# A Hybrid Genetic Algorithm for Resolving Closely Spaced Objects

5/2-63
47262
P. 8

**R. J. Abbott**
*The Aerospace Corporation and California State University, Los Angeles*

**W. E. Lillo**
*The Aerospace Corporation*

**N. Schulenburg**
*The Aerospace Corporation*

**A hybrid genetic algorithm is described for performing the difficult optimization task of resolving closely-spaced objects appearing in space-based and ground-based surveillance data. This application of genetic algorithms is unusual in that it uses a powerful domain-specific operation as a genetic operator. Results of applying the algorithm to real data from telescopic observations of a star field are presented.**

## 1.0 Introduction

Extracting information on individual visual point sources in a closely-spaced object (CSO) cluster is a fundamental problem for such applications as astronomy and ballistic missile defense. The problem is difficult because objects within closely-spaced object clumps cannot be resolved directly. Instead, one hypothesizes overlapping point sources to create a model of the clump. Then one parametrically solves for the number of sources along with their amplitudes and locations.

An objective function is formed based on the sum of the squared residual errors between the data and model employed in Bayes' Theorem. This Bayesian approach has been described by [Sc93] and[Li94]. The best model is that which minimizes the residual errors and thus maximizes the probability that the model represents the data.

This estimation approach presents a difficult optimization task since the probability function to be maximized is rugged, i.e., has many local maxima.

Traditional approaches were found to be inadequate to solve this problem. Hill-climbing techniques were found to be highly dependent upon the initialization of the parameters; different solutions would be obtained from different initial guesses. Further, convergence was often slow, prohibitively so when the number of sources in the closely-spaced object clump reached four or more. To avoid these deficiencies, we developed a hybrid genetic algorithm.

The remainder of this paper is organized as follows. Section 2 describes the problem in slightly more detail. Section 3 provides an overview of genetic algorithms. Section 4 describes the genetic algorithm developed for the CSO problem. Section 5 presents our results.

## 2.0 Problem Statement

Every optical system has a point response function (PRF), which is the image generated by a sensor from a point source located at infinity. The PRF width is due to diffraction of the input radiation through the system aperture and the presence of aberrations in the optical system. Because their geometrical angular



PRECEDING PAGE BLANK NOT FILMED

PAGE 154 INTENTIONALLY BLANK

subtense is much less than the width of the sensor PRF, many objects viewed by optical sensors, such as stars or distant space vehicles, appear as point sources.

If multiple point sources are located within the resolution limit of the sensor, the optical system will produce an image which appears as a *clump*. The objects in this case are referred to as a cluster of closely-spaced objects. The individual source amplitudes and locations determine the amount of overlap between the responses and thus the shape of the clump. Given the input clump of data and knowledge of the PRF, the sensor processing software must properly count and recover location and amplitude information of the individual objects.

# 3.0 Overview of Genetic Algorithms

The term *Genetic Algorithms* [Ho75] includes a broad class of iterative optimization techniques that employ methods that are modelled after the way evolution occurs by natural selection in biological systems.

## 3.1 The Genetic Algorithm

A genetic algorithm begins with a set of (sub-optimal) solutions, called a population. The initial population may be arbitrarily or randomly chosen, or it may be given as an external input.

An application-specific objective function is applied to each member of the population, thereby ranking the solutions.

The following selection/transformation/replacement cycle is repeated until a termination condition is met. (See Figure 1.)

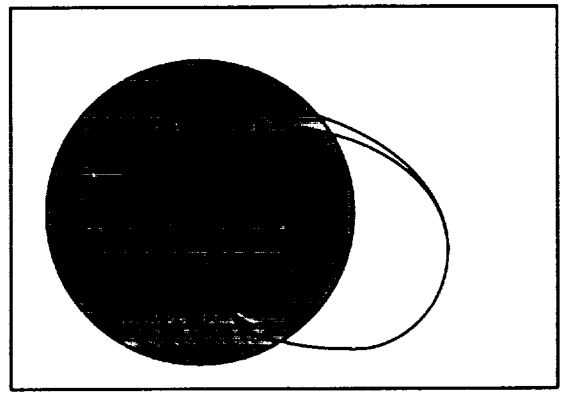

**Figure 1. A generic genetic algorithm**

1. **Selection.** Select one or more elements from the population using the following rule: the higher an element's score on the objective function, the *more* likely it is to be selected.
2. **Transformation.** Modify the selected element or elements to produce a new, possibly higher ranked element. The operators used to modify the elements are called genetic operators.
   - A single element may be modified in a process called *mutation.*
   - Two or more elements may be combined to produce a new element. The process of combination can create new elements that combine the best attributes of their predecessors in ways that are very unlikely under purely random stochastic methods. This is widely considered as one of the sources of the efficiency and broad applicability of genetic algorithms.
3. **Replacement.** Put the new element back into the population, replacing some element currently in the population. The higher a population element's score on the objective function, the *less* likely it will be selected to be replaced.

When the process terminates, the best ranked solution is the reported solution.

Genetic algorithms have been successfully applied to a wide range of optimization problems including the travelling salesman problem [Gr85], communication network design [Da87], natural gas pipeline control [Go83], image processing [Fi84], and training artificial neural networks [Wh89].

### 3.2 Hybrid Genetic Algorithms

As originally formulated, genetic algorithms were applied strictly to populations of fixed-length bit strings. All problem-specific information was encoded as bits. Within that framework, there are two genetic operators: mutation and crossover. The mutation operator changes one of the bits of the element to which it is applied. The crossover operator creates a new element by selecting, for each bit position, a bit in that position from the parents.

Hybrid genetic algorithms [Da91] move away from the bitstring representation in two ways.

1. Population elements are represented in ways that may be specific to the problem domain. Any data structure is allowed.
2. Genetic operators are defined which operate on the elements as represented. The primary operations are still generically mutation (change a single population element) and combination (combine pieces of multiple population elements to produce a new element). But mutation and combination are now tailored to the particular problem.

## 4.0 Application of Genetic Algorithms to the CSO Problem

In our application of hybrid genetic algorithms, we are searching for a set of objects at particular positions such that those objects would generate the given image. Since the number of objects to be resolved is unknown, the number of objects represented by each population element is not fixed. The positions of the objects are also unknown. The objective of the GA is to find both the number and placement of objects that best matches the received signal.

For the purposes of this discussion we need to distinguish terminologically between *elements in the population* and *objects to be resolved.* We use *element* and *object* to make this terminological distinction. *Element* refers to an element of the population. *Object* refers to a signal generating object. A population *element* thus consists of a number of *objects*.

### 4.1 Element representation

We have simplified the problem in a number of ways.

1. Instead of attempting to find the location of the objects in 3-dimensional space, we limit the problem to finding positions of the objects in the 2-dimensional field of view of the sensor.
2. As a result of orthonormalization (see [Br87]), the brightness (or amplitude) of the hypothesized objects need not be considered during the search. The amplitudes are computed after a best solution is found. This is consistent with the principle of maximum entropy as explained in [Li94].
3. The PRF generated by each object is assumed known.

As a consequence of these simplifications, each object may be represented by its centroid, a pair of numbers, representing the <x, y> position of the hypothesized object in the two-dimensional field of view of the sensor.

Our population therefore consists of sets of elements, each of which is represented by its 2-dimensional centroid.

To simplify comparing one element with another, we order the objects in each element by their x-coordinate. Thus each population element is an ordered list of <x, y> pairs of numbers.

## 4.2 Smoothing the Ruggedness of the Search Space

The search space for this problem turns out to be very *rugged*. By this we mean two things.

1. A very small change in the <x, y> position of an object can make a very big difference in the value of the objective function of the element within which that object appears.
2. There are a great many local maxima.

The more rugged the search space, the more difficult the optimization problem. To make matters worse, we do not know in advance how many elements will best approximate the given signal. Thus the problem is not only to find the best solution in a rugged n-dimensional search space, it is also to find the best dimensionality.

The naive approach would be to let the genetic algorithm determine both the dimensionality and the position of the elements within that n-dimensional space. With no information, this approach would work, but it would be quite slow.

We did some experiments and found that we could "climb the dimensionality ladder" as follows.

- First run a genetic algorithm to determine the best 1-dimensional (i.e., one object) solution.
- Using that 1-dimensional solution to help seed the population, run the genetic algorithm again to find the best 2-dimensional solution.
- Proceeding in this way, find solutions with increasingly higher dimensionality. Stop, when the solution with dimensionality n+1 is a worse approximation than the solution with dimensionality n.

This approach is successful because the solution with dimensionality n is always an *approximate superset* of the solution with dimensionality n-1. By this we mean that n-1 of the objects in the solution with dimensionality n are always *very close* to the objects in the solution with dimensionality n-1. *Very close* in this case means that it is generally a matter of hill-climbing to move from the positions of the n-1 objects in the n-1 dimensional solution to the optimum positions of the "corresponding" n-1 objects in the n-dimensional problem. Thus the biggest challenge in moving from dimensionality n-1 to dimensionality n is to determine where to put the additional object.

## 4.3 Genetic Operators

We defined the following genetic operators. The first two are combination operators; the last two are mutation operations

1. **Cross-over.** Select two elements. Select objects from each to include in the third element. Recall that during each run of the genetic algorithm, the population is homogeneous in size: each element has the same number of objects. Furthermore, the objects are sorted by x-position. This makes cross-over a more meaningful operation since comparable objects are being substituted for each other.

2. **Weighted-average of elements.** This operator is similar to crossover. But instead of selecting objects randomly from either of the two parents, create a new object by taking the weighted average of the corresponding elements in the parents. The weights depend on how good an approximation the parents are. This turns out to be a powerful hill-climbing operation.

3. **Line-optimize an object.** Given an element, select at random both an object and a direction in the x-y plane. Using standard line-optimization techniques, move the object along the selected direction until one finds a position that maximizes the element's overall value.

4. **Bretthorst's optimization technique.** Bretthorst [Br87] has developed a powerful optimization technique for problems similar to this one. In many cases, it finds the optimum value on its own. Our experience was that in some cases it found only a local optimum. We therefore incorporated it as an operator in our genetic algorithm framework.

   To use it, select an element from the population, which is used to seed the Bretthorst algorithm. The result produced by the Bretthorst algorithm is taken as the result of the operator.

The integration of known optimization techniques into a genetic algorithm framework poses a challenge. That challenge and our approach to its resolution is discussed in the following section.

## 4.4 Maintaining Population Diversity

A fundamental principle of genetic algorithms population management is: the better an individual, the better its chance of being retained in the population. A consequence of this principle is that over time, even if new population elements were selected from the search space at random, the average fitness of the population would increase.

A second fundamental (and countervailing) principle of genetic algorithms is the need to maintain diversity. Premature population convergence to a suboptimal solution is exactly what genetic algorithms are intended to avoid.

We have developed an approach to population management that attempts to satisfy both objectives. Our approach is based on the use of two techniques: continual injection of new, random elements into the population and tournament selection with varying competition levels.

Continual injection of new, random elements is just as it sounds. Instead of transforming an existing schedule, an entirely new schedule is generated. This ensures that the population will never be completely isolated in one part of the search space. In addition, once a new random element is generated one of the two mutation operations are applied to it to allow it climb to a local maximum. This increases the probability that the element will be retained in the population long enough to participate in additional transformations.

Tournament selection is used in the transformation and replacement step. It is used to select both the element(s) to be transformed and the element to be discarded.

To select an element for transformation, a subset of the population, *the selection pool,* is chosen randomly and uniformly from the entire population. The best (or best two) element(s) of that pool are selected. To select an element to be discarded, we again choose a subset of the population; the worst element of the selection pool is selected for deletion.

Since elements are included in the selection pool with equal probability, the size of the selection pool is *inversely related* to the selectivity of the search. If the pool size were 1, one would be selecting (for transformation or deletion) an element uniformly from the population, i.e., with no regard for how well the element solved the problem. This would minimize convergence, but it would also minimize the likelihood that good features would be exploited.

On the other hand, were the pool to be the entire population, one would always select the best element(s) for transformation and the worst for deletion. This would maximize convergence, but it would virtually eliminate significant diversity.

Our strategy is to allow the size of the selection pool to vary based on the extent to which the population has converged. Convergence is measured by the difference between the best element of the population and the median element of the population. As they approach each other, the size of the selection pool is decreased, thereby promoting population divergence. As they move apart, the size of the selection pool is increased, thus promoting population convergence.

This yo-yoing effect tends to ensure that the population will not converge to a single area of the search space.

## 5.0 Results

The genetic algorithm strategy was applied to real data from a staring visible CCD sensor attached to a 24-inch telescope on Table Mountain, California. Figure 2 shows the center 256X256 pixel scene of NGC 6819 measured September 19, 1992. We chose four stars as model PRFs and ran 1-4 source models on 32 different clumps. The clumps were chosen to include single and multiple sources with a variety of amplitudes at locations.

Three cases are discussed. In these examples, the bright star at ~(250, 320) was used as the PRF. The three clumps are displayed in Figure 3 as detailed contour plots.

- **Star clump 423 at ~(300, 190)** is commonly thought to be a single source and is used for calibration photometry. Our algorithm agreed with this hypothesis with an extremely high confidence of 99%. The estimated amplitude was 1198 counts with an error bar of only 3 counts. The location was 9.704 +/- 0.006 pixels east and 11.493+/-0.006 pixels north. The small error bars result from the high signal to noise ratio of ~150.
- **Star clump 416 at ~(200, 360)** is also commonly taken to be a single source. Our technique, however, assigned virtually no probability to a one- source model compared to a two source model. The two source model was also preferred over the three source model by a factor of 100. The two source model put a source about 11

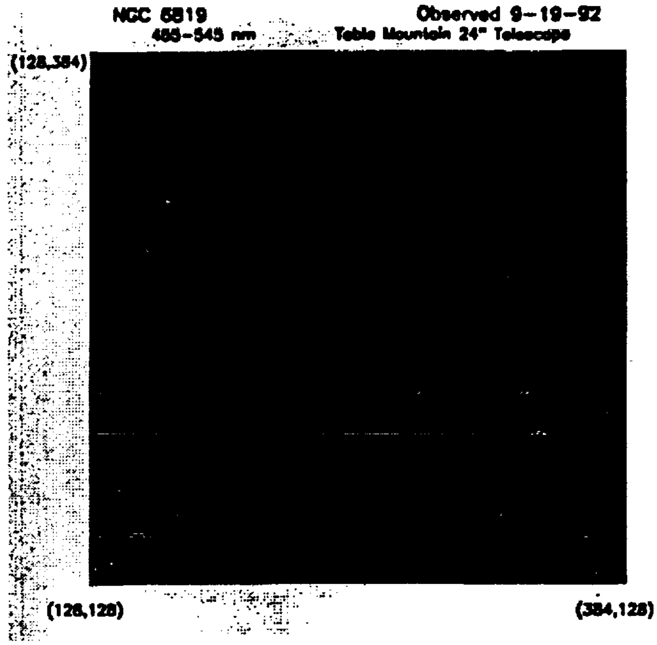


**Figure 2. View from Table Mountain**

times dimmer than the other separated by 2.2 pixels to the east and 2.5 pixels to the south. The location error bars are greater than ~1/6 pixel for the dimmer source compared to 0.013 pixel for the brighter source. The amplitude error bars for the dimmer source were about 6% compared to 0.5% for the brighter source.

- **Star clump 414 at ~(340, 210)** looked interesting because the bulge to the south and west of the doublet gives evidence for another source. The technique, in fact, strongly preferred a three source model with a dim source located at 8.8 pixels east and 6.9 pixels north. It was separated by 4.6 pixels east and 0.5 pixels south from one source and 0.6 pixels east and 5.2 pixels south from the other pulse.

For all of the above clumps, we ran the code several times with different initial guesses. In all cases the genetic algorithm converged on the indicated solution in a reasonable amount of processing time. A standard hill-climbing

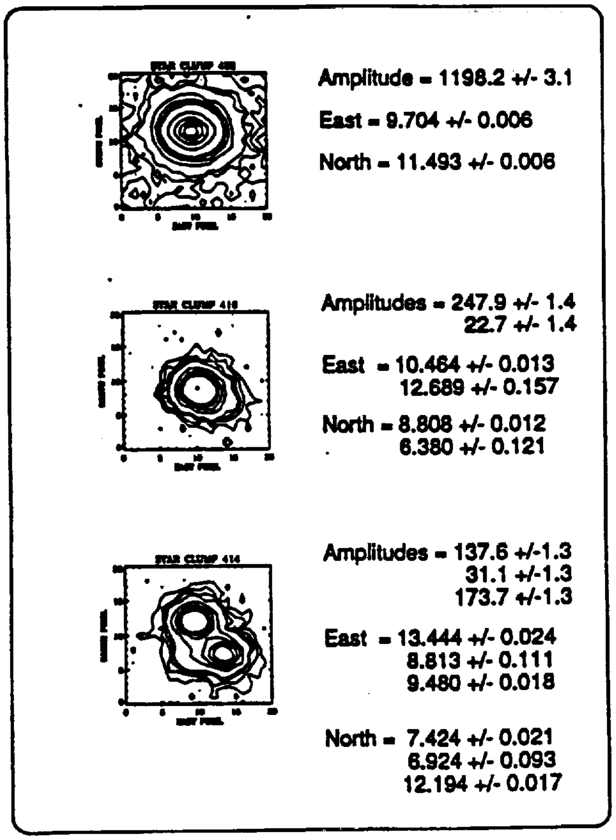


**Figure 3. Three Clumps**

technique gave different solutions for different initial guesses.

## 6.0 Conclusion

This work has shown that a hybrid genetic algorithm can be developed to solve a difficult optimization problem arising from image processing. Our experience has convinced us that neither traditional optimization techniques nor traditional genetic algorithm techniques would have allowed us to solve this problem. As our genetic operators show, a hybrid genetic algorithm approach allows one to incorporate known optimization techniques into a genetic algorithm framework. This strategy demonstrates that one need not sacrifice known, powerful techniques when one employs genetic algorithms.

**Acknowledgments**

This work was performed at the Aerospace Corporation under sponsorship of its Aerospace Sponsored Research Program. We thank M. Campbell for many fruitful discussions.

**References**

[Br87]. Bretthorst, L, *Bayesian Spectrum Analysis and Parameter Estimation,* Ph.D. Dissertation, Washington University, St. Louis, 1987.

[Da87]. Davis, L and S Coombs, "Genetic algorithms and communication link speed design: theoretical considerations," *Proceedings of the Second International Conference on Genetic Algorithms,* pp 252-256, 1987.

[Da91]. Davis, L (Ed), *Handbook of Genetic Algorithms,* Van Nostrand Reinhold, New York, 1991.

[Fi84]. Fitzpatrick, JM, JJ Grefenstette, and D Van Gucht, "Image registration by genetic search," *Proceedings of the IEEE Southeast Conference,* pp 460-464, 1984.

[Go83]. Goldberg, DE, *Computer-Aided Gas Pipeline Operation Using Genetic Algorithms and Rule Learning,* Doctoral Dissertation, University of Michigan, 1983.

[Gr85]. Grefenstette, JJ, R Gopal, B Rosamita, DV Gucht, "Genetic algorithms for the traveling salesman problem," *Proceedings of International Conference on Genetic Algorithms and their Application,* pp 160-165, Lawrence Erlbaum & Assoc., NJ 1985.

[Ho75]. Holland, JH, *Adaption in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, Michigan, 1975.

[Li94]. Lillo, W and NW Schulenburg, *A Bayesian Closely Spaced Object Resolution Technique,* Aerospace Technical Report ATR-94(8025)-1, 1984.

[Sc93]. Schulenburg, NW, and JA Hackwell, *Bayesian Approach to Image Recovery of*

*Closely Spaced Objects,* SPIE Conference, Orlando, Florida, April 12-14, 1993.

[Wh89]. Whitley, D and T Hanson, "Optimizing neural networks using faster, more accurate genetic search," *Proceedings of the Third International Conference on Genetic Algorithms,* pp. 391-396, Morgan Kaufmann, San Mateo, Ca. 1989.