# Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

## ABSTRACT

# Point Pattern Matching by Heuristic Methods: A Genetic Algorithm and Simulated Annealing

by
**Erh-Chin Chen**

The problem we consider is to find a subset of points in a pattern that best match to a subset of points in another pattern through a transformation in an optimal sense. Exhaustive search to find the best assignment mapping one set of points to another set is, if the number of points that are to be matched is large, computationally expensive. We propose two stochastic searching techniques — a genetic algorithm and simulated annealing to search for the best ("almost the best") assignment efficiently. To make the comparison between GA and SA fair, we introduce a piece-wise linear cooling schedule for the SA. As compared to conventional searching techniques such as simple hill climbing and random search techniques, the proposed methods are able to attain better solutions much faster. The proposed methods can be applied to n-dimensional point patterns and any transformation, but we only present results for two-dimensional point patterns and similarity transformations.

# POINT PATTERN MATCHING BY HEURISTIC METHODS: A GENETIC ALGORITHM AND SIMULATED ANNEALING

by
Erh-Chin Chen

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science
Department of Electrical and Computer Engineering
October 1992

# APPROVAL PAGE

POINT PATTERN MATCHING BY HEURISTIC METHODS:
A GENETIC ALGORITHM AND SIMULATED ANNEALING

by

Erh-Chin Chen

7/22/92

Dr. Nirwan Ansari, Thesis Advisor
Assistant Professor of Electrical and Computer Engineering
New Jersey Institute of Technology

Dr. Edwin S.H. Hou, Committee Member
Assistant Professor of Electrical and Computer Engineering
New Jersey Institute of Technology

Dr. Zoran Siveski, Committee Member
Assistant Professor of Electrical and Computer Engineering
New Jersey Institute of Technology

## BIOGRAPHIC SKETCH

**Author:** Erh-Chin Chen

**Degree:** Master of Science in Electric and Computer Engineering

**Date:** October, 1992

**Date of Birth:**

**Place of Birth:**

**Undergraduate and Graduate Education:**

- Master of Science in Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, 1992

- Bachelor of Science in Material Engineering, National Cheng-Kung University, Taiwan, R. O. C., 1987

**Malor:** Electrical and Computer Engineering

This thesis is dedicated to

Dr. Nirwan Ansari

## ACKNOWLEDGMENT

The author wishes to express his sincere gratitude to his supervisor, Dr. Nirwan Ansari, for his guidance, timely help and suggestions throughout this research.

Special thanks go to Dr. Edwin S. H. Hou and Dr. Zoran Siveski for serving as members of the committee.

Finally, I would like to thank my parents for their patience and assistance.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

**Figure** **Page**

# CHAPTER 1
# INTRODUCTION

Shape recognition is an important task in computer vision and pattern recognition. We will use the term *shape* to refer to the invariant geometrical properties of the relative distances among a set of static spatial features of an object. These static spatial features are known as the *shape features* of the object. For the purpose of recognition, much of the visual data perceived by a human being is highly redundant. It has been suggested from the view point of the human visual system (2) that some dominant points along an object contour are rich in information content and sufficient to characterize the shape of the object. Thus, point pattern matching in which points are used as shape features is a crucial vision task.

The problem we address in this paper is that of recognizing and locating objects which are represented by a set of points. That is, each object is represented by a set of dominant points (shape features). Information about the sequential order of these points is not known or provided. The problem is to find a subset of points in a point pattern that match to a subset of points in another point pattern through a transformation in a certain optimal sense with the constraint that the mapping is single. In a general setting, the points are arranged in n-dimensional space, and the transformation is specified according to the geometric and environmental constraints of the problem. In this paper, we only consider two-dimensional point patterns and similarity transformations because images are inherently 2-D and similarity transformations can be used to indicate the similarity between two point patterns. However, the algorithm itself is not restricted to 2-D point patterns and similarity transformations.

Many studies on planar object recognition have been done. The recognition task can be modeled as searching for an assignment between two features.

Commonly used features are holes and points (3), (4), (8), (17), (18), (28), (26), line segments (5), (7), (6), (10), (25), curve segments (11), (14), (15), (19), (21), (29), or a combination of these features (23), (24). The features are obtained by a preprocessing step such as edge detection, polygonal approximation, and corner extraction. We have taken a similar view by posing the recognition task as a point pattern-matching problem. Even methods that use points as their features usually require *a priori* knowledge on the sequential order of the arrangement of the points. The point pattern matching problem we are addressing is more general, and assumes no knowledge on the sequential order of the points.

Among the methods mentioned above, (18),(26) which use relaxation labeling for point pattern matching do not assume, similar to our proposed algorithm, knowledge on the order of the points. However, a good estimate of the initial assignment between the points of the two point patterns is important relative to the convergence of the algorithm and the validity of the result. These methods (18),(26), inheriting the drawback of relaxation labeling, are complex, and computationally expensive because of their sequential nature.

Comparing the algorithms addressed here to conventional search techniques such as random search and simple hill climbing technique which evaluate the fitness value sequentially, a genetic algorithm evaluates a set of samples in each generation. Thus, the genetic algorithm can achieve faster convergence and can escape local maxima. On the other hand, though simulated annealing is a sequential search technique, it provides a mechanism from getting stuck in local minima (9).

After having formulated the problem in the next Chapter, we will describe a genetic algorithm for point pattern matching in Chapter 3, and the simulated annealing approach in Chapter 4. Experimental results and comparative studies are described in Chapter 5. Concluding remarks are drawn in Chapter 6.

# CHAPTER 2
# PROBLEM FORMULATION

The point pattern matching problem can be formulated as follows: Given two sets of point patterns,

$$\tilde{P} = \{p_i : p_i \in R^n, i = 1,2,3,...,m\} \text{ and } \tilde{O} = \{o_i : o_i \in R^n, i = 1,2,3,...,n\},$$

we want to find an assignment, $P \rightarrow O$, where $\tilde{P} \supseteq P$ and $\tilde{O} \supseteq O$, such that the match error between $T\{P\}$ and $O$ is minimized; the match error which will be defined later indicates the quality of match, the smaller the error, the better the match between $P$ and $O$, and $T$ is a predefined transformation. In this paper, we only consider two-dimensional point patterns, $N=2$, and that $T$ is a similarity transformation, $T$={rotation, scaling, translation}.This problem is different from the image registration problem [13] in which the objective is to align two images through a geometric transformation. Let $\tilde{O}$ be an observed 2-D point pattern, and $\tilde{P}$ be a model 2-D point pattern. The 2-D similarity transformation is defined by the mapping $\underline{x} \rightarrow \underline{u}$, where $\underline{x}$, $\underline{u} \in \boldsymbol{R}^2$ such that

$$\begin{bmatrix} u \\ v \end{bmatrix} = S \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix},$$

where $\underline{x}=\begin{bmatrix} x \\ y \end{bmatrix}$, $\underline{u}=\begin{bmatrix} u \\ v \end{bmatrix}$, $S$= scale factor, $\theta$= angle of rotation, $e$= translation in the x-axis, and $f$= translation in the y-axis.

By letting $a = S\cos\theta$ and $b = S\sin\theta$, the similarity transformation can be rewritten as

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}.$$

Let $P=\{(x_i,y_i): (x_i,y_i) \in \boldsymbol{R}^2,\ i=1,2,3,...,k\}$ be a subset of model points. To find how well $P$ is assigned to $O$ under a similarity transformation, we need to find the parameters of the similarity transformation which maps $P$ to $O$ in an optimal sense, in

our case, in the minimum least squared error sense. Denote $(\hat{u}_i,\hat{v}_i)$ as the result obtained by transforming $(x_i,y_i)$ under $T$; i.e., $T\{(x_i,y_i)\}=\{(\hat{u}_i,\hat{v}_i)\}$. Mapping $P$ by this transformation, we have

$$\begin{bmatrix}\hat{u}_i\\ \hat{v}_i\end{bmatrix}=\begin{bmatrix}a & b\\ -b & a\end{bmatrix}\begin{bmatrix}x_i\\ y_i\end{bmatrix}+\begin{bmatrix}e\\ f\end{bmatrix}.$$

The squared error between the two sets of points $T\{P\}=\{(\hat{u}_i,\hat{v}_i),\ i=1,2,...,k\}$ and $O=\{(u_i,v_i),\ i=1,2,...,k\}$ is defined by:

$$\varepsilon^2=\sum_{i=1}^{n}((u_i-\hat{u}_i)^2+(v_i-\hat{v}_i)^2$$

$$=\sum_{i=1}^{n}((u_i-ax_i-by_i-e)^2+(v_i+bx_i-ay_i-f)^2$$

To find the parameters of the transformation which will achieve the minimum least squared error, we simply take the derivatives of the error ε will respect to each parameter, and set these derivatives to zero. That is,

$$\frac{\partial\varepsilon^2}{\partial a}=\sum_{i=1}^{n}(2(u_i-ax_i-by_i-e)(-x_i)+2(v_i+bx_i-ay_i-f)(-y_i)=0,$$

$$\frac{\partial\varepsilon^2}{\partial b}=\sum_{i=1}^{n}(2(u_i-ax_i-by_i-e)(-y_i)+2(v_i+bx_i-ay_i-f)(-x_i)=0,$$

$$\frac{\partial\varepsilon^2}{\partial e}=\sum_{i=1}^{n}2(u_i-ax_i-by_i-e)(-1)=0,\quad and$$

$$\frac{\partial\varepsilon^2}{\partial f}=\sum_{i=1}^{n}2(v_i+bx_i-ay_i-f)(-1)=0.$$

Rewrite this in matrix form, we have A $\begin{bmatrix}a\\ b\\ e\\ f\end{bmatrix}$ = C, where

$$A=\begin{bmatrix}\sum_{i=1}^{n}(x_i^2+y_i^2) & 0 & \sum_{i=1}^{n}x_i & \sum_{i=1}^{n}y_i\\ 0 & \sum_{i=1}^{n}(x_i^2+y_i^2) & \sum_{i=1}^{n}y_i & -\sum_{i=1}^{n}x_i\\ \sum_{i=1}^{n}x_i & \sum_{i=1}^{n}y_i & \sum_{i=1}^{n}1 & 0\\ \sum_{i=1}^{n}y_i & -\sum_{i=1}^{n}x_i & 0 & \sum_{i=1}^{n}1\end{bmatrix},\ and\ C=\begin{bmatrix}\sum_{i=1}^{n}(u_ix_i+v_iy_i)\\ \sum_{i=1}^{n}(u_iy_i-v_ix_i)\\ \sum_{i=1}^{n}u_i\\ \sum_{i=1}^{n}v_i\end{bmatrix}.$$

The above least squared error derived from the similarity transformation quantifies how well $P$ is mapped to $O$. That is, it only quantifies how well a portion of the model point pattern $\tilde{P}$ matches to the corresponding portion of the observed point pattern $\tilde{O}$. A small error indicates that the portion of the model points match well to the corresponding observed points. It does not, however, account for the overall goodness of match between point pattern $\tilde{P}$ and point pattern $\tilde{O}$. To account for the overall goodness of the match between model and observed point patterns, we use the following heuristic measure (20), (21) which penalizes incomplete matching of the model point pattern:

$$\hat{\varepsilon} = \begin{cases} \dfrac{\varepsilon}{S * k}\left(1 + \left(\dfrac{m-2}{k-2}\right)\log_2\left(\dfrac{m-2}{k-2}\right)\right) & k \geq 3 \\ \infty & k = 0,1,2. \end{cases}$$

where $k$ is the number of model points that match the observed points, $m$ is the number of model points, and $S$ is the scale factor.

The heuristic measure, which can be regarded as an error measure for the overall goodness of match between the model and the observed point pattern, is referred to as the *match error*. If there are only two or less pairs of matches between the model and the observed point patterns, the least squared error is always zero because there always exists a similarity transformation that perfectly maps a set of one or two points to another set. We consider such cases where two or less model points match to those in the scene as undetermined cases; i.e., these cases have insufficient evidence of match between the model and the observed point pattern. The logarithmic term of the match error serves as a penalty factor for incomplete matching of the model points. When all model points match those in the observed points ($k$=$m$), the match error equals the normalized least squared error. The penalty is larger when less model points match those in the observed point pattern. The smaller the error, the larger the fitness value. The reciprocal value of the match error is thus used as the

fitness function in our algorithm. That is, the fitness function is

$$f = \begin{cases} 1/\hat{\varepsilon} & for \ \ \hat{\varepsilon} > 0 \\ 0 & else \end{cases}.$$

# CHAPTER 3
# POINT PATTERN-MATCHING BY A GENETIC ALGORITHM

Genetic algorithms were first introduced by John Holland (11) and received much attention as efficient searching methods for various applications. This searching scheme is based on the mechanics of natural selection and natural genetics that combines the notion of survival of the fittest, and the parallel evaluation of nodes in the entire search space (10).

A genetic algorithm consists of a string representation of the node in the solution space, a fitness function to evaluate the nodes, a set of genetic operators for generating new nodes, and a stochastic assignment to control the genetic operators.

To map the matching problem onto a genetic algorithm framework, we need to define a coding scheme (Section 3.1), a fitness function (Section 3.2), a set of genetic operators (Section 3.4), and the rules to control the operators that are suitable to this matching problem (5).

## 3.1 String Representation

Each string is an idealized "chromosome" consisting of a number of idealized "genes" (items in the string), and hence ending with the name "genetic algorithm." Choosing a coding scheme to encode the parameters of the problem into a string is problem dependent and not unique. In general, a proper coding scheme should

(1) be as simple as possible,

(2) span the range of the parameter to avoid illegal codes, and

(3) be easily manipulated by genetic operators.

In this paper, the parameters which we want to encode are assignments between two sets of points; each cell in a string indicates which model point is assigned a specific observed point. Suppose we have $m$ model points and $n$

observed points. For $m$ model points, we choose a code consisting of m cells. Each cell takes on an integer value between *0* to $n$. The value of each cell indicates an assignment of match from the model point (that correspond to this cell) to an observed point. For our point pattern matching problem, the assignment between a set of model points and a set of observed points must satisfy the constraints that the match is single; that is,

(1) A model point cannot be assigned to more than one observed point, and

(2) An observed point cannot be assigned to more than one model point.

A value of *0* indicates that no observed point is assigned to this corresponding model point. Note that the cell value in each string (except a cell value of *0*) must be distinct because the mapping from the model points to the observed points is single. A string can, however, have several *0*'s because it is possible that a number of model points do not match any observed points.

Consider the following code with 12 cells: 2 5 7 0 9 1 3 4 6 11 10 12. The cell position from the left indicates the label of a model point. For example, the sixth cell corresponds to the sixth model point. In this code, the first model point is assigned to the second observed point, the second to the fifth, the third to the seventh, the fourth is not assigned, the fifth to the ninth, and so on.

### 3.2 Fitness Function

The fitness function must be relevant to the problem to be optimized. A well-defined fitness function is necessary to ensure success and to provide the payoff information indicating how good each sampled assignment is. Each string is thus evaluated by the fitness function to get a fitness value, the smaller the error, the larger the fitness value. We use a fitness value that is inversely proportional to the match error defined in Chapter 2. That is

$$fitness = \begin{cases} \dfrac{S * k}{\varepsilon(1 + (\dfrac{m-2}{k-2})\log_2(\dfrac{m-2}{k-2}))} & k \geq 3 \\ 0 & k = 0,1,2. \end{cases}$$

## 3.3 Population Size

The population size is the number of strings to be evaluated in each generation. The larger the population size, the faster the convergence in terms of the number of generations to reach the optima, but the required computation for each generation becomes more intense. There is no fixed rule to select the size of the population. It is usually determined by trial and error, and we have chosen a population size of 30 for our experiments.

## 3.4 Genetic Operators

### 3.4.1 Reproduction

The purpose of reproduction is to pass on good strings to which genetic operators such as crossover and mutation will be applied. For the point pattern matching problem, the reproduction operator is performed as follows:

(1) Normalize the fitness value of each string such that the sum of the fitness value of all the string in the current population equal 1.

(2) Partition a unit-length scale into thirty slots, each slot size is in proportion to the normalized fitness value of a string in the current population.

(3) A new population of strings is reproduced by picking 29 random number (which are uniformly distributed on [0,1]), and see where the number falls on the scale. The string corresponding to the division where a random number falls is selected to be a member for the new population. We pick 29 random numbers because we always pass the best string in

the current generation. Then, a total of 30 strings including the best string that is passed to the new population form a reproduction pool.

### 3.4.2 Crossover

The crossover operator is the most important operator in genetic algorithms. It is the "mating" operator which allows production of new strings through combination of parts of strings. It combines partial solutions that have been judged to be relatively good. One simple way of performing the crossover operator is:

(1) Pairs of members of the newly reproduced strings are randomly selected for mating.

(2) For each pair of selected strings, we swap parts of the strings to form a pair of new strings. The position of a string to which the swapping takes place is randomly selected.

Each new string produced contains partial information from its parents. The mechanics of reproduction and crossover are surprisingly simple, involving only random number generation and string coping.

Consider the following two strings A1 and A2

A1:12345678

A2:25781436

Using simple crossover at cross site 4, the following two new strings are obtained.

A1':1234|1436

A2':2578|5678

The above simple crossover operator may produce illegal strings for our point matching problem. For example, Al' violates the constraint that the mapping must be single. We thus need to modify the crossover operator with some type of reordering. We introduce a new operator known as *mixed-type partial matching crossover (MPMX)*. The MPMX operator is introduced for the following reasons:

(1) it inherits the characteristics of the crossover operator in combining partial

solutions.

(2) it avoids illegal strings by reordering.

(3) it differs from mutation because no new "genes" are introduced.

We shall illustrate the MPMX operator by means of an example. Let m= 12 and n= 15. Thus, each string consists of 12 cells, and each cell takes on an integer value between 0 and 15. Consider the following two strings:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A1: | 1 | 2 | 7 | 6 | 9 | 13 | 8 | 0 | 4 | 11 | 12 | 5 |
| A2: | 0 | 15 | 2 | 4 | 5 | 6 | 7 | 1 | 3 | 8 | 9 | 0 |

The MPMX operator works as follows:

(1) a. Determine the common genes between the two strings,

$$\{A1\} \cap \{A2\} = \{0,1,2,4,5,6,7,8,9\}.$$

b. Determine the genes contained in A1 but not in A2,

$$\{A1\} \setminus \{A2\} = \{11,12,13\}.$$

c. Determine the genes contained in A2 but not in A1,

$$\{A2\} \setminus \{A1\} = \{15,3\}.$$

(2) a. Randomly select N1 cells from the pool of common genes, say, N1=4, and the cells are {1,5,6,7}.

b. Randomly select N2 cells from the two pools of genes, say, N2=1, and the cells are {12} and {3}, respectively.

(3) Form the matching section from the selected common and uncommon genes.

a. the first matching section M1=1,5,6,7,12.

b. the second matching section M2=1,5,6,7,3.

(4) Permute the first matching section to the new matching section say

$$M1' = 5,6,7,12,1.$$

(5) According to the position wise mapping between the new matching section and

the second matching section (e.g. cell value of 5 in string A1 becomes 1, cell value of 6 in string A1 becomes 5, and so on; likewise, cell value of 1 in string A2 becomes 5, and so on), the two new strings are generated:

A1 3, 2, 6, 5, 9, 13, 8, 0, 4, 11, 7, 1.

A2 0, 15, 2, 4, 6, 7, 12, 5, 1, 8, 9, 0.

The operator satisfies the constraint that each cell has a unique cell value in a string except cell value *0*.

### 3.4.3 Mutation

Mutation is the occasional (with small probability) random alteration of the value of a string position. For a binary code, this simply means changing a *1* to a *0*, and vice versa. By itself, mutation is a random walk through the string space. Mutation is needed because reproduction and crossover never introduce new genetic material (which might be needed even if they never actually lose any genetic material). In an artificial genetic system, the mutation operator protects against such irrecoverable lose, and provides a means of escaping from local minima.

Since the string used for the point pattern matching problem is not binary, we cannot simply do bit inversion. In addition, we are constrained that cell values (except *0*) in a string cannot be repeated. Thus, we adopt the following mutation procedure:

(1) Check the cell values of the string. If some cell values (out of all the possible cell values) are not assigned in the string, we randomly pick a cell in the string, and replace this cell with a cell value which is randomly picked from those unassigned cell values.

(2) If the string contains all the possible cell values, we randomly swap the values between two cells in a string.

Note that, as in natural selection, mutation rarely occurs, and thus artificial mutation operation is not always carried out. That is, mutation occurs with a small probability in each generation.

### 3.5 Summarizing the Procedure

The point pattern matching procedure can be summarized as follows:

(1) Select a fixed population size of strings, and randomly select an initial population. In our experiments, we use 30.

(2) Assign a probability, each for the MPMX operation and the mutation operation. Usually the probability for the mutation operation is very small.

(3) Perform reproduction as described in Section 3.4.1.

(4) Reordering: Randomly pair up 28 strings (the best string is not paired up). For each pair of strings, generate a random number distributed on [0,1]. If the number is less than the assigned crossover probability, perform the MPMX operator on the present pair of strings.

(5) Mutation: For each string in the population, generate a random number distributed on [0,1]. If the value is less than the assigned mutation probability, perform the mutation operation on the current string.

(6) Repeat Steps (3)-(5) until convergence or a predefined number of generations has been reached.

# CHAPTER 4
# POINT PATTERN-MATCHING BY SIMULATED ANNEALING

First introduced by Kirkpatric *et al* (1983) (20), simulated annealing is a stochastic searching technique derived from statistical mechanics for finding solutions to large optimization problems (28). The concept of simulated annealing is analogous to the way liquids freeze and crystallize. At high temperature, the molecules of a liquid move freely, and thermal mobility is lost as the liquid is cooled slowly. A pure crystal is formed when it is at the state of minimum energy.

In simulated annealing, there are two conceptual operations involved: a thermostatic operation which schedules the decrease of the temperatures (an algorithm parameter), and a random relaxation process which search for the equilibrium solution at each temperature (1).

To map the point pattern matching problem onto the simulated annealing framework, we use the same coding scheme as in the genetic algorithm discussed earlier. Each code (an assignment) is analogous to the state of a liquid. The cost of an assignment (the match error) is analogous to the energy of a state of the liquid. Thus, we let the energy function of the SA procedure be the match error. A perturbation rule for generating new assignments (configurations, states), the acceptance rule, the cooling schedule, and the stopping criterion will all be discussed next (30) (9).

## 4.1 The Energy Function

As mentioned earlier, the match error for the point pattern matching is analogous to the state energy. Thus the energy function E of an assignment is equal to the match error of the assignment.

$$E=\hat{\varepsilon}=\begin{cases}\dfrac{\varepsilon}{S*k}(1+(\dfrac{m-2}{k-2})\log_2(\dfrac{m-2}{k-2})) & k\geq 3\\ \infty & k=0,1,2.\end{cases}$$

### 4.2 The Perturbation Rule

Various perturbation rules are applicable. For simplicity, a simple perturbation rule is applied here to generate a new assignment from the current assignment. Consider the following string assignment with eight number:

A1: 25781436

We randomly generate two numbers, say, "1" and "3." The first random number indicates the element of the string to which its value will be replaced by the second random number. Thus, the new string is generated by replacing the first element with 3.

A1':35781436

Since there exists another "3" in the string (in position 7), this violates the constraint that the match must be single. Thus, the *7* th element is replaced by the original value of the first element. Thus, we obtain

A1': 35781426

### 4.3 The Acceptance Rule

The acceptance rule decides whether the new search node is accepted. Here, the new assignment with a lower energy is always accepted. To provide a mechanism from getting stuck in a local minima, a new assignment with a higher energy is occasionally accepted. The acceptance of a new assignment with a higher energy follows the Boltzmann distribution.

$$P=\begin{cases}1 & if\ \ \Delta E\leq 0\\ \exp(\dfrac{-\Delta E}{T}) & if\ \ \Delta E>0\end{cases}$$

Here, P is the probability for accepting a new assignment. This provides the

mechanism to avoid getting stuck in local minima. As the temperature decreases, the probability of accepting new assignments with higher energy values is greatly reduced.

### 4.4 Cooling Schedule

In order to make a fair comparison with GA, and to see if SA will acquire the (near) optimal solution as fast as GA, for each experiment, if GA acquires the (near) optimal solution at the *N*th generation with population size 30 (i.e. 30 N iterations), we allow SA to run for 30 N iteration.

Let $T_O$=1 be the initial temperature. The cooling schedule:

$$T_{n+1} = T_n - \Delta T_n \quad where \quad \Delta T_n = \frac{I_n}{30N}$$

$I_n$= the number of runs (iterations) executed by SA at $T_n$.

At each temperature, new assignments are generated and accepted according to the perturbation rule and the acceptance rule described above until thermal equilibrium is reached at that temperature or a predefined maximum number of iterations allowed at each temperature is reached. Thermal equilibrium is reached if the energy for a number of new consecutive assignments are close to each other. In our experiments, if $|\Delta E_k| = |E_k - E_{k-1}| < 0.5$ for 20 consecutive iterations, thermal equilibrium is said to be attained, where $E_k$ is the energy of the *k*th assignment.

### 4.5 Stopping Criterion

The annealing procedure is terminated when the temperature reaches 0 or an optimal solution (with match error=0) is found.

### 4.6 The Summarized Procedure

The procedure can be summarized as follows:

(1) Set $T_O$=1

(2) Randomly select an assignment.

(3) Generate a new assignment according to the perturbation rule.

(4) Determine if the new assignment is accepted according to the acceptance rule.

(5) Repeat steps (3) & (4) until thermal equilibrium or a predefined number of iterations is reached at that temperature.

(6) Reduce the temperature according to the cooling schedule.

(7) Repeat steps (3) to (6) until the stopping criterion is met.

# CHAPTER 5
# EXPERIMENTAL RESULTS

In this section, we present experimental results to demonstrate the efficiency (how well and how fast) of our proposed methods for point pattern matching. We have run a large number of simulations which can be categorized into several cases. In the following set of simulations, we create a library of four model point patterns, each consisting of 12 points as shown in Figure 1. The observed point pattern are made up of points belonging to one or more model objects which have undergone a similarity transformation. In addition, as compared to the model point patterns, extraneous points or missing points may be introduced in the observed point patterns. We also consider cases where the observed point pattern is derived from one or more model point patterns with noise added on the points. Extensive comparative studies are also made among GA, SA the simple hill climbing technique and the random search technique. The results are summarized in Table 1.

Since Case (7) is the most difficult case in which some points are missing, extraneous points are added, and all points are corrupted with noise, this case is used as a test bed for comparison among GA with various cross-over and mutation probabilities, SA, the simple hill climbing method and the random search method.

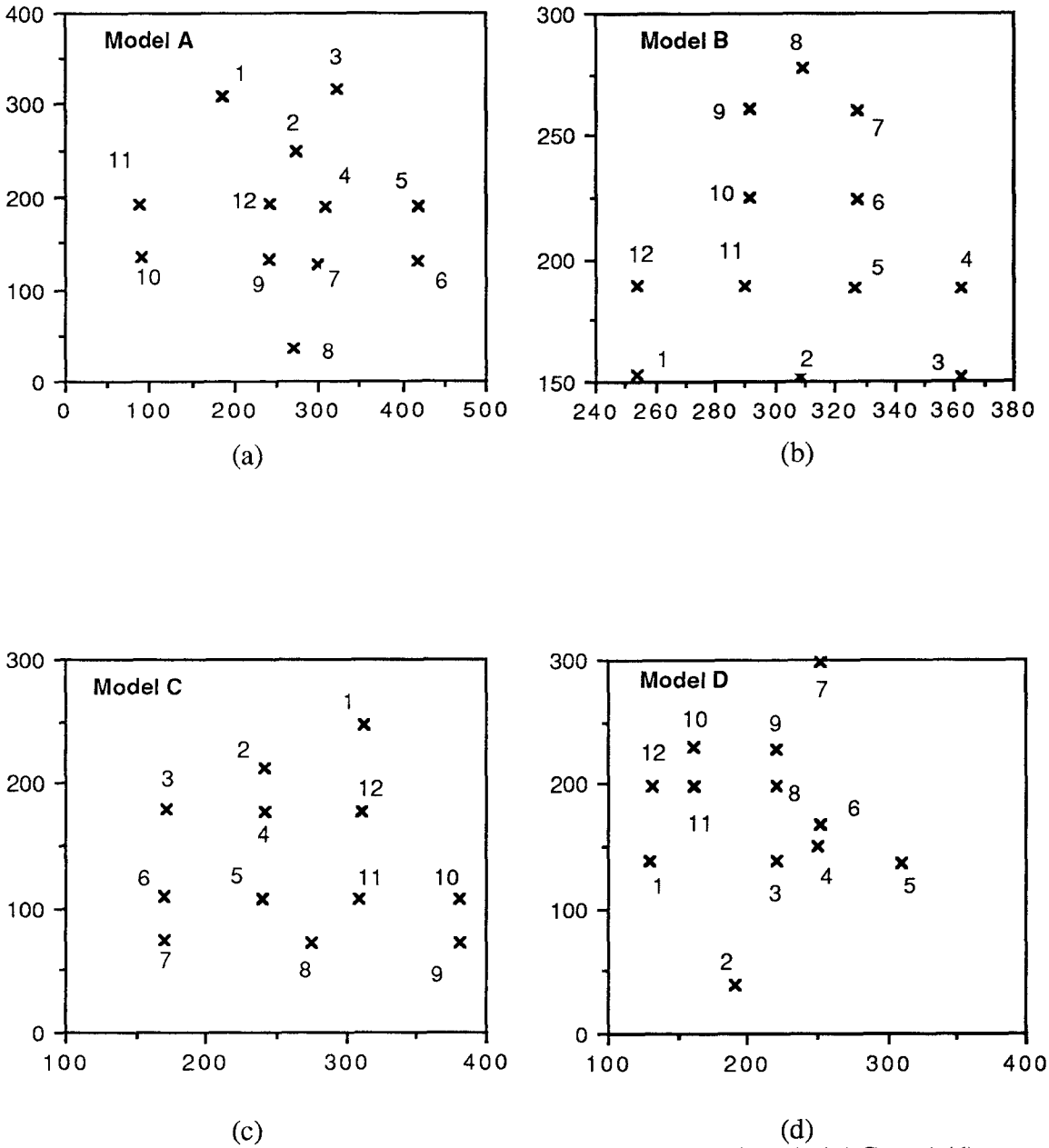
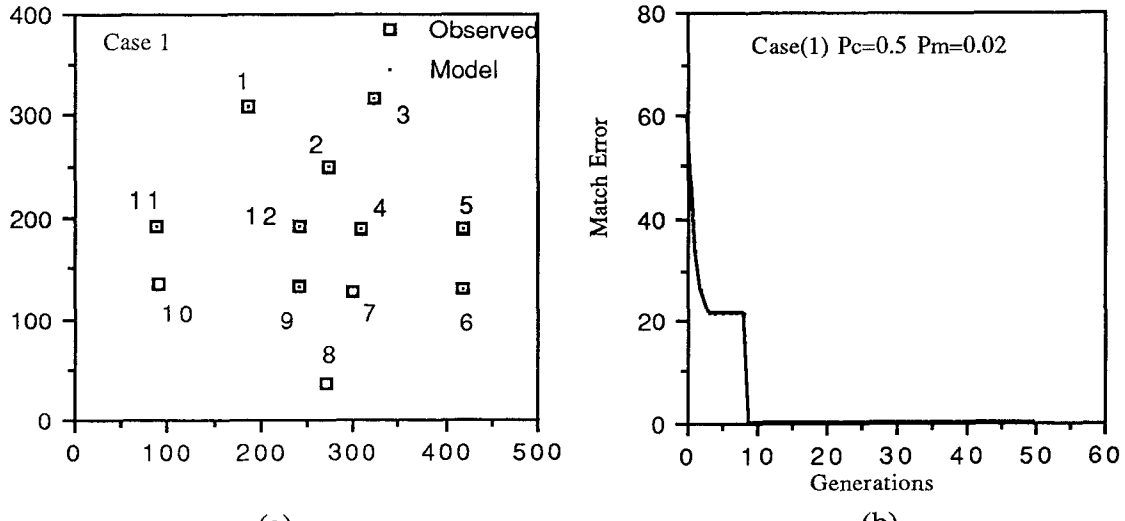
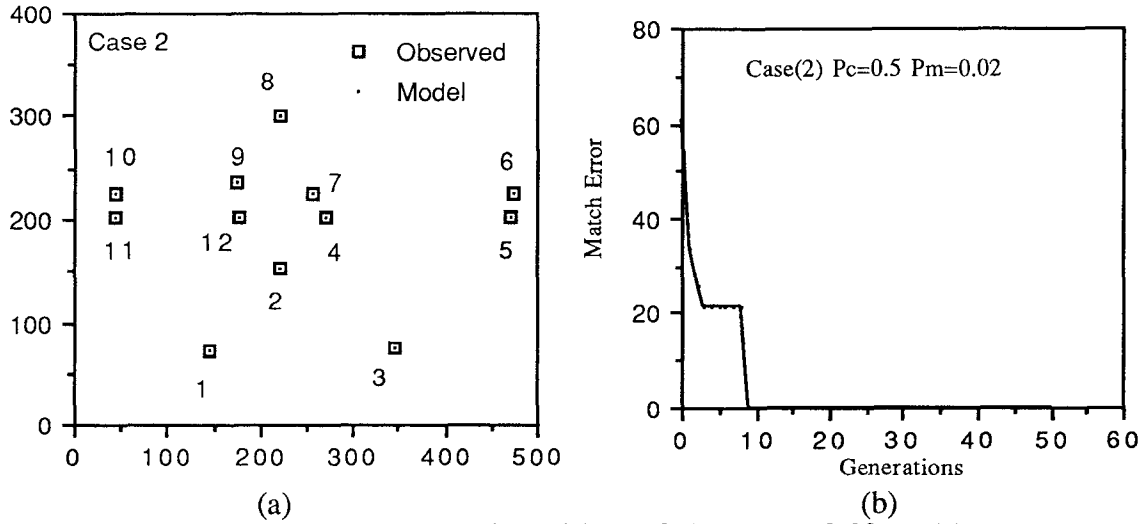



**Figure 1.** Four model point patterns:(a) Model A,(b) Model B,(c) Model C, and (d) Model D.




(a) (b)

**Figure 2.** Experiment Case (1) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model A to the observed pattern, (b) the convergence plot.




(a) (b)

**Figure 3.** Experiment Case (2) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model A to the observed pattern, (b) the convergence plot.

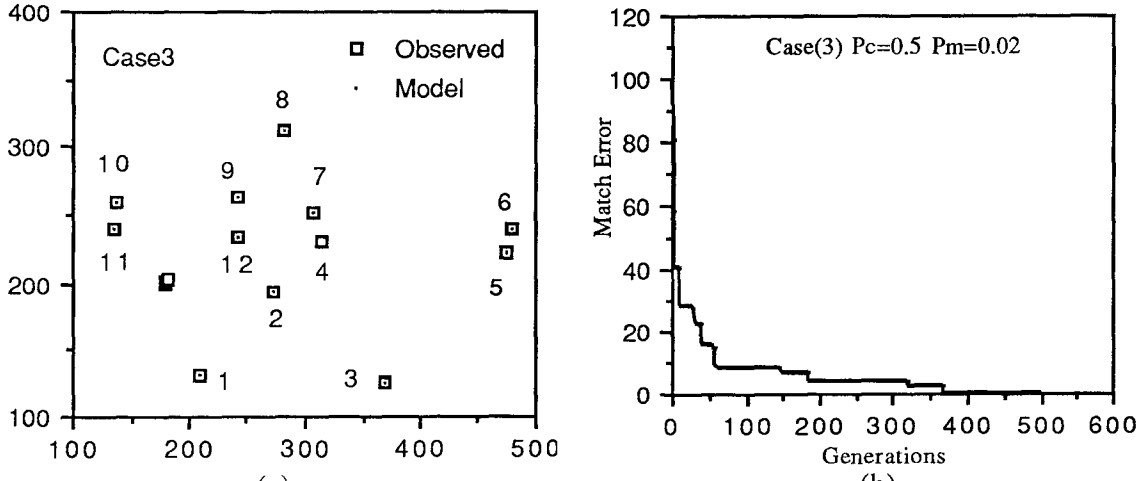



**Figure 4.** Experiment Case (3) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model A to the observed pattern, (b) the convergence plot.

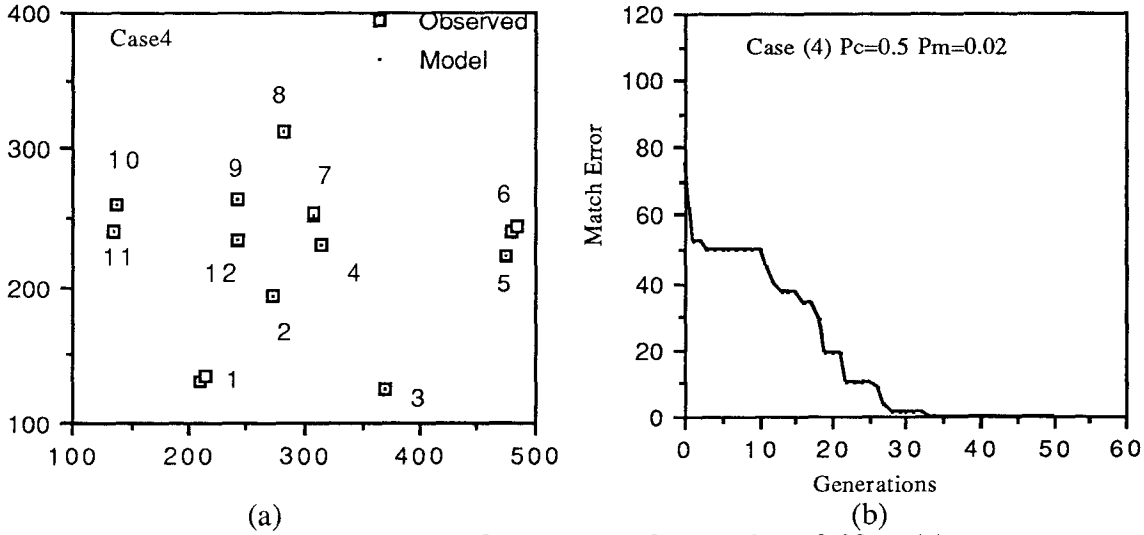



**Figure 5.** Experiment Case (4) using GA with Pc=0.5 and Pm=0.02— (a) Mapping Model A to the observed pattern, (b) the convergence plot.

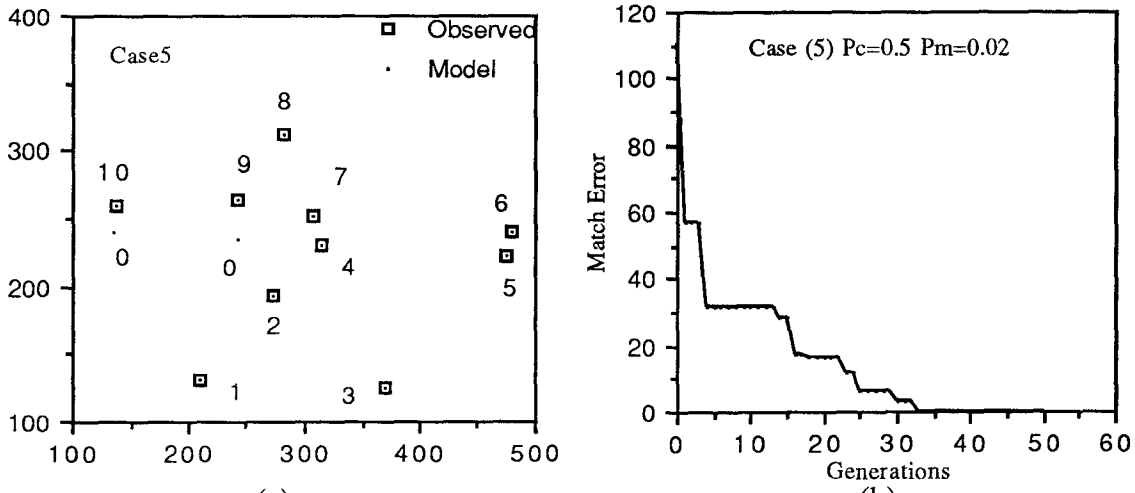



**Figure 6.** Experiment Case (5) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model A to the observed pattern, (b) the convergence plot.

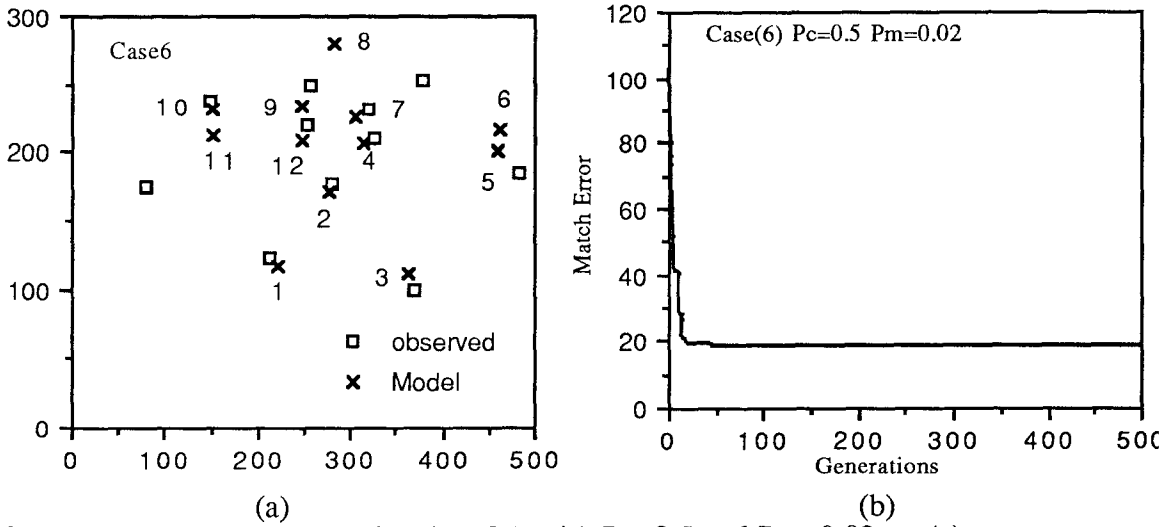



**Figure 7.** Experiment Case (6) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model A to the observed pattern, (b) the convergence plot.

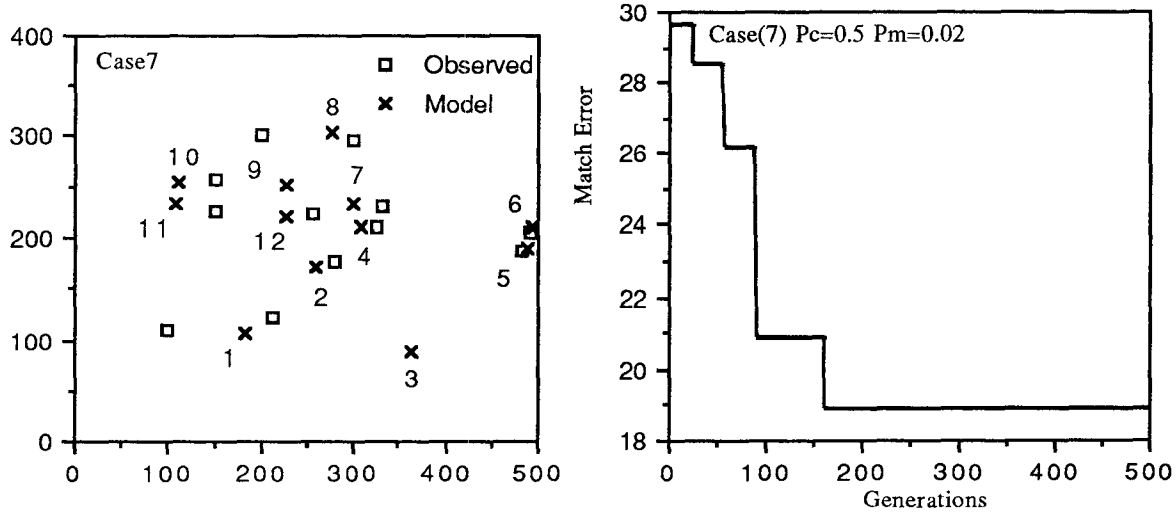



**Figure 8.** Experiment Case (7) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model A to the observed pattern, (b) the convergence plot.

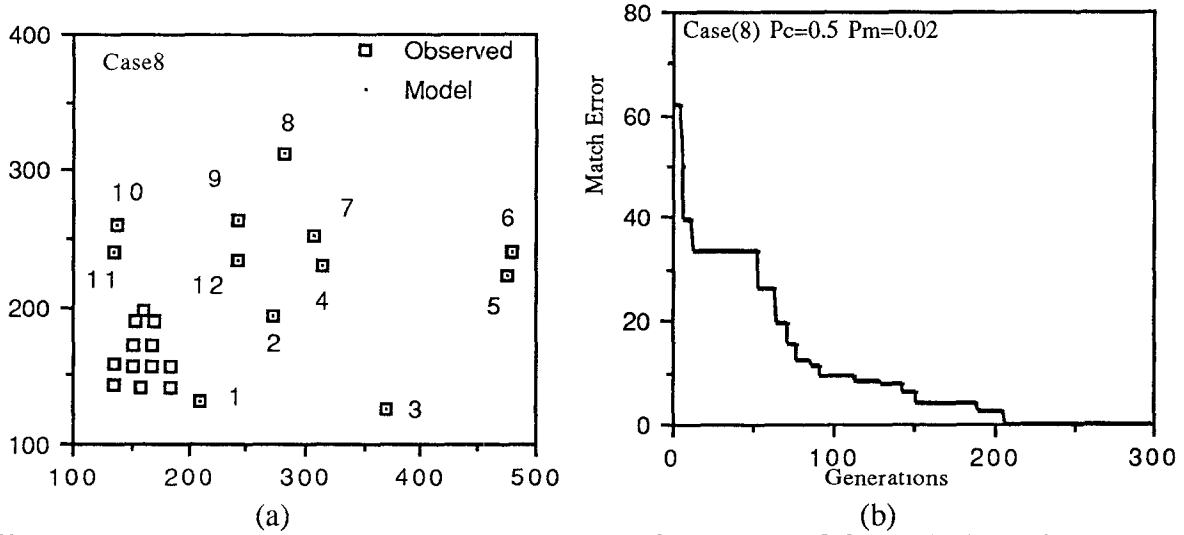



**Figure 9.** Experiment Case (8) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model A to the observed pattern, (b) the convergence plot.

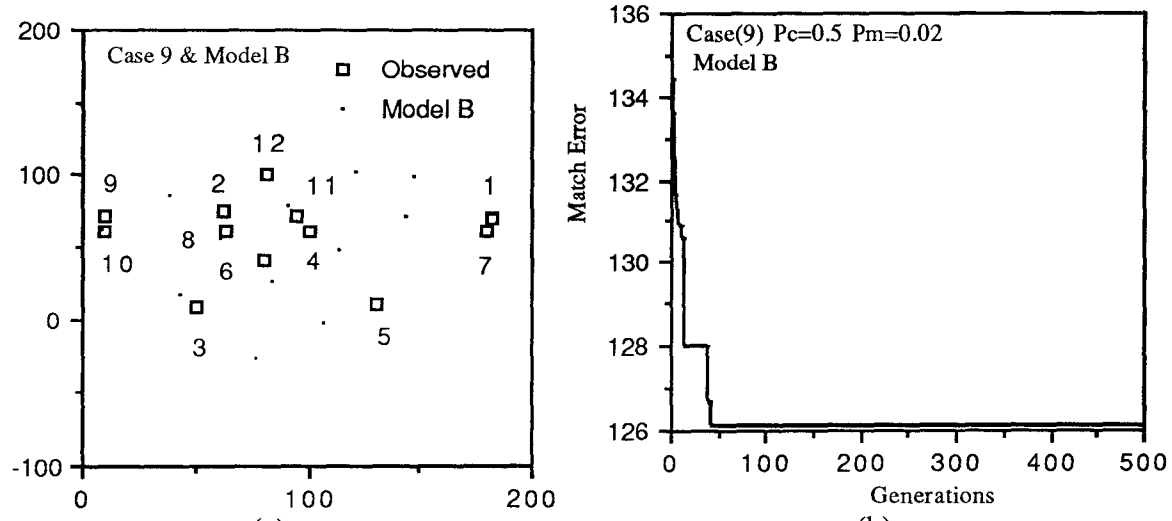



**Figure 10.** Experiment Case (9) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model B to the incompatible observed pattern, (b) the convergence plot.

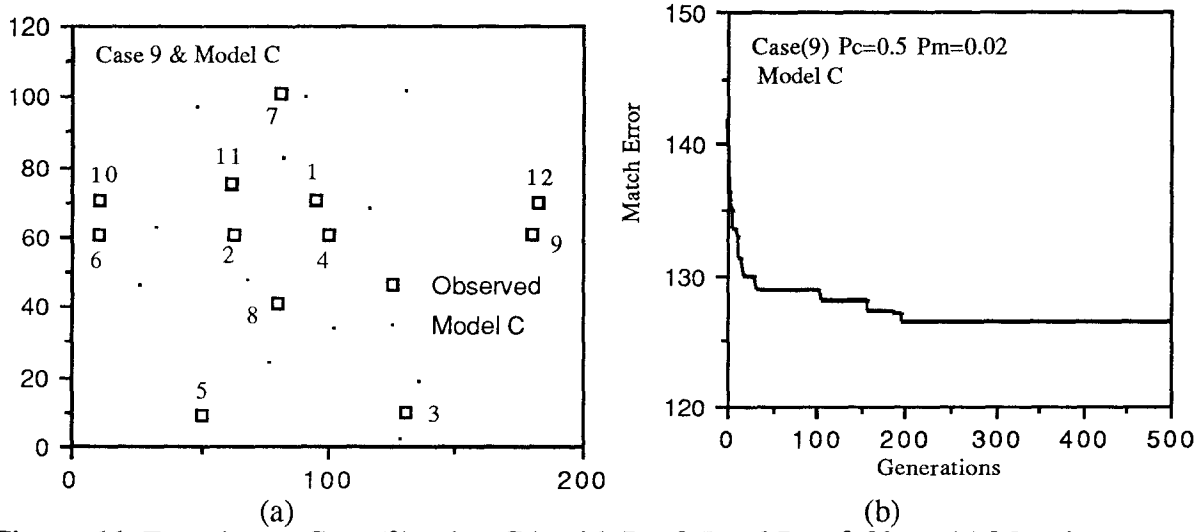



**Figure 11.** Experiment Case (9) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model C to the incompatible observed pattern, (b) the convergence plot.

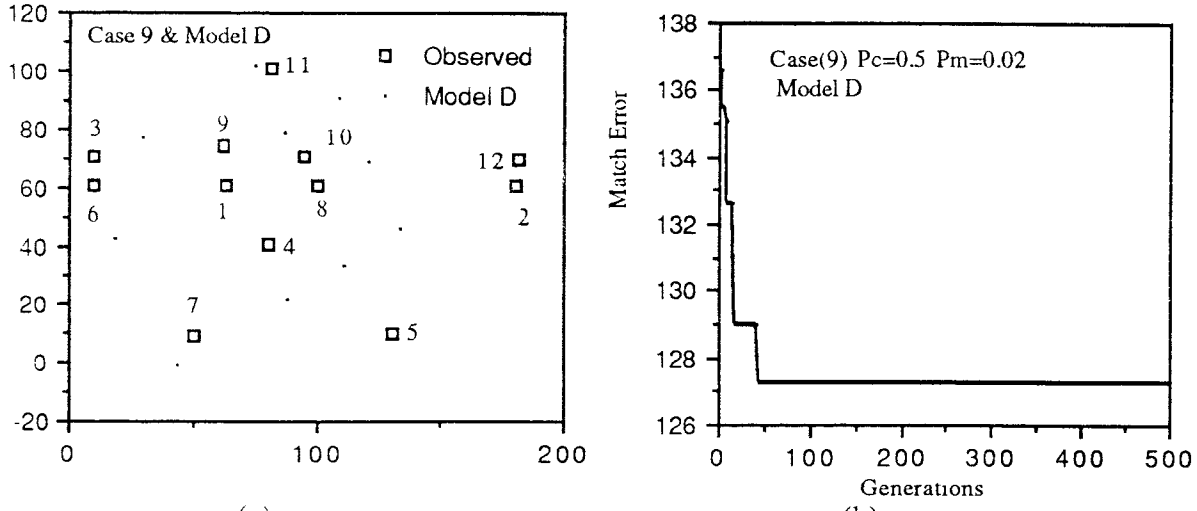



**Figure 12.** Experiment Case (9) using GA with Pc=0.5 and Pm=0.02 — (a) Mapping Model D to the incompatible observed pattern, (b) the convergence plot.

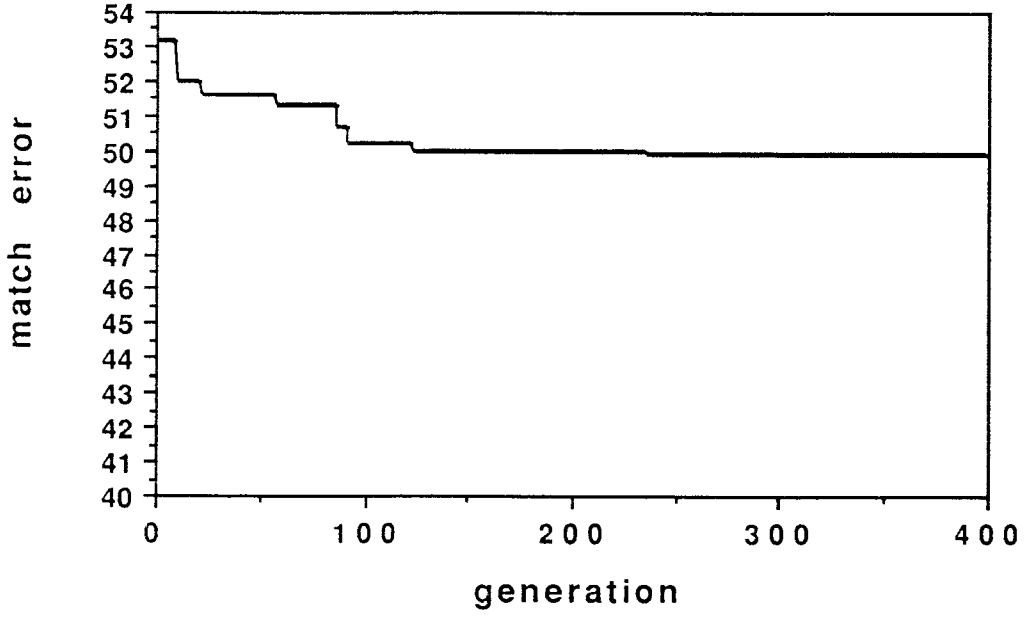



**Figure 13.** The convergence plot for Experiment Case(7) using DA with Pc=0 and Pm=0.03.

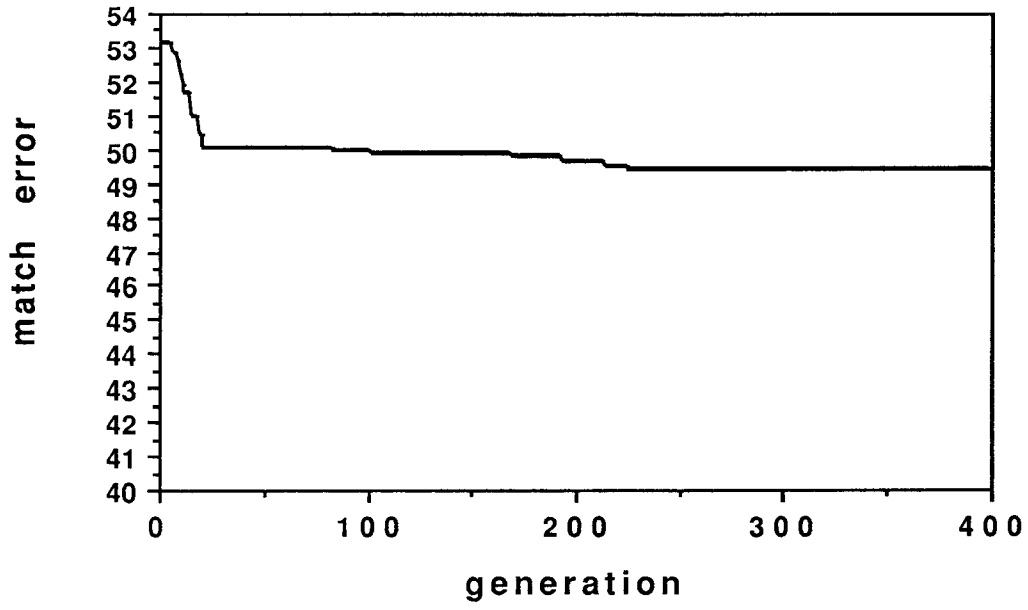



**Figure 14.** The convergence plot for Experiment Case(7) using GA with Pc=0 and Pm=1.

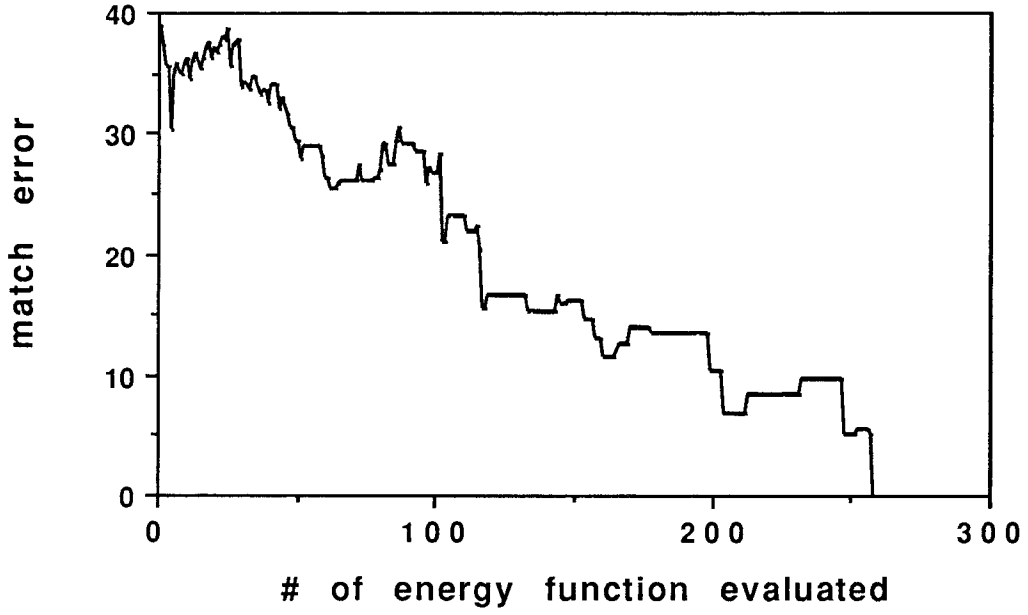



**Figure 15.** The convergence plot for Experiment Case(1) using simulated annealing.

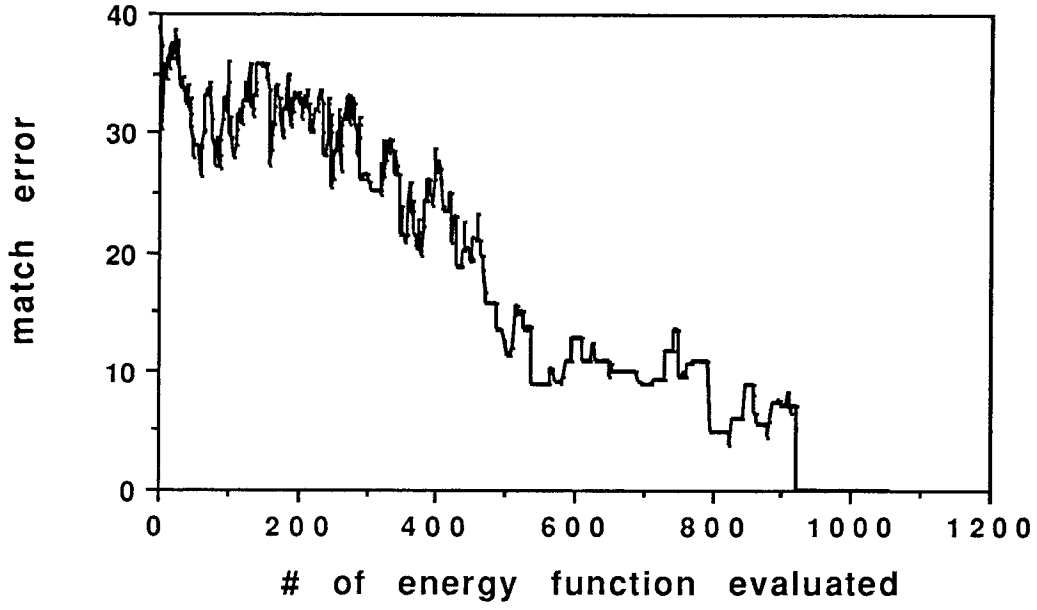



**Figure 16.** The convergence plot for Experiment Case(2) using simulated annealing.

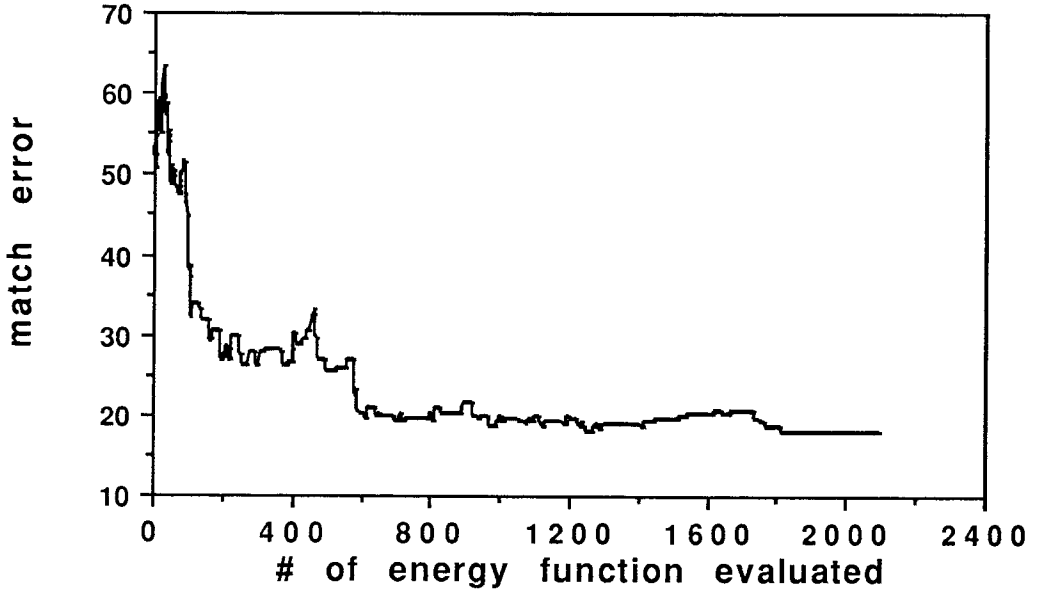



**Figure 17.** The convergence plot for Experiment Case(7) using simulated annealing.

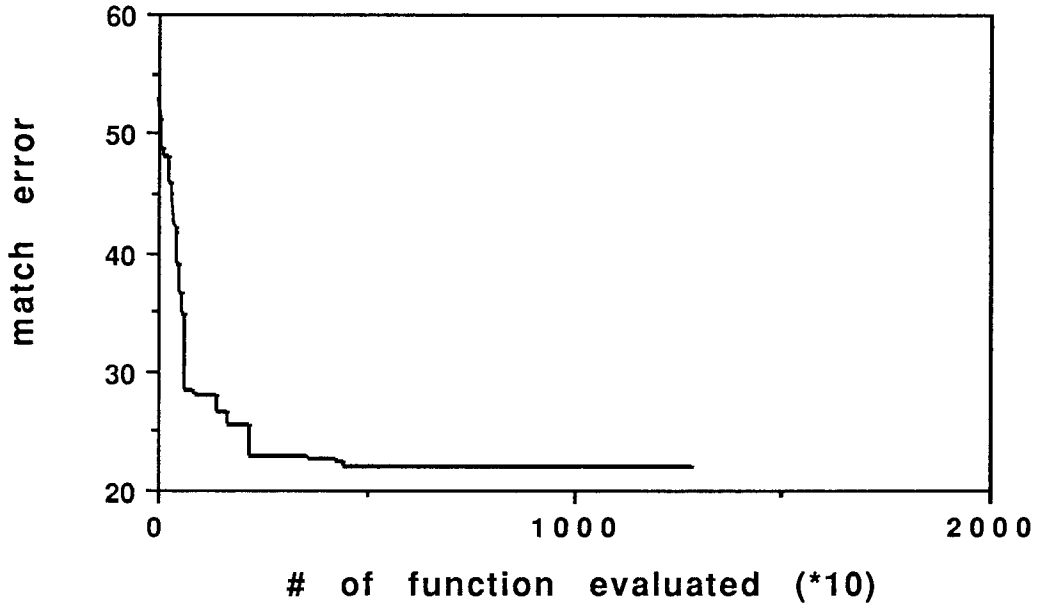



**Figure 18.** The convergence plot for Experiment Case(7) using the simple Hill Climbing technique.

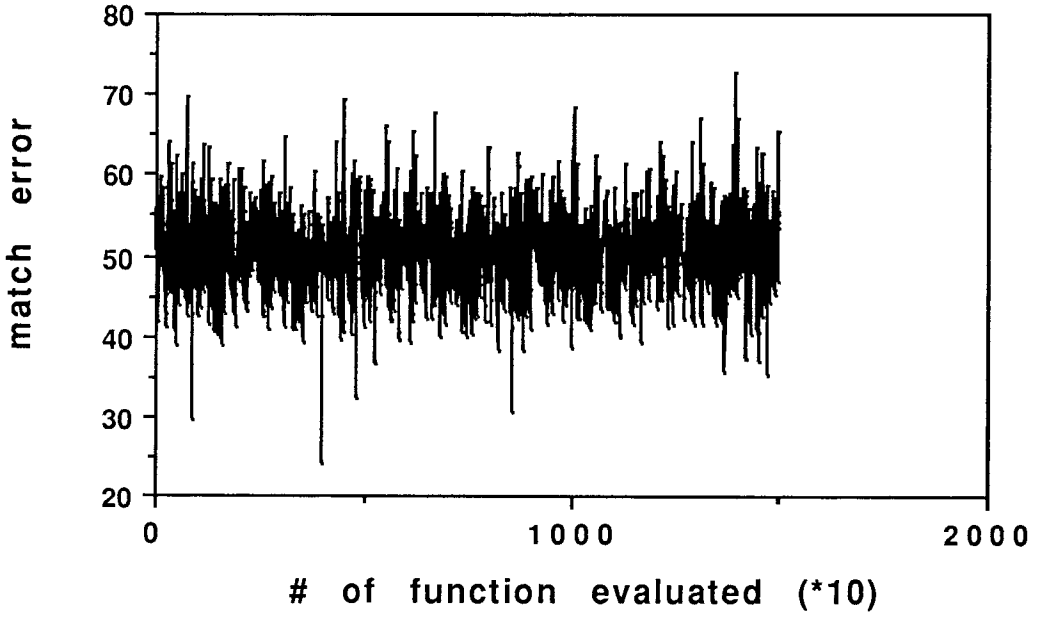



**Figure 19.** The convergence plot for Experiment Case(7) using the random search technique.

Table. 1. Summarized experimental results.

| Methods | Experiments<br>T{Model} — Mapping Model through a similarity transformation.<br>Noise — points are corrupted by additive noise. | Number† of Iterations | Final Match Error | Figures depicting the results |
|---|---|---|---|---|
| **GA**<br>$P_c$=0.5<br>$P_m$=0.02 | Case (1) Observed={Relabelling Model A} | 270 | 0 | Figure 2a, 2b |
| | Case (2) Observed=T{Model A} | 270 | 0 | FIgure 3a, 3b |
| | Case (3) Observed=T{Model A}+Extraneous Points | 10,500 | 0 | Figure 4a, 4b |
| | Case (4)Observed=T{Model A}+Extraneous Points {close to observed points} | 1,500 | 0 | Figure 5a, 5b |
| | Case (5) Observed=T{Model A - Missing Points} | 1,500 | 0 | Figure 6a, 6b |
| | Case (6) Observed=T{Model A}+Noise | 1,500 | 19.2 | Figure 7a, 7b |
| | Case (7) Observed=T{Model A - Missing Points}+Extraneous Points+Noise. | 4,500 | 18.9 | Figure 8a, 8b |
| | Case (8) Observed=T{Model A}+T{other models} | 6,000 | 0 | Figure 9a, 9b |
| | Case (9) Incompatible Observed Point Pattern | >10,000 | ≠0 | Figure 10a, 10b<br>Figure 11, 12 |
| GA | Same as Case (7) — $P_c$=0, $P_m$=0.02 | > 10,000 | 50 | Figure 13 |
| | Same as Case (7) — $P_c$=0, $P_m$=1 | > 10,000 | 50 | Figure 14 |
| SA | Same as Case (1) | 300 | 0 | Figure 15 |
| | Same as Case (2) | 1,000 | 0 | Figure 16 |
| | Same as Case (7) | 4,000 | 20 | Figure 17 |
| Simple Hill Climbing | Same as Case (7) | > 10,000 | 22 | Figure 18 |
| Random Search | Same as Case (7) | >10.000 | ≈24 | Figure 19 |

† For GA, the number of iterations to reach (near) optimal solution = 30×(number of generations).

# CHAPTER 6
# CONCLUSIONS

We have introduced the two heuristic methods for point pattern recognition. The robustness and fast convergence of our algorithms have been demonstrated through experimental results. To find the "best" match between a set of m model points and a set of n observed points, our GA only needs to evaluate NG search nodes where N is the population size and G is the number of generations at which the algorithm converges or stops. In our experimental results, it usually takes less than 200 generations, and in some cases less than 10 generations, for GA to converge as compared to a total of

$$\sum_{i=0}^{m}\binom{n}{m-i}\binom{m}{i}(m-i)! \qquad \text{for } n \geq m \qquad \text{where} \quad \binom{n}{k} = \frac{n!}{(n-k)!(k!)}$$

$$\sum_{i=0}^{n}\binom{m}{m-n+i}\binom{n}{n-i}(n-i)! \qquad \text{for } m > n$$

search nodes that are to be evaluated using exhaustive search.

Simulated Annealing is sequential in nature , but it avoids the search getting stuck in a local minima. If GA can be implemented in parallel, it converges faster than SA even though they may takes the same number of iterations to reach the steady state. In our implementation, each generation of GA evaluated 30 search nodes that is equivalent to 30 iterations in SA. It has been demonstrated through experiments that the proposed heuristic methods perform the point pattern matching task well.

# References


1. Ackley, D. I., G. E. Hinton, and T. J. Seijnowski, "A Learning Algorithm for Boltzmann Machines." *Cognitive Science*, Vol.9 (1983): 147-169,.

2. Attneave, F. "Some Informational Aspects of Visual Perception." *Psychol. Rev.*, vol.61, no.3 (1954): 183-193, .

3. Ansari, N. and E. J. Delp, "Partial Shape Recognition: A Landmark-Based Approach." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.12, no.5, May (1990): 470-483.

4. Ansari, N. and E. J. Delp, "Recognizing Planar Objects in 3-D Space." *Proc. SPIE Automated Inspection and High-Speed Vision Architectures III*, vol.1197, Nov. 5-10, Philadelphia, PA (1989): 127-138.

5. Ayache, N. and O. D. Faugeras, "Hyper: a New Approach for the Recognition And Positioning of Two-Dimensional Objects." *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol.PAMI-8, no.1, Jan. (1986): 44-54.

6. Bhanu, B. and J. C. Ming, "Recognition of Occluded Object: a Cluster-Structure Algorithm." *Pattern Recognition*, vol.20, no.2, (1987): 199-211, .

7. Bhanu, B. and O. D. Faugeras, "Shape Matching of Two-Dimensional Objects." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.PAMI-6, no.2, Mar. (1984): 137-156, .

8. Bolles, R. C. and R. A. Cain, "Recognizing and Locating Partially Visible Objects: the Local-Feature-Focus Method." *Int. J. Robotics Res.*, vol.1, no.3, Fall (1982): 57-82.

9. Davis, L., Genetic Algorithms and Simulated Annealing. Las Altos, CA: Morgan Kanfmann, (1987).

10. Davis, L. S., "Shape Matching Using Relaxation Techniques." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.PAMI-1, no.1, Jan. (1979): 60-72.

11. Ettinger, G. J., "Large Hierarchical Object Recognition Using Libraries of Parameterized Model Sub-Parts." in *Proc. IEEE Comput. Soc. Conf. Computer Vision and Pattern Recognition*, Ann Arbor, MI, June 5-9 (1988): 32-41.

12. Fitzpatrick, J. M. and J. J. Grefenstette, "Genetic Algorithms in Noisy Environments." Machine Learning, Vol.3. (1988): 101-120.

13. Goldberg, D. E., *Genetic Algorithms in Search, Optimization & Machine Learning.* Reading, MA: Addison Wesley, (1989).

14. Gorman, J. W., O. R. Mitchell, and F. P. Kuhl, "Partial Shape Recognition Using Dynamic Programming." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.PAMI-10, no.2, Mar. (1988): 257-266.

15. Grimson, W. E. L., "On Recognition of Curved Objects." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.PAMI-11, no.6, June (1989): 632-643.

16. Holland, J. H., Adaptation in Nation and Artificial Systems. Ann Arbor, MI: University of Michigan Press, (1975).

17. Huttenlocher, D. P. and S. Ullman, "Object Recognition Using Alignment." in *Proc. IEEE 1st Int. Conf. Computer Vision*, London (1987): 102-111.

18. Ibison, M. C. and L. Zapalowski, "On the Use of Relaxation Labelling in the Correspondence Problem." *Pattern Recognition Letters*, April (1986): 103-109.

19. Kalvin, A. E., Schonberg, J. T. Schwartz, and M. Sharir, "Two-Dimensional Model-Based, Boundary Matching Using Footprints." *Int. J. Robotics Res.*, vol.5, no.4, Winter (1986): 38-51.

20. Kirpatrick, S., C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by Simulated Annealing." *Science* Vol.220 (1983): 671-680.

21. Knoll, T. F. and R. C. Jain, "Recognizing partially visible objects using feature indexed hypotheses." *IEEE Trans. Robotics and Automation*, vol.RA-2, no.1, Mar. (1986): 3-13.

22. Koch, M. W. and R. L. Kashyap, "Using Polygons to Recognize and Locate Partially Occluded Objects." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.PAMI-9, no.4, July (1987): 483-494.

23. Lamdan, Y., J. T. Schwartz, and H. J. Wolfson, "Object Recognition by Affine Invariant Matching." in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Ann Arbor, MI., June 5-9 (1988): 335-344.

24. Lamdan, Y., J. T. Schwartz, and H. J. Wolfson, "On Recognition of 3-D Objects from 2-D Images." in *Proc. IEEE Int. Conf. Robotics and Automation*, Philadelphia, PA. Apr. (1988): 1407-1413.

25. Price, K. E., "Matching Closed Contours." *Proc. Seventh Int. Conf. Pattern Recognition*, Montreal, P.Q., Canada, July 30-Aug. 2, (1984): 990-992.

26. Ranade, S. and A. Rosenfeld,"Point Pattern Matching By Relaxation." *Pattern Recognition* , vol.12 (1980): 269-275.

27. Rawlins, G. J. E., Fundations of Genetic Algorithms. San Mateo, CA: Morgan Kanfmann, (1991).

28. Ripley, B. D., *Stochastic Simulation*, New York: John Wiley & Sons, (1987)56-88.

29. Turney, J. L., T. N. Mudge, and R. A. Volz, "Recognizing Partially Occluded Parts." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol.PAMI-7, no.4, July (1985): 410-421.

30. Van Laarhoven, P. J. M. and E. H. L. Aarts, Simulated Annealing: Theroy and