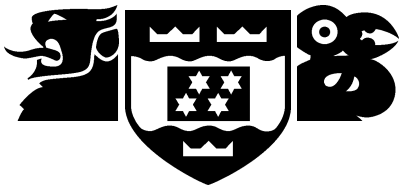
VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui



School of Mathematical and Computing Sciences
Computer Science

New Fitness Functions in Genetic Programming for Object Detection

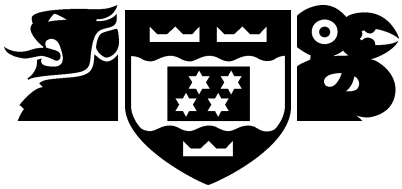
Malcolm Lett, Mengjie Zhang

Technical Report CS-TR-04/12
November 2004

School of Mathematical and Computing Sciences
Victoria University
PO Box 600, Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Email: Tech.Reports@mcs.vuw.ac.nz
http://www.mcs.vuw.ac.nz/research

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui



School of Mathematical and Computing Sciences
Computer Science

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341, Fax: +64 4 463 5045
Email: Tech.Reports@mcs.vuw.ac.nz
http://www.mcs.vuw.ac.nz/research

New Fitness Functions in Genetic Programming for Object Detection

Malcolm Lett, Mengjie Zhang

Technical Report CS-TR-04/12
November 2004

Abstract

Object detection is an important field of research in computer vision which genetic programming has been applied to recently. This paper describes two new fitness functions in genetic programming for object detection. Both fitness functions are based on recall and precision of genetic programs. The first is a tolerance based fitness function and the second is a weighted fitness function. The merits and effectiveness of the two fitness function are discussed. The two fitness functions are examined and compared on three object detection problems of increasing difficulty. The results suggest that both fitness functions perform very well on the relatively easy problem, the weighted fitness function outperforms the tolerance based fitness function on the relatively difficult problems.

Keywords Genetic programming, object detection, object localisation, fitness function.

Author Information

Malcolm Lett is a postgraduate student in computer science and Mengjie Zhang is an academic staff member in computer science. Both authors are in the School of Mathematical and Computing Sciences, Victoria University of Wellington, New Zealand.

1 Introduction

Object detection and classification in computer vision is an important field of research and genetic programming is an exciting and relatively new technique. Object detection is important for many modern tasks, from detecting cancerous cells to human face recognition. It is the process of finding "objects of interest" within images (*Localisation*) and determining the type, or class, of the objects found (*Classification*). The task for localisation is to find the centres of these objects, preferably with high "positional accuracy" (close to the exact centre of the objects).

Genetic programming (GP) is a method which automatically generates programs to solve a particular task [4, 5, 1]. Machine learning methods, such as GP, are generally used in object recognition because the problem is too hard for a human to devise a good solution. The resultant programs in GP, called "genetic programs", are usually represented by simple algebraic expressions which are evolved over a number of "generations". The evolution is guided by a "fitness function" which measures the performance of each genetic program and allows the evolutionary engine to select the best of a population of programs. The evolutionary system is based on Darwinian evolution. Mutation is applied to the genetic programs to introduce new information (the inspirational force). Crossover transforms two programs by switching part of one program with part of another. Reproduction allows the best programs to survive, ensuring that good solutions are not lost.

Finding a good fitness function for a particular task is an important but difficult problem in developing a GP system. Various fitness functions have been devised for object detection, with varying success [1, 7, 6, 3, 8]. These tend to combine many parameters using scaling factors which specify the relative importance of each parameter, with no obvious indication of what scaling factors are good for a given problem. The majority of fitness functions for localisation require clustering to be performed to group multiple localisations of single objects into a single point before the fitness is determined [2, 3, 6]. Other measures are then incorporated in order to include information about the pre-clustered results (such as how many points have been found for each object). The requirement of clustering can make the calculation of the fitness require more processing time than one which does not need this.

The goal of this paper is to investigate two new fitness functions in GP for object detection, in particular localisation. The two fitness functions will be examined and compared on a sequence of object detection/localisation problems of increasing difficulty.

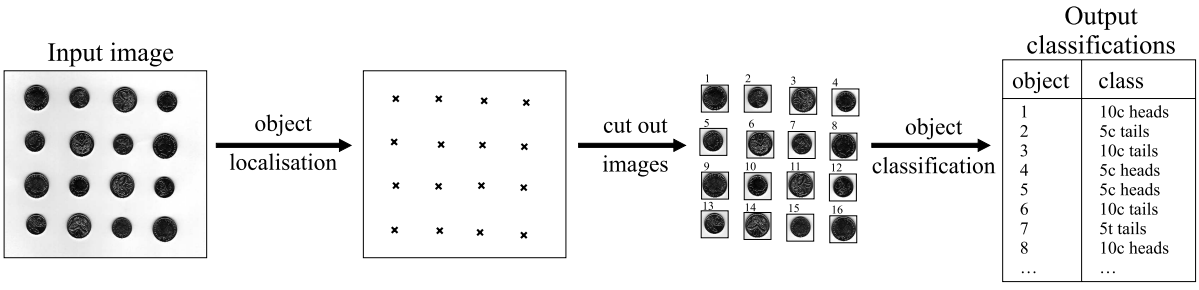


Figure 1: Object Detection Process

The remainder of this document is structured as follows. Section 2 describes the basics of object detection. Section 3 describes the GP approach with the two new fitness functions. Section 4 describes the object detection tasks and the experiment setup. Section 5 presents the results with discussions and we draw conclusions on these in section 7.

2 Object Detection

The process we used for object detection is shown in Figure 1. A raw image is taken and a trained Localiser applied to it, producing a set of points found to be the positions of these objects. Single objects could have multiple positions ("localisations") found for them, however ideally there would be exactly one localisation per object. Sections of the image are then "cut out" at each of the positions specified. Each of these cutouts are then classified using the trained Classifier.

This method treats all objects of multiple classes as a single "object of interest" class for the purpose of localisation, and the classification stage handles attaching correct class labels. This has the advantage that the training is easy for both stages as different fitness functions are used for the training of the two stages. The first is tailed to achieving results as close to the object centres as possible (to achieve high "positional accuracy"), while the second is tailored to making all classifications correct (high "classification accuracy").

The accuracy of the positions found by the localiser are important, as incorrect results from this first stage will need to be handled by the second stage. Any false alarms from the localisation stage could cause problems with the classification stage, unless it is able to classify these as "background". If the positions found are not close enough to the object centres then the classification stage will likely not handle it well.

The object localisation stage is performed by means of a window which sweeps over the whole image, and for each position extracts the features and passes them to the trained localiser. The localiser then determines whether each position is an object or not.

3 Genetic Programming Applied to Object Detection

Our work will focus on object localisation using genetic programming. This approach has a learning process and a testing procedure. In the learning/evolutionary process, the evolved genetic programs use a square input field which is large enough to contain each of the objects of interest. The programs are applied at many sampled positions within the images in the training set to detect the objects of interest. If the program localiser returns a value greater than or equal to zero, then this position is considered the centre of an object of interest; otherwise it is considered background. In the test procedure, the best evolved genetic program obtained in the learning process is then applied, in a moving window fashion, to the whole images in the test set to measure object detection performance.

In this system, we used tree structures to represent genetic programs [4]. The ramped half-and-half method [1] was used for generating programs in the initial population and for the mutation operator. The proportional selection mechanism and the reproduction [6], crossover and mutation operators [1] were used in the learning process.

Similarly to other GP systems for object detection, this system used simple statistical features as the terminal set and the four standard arithmetic operators and a conditional operator as the function set. Rather than using clustering and accurate detection accuracy in the fitness function in previous approaches [2, 8], we developed two new fitness functions based on tolerance and weighted detection measures without clustering, which are discussed in the rest of this section.

3.1 Tolerance Based Fitness Function

Object localisation/detection performance is usually measured by precision and recall of a localisation system or similar measures such as detection rates and false alarm rates. Precision refers to the number of objects correctly localised/detected by a system as a percentage of

the total number of object localised/detected by the system. Recall refers to the number of objects correctly localised by a system as a percentage of total number of target objects in a data set.

During object localisation process, a genetic program might consider many pixel positions in an image as object centres, particularly for those pixels surrounding the target object centres. For presentation convenience, we call each object centre localised in an image by a genetic program a *localisation*.

Our first fitness function is based on the F-measure, a combination of precision and recall, of a program. Rather than using the exact match to measure the number of correctly localised/detected objects, we apply a tolerance of three pixels for this purpose. In other words, if a genetic program produces any localisations within three pixels of a target object centre, then we consider that the program has correctly localised this object; otherwise, the program did not successfully localise this object. Consequently, precision here is the number of objects correctly localised by the program as a percentage of total number of localisations made by the program, and recall is the number of objects correctly localised by the program as a percentage of total number of target objects in the image data set. Thus, this fitness function is called "Within 3-pixel Tolerance of object Centre", and we use WTC_3 for short, it is shown in Equation 1.

$$\text{fitness}_{WTC_3} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{1}$$

where $\mathit{fitness}_{WTC_3}$ is in range $[0, 1]$, and 1.0 means perfect fitness: all objects have been localised and all localisations are within the 3-pixel tolerance of some object.

This method attempts to produce all localisations fitting within some tolerance which is inherent to the ability of the classification stage. Three pixels seem a good choice because a classifier which expects cutouts to be taken at exactly the correct positions may not be realistic.

3.2 Weighted Fitness Function

Unlike the first fitness function, our second fitness function is based on a "Localisation Fitness Weighted 'F' measure" (LFWF), which attempts to acknowledge the worth/goodness of individual localisations made by the genetic program. Instead of using either correct or incorrect to represent a localisation, each localisation is allocated a weight (referred to as the "localisation fitness") which represents its individual worth and counts towards the overall fitness. If all localisations have high localisation fitness, then the overall fitness of this genetic program will be high.

Each weight is calculated based on the distance of the localisation from the centre of the closest object, as shown in Equation 2.

$$\text{localisationFitness} = \begin{cases} 1 - d/r & \text{if } d \leq r \\ 0.0 & \text{otherwise} \end{cases} \tag{2}$$

where d is the distance from object centre, and r is called the "localisation fitness radius", defined by the user. In this system, r is set to 35 pixels, half the square size of the input window, which is also the radius of the largest object.

In order to deal with multiple localisations of objects, the single localisation closest to each object centre is chosen for calculation of localisation fitness, and all other localisations of those object are treated as false alarms. Equations 3 to 5 describe the fitness function. The precision and recall are calculated by taking the localisation fitness for the best localisation of

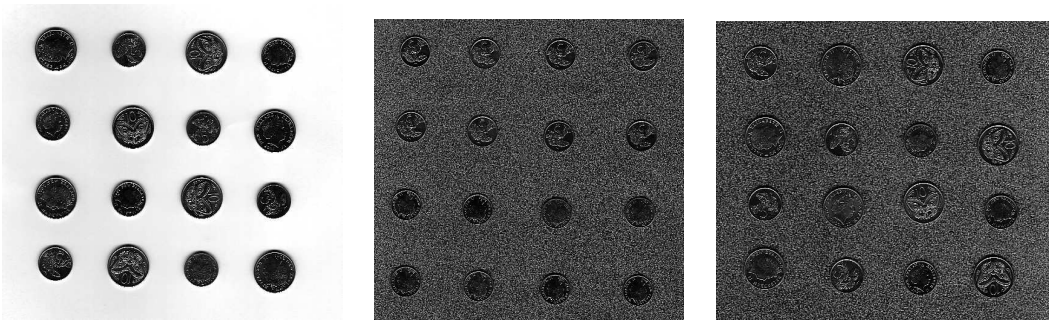
each object and dividing this by the total number of localisations or total number of objects respectively.

$$\text{weightedPrecision} = \frac{\sum_{i=1}^{N} \max(\{\text{localisationFitness}_{i,j}, j = 1..L_i\})}{L} \tag{3}$$

$$\text{weightedRecall} = \frac{\sum_{i=1}^{N} \max(\{\text{localisationFitness}_{i,j}, j = 1..L_i\})}{N} \tag{4}$$

$$\text{fitness}_{LFWF} = \frac{2 \times \text{weightedPrecision} \times \text{weightedRecall}}{\text{weightedPrecision} + \text{weightedRecall}} \tag{5}$$

where N is the total number of objects, L is total number of localisations made, $localisationFitness_{i,j}$ is the localisation fitness of the j-th localisation of object i, L_i is number of localisations made to object i, *weightedPrecision* is the precision in range $[0, 1]$, and 1.0 means all localisations where correct, with no extra localisations of objects, *weightedRecall* is the recall in range $[0, 1]$, and 1.0 means all objects are localised, and $fitness_{LFWF}$ is in range $[0, 1]$, and 1.0 means perfect fitness: all localisations are at the exact object centres with no extra localisations of objects and no false alarms.



(a) (b) (c)

Figure 2: Sample images in the three data sets. (a) Easy; (b) Medium difficulty; (c) Hard.

The fitness function has a number of advantages. Firstly, the main parameter in this fitness function is the *localisation fitness*. This can by easily determined in the way presented here. This has an advantage over other methods which have many parameters. Secondly, it has an advantage over other methods which also apply tolerances on radius, but must use very small tolerances to achieve good results. We will compare it with our first fitness function to check this point. Thirdly, in the previous approaches, before cutouts can be taken for classification, the multiple localisations of each object must be clustered into one

group and its centre found. While this is a relatively easy task, it is time consuming to do during training. Our fitness function does not require clustering to be performed before it is calculated. We expect that this fitness function can produce small areas of localisations which should make clustering efficient, as opposed to large unconstrained areas which could be produced if the fitness is calculated after clustering is performed.

4 Object Detection Tasks and Experimental Setup

We used three image data sets of New Zealand 5 and 10 cent coins in the experiments. Examples are shown in Figure 2. The data sets are intended to provide object localisation problems of increasing difficulty. The first data set (*easy*) contains images of tails and heads of 5 and 10 cent coins against an almost uniform background. The second (*medium difficulty*) is of 5 cent coins against a noisy background, making the task much harder. The third data set (*hard*) contains tails and heads of both 5 and 10 cent coins against a noisy background.

We used 24 images for each data set in our experiments and equally split them into three sets: a training set for learning good genetic programs, a validation set for monitoring the training process to avoid overfitting, and a test set to measure object detection performance. We used a population of 500 genetic programs for evolution in each experiment run. 100 runs were performed for each fitness function on each data set and average results are presented.

To give a fair comparison, the "localisation recall (LR) and precision (LP)" were used to measure the final performance of the genetic programs with the two fitness functions. LR is the number of objects with any correct localisations within the localisation fitness radius as a percentage of the total number of objects, and LP is the number of correct localisations which fall within the localisation fitness radius of any object centres as a percentage of the total number of localisations made. In addition, we also check the "Extra Localisations" (ExtraLocs) for each system to measure how many extra localisation were made for each object.

So LR and LP measure the ability of a program to locate objects to within the localisation fitness radius of their centres, a tolerance of 35 pixels.

5 Results

Table 1 shows the results of the GP systems with the two fitness functions. The first line shows that the GP system with the first fitness function, WTC_3, achieved an average of localisation precision 99.99%, an average localisation recall 98.98%, and resulted in 126.71 extra localisation on average.

Table 1: Results of the GP systems with the two fitness functions.

Dataset	Fitness function	Accuracy		
		LP (%)	LR (%)	ExtraLocs
Easy	WTC_3	99.99	98.98	126.71
	LFWF	99.98	99.35	113.88
Medium	WTC_3	99.80	90.45	135.45
	LFWF	99.90	93.11	133.56
Hard	WTC_3	98.32	78.34	185.39
	LFWF	98.53	87.65	196.61

The results on the easy data set show that both fitness functions achieved almost perfect

performance. Almost all the objects of interest in this data set were successfully localised by the GP system with very few false alarms (both LR and LP are very close to 100%). Although both fitness functions resulted in a number of extra localisations, almost all of them were located within the localisation fitness radius. In particular, the weighted fitness function resulted in a better recall and fewer extra localisations.

The results on the medium difficulty data set show that the GP system with the weighted fitness function achieved better performance in terms of all of the precision, recall, and extra localisations. In particular, the recall performance produced by the weighted fitness function was more than 3% better than the tolerance based fitness function.

The results on the hard data set show a similar pattern to the medium difficult data set, except that the improvement of recall by the weighted fitness function over the tolerance based fitness function was significant (8.7%). Although it resulted in a slightly more extra localisations, almost all of them were within the localisation fitness radius range.

As expected, the performance on the three data sets deteriorated as the degree of difficulty of the localisation problem was increased.

These results suggest that the weighted fitness function is similar to or slightly better than the tolerance based fitness function on relatively easy object localisation problems, but can achieve much better results, particularly recall, on relatively hard object localisation problems. On situations where it is more important to detect all objects of interest than to avoid extra localisations, the weighted fitness function should be used.

6 Conclusions

This paper described two new fitness functions in genetic programming for object detection, particularly object localisation problems. The first fitness function was developed based on precision and recall of a genetic program within a range of three-pixel tolerance, and the second fitness function was developed based on weighted precision and weighted recall of the genetic program. The GP systems with the two fitness functions were examined and compared on a sequence of object localisation problems of increasing difficulty. The results suggest that both fitness functions performed very well on the easy data set and the weighted fitness function outperformed the tolerance based fitness function on all problems particularly the two relatively difficult object localisation problems.

The results also showed that the weighted fitness function achieved a much better recall on the medium difficulty and the hard problems. This suggests that the weighted fitness function has an advantage of preserving the objects of interest. In the situation that recall is important, the weighted fitness function should be applied.

For the future work, we will compare the results of this approach with those of the existing approaches with a time consuming clustering as part of the fitness function. We will also investigate better ways to organise training patterns for object detection.

Acknowledgement

We would like to thank the members of the Genetic Programming Research Group and the Artificial Intelligence Research Group at Victoria University of Wellington for a number of useful discussions.

References

[1] Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. *Genetic Programming: An Introduction on the Automatic Evolution of computer programs and its Applications*. San Francisco, Calif. : Morgan Kaufmann Publishers; Heidelburg : Dpunkt-verlag, 1998. Subject: Genetic programming (Computer science); ISBN: 1-55860-510-X.

[2] Urvesh Bhowan. A domain independent approach to multi-class object detection using genetic programming. Master's thesis, BSc Honours research project/thesis, School of Mathematical and Computing Sciences, Victoria University of Wellington, 2003.

[3] Daniel Howard, Simon C. Roberts, and Richard Brankin. Target detection in SAR imagery by genetic programming. *Advances in Engineering Software*, 30:303–311, 1999.

[4] John R. Koza. *Genetic programming : on the programming of computers by means of natural selection*. Cambridge, Mass. : MIT Press, London, England, 1992.

[5] John R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs*. Cambridge, Mass. : MIT Press, London, England, 1994.

[6] Will Smart and Mengjie Zhang. Classification strategies for image classification in genetic programming. In Donald Bailey, editor, *Proceeding of Image and Vision Computing Conference*, pages 402–407, Palmerston North, New Zealand, November 2003.

[7] Walter Alden Tackett. Genetic programming for feature discovery and image discrimination. In Stephanie Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, pages 303–309, University of Illinois at Urbana-Champaign, 17-21 July 1993. Morgan Kaufmann.

[8] Mengjie Zhang, Victor Ciesielski, and Peter Andreae. A domain independent window-approach to multiclass object detection using genetic programming. *EURASIP Journal on Signal Processing, Special Issue on Genetic and Evolutionary Computation for Signal Processing and Image Analysis*, 2003(8):841–859, 2003.