

# GENETIC ALGORITHMS FOR OBJECT RECOGNITION IN A COMPLEX SCENE

*Daniel L. Swets, Bill Punch, and John Weng*

A714 Wells Hall  
Michigan State University  
East Lansing, Michigan 48824  
{swetsd, punch, weng}@cps.msu.edu

## ABSTRACT

A real-world computer vision module must deal with a wide variety of environmental parameters. Object recognition, one of the major tasks of this vision module, typically requires a preprocessing step to locate objects in the scenes that ought to be recognized. Genetic algorithms are a search technique for dealing with a very large search space, such as the one encountered in image segmentation or object recognition. This work describes a technique for using genetic algorithms to combine the image segmentation and object recognition steps for a complex scene. The results show that this approach is a viable method for successfully combining the image segmentation and object recognition steps for a computer vision module.

## 1. INTRODUCTION

A central task of the computer vision module is to recognize objects from images of the machine's environment. Navigation systems require the localization and recognition of landmarks or threats; robotic systems must find objects to manipulate; image retrieval systems require the localization and recognition of objects in images in order to find database records of interest.

Image segmentation is often a prerequisite to object recognition [1]; various techniques have been proposed which combine the segmentation step with the object recognition itself [2] [3].

Genetic algorithms [4] have been used in computer vision systems for such tasks as parameter tuning (e.g., [5]) and feature extraction [6]; typically the actual image processing tasks have been handled by other standard methods.

This work describes using genetic algorithms for object recognition, combining the image segmentation task with the object recognition task to be solved in its entirety by the genetic algorithm. Before the approach

is laid out, a brief review of genetic algorithms is given. Results are shown on this segmentation/recognition combination which demonstrate the utility of the approach.

## 2. BRIEF REVIEW OF GENETIC ALGORITHMS

This section gives a brief overview of genetic algorithm fundamentals. Goldberg [4] gives a wonderful introduction to genetic algorithms, and the reader is referred to this source for further information.

A genetic algorithm is an optimization technique that operates on a *population* of individual solutions. Each individual solution, also called a *string* in the population, represents a proposed solution to the problem being solved. The theories of natural selection are applied to this population, and subsequent generations of the population are obtained by applying selection, reproduction, and mutation operators (among possible others) to the population. With these operators, the population of solutions is gently pushed towards a good—and hopefully optimum—solution to the problem.

The GA designer provides a fitness function to evaluate the fitness of each individual solution; this fitness function is used to propagate “good” individuals into the next generation. Some set of these fit individuals are chosen for a *crossover* operation, which recombines the strings of the parents into new children solutions, trying to build up healthier strings in the process. The *mutation* operator randomly alters some element of an individual in order to further enhance the population, though typically only rarely in comparison to the crossover operation.

*Crossover* is the name given to a simple reproduction operation because of the way that the parent strings are recombined. Usually one or two common points in a pair of parents are chosen at random. For a one-point crossover, the portion of the parent strings to the right

of the crossover point can be called the *crossover area*; for a two-point crossover, the area between the points is the crossover area. One child is formed by taking the crossover area from one parent and the non-crossover area from the other parent—recombining the parent strings. The other child is formed by reversing the process. The theory behind doing this operation is called the *schema theorem* [4]; short, low-order portions of strings contributing to fit individuals are thus created.

There are several issues of paramount importance in the design of a genetic algorithm. First and foremost, the fitness function of the system must be designed appropriately to select good individuals, since it is the only window that the population has to the outside world. Furthermore, the representation of the strings making up the population of individuals must be selected with great care. The way that strings recombine and are propagated into the next generation is inherently linked to the way they are represented and interpreted. Finally, the selection of appropriate genetic operators for the system is fundamental to obtaining a good solution in a reasonable time.

### 3. GENETIC ALGORITHMS FOR OBJECT RECOGNITION

The task of locating and recognizing a particular object of interest in a complex scene is quite simple when cast in the framework of genetic algorithms. A brute-force method for finding an object in a scene is to examine all possible subimage positions and sizes. A subimage extracted in this way could then be used as a database query for the model database; the subimage/model pair with the highest confidence is taken to be a good segmentation and recognition result. Because of the enormous computational complexity of this approach, however, intelligent researchers immediately dismiss this approach as far too expensive. When cast in the framework of the genetic algorithm methodology, however, this brute-force method can be elevated to the level of an elegant solution to the difficult image segmentation and object recognition problem.

Since the genetic algorithm approach does well in very large search spaces by working with only a sample of the available population, the computation limitation of the brute-force method using full search space enumeration does not apply. It is this brute-force method, cast in the genetic algorithm methodology, that is explored in this work.

#### 3.1. Genetic Algorithm Parameters

Let  $(u_x, u_y)$  and  $(b_x, b_y)$  represent the upper left and lower right corner of the object of interest, respectively.

Then for an input image whose dimensions are  $h \times w$ ,  $u_x \in [0, w]$ ,  $u_y \in [0, h]$ ,  $b_x \in (u_x, w]$ , and  $b_y \in (u_y, h]$ .

The encoding recommended by Punch *et al.* [7] provides a “relative encoding” of the coordinates on the strings of the genetic algorithm population, and is used for this work. The layout of each chromosome is given in figure 1. The  $y$ -component of the lower-right corner is calculated using a standard aspect ratio for the images being examined.

The evaluation function extracts the coordinates from each chromosome in the population, performs the subimage extraction, subimage resize, and probes the image database for a likely match. The fitness of the individual that defines the subimage is given as the confidence that the database retrieval utility has for its best match.

#### 3.2. Database Retrieval Utility

The database retrieval utility described in [2] is used for this work. This content-based image retrieval system is broken into two phases, a training phase and a recognition phase.

In the training phase, a labeled set of images are given to the system. Each of these images has been previously pre-segmented such that only a single object of interest is contained in the image. Furthermore, this object in the image has a standard size, position, and orientation; the image dimensions for all of the images are identical in this phase. The training of the system consists of generating a hierarchical set of *Most Expressive Features* (MEFs) and *Most Discriminating Features* (MDFs) [2] [8] for each object placed into the model database. The theories of optimal linear projection are used in order to fully automate the selection of these features. The MEFs best describe a particular object class; the MDFs optimally discriminate among the various classes in a linear sense. These feature sets are used to build a hierarchical space decomposition of the image database.

In the test phase, an image probe is presented to this hierarchical space decomposition. The Most Discriminating Features selected during the training phase provide the basis for selecting which branch of the database hierarchy to explore at each level. When the database hierarchy exploration reaches a leaf node, the retrieval technique provides a confidence measure by comparing the search probe and the database object class defined by the leaf node in the space of the Most Expressive Features.

This confidence measure returned from the database retrieval algorithm for test probe  $x$  and database sam-

bits	$0 - > k - 1$	$k - > 2k - 1$	$2k - > 3k - 1$
Meaning	UL x	UL y	BR x
Interpretation	$x, y$ -offset from image center $[-2^k, 2^k]$		$x$ offset $[0, 2^k]$

Figure 1: Layout of chromosomes.  $k$  is the number of bits required to store the maximum of  $\{UL_x, UL_y, BR_x\}$

ple  $\mu$  is given by

$$c(x, \mu) = e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} + e^{-p}$$

where  $x, \mu$  are the MDF or the MEF subspace vectors, and  $\Sigma$  is a diagonal matrix where each diagonal element  $d_i$  is the variance in the  $i$ th dimension of the subspace being used.  $0 < p$  is a penalty function for illegal point combinations and ensures that invalid solutions receive little weight in the population of possible strings. We do not wish to simply remove these invalid solutions because they may contain building blocks useful for obtaining fitter individuals in subsequent generations.  $c(x, \mu)$  always falls in the range of  $[0, 2]$ , and is maximized when the test probe most closely matches a database sample in the feature space.

#### 4. EXPERIMENTS

The experiments were performed using the Genetic Algorithm Optimized for Portability and Parallelism System (GAIOPPS) developed at the Michigan State University Intelligent Systems Laboratory [9].

Among the experiments performed was an unconstrained natural scene experiment. This experiment allowed the genetic algorithm to operate on a natural scene to try to locate and recognize a face in a crowd. The crowd image with the face located and recognized by the genetic algorithm is shown in figure 2. The genetic algorithm parameters used are shown in figure 3. As this experiment shows, the genetic algorithm is capable of finding a valid segmentation in a complex, natural scene.

Because the GAIOPPS [9] system does not re-evaluate individuals which have already been examined, this genetic algorithm approach examined a total of less than 750,000 subimages. For the brute force method to enumerate and examine the entire search space explored by the genetic algorithm, a total of four billion ( $2^{32}$ ) subimages would need to be checked. So, though a large computational effort was expended, less than 0.02% of the possibilities were considered using the genetic algorithm sampling technique. Furthermore, no

constraints whatsoever were placed on the images or the objects being considered for the recognition.

#### 5. CONCLUSIONS AND FUTURE WORK

This work proposes a technique for image segmentation and object recognition using genetic algorithms on the images directly. The experiments show that the genetic algorithm performs well in finding areas of interest even in a complex, real-world scene. Genetic Algorithms are adaptive to their environment, and as such this type of a method is appealing to the vision community who must often work in a changing environment.

Clearly further experimentation could be performed in order to show the general applicability of the technique for a widely varying set of images and objects in the images. Grey coding the fields would greatly improve the mutation operation by ensuring that flipping a bit only changes a field's value by one. The way the described system is implemented, a bit changed by the mutation operator could cause a large change in the value.

DeJong-style crowding [4] could be used to find multiple objects of interest in an image. Since a natural scene probably contains many objects, the segmentation and recognition is necessary to make the technique useful for real-world images.

Timing improvements could be made by utilizing the implicit parallelization of multiple independent generations evolving at the same time. Different areas of the search space are explored in each of these populations; occasional swapping of individuals among populations could be done to improve diversity and gently push all the populations to a global maximum.

#### 6. REFERENCES

- [1] J. J. Weng, "SHOSLIF: The hierarchical optimal subspace learning and inference framework," Tech. Rep. CPS 94-15, Michigan State University, Department of Computer Science, A714 Wells Hall, East Lansing, Michigan 48824, March 1994.



Figure 2: Face segmented and recognized by the genetic algorithm during the test phase.

$P_c$	0.60
$P_m$	0.09091
$l$	33 bits
Scale	1.5
Pop. size	1000
Max gens	750

Figure 3: Parameters used for the natural scene experiment.  $P_c$  is the probability of crossover;  $P_m$  is the probability for mutation;  $l$  is the length of the chromosome used; Scale is the scaling factor used for the ratio of the best : mean individual in the population. The mutation rate was set so that on average, three bits will be flipped per individual in each generation. The crossover was restricted to crossing between fields on the chromosome.

- [2] D. L. Swets and J. J. Weng, "SHOSLIF-O: SHOSLIF for object recognition (phase I)," Tech. Rep. CPS 94-64, Michigan State University, Department of Computer Science, A714 Wells Hall, East Lansing, Michigan 48824, December 1994.
- [3] J. Weng, N. Ahuja, and T. S. Huang, "Learning recognition and segmentation using the creceptor," in *Proc. International Conference on Computer Vision*, pp. 121-128, May 1993. Berlin, Germany.
- [4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.
- [5] B. Bhanu, S. Lee, and J. Ming, "Self-optimizing image segmentation system using a genetic algorithm," in *Proceedings, 4th Int. Conf. Genetic Algorithms*, (San Diego), 1991.
- [6] W. A. Tackett, "Genetic programming for feature discovery and image discrimination," in *Proceedings, 5th Int. Conf. on Genetic Algorithms*, (Urbana-Champaign), 1993.
- [7] W. Punch, E. Goodman, R. Averill, M. Pei, L. Chia-Shun, D. Ying, D. Redder, and V. Kureichik, "Research applications using genetic algorithms," Tech. Rep. CAD-94, Michigan State University Genetic Algorithm Group, East Lansing, Michigan 48824, 1994.
- [8] Y. Cui, D. L. Swets, and J. J. Weng, "Learning-based hand sign recognition using SHOSLIF-M," in *Proc. International Conference on Computer Vision*, 1995.
- [9] E. D. Goodman, *GALOPPS-Genetic ALgorithm Optimized for Portability and Parallelism System*. Genetic Algorithms Research and Applications Group, Michigan State University, East Lansing, Michigan 48824, 2.20 ed., 1994.