Genetic algorithm approach to image segmentation

by

Asma Akhter Saudagar

A Thesis Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Department: Electrical and Computer Engineering
Major: Electrical Engineering

Iowa State University
Ames, Iowa
1995

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I want to express my deepest gratitude to my advisor Dr. Lalita Udpa who has guided me through this research. All along, she encouraged me and provided new ideas and suggestions that helped me with my study. Her motivation, enthusiasm and her faith in me are mainly responsible for my achievements.

I want to thank Dr. Satish Udpa and Dr. Kothari for being on my committee and being supportive and encouraging.

I want to thank my father, my mother and my sisters for making the opportunity of doing a Masters possible to me. I owe my thirst of knowledge to my parents who are themselves professors, especially to my father who after his two Ph.Ds and two D.Litts continues his quest for learning.

Most importantly, I want to thank my dear husband, Irfan. He has not only been my pillar of strength and my source of inspiration, but has helped and supported me in ways more than one, boosting my self esteem and increasing my self confidence.

Special thanks to my office-mates, collegues, and friends for their cooperation and help.

CHAPTER 1. INTRODUCTION

Digital image processing and analysis has seen an impressive growth in the past decade in terms of theoretical developments and application. These techniques constitute a leading technology in a number of very important areas, e.g. digital telecommunications, medical imaging, multimedia systems, robotics, remote sensing via satellites, automated inspection of industrial parts, graphic arts, radar, sonar and acoustic image processing etc. Interest in digital image processing arises from the principal application areas such as, the enhancement of pictorial information for human interpretation and processing of scene data for autonomous machine perception. Image processing systems play an important role in scientific, industrial, biomedical and space applications. A typical image processing system generally performs the following operations: (1) aquisition, (2) storage, (3) processing, (4) communication, and (5) display.

Digital image processing deals with the transformation of an image to a digital format and its processing by digital computers. The basic image processing requires:

1. Digital image formation

2. Digital image restoration

3. Digital image enhancement

4. Digital image coding
5. Digital image compression
6. Digital image analysis

Digital image formation is the first step in a digital image processing application. Each digital image formation subsystem introduces a deformation or degradation to the digital image (e.g. geometrical distortion, noise, nonlinear transformation). Digital image restoration is commonly defined as the processing of the measured image data to compensate for artifacts introduced by the imaging system. The image enhancement techniques deal primarily with improvement of the quality of the digital image. This usually involves contrast enhancement, digital image sharpening and noise reduction. Digital images usually require a very large amount of memory for their storage. The reduction of the memory requirements is of utmost importance in many applications like image storage or transmission. Digital image coding and compression take advantage of the information redundancy existing in the image in order to reduce and compress its information content. Image compression plays a vital role in applications involving image databases, digital image transmission, facsimile, digital video etc.

Finally, image analysis is the interpretation of the information content in an image data. Image analysis comprises of many functions such as:

- Feature extraction
- Segmentation
- Classification

Feature extraction as the name suggests, involves extraction of certain features from an image. These include spatial features, transform features, edges and boundaries, sharp features, moments, and textures. Segmentation entails division of an image into regions of similar attributes. The methods for segmentation include, besides other techniques, template matching, thresholding, boundary detection, clustering, quad-trees, texture matching. Classification deals with associating objects having identical features with certain properties and grouping together such objects under a particular class. The techniques for classification include clustering, decision trees, similarity measures, and spanning trees.

Image segmentation is generally the first step in image analysis. Segmentation, in a broad sense partitions the input image into its constituent parts or objects.

A multitude of algorithms have been presented in the literature in all the above-mentioned areas. Due to the plethora of applications of image analysis, the algorithms for object identification and image segmentation are of prime interest. Space applications require recognition and analysis of objects in images obtained from space missions. In medical imaging, segmentation procedures are mainly used for processing a variety of images obtained using MRI, PET, X-rays, angiograms and other radiological images. These images are used for detection of tumors and other disorders. Radar and sonar images are used for detection and recognition of various types of targets. Several classical methods for image segmentation exist, some of them being thresholding, template matching etc. However, these methods are more or less heuristic and specific to a particular application. For example, in case of thresholding it is difficult to decide a threshold to segment an image into foreground and background. The results obtained from thresholding depends to a large extent on the image under consideration.

This thesis proposes a new new simple yet robust method for image segmentation that is based on genetic algorithm. Although GA techniques have been used extensively in the optimization problems, the application of genetic algorithm in image processing is fairly recent and the results are encouraging.

The rest of the chapters are organized as follows. Chapter 2 provides an introduction to genetic algorithms and attempts to explain their important elements. The image segmentation algorithm based on genetic algorithm is explained in chapter 3. The results of implementation of the algorithm developed in this thesis, on simple simulated images are first described in chapter 4. These results help in developing an insight into the performance of the algorithm. A discussion of the performance and conclusions regarding feasibility of the approach are finally presented in chapter 5.

CHAPTER 2. GENETIC ALGORITHM: AN OVERVIEW

Genetic algorithms are stochastic search methods, the principles of which are inspired from the laws of genetics, natural selection and evolution of organisms. Their main attractive characteristic is their ability to efficiently deal with hard combinatorial search problems, where the parallel exploration of the search space eliminates to a large extent the possibility of getting stuck in local extrema. The approach is based on the fact that individuals tend to pass on their traits to their offsprings. The fittest of the individuals tend to have more offsprings. In effect, the tendency is to drive the population towards favorable traits. Over long periods of time, entirely new species are produced which are better adapted to particular ecological conditions. The genetic algorithm is based on the mechanisms exhibited by nature incorporating the robustness, the efficiency, and the flexibility of biological systems. Genetic algorithms was developed by John Holland, his colleagues and his students at the University of Michigan.

Source of Inspiration: Natural Evolution

This section considers some features of biological evolution that inspired John Holland's invention of GA [3]. GA was proposed as an attempt to mimic some of the processes observed in natural evolution. Biologists have been intrigued with the mechanics of evolution since the evolutionary theory of biological change gained acceptance. It is indeed very surprising that life at the level of complexity that we observe, could have evolved in the relatively short time suggested by fossil records. The mechanisms that triggered this evolution are not fully understood, but some of its features are known. Evolution takes place at the chromosomes - organic devices for encoding the structure of living beings. A living being is created partly by a process of decoding the chromosomes. The specifics of chromosomal encoding and decoding processes are not fully understood, but here are some general features of the evolutionary theory that are widely accepted:

- Evolution is a process that operates on chromosomes rather than on the living beings they encode.
- Natural selection is the link between chromosomes and the performance of their decoded structures. Processes of natural selection cause those chromosomes that encode successful structures to reproduce more often than those that do not.
- The process of reproduction is the point at which evolution takes place. Mutations may cause the chromosomes of biological children to be different from those of their biological parents, and recombination processes may create quite different chromosomes in the children by combining material from the chromosomes of two parents.
- Biological evolution has no memory. Whatever it knows about producing individuals that will function in their environment is contained in the gene pool -

the set of chromosomes carried by the current individuals - and in the structure of the chromosomes decoders.

These features of natural evolution intrigued John Holland in the early 1970's, and he believed that if this was appropriately incorporated in a computer algorithm, it might yield a technique of solving difficult problems, the way nature has done - - through evolution. These algorithms, using simple encodings and reproduction mechanisms, did solve some extremely difficult problems. In reference to their origin in the study of genetics, Holland named this field *genetic algorithms*

Analogy to Genetics

Every living being is composed of one or more cells each of which contains a nucleus. Each nucleus consists of one or many chromosomes. A chromosome is identified as custodians of the trait-determining factors, called genes, that are passed on when cells divide and when offsprings are produced. Before we delve into further discussion on GA, some of the terms often used are defined with respect to their biological counterparts.

- Gene: Gene is the basic unit of a chromosome responsible for the characteristic feature in an individual.
- Chromosome: A chromosome is an organic device for encoding the structure of living beings and is composed of genes.
- Schema: A schema is a similarity template describing a subset of strings with similarities at certain position.

- Individual: A living being that is characterized by distinct features dependent on the chromosomes.
- Population: A group of individuals in the same domain.

What are Genetic Algorithms

Genetic algorithms [2] are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among individuals with a structured yet randomized information exchange. In every generation, a new set of artificial individuals (strings) is created using bits and pieces of the fittest, found in the earlier generation. While randomized, a genetic algorithm is not a simple random walk. These algorithms efficiently exploit the historical information to speculate on new search points with expected improved performance. Genetic algorithms are theoretically and empirically proven to provide robust search in complex spaces. These algorithms are computationally simple yet powerful in their search for improvement. The GA is an example of search procedure that uses random choice as a tool (as is exhibited by nature) to guide a highly exhaustive search through a coding of the parameter space. An important aspect to recognize is that randomized search does not necessarily imply directionless search. Many papers and dissertations establish the validity of the technique in function optimization and control applications. Having been established as a valid approach to problems requiring efficient and effective search, genetic algorithms are now finding more widespread application in business, scientific and engineering circles. This thesis presents yet another application of GA to Image Processing.

Elements of Genetic Algorithms

A simple genetic algorithm that yields good results in many practical problems is composed of three main mechanisms namely: a way of *encoding* or representing solutions to the problem, to chromosomes, a *fitness function* that returns a figure of merit of a chromosome in the context of the problem, and *reproduction operators* that help in evolving new generations.

1. Encoding - The technique for encoding the solutions may vary from problem to problem. In most of the work done so far, encoding is carried out using bit strings. A certain amount of art is required in selecting a good encoding technique that is appropriate to an application.

2. Fitness function - The fitness function is the link between the GA and the problem to be solved. An evaluation function takes a chromosome as input and returns a number that is a measure of the chromosome's performance on the problem to be solved. Fitness functions play the same role in GAs as that of environment in natural evolution. The interaction of an individual with its environment provides a measure of fitness, and the interaction of a chromosome with a fitness function provides a measure of fitness that the GA uses when carrying out reproduction.

3. Reproduction operators - The reproduction operators are essential in evolving from one generation to another. Three basic reproduction operators that are commonly used are described below:

- **Reproduction**

  Reproduction [2] is a process in which individual strings are copied according to their fitness function values. Intuitively, we can think of the fitness function as some measure of profit, utility, or goodness that we want to maximize. Copying strings according to their fitness values means that strings with higher value have a higher probability of contributing one or more offsprings in the next generation. This operator, of course, is an artificial version of natural selection, a Darwinian survival of the fittest among individuals. In natural populations, fitness is determined by a creature's ability to survive predators, pestilence, and other obstacles to adulthood and subsequent reproduction. In the artificial setting, the fitness function is the final arbiter of the individuals life or death.

- **Crossover**

  Crossover [2] may proceed in two steps. First, members of the newly reproduced strings in the mating pool are mated at random. Second, each pair of strings undergoes crossing over as follows: an integer position k along the string is selected uniformly at random between 1 and the string length less one $[1, l-1]$. Two new strings are created by swapping all the characters between positions $k+1$ and l inclusively. For example, consider individuals I_1 and I_2 of string length 6:

$$I_1 = 0110|10$$

$$I_2 = 1100|11$$

Suppose in choosing a random number between 1 and 5, we obtain a $k = 4$ (as indicated by the separator symbol |). The resulting crossover yields two new individuals where the prime (') means the strings are a part of the new generation:

$$I_1' = 0110|11$$

$$I_2' = 1100|10$$

The crossover produces children that are radically different from their parents. A point to be noted is that the crossover technique will not introduce differences for a bit in a position where both parents have the same value. An extreme instance occurs when both parents are identical. In such cases, crossover can introduce no diversity in the children.

- **Mutation**

  Mutation plays a secondary role in the operation of genetic algorithms. Mutation [2] is needed because, even though reproduction and crossover effectively search and recombine the better features of the parents, occasionally they may become overzealous and lose some potentially useful genetic material. In the artificial genetic systems, the mutation operator protects against such an irrecoverable loss. It mainly deals with flipping the bit from 1 to 0 or vice versa. By itself, it is random walk through the string space. When used sparingly with reproduction and crossover, it is an insurance policy against premature loss of important notions.

The mechanics of reproduction and crossover are surprisingly simple, involving random number generation, string copies, and some partial string exchanges. Nonetheless, the combined emphasis of reproduction and the structured, though randomized, information exchange of crossover give genetic algorithms much of their power. The process of reproduction and crossover in a GA is a kind of exchange in search for better and better performance.

Given these initial components - a problem, a way of encoding solutions to it, a function that returns a measure of how good an encoding is, and reproduction operators - we can use a GA to carry out simulated evolution on a population of solutions. The next section describes the top-level description of the GA itself - the algorithm that uses these components to simulate evolution.

The Genetic Algorithm

The Genetic Algorithm consists of the following steps.

1. Generate the initial population.
2. Evaluate the fitness of each individual according to a fitness function.
3. Select the fittest individuals for mating.
4. Apply reproductive operators (e.g. cross-over, mutation) to create offsprings.
5. Calculate the fitness of the offsprings.
6. Select the best fit individuals from the pool of individuals in the current generation and the offsprings. These form the population of the next generation.
7. If stopping criterion is not met go to step 3, else stop.

In every iteration, we move from one generation to the other. By transforming the previous set of good individuals to a new one, the operators generate a new set of individuals that have a better than average chance of also being good. When this cycle of evaluation, selection, and genetic operations is iterated for many generations, the overall fitness of the population generally improves and the individuals in the population represent improved "solutions" to the problem at hand. The genetic algorithm can be viewed as a procedure [6] for searching the space of all possible binary strings of a fixed length l (denoted as $\{0,1\}^l$) which is a 2^l-dimensional hyperspace.

The Modules in GA

The GA [3] can be divided into different modules:

- Population module: To start with, an initial population is required and there are different ways to generate it. The size of the population can be fixed. Each member in the population is an individual made up of bits (1 or 0).
- Evaluation module: This module defines the fitness function which is used to evaluate the fitness of each individual.
- Reproduction module: Different reproduction operators can be defined, the most common being crossover and mutation which create offsprings (new individuals). The Population Module asks the Reproduction Module for a new population. The Reproduction Module asks the Population Module for parents to be used in the reproduction events. In each reproduction event, the Reproduction Module gets parents from the Population Module, applies the reproduction operators and sends the children created to the Population Module

until enough children have been generated. The Population Module interacts with the Evaluation Module whenever a new set of children has been produced, and asks it to evaluate each new child before the the population of the next generation is chosen.

The interactions between these modules can be explained with Figure 2.1 shown below.

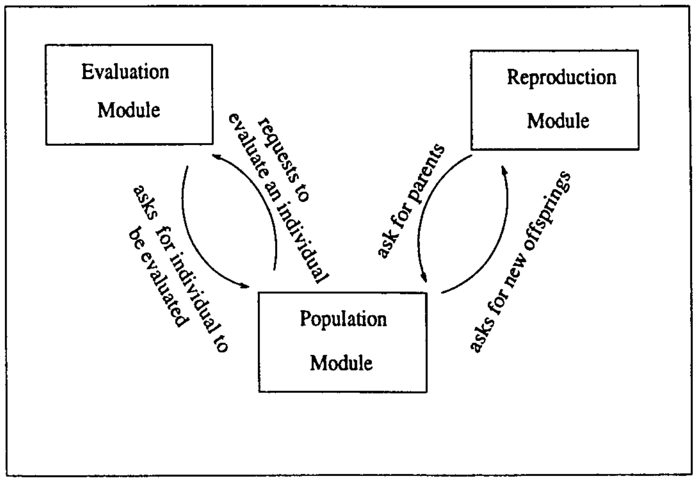


Figure 2.1: Interactions between the modules

Schema Theorem

A schema is a similarity template describing a subset of strings with similarities at certain position. The schema theorem states states a schema occurring in chromosomes with above-average evaluations will tend to occur more frequently in the next generation, and the one occurring in chromosomes with below-average evaluations will tend to occur less frequently (ignoring the effects of mutation and crossover). The details of Schema Theorem and other theoretical results are treated in detail in [2]. This feature of GAs has been described as one of *intrinsic parallelism*, in that the algorithm is manipulating a large number of schema in parallel. The reproduction mechanisms cause the best schemata to proliferate in the population, combining and recombining to produce high-quality combinations of schemata on single chromosomes.

What's so special about GA?

The search involved in genetic algorithms is from a population of points and not from a single point. Moreover, GA uses probabilistic transition rules, instead of deterministic rules to guide their search. By working from a population of well-adapted diversity instead of a single point, the GA adheres to the old adage that there is safety in numbers. GAs are blind, i.e. they do not require any auxiliary information. To perform an effective search for moving towards better solution, GAs only require a function value associated with each individual. This characteristic makes GA a more canonical method than many search schemes. The mechanics of a GA is astonishingly simple, involving nothing more complex than copying strings and swapping partial strings. The explanation of why this simple process works is much more subtle and powerful. Simplicity of operations and power of effect are two of the main attractions of genetic algorithm approach.

Crossover is regarded as the distinguishing feature of GA and as a critical accelerator of the search process. If only mutation operator is used for reproduction, the performance of the GA is degraded. This [2] is analogous to reproduction involving only one parent (asexual reproduction) as mutation is performed on a single parent to produce one or more children. Sexual reproduction on the other hand, is employed by more complicated creatures on our planet and it involves more than one individual. Those individuals must differ in important respects, and they must spend time and energy finding each other when reproduction is to occur. Since this type of reproduction has won out in the arena of natural selection, there must be some respects in which this overhead is repaid. Crossover allows rapid combination of beneficial new traits in a way that cannot be duplicated by mutation.

CHAPTER 3. IMAGE SEGMENTATION

Image segmentation is the fundamental process in image analysis. Segmentation entails division or separation of an image into regions of similar attributes. The most basic attribute for segmentation is the image intensity (luminance for a monochromatic image). Segmentation algorithms basically identify homogeneous image regions, each corresponding to objects or to background. Regional segmentation techniques can be grouped in three classes. Local techniques employ the local properties within the image neighborhood. Global techniques segment the image on the basis of global information. Split and merge techniques employ both pixel proximity and region homogeneity in order to obtain good segmentation results. Segmentation is one of the most important tasks in image processing as it determines the eventual success or failure of the analysis. The well-known image segmentation techniques are thresholding, edge-detection, boundary extraction etc.

Problem Statement

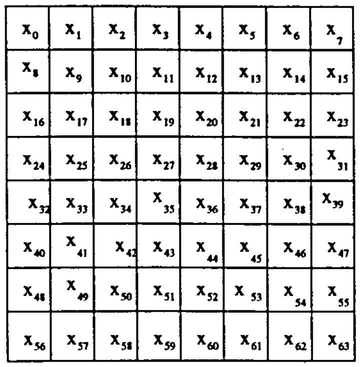
In this thesis, we address the problem of detecting homogeneous image regions, corresponding to image objects or background. In other words, we are concerned with identifying objects in a raw (noisy) image consisting of multiple objects.

Image Model

A digitized image is typically represented by a matrix X with components, x_{ij}, whose values represent the intensity of pixel (i,j). A raw image is assumed to have a picture component superimposed by a noise component. Such a noisy image is modeled by the equation

$$x_{ij} = f_{ij} + n_{ij} \tag{3.1}$$

Here, f_{ij} is the image intensity and the noise n_{ij} is assumed to be normally



x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7
x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
x_{16}	x_{17}	x_{18}	x_{19}	x_{20}	x_{21}	x_{22}	x_{23}
x_{24}	x_{25}	x_{26}	x_{27}	x_{28}	x_{29}	x_{30}	x_{31}
x_{32}	x_{33}	x_{34}	x_{35}	x_{36}	x_{37}	x_{38}	x_{39}
x_{40}	x_{41}	x_{42}	x_{43}	x_{44}	x_{45}	x_{46}	x_{47}
x_{48}	x_{49}	x_{50}	x_{51}	x_{52}	x_{53}	x_{54}	x_{55}
x_{56}	x_{57}	x_{58}	x_{59}	x_{60}	x_{61}	x_{62}	x_{63}

Figure 3.1: The subimage matrix

distributed with zero mean and variance σ^2. The images considered in this study are composed of a single object of intensity *Ro* embedded in a background of intensity *Rb*. The segmentation algorithm is performed on the 8 X 8 subimages, and the resulting subimages are then combined to obtain the entire segmented image. The 2-D 8X8 subimages are represented as a vector in 1-D as $\bar{x} = [x_0, x_1, ..., x_{63}]$, $0 \le x_i \le 255$, where x_i represents the intensity of the pixel in the *ith* position. This is illustrated in Figures 3.1 and 3.2.

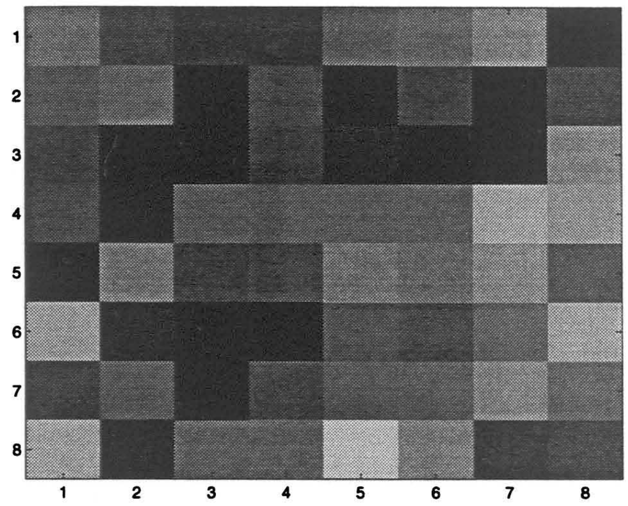


Figure 3.2: 2-D 8 X 8 raw subimage

Approach

The raw image that has to be segmented can be of any arbitrary size. It is divided into subimages and the algorithm for image segmentation is applied to each subimage. The algorithm comprises of generating initial population, evaluating every individual in the population on the basis of a fitness fuction, selecting eligible candidates for reproduction, generating offsprings by appling reproductive operators, and finally moving to the next generation by allowing only the fittest individuals to survive. This procedure is iterated until the stopping criterion is met. The fittest individual is the segmented image. The resulting segmented subimages are then combined to obtain the entire segmented image. There is no overlap in the subimages. More specifically, a pixel in a given image occurs in one and only one subimage. The steps in the algorithm mentioned above are explained in detail next.

Initial Population

The first step in the GA is to create an initial population. A set of individuals have to be created. Each candidate or individual is a string of *Ro* (object-intensity) and *Rb* (background-intensity). This represents a binary subimage. The initial population can be represented by the set,

$$\{Y_i^k\}, \quad i = 0, 1, ..., 63; \quad k = 1, 2, ..., N$$

where N is the population size. Two methods of generating initial population are discussed next.

Method 1

In this method every candidate in the population is generated randomly. Each individual in the population is a vector of length 64. Each element in the vector is randomly chosen to be *Ro* or *Rb*.

$$Y_i^k = Ro \ or \ Rb \ (randomly) \quad i = 0, 1, ..., 63; \quad k = 1, 2, ..., N$$

In effect, each individual represents a subimage and N such subimages are generated which represent the initial population. Figure 3.3 shows the randomly generated subimage which is a candidate in the initial population

Method 2

This method uses some heuristic to obtain the initial population. Since the noise is assumed to be gaussian with a spread of 3σ from the mean, every individual in the initial population is generated as described below. For every element x_i in the original image, the value in the *ith* position of the individual k is

$$Y_i^k = \begin{cases} Ro & \text{if } x_i > Rb + 3\sigma; \\ Rb & \text{if } x_i < Ro - 3\sigma; \\ Ro \text{ or } Rb \ (randomly) & \text{otherwise} \end{cases}$$

Figure 3.4 explains the method of generation of initial population based on Method 2. Figures 3.5 and 3.6 show a raw subimage and a candidate for an initial population generated by Method 2.

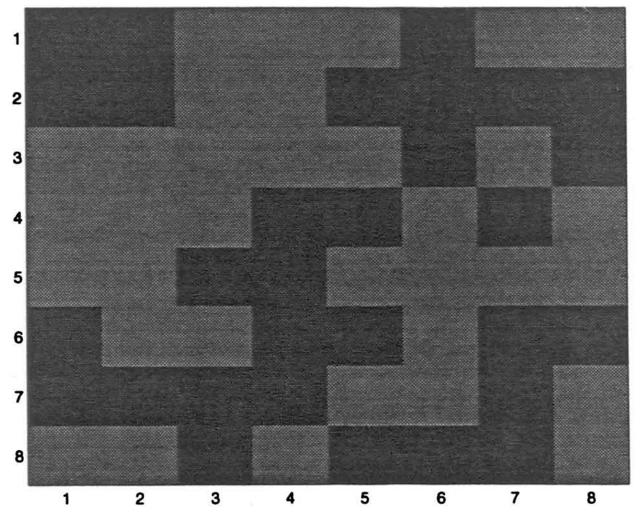


Figure 3.3: A candidate of initial population generated by Method 1

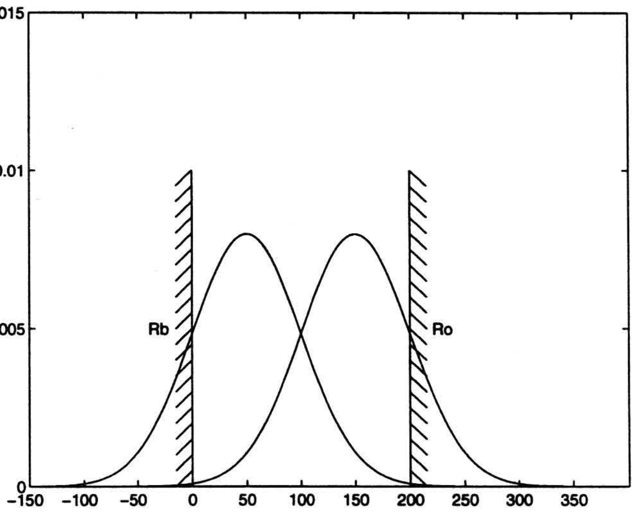


Figure 3.4: Heuristic used in Method 2

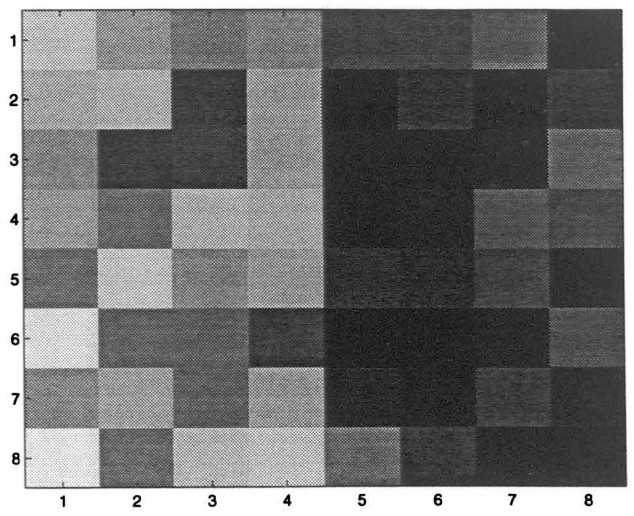


Figure 3.5: The raw subimage

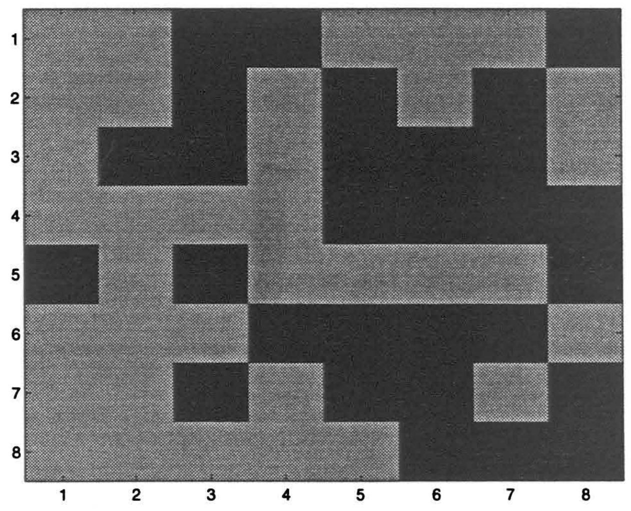


Figure 3.6: A candidate of initial population generated by Method 2

Method 1 vs Method 2

Method 1 does not depend on the knowledge of noise variance. In most of the Image Segmentation algorithms, noise variance is one of the parameters to be provided by the user. This method of generation of initial population does not require knowledge of this parameter. Since the method is totally random, the initial population size is kept high so that it has a large enough solution space. In method 2, N comparisons with the raw subimage are required to generate N candidates for the initial population. This method requires one to know the variance. However, because some initial heuristic is used, the number of individuals in the population, i.e. the population size can be decreased. This effectively decreases the run-time of the algorithm. Thus there is a tradeoff between reduced user input parameters and the run-time of the segmentation algorithm.

Fitness Evaluation

A fitness function is defined in order to evaluate the individuals of the population. The fitness function plays an important role in the genetic algorithm and helps decide whether:

1. The individuals that are capable of producing offsprings.
2. The offsprings that are fit enough to survive.
3. The individuals in the current generation that are capable of existing in the next generation.

If the original image is $\bar{x}$ and the initial population is

$$\{Y_i^k\}; \qquad 0 \le i \le 63; \qquad 0 \le k \le N$$

The fitness of an individual $f(Y^k)$ of the population is calculated as:

$$f(Y^k) = E(Y^k) + \alpha * T(Y^k) \qquad k = 1, 2, ..., N \tag{3.2}$$

$$E(Y^k) = \frac{1}{\sum_{i=0}^{63} |Y_i^k - x_i|} \tag{3.3}$$

$$T(Y^k) = \frac{1}{\sum_{i=0}^{63} |Y_i^k - w_i|} \qquad w_i \in \mathit{neighboring\ pixel\ of}\ \ Y_i^k \tag{3.4}$$

$E(Y^k)$ is the measure of similarity between the individual Y^k and the original noisy subimage. The idea here is to minimize the difference between the raw subimage of varying gray intensity and the binary subimage. The candidate subimage which has the least hamming distance to the raw subimage has the highest $E(Y^k)$. For instance, for the raw image in Figure 3.7, two candidates along with the similarity measure value $E(Y^k)$ are shown in Figures 3.8 and 3.9. The second term in the fitness, $T(Y^k)$ is the reciprocal of the transition count in the horizontal and vertical direction. This term is introduced in order that the images with homogeneous regions have a better fitness as compared to those images with discontinuities. The fitter individuals will have a high $T(Y^k)$ since the sum of the transition count will be low. The procedure for calculating the transition count is described below. For every pixel, the 4-neighbors are observed.

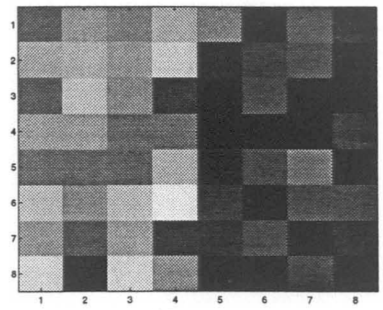


Figure 3.7: 8 X 8 raw image: SNR = 2

If the pixel under consideration has a value *Rb* and three out of the four neighbors have values *Ro* then the transition count of that pixel is $3 * |Rb - Ro|$. The corner pixels have only two 4-neighbors while the pixels on the edge of the image have only three 4-neighbors. $T(Y^k)$ is the reciprocal of the sum of the transition counts of all the pixels. As can be seen from the figures below, the candidate in Figure 3.11 is homogeneous and has a higher $T(Y^k)$ than the candidate shown in Figure 3.10 which shows many discontinuities. In these images Ro = 150, and Rb = 50. α is the weighting factor that normalizes the two quantities $E(Y^k)$ and $T(Y^k)$.

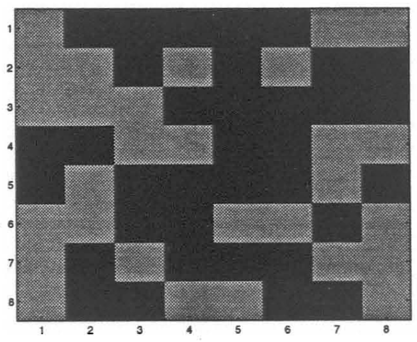


Figure 3.8: $E(Y^k)$ = 2.3595e-04

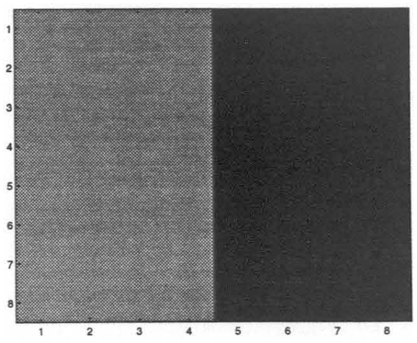


Figure 3.9: $E(Y^k)$ = 3.8569e-04

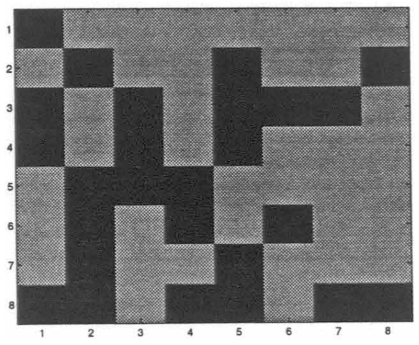


Figure 3.10: $T(Y^k)$ = 9.1743e-05

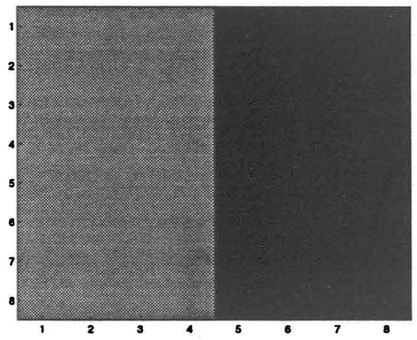


Figure 3.11: $T(Y^k)$= 6.2500e-04

Selection of Fittest

Once the fitness of the individuals is evaluated, the fitter individuals must be selected so that they can be mated to produce offsprings in the next generation. A very simple way to do the selection is to define a threshold θ,

$$\theta = \frac{Max(f) + Min(f)}{2} \tag{3.5}$$

$Max(f)$ and $Min(f)$ are the maximum and minimum values of the fitness function respectively. The individuals with fitness greater than the threshold are selected to produce offsprings. In order that the search space does not get limited, we define a minimum number of offsprings, ζ that must be generated. If the number of offsprings generated is less than ζ the threshold θ is decreased so that more candidates are selected for the reproduction and the number of offsprings generated by the selected candidates is more than the minimum allowed.

Reproduction

The reproduction phase is critical because it is responsible for the evolution. The individuals capable of reproduction are brought together in the mating pool. The basic operators of reproduction, namely crossover and mutation are explained in chapter 2. However, the way in which they are applied may depend on the problem at hand. The implementation of these operators are defined in the application of image segmentation is explained next.

1. **Crossover:** Let us assume that the mating pool consists of K individuals. Since all are capable of reproduction, we assume that each of the K individuals mates with each of the remaining $(K-1)$ individual and produces $(K-1)$ offsprings. So in effect, the K individuals will produce a total of $K(K-1)$ offsprings. Let S denote the set of K individuals selected from the current generation.

$$S = \{Y_0^1, Y_1^1, ..., Y_{63}^1$$

$$Y_0^2, Y_1^2, ..., Y_{63}^2$$

$$Y_0^K, Y_1^K, ..., Y_{63}^K\}$$

Every individual in the set S generates $(K-1)$ new offsprings as follows. For every individual Y^i, $i = 1, 2, ..., K$ a cross-over point p $(0 < p < 63)$ is randomly chosen. If Z^i is any individual in { S }, and the crossover point is p then the first $p+1$ elements of Z^i, $\{ Z^0, Z^1, ..., Z^p\}$ are concatenated with the remaining $64-(p+1)$ elements from position $p+1$ to 63, of all the individuals in the set {

S - Z^i } The $K-1$ new offsprings created from Z^i in { S } can be represented as,

$$\{Z_0^i, ..., Z_p^i, W_{p+1}^j, ..., W_{63}^j\}$$

$$j = 1, 2, ..., K-1. \qquad W \in \{S - Z^i\}$$

The total number of individuals after the crossover is $N + K(K-1)$. N individuals belong to the current generation and $K(K-1)$ offsprings are produced by the crossover operator. A note to be made at this point is that not all of the offsprings survive.

2. **Mutation:** The mutation operator is applied to the individuals selected for the next generation. Mutation can be explained as follows. A living being undergoes some changes in its characteristics due to the influence of the surroundings. During its lifespan, every creature develops and grows by adapting itself to the existing environment so that it is more suited for survival. In a very similar way, the mutation operator applied to the individuals in our study, does not really produce new offsprings, but allows the existing individuals to develop traits that help them survive. The mutation operator is applied to an individual as follows:

   Each gene (pixel value) is modified depending on the value possessed by the majority of its neighbor. A neighborhood C is defined (similar to the 8-pixel neighborhood or 4-pixel neighborhood). For each bit x_i in the individual Y_i^k the neighborhood C is considered. If N_b is the number of background pixels and N_o is the number of object pixels in C then

$$Y_i^k = \begin{cases} Rb & \text{if } N_b > N_o \\ Ro & \text{if } N_o > N_b \end{cases}$$

$$i = 0, 1, ..., 63. \qquad k = 1, 2, ..., N.$$

The Figure 3.12 shows the typical 4-pixel and 8-pixel neighborhoods. In both cases the center pixel is the pixel under consideration and since the N_b is more than N_o in both cases, the pixel under consideration is changed from Ro to Rb

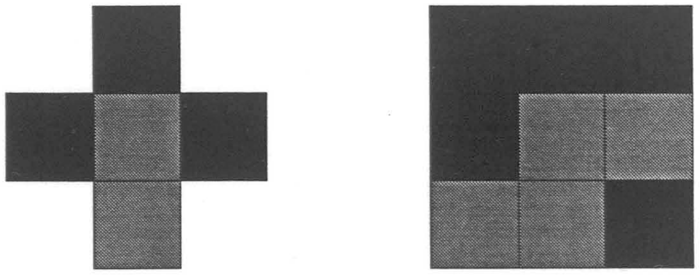


Figure 3.12: Different neighborhoods

Next Generation

Once the reproductive operators have been applied, there are $N + K(K-1)$ individuals. The fitness of each is evaluated using the fitness function described in the equation 3.2. The N best individuals, are selected to form the next generation.

In order that the genetic algorithm does not get stuck in the local extrema, we do not take two individuals with the same fitness value. In case there are more than one individuals with same fitness value, one of them is selected randomly.

Stopping Criterion

The algorithm proceeds from one generation to another and evolution takes place. One would expect this to go on indefinitely. However, the evolution is directed towards improving the individual characteristic. After a number of generations, all the individuals of the population will have the same fitness, indicating that convergence is reached. Further processing will not cause improvement and typically, the algorithm is stopped at this point. In this study, the algorithm is made to stop after a fixed number of generations. This is done to ensure that the algorithm does not take an indefinitely long time to come to an end.

Parameters that Affect the Performance of the GA

In order that the genetic algorithm works effectively, the various components of the algorithm have to be chosen with care. A brief discussion of the parameters that influence the performance of the GA has been presented next.

1. **Population size**

   The number of individuals at a given time in the population determines its size. If the population size is large, the algorithm will take fewer generations to reach a good "solution". The run-time of the algorithm for a generation is directly proportional to the population size which in other words means that if the population size is small then the time required for running one generation is less and vice versa. However, the overall runtime is not affected since a smaller population size means large number of generations. The reverse also holds good.

2. **Generation of initial population**

   We discussed two methods of generation of initial population. There are various other ways in which one can generate an initial population. In most of the literature on genetic algorithm, random method of generation of initial population is preferred.

3. **Subimage size**

   The size of the subimage is another parameter that can be varied. The subimage size also plays an important role in the determining the runtime of the algorithm. If the subimage size is increased then the length of the chromosome will automatically increase. Intuitively, we can expect that the time for processing will increase. But then again the entire image will be divided into fewer subimages. If the subimage size is $(m\ X\ n)$ then the string length of every candidate will be $(m\ X\ n)$ and the solution space will increase to $2^{(mXn)}$. The results of the experiments with different subimage size are presented in the next chapter.

4. **Fitness Function**

   The fitness function chosen decides how well the algorithm performs. The genetic algorithm provides a lot of flexibility in the sense that there can be numerous fitness functions for a given problem. The choice of a wrong fitness function can lead to total failure. It may so happen that the average fitness function increases from one generation to another, but the solution obtained may be going further away from the desired solution. Care should be taken while designing a fitness function.

5. **Selection Criteria**

   A selection criterion is required to decide which of the individuals can be selected for application of reproduction operators. One could decide on a fixed number of individuals to be selected every time. Here a fixed number of individuals with best fitness can be selected. This makes the algorithm very rigid. In order to bring more flexibility, we can choose the best fit individuals with a certain probability. The selection criteria described in the current algorithm is based on a threshold as explained before. This method allows a lot of flexibility, that is, every time a different number of individuals are selected for the mating pool.

6. **Crossover techniques**

   There can be various methods of applying crossover. The method of crossover described in this algorithm forces each individual in the mating pool to produce an offspring with every other individual. This could be modified in a way that each individual mates with the remaining individuals with a certain probability. Another variation could be that two individuals are selected randomly from the mating pool and made to produce two children. The crossover discussed so far is concerned with selecting only one crossover point (one-point crossover) randomly. A modification to it could be selecting two (two-point crossover) or more (multi-point crossover) crossovers. However this would unnecessarily complicate the algorithm and the inherent simplicity of the current algorithm will be lost.

7. **Mutation techniques (different neighborhood)**

   The mutation technique described in the section on mutation is based on flipping the selected bits based on some heuristic (looking at the neighborhood). We could simply define a mutation rate (number of bits to be mutated) and flip bits randomly. Application of any kind of information that is available (knowledge of neighboring pixels in this case) to enhance the genetic algorithm is referred to as hybridizing the algorithm. Different neighborhoods can be used and bits can be flipped with some induced uncertainity to modify the current mutation operator.

8. **Variable population**

   In the algorithm described above, the number of individuals in the population is kept fixed for every generation. This parameter can be made variable by allowing individuals with fitness above a certain threshold to move on to the next generation instead of allowing a fixed number of best fit individuals to form the next generation.

9. **Stopping criteria**

   Apart from the stopping criterion used in the algorithm in our study, several choices of stopping criteria could be used for terminating the algorithm. The algorithm can be allowed to run until convergence is achieved. The algorithm converges when the average fitness remains constant from one generation to another. Alternatively the algorithm could be stopped if the threshold θ remains constant from one generation to another.

An investigation of the effects of the above mentioned parameters on the performance on GA are presented next using some simple simulated images. The algorithm is then implemented on an X-ray image for flaw detection.

CHAPTER 4. RESULTS AND DISCUSSION

This chapter is organized in the manner the algorithm evolved over the duration of the study. Little was known when this work was started about application of genetic algorithm to the problem of object identification in noisy images. In order to explore the field, some initial confidence was needed to prove the feasibility of the approach and take it further. To start with, some very simple images (horizontal and vertical boundaries) were considered. The results on the data looked promising and the algorithm was then applied to more complex images with smooth curves and other jagged edges the results of which are also presented. The different parameters were varied and their role in the performance of the algorithm was studied. All these results are presented and discussed in the following sections.

Effect of Subimage Size

The study was first conducted on elementary images of size 8 X 8 with Ro = 150 and Rb = 50. The noisy input image was synthesized by superimposing a random gaussian noise on the clean binary image. The variance of the noise controlled the signal to noise ratio (SNR) of the raw noisy image. The noise variance σ^2 was 2500 and the SNR was 2.

The initial population was generated randomly (Method 1) and the population size was selected as 30. The algorithm was made to stop after 100 generations. Figure 4.1 shows: (a) the original image (Horizontal Cut), (b) the noisy image, (c) one candidate of the initial population and finally (d) the resulting segmented image.

The experiments with a subimage of size 8 X 8 proved the feasibility of the approach. It seemed reasonable to extend this technique to a larger subimage size. The next subimage size that was tried was of size 16 X 16. In the results shown in Figure 4.2 the image under considerations the object intensity Ro = 150 and background intensity Rb = 50. The noise superimposed on the original images had a variance of $\sigma^2 = 4000$ resulting in noisy images of $SNR = 1.58$ The plot of average"weakness" with respect to number of generations is shown in Figure 4.3. The population size is chosen to be 35 while it is made to stop after 100 generations. Figure 4.4 shows the plot of the weakness of the fittest individual in the population for the 100 generations. From the plot shown in Figure 4.4 it is clear that the fitness of the best individual does not increase after 57 generation. The average fitness increases, but the same best fit individual continues to live from one generation to another. The increase in the average fitness is due to the fact that the fitness of most of the individuals in the population increases. The optimum solution is not reached. This can be explained as follows. The subimage size is increased to 16 X 16 which causes the solution space to be increased to 2^{16}. This makes the search for global minimum more difficult. Instead of increasing the number of generations, one alternative would be to increase the size of the population. At the same time, the number of generations can be also reduced.

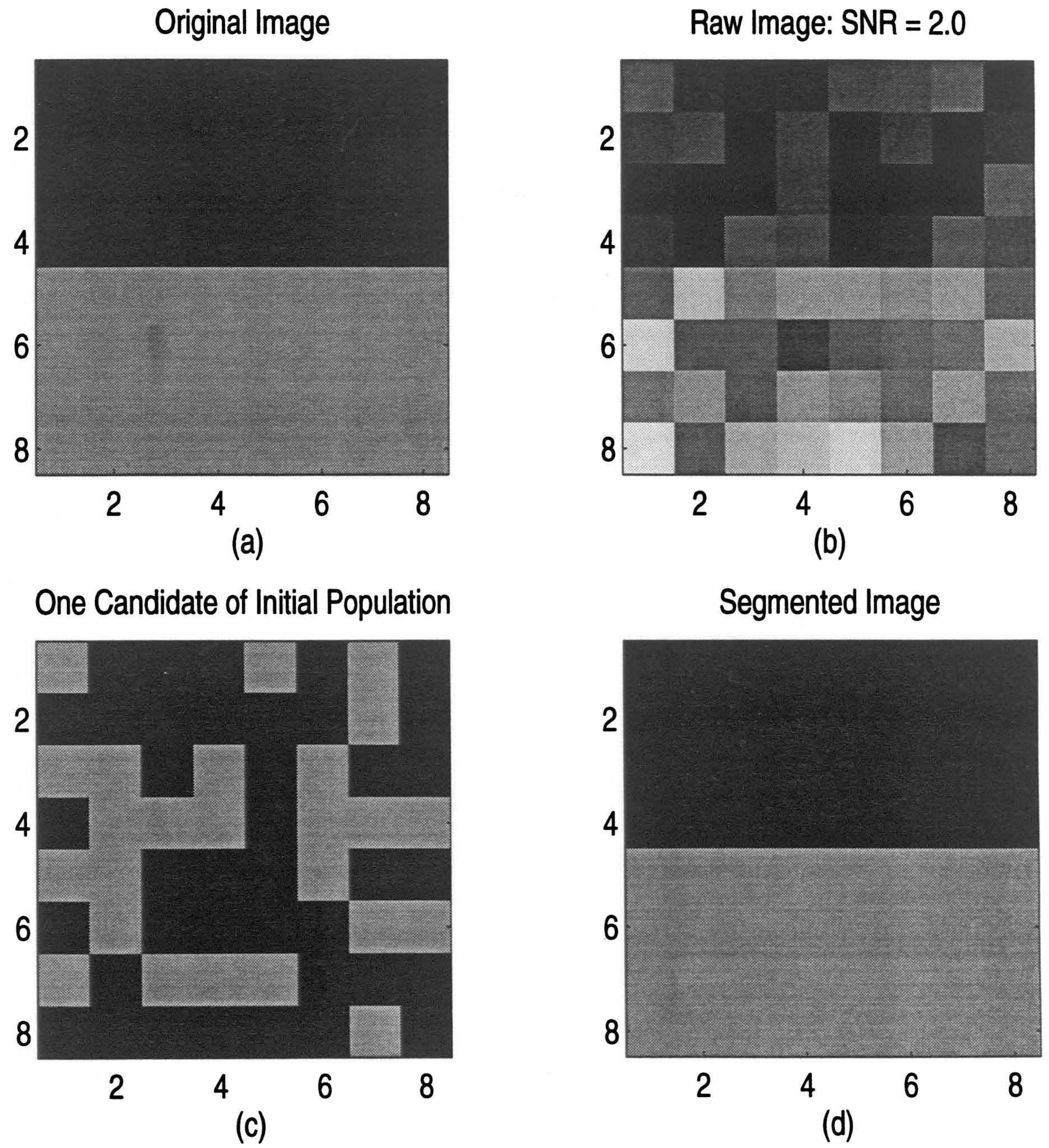


Figure 4.1: Results for 8 X 8 subimage

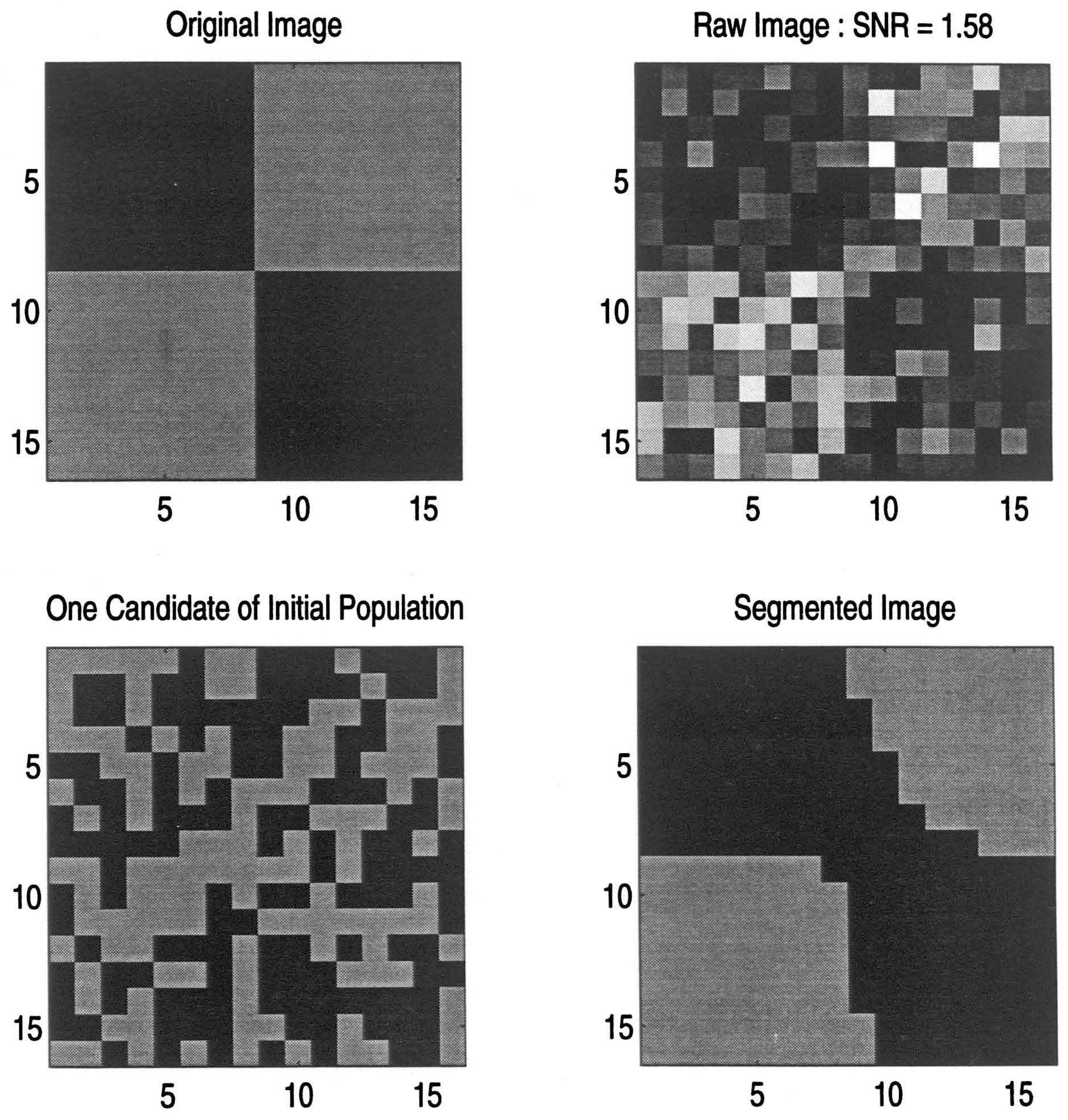


Figure 4.2: Results for 16 X 16 subimage

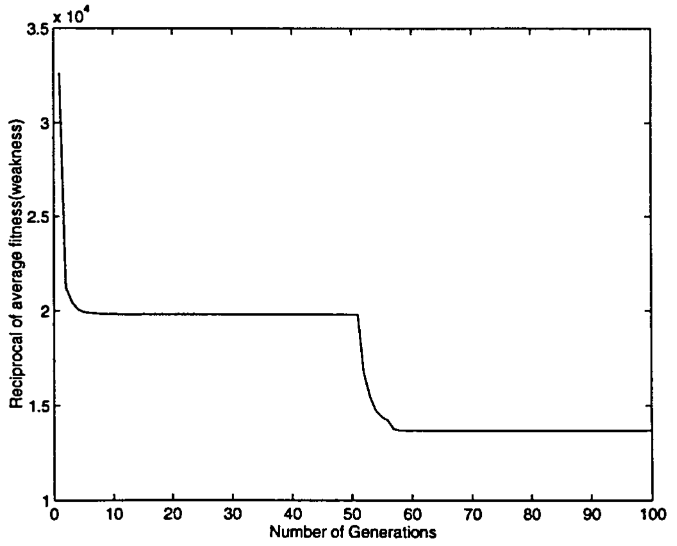


Figure 4.3: The plot of "average weakness" versus the number of generations (100)

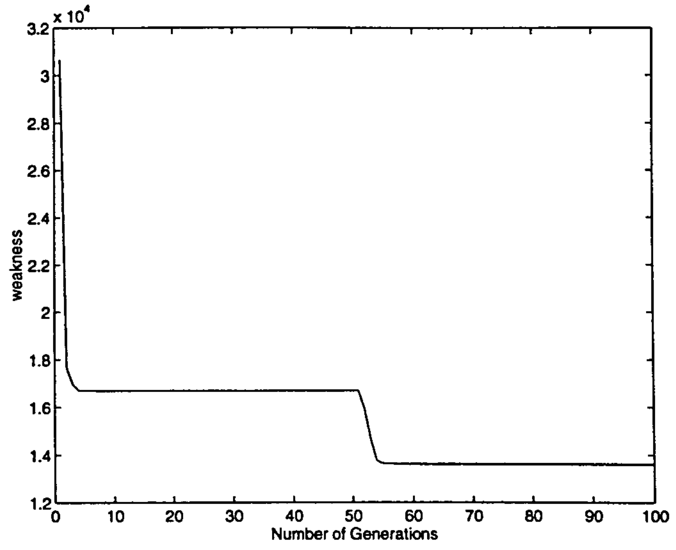


Figure 4.4: The plot of the reciprocal of the fitness of the fittest individual with respect to the number of generations

The algorithm was tried with a population size of 50 which was allowed to evolve for 50 generations. The results obtained are presented in Figure 4.5. The subimage size is further increased to 32 X 32, and the algorithm was implemented on the image, with Ro = 150, Rb = 50, and SNR = 1.58. The population size is 50 and the algorithm is made to stop after 30 generations. The results are not very encouraging. Moreover, the time complexity also increased as the individual string was 1024 long. The result is shown in Figure 4.6. The subimage size of 16 X 16 was found to be optimal.

Initial Population

In this section, a comparison between the two methods of generation of the initial population is discussed. The examples chosen were complex in that the images had multiple objects with sharply varying object boundaries. The signal to noise ratio of the raw images is 1.58. The population size is maintained at 50 and the algorithm is stopped after 10 generations.

In the results presented in Figures 4.7, 4.8 and 4.9 the initial population is generated randomly using Method 1. Figures 4.10 4.11 and 4.12 show the results of the algorithm when Method 2 is used to generate the initial population. In the second set of figures, the candidate of the initial population resembles the original image to some extent. There is an additional time required in generating the initial population using the Method 2. The set of initial population has to be generated for each of the subimage before processing it further.

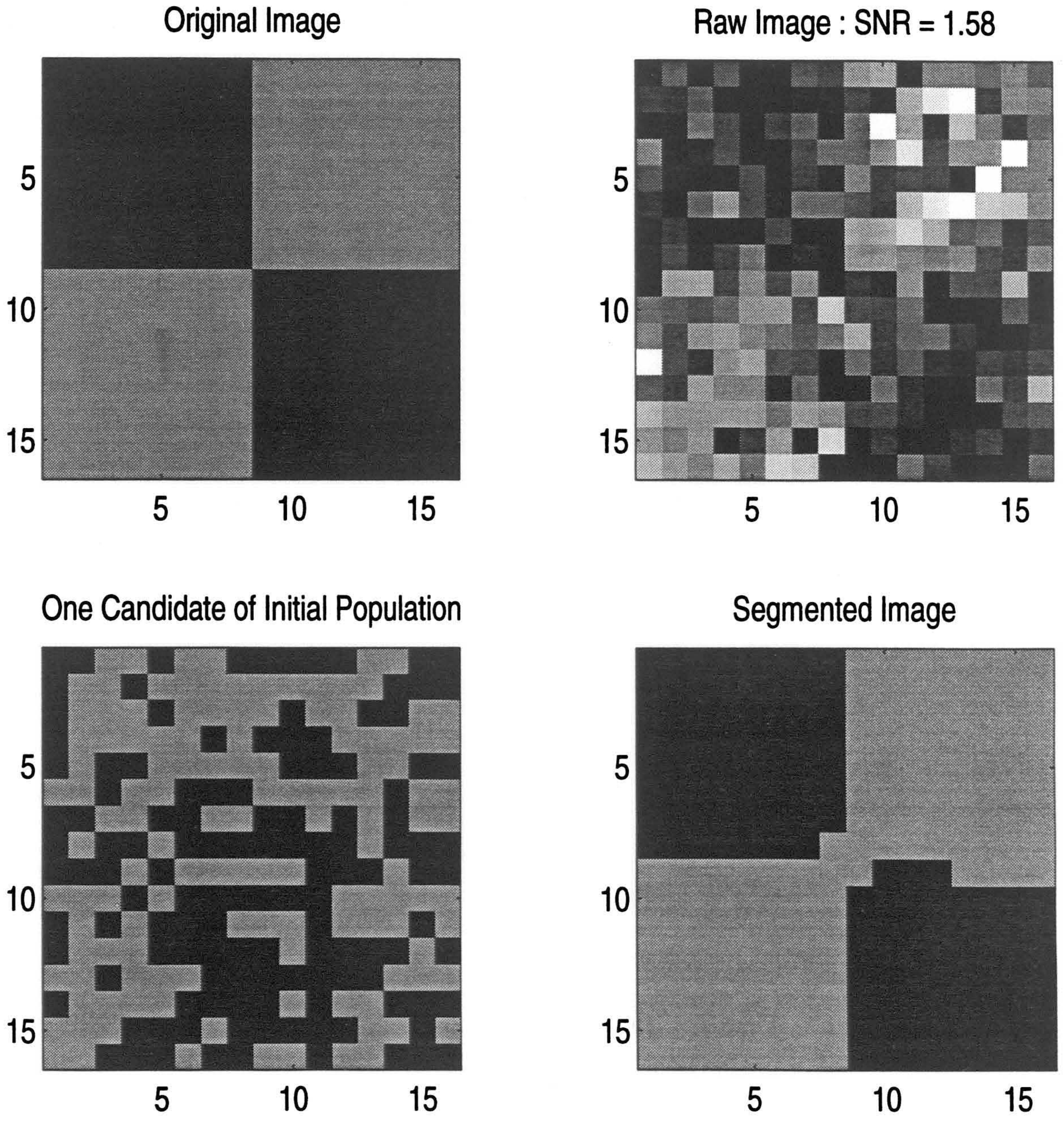


Figure 4.5: Results for 16 X 16 subimage with increased population size

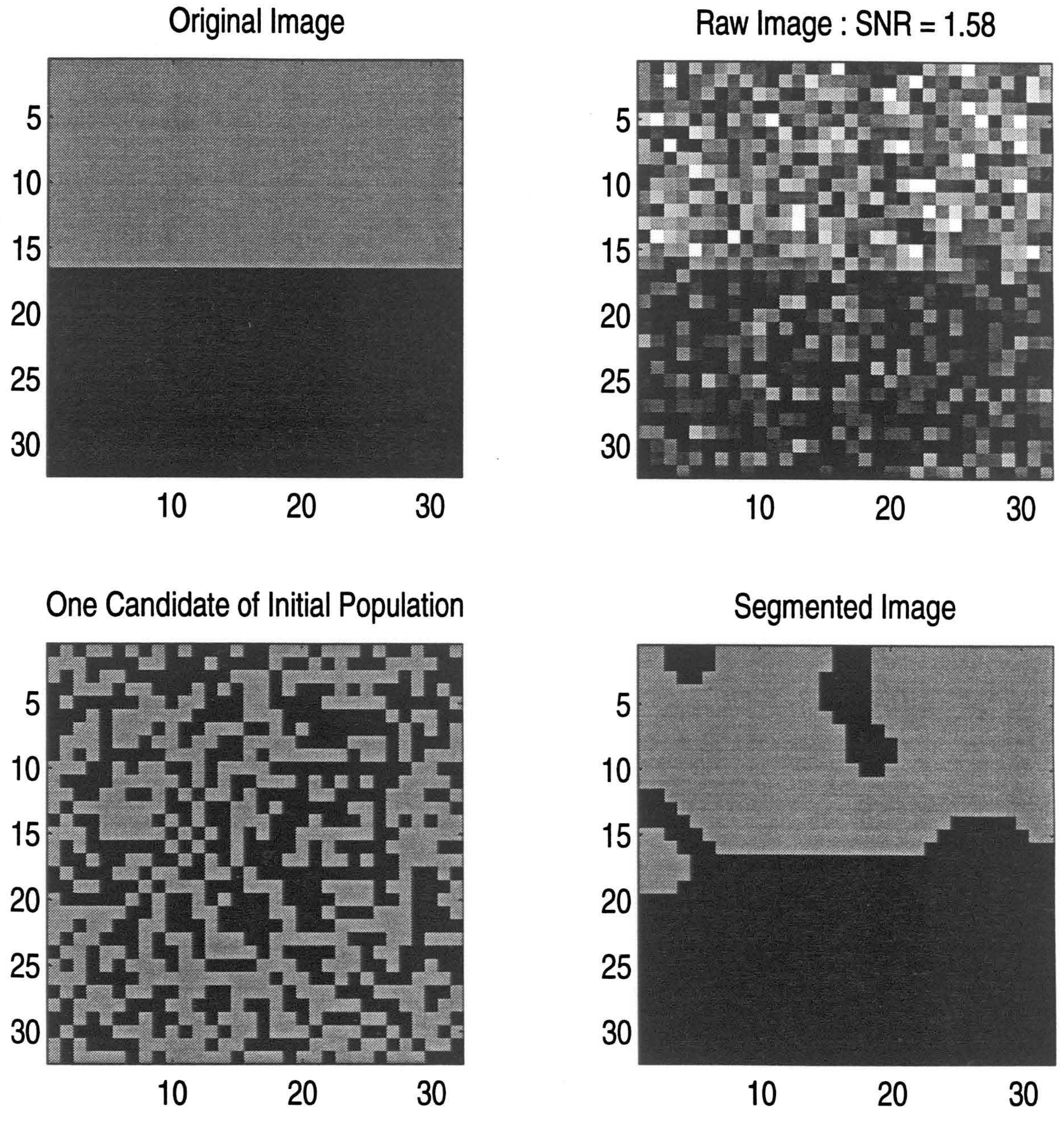


Figure 4.6: Results for 32 X 32 subimage

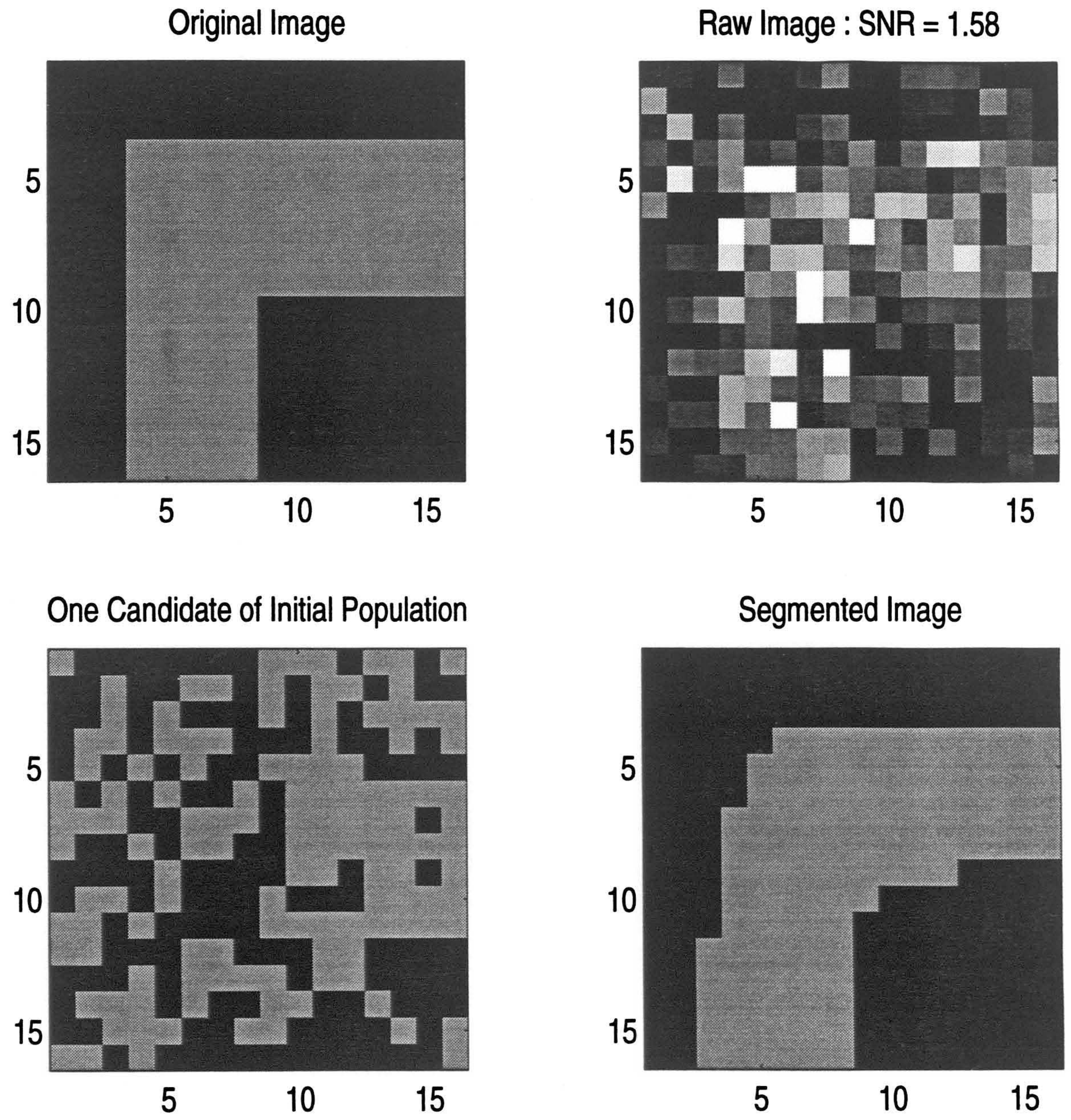


Figure 4.7: Results for 16 X 16 image with sharp edges

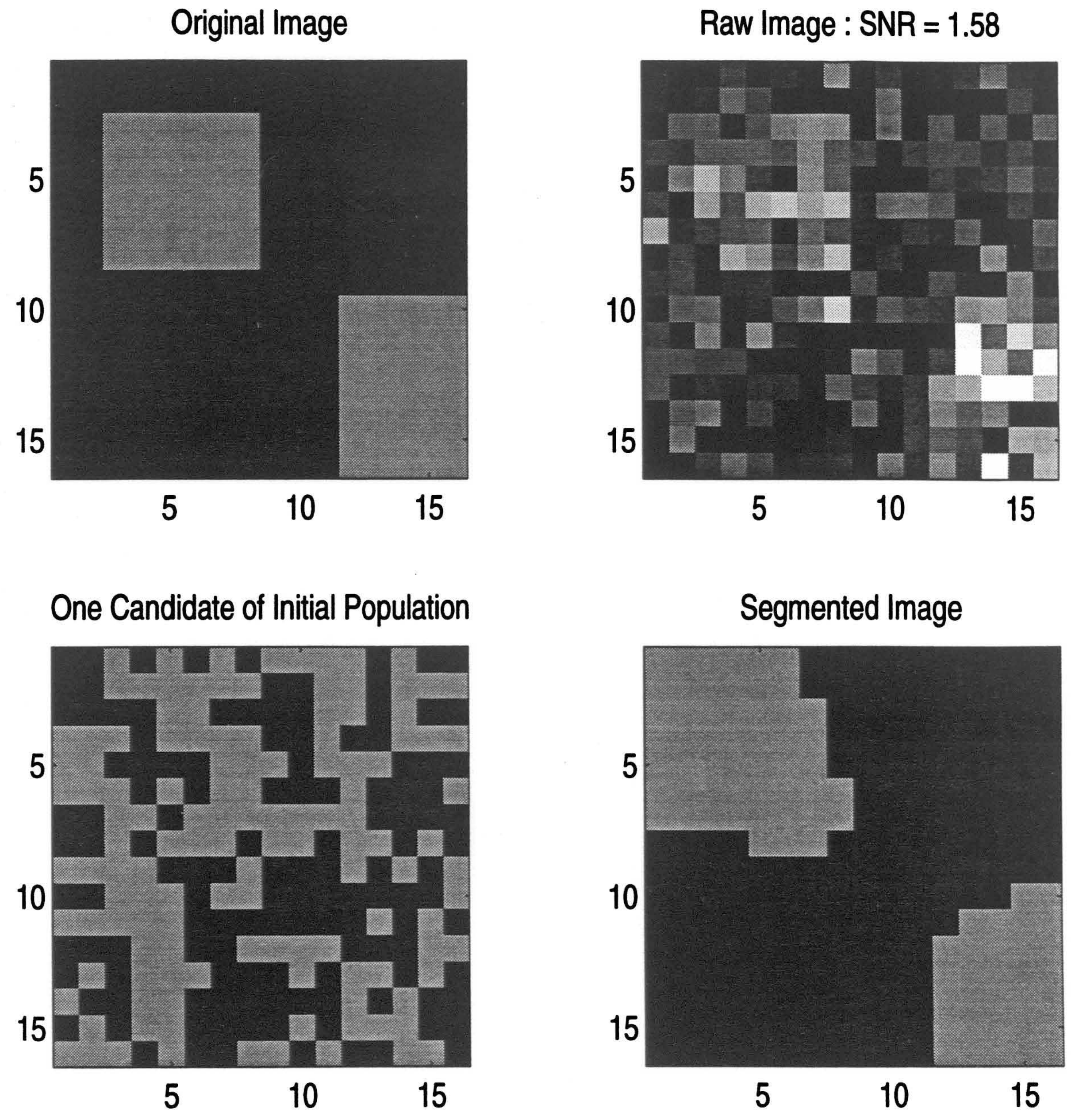


Figure 4.8: Results for 16 X 16 image with two objects

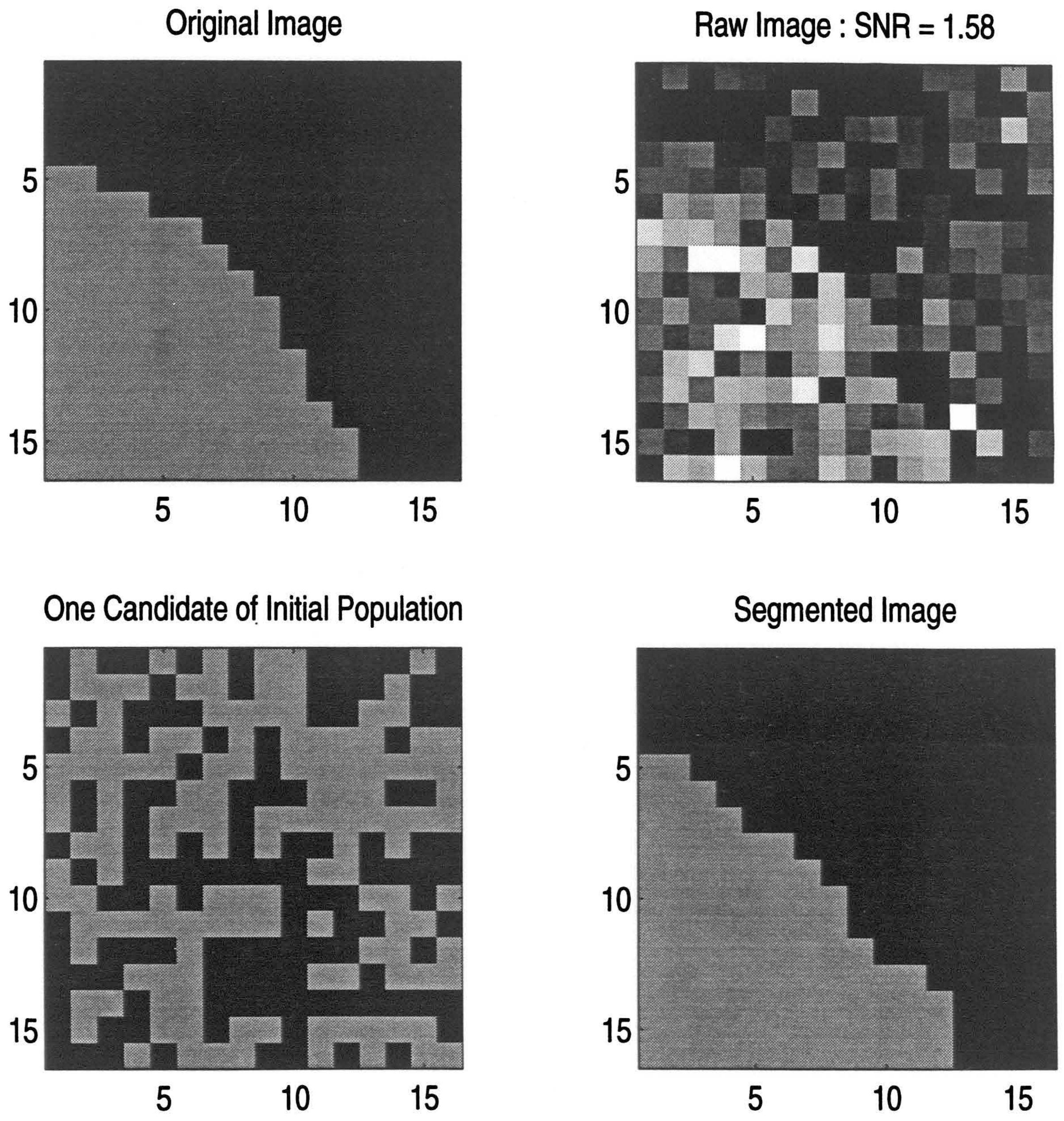


Figure 4.9: Results for 16 X 16 image with smooth curve

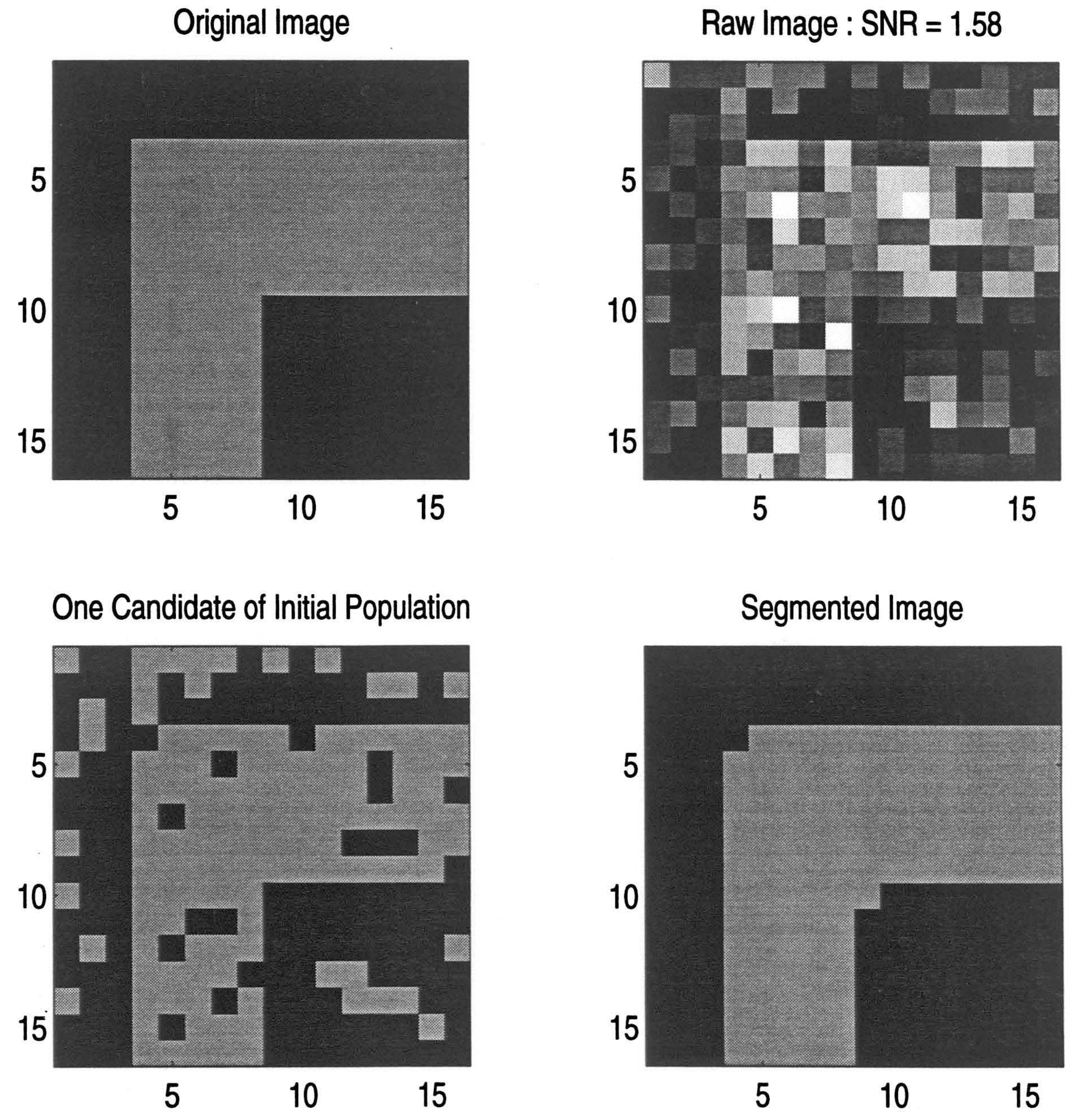


Figure 4.10: Results for 16 X 16 image with sharp edges using Method 2 for generating initial population

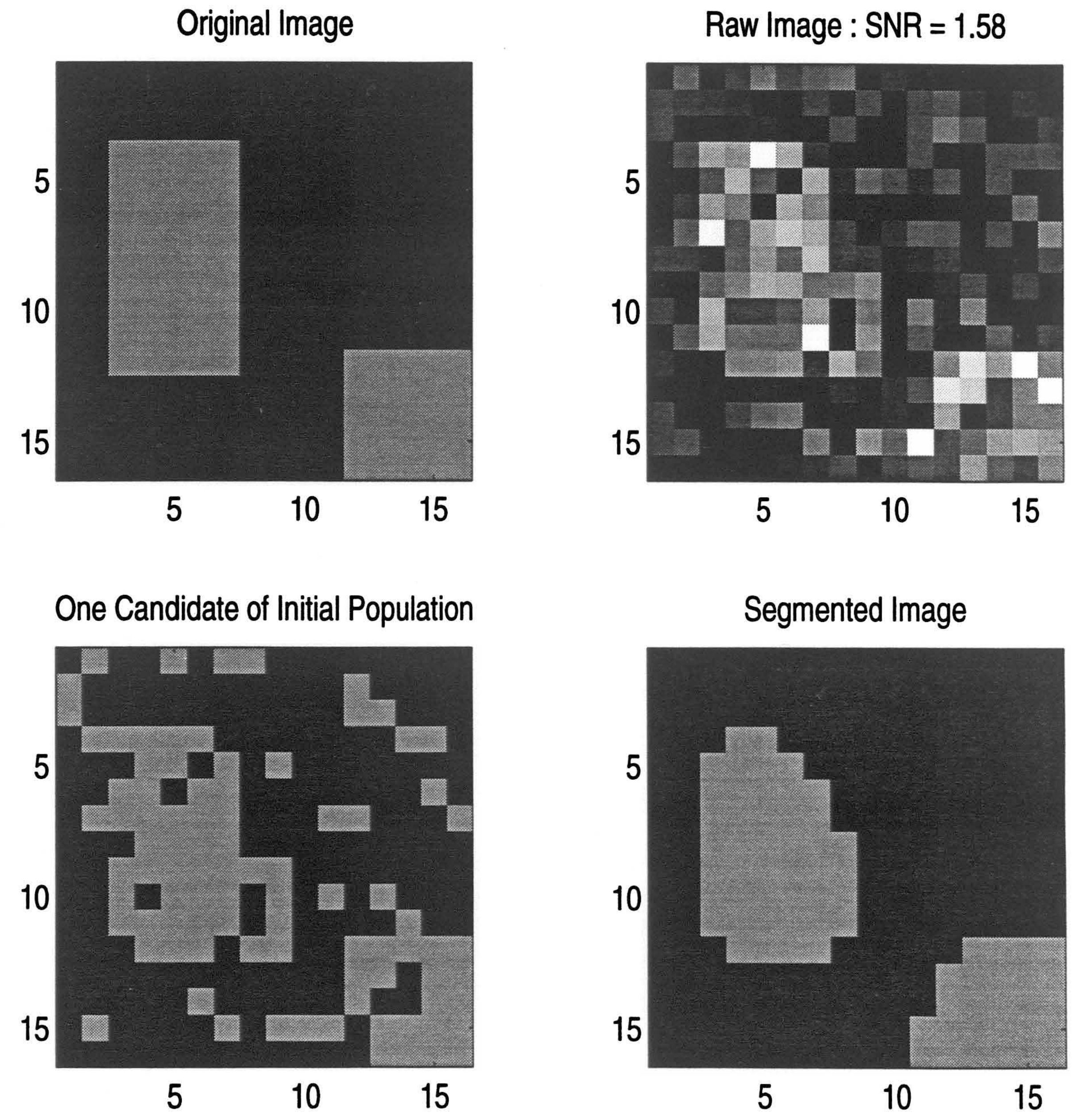


Figure 4.11: Results for 16 X 16 image with two objects using Method 2 for generating initial population

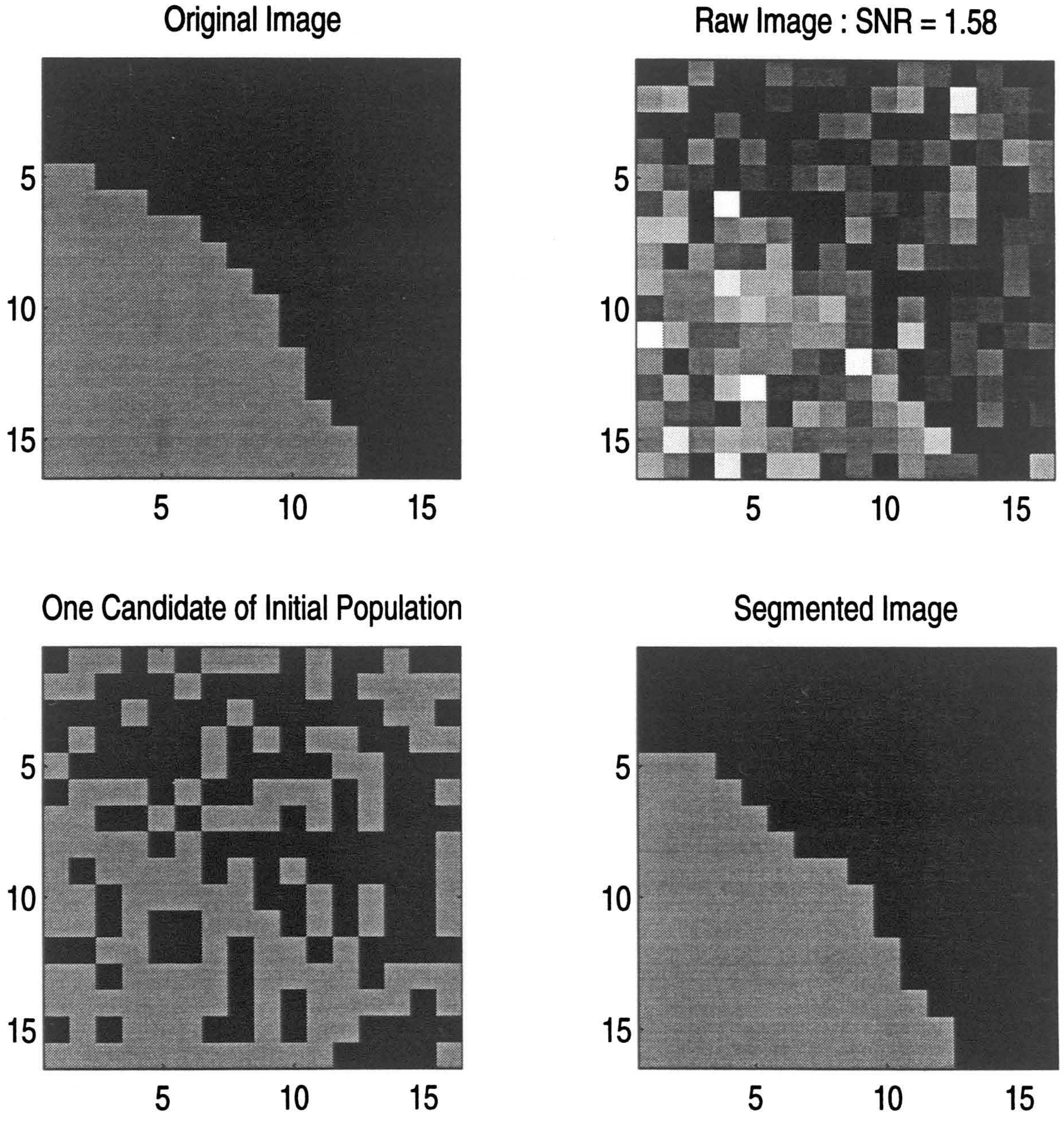


Figure 4.12: Results for 16 X 16 image with smooth curve using Method 2 for generating initial population

This additional time for generating initial population increases the time complexity if the image size is large and consequently larger number of subimages. On the contrary, if Method 1 is used to generate the initial population, the same set of initial population is be used for processing each of the subimage. Another alternative is to create a large number of individuals (using Method 1) and choose N of them as the initial population for every subimage. Since the results obtained by both the methods are comparable, and Method 1 is more efficient and simple, it is used for generating initial population hereafter.

Effect of Number of Generations

The study was first conducted on images of size 8 X 8 with Ro = 150 and Rb = 50. The noise that was superimposed had variance (σ^2) of 2500 which resulted in signal to noise ratio (SNR) of 2. The initial population was generated randomly (Method 1) and the population size was selected as 30. The algorithm was made to stop after 100 generations. Figure 4.13 shows: (a) the original image (Horizontal Cut), (b) the noisy image, (c) one candidate of the initial population and finally (d) the resulting segmented image. As can be seen, the algorithm was successful in finding the global extrema. Figure 4.14 shows the plot of the reciprocal of the average fitness or the "weakness" versus the number of generations. As can be seen from the plot, the average weakness continues to decrease with the number of generations and the reduction is very steep in the first five generations. After that the function remains steady until the fiftieth generation. The weakness function falls sharply and once again remains constant until the 100th generation. In other words the the maximum fitness was attained after 50th generations and the "solution" was reached.

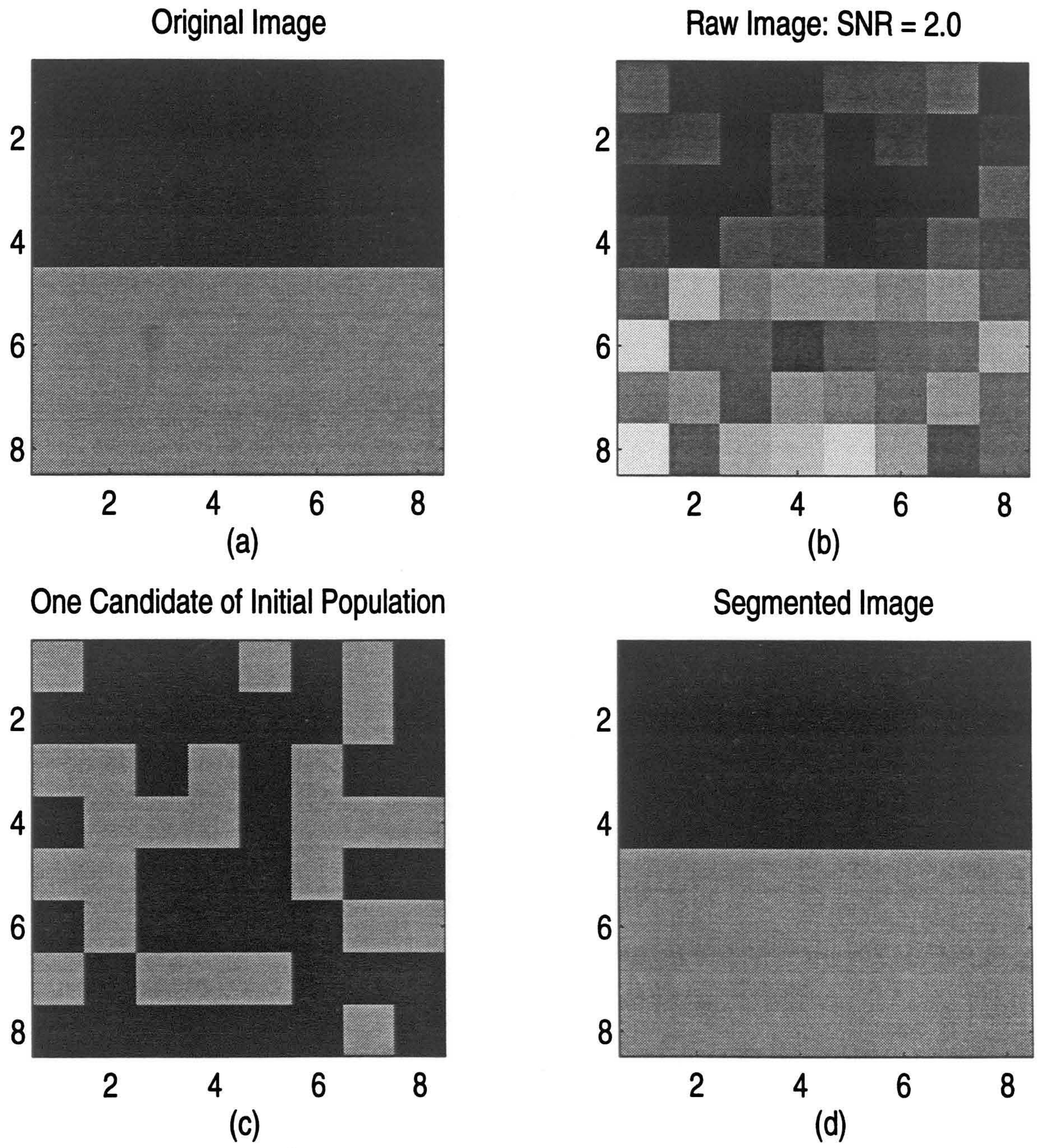


Figure 4.13: Initial results for 8 X 8 subimage with a horizontal boundary

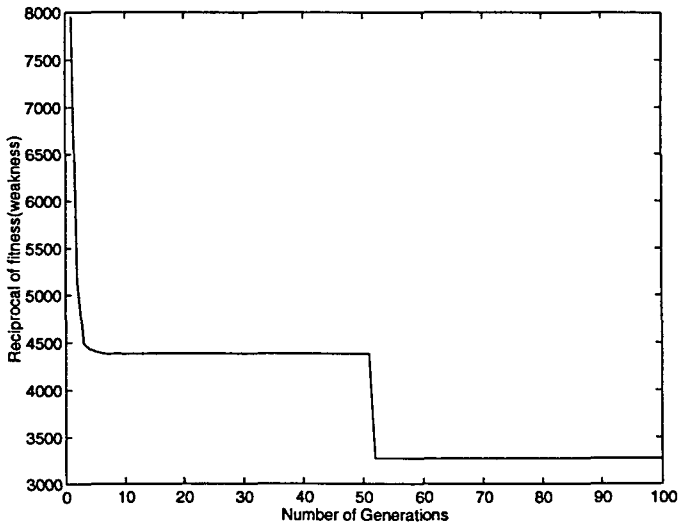


Figure 4.14: The plot of "weakness" versus the number of generations (100) for Figure 4.13

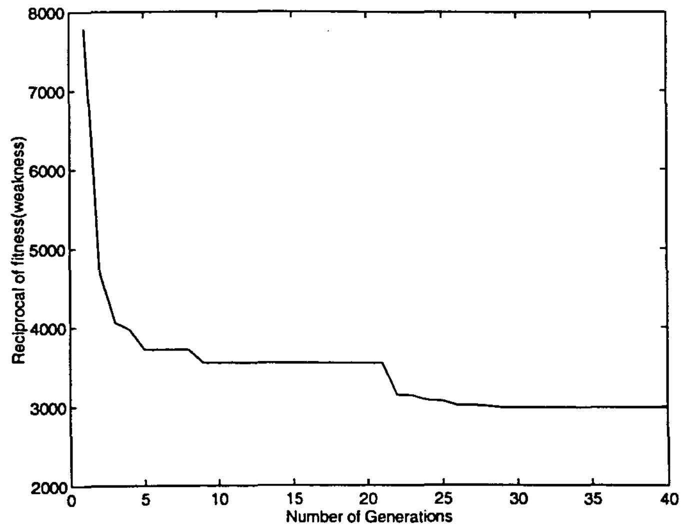


Figure 4.15: The plot of "weakness" versus the number of generations (40) for Figure 4.13

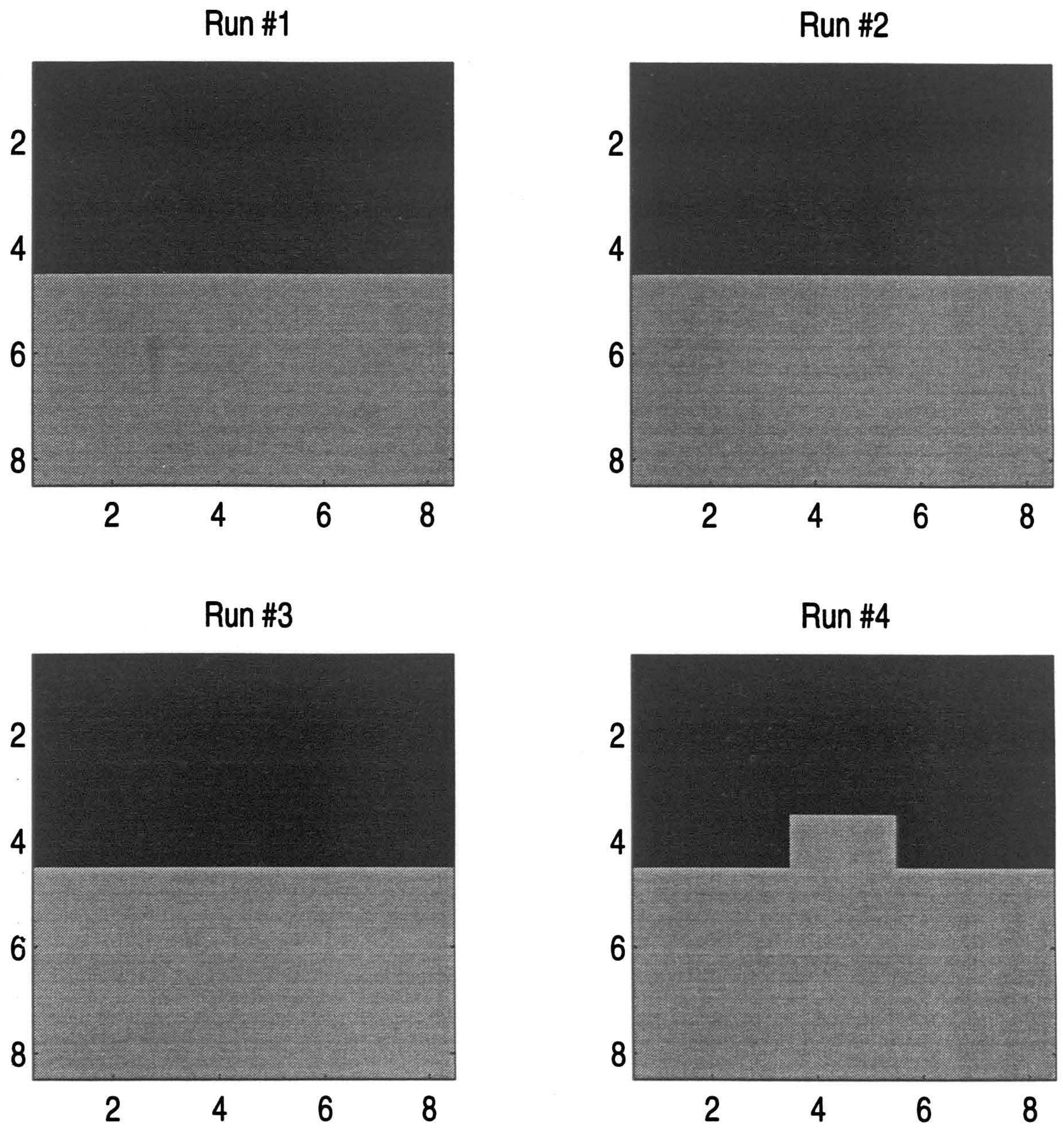


Figure 4.16: Results obtained in 4 different trials with the raw image of Figure 4.13

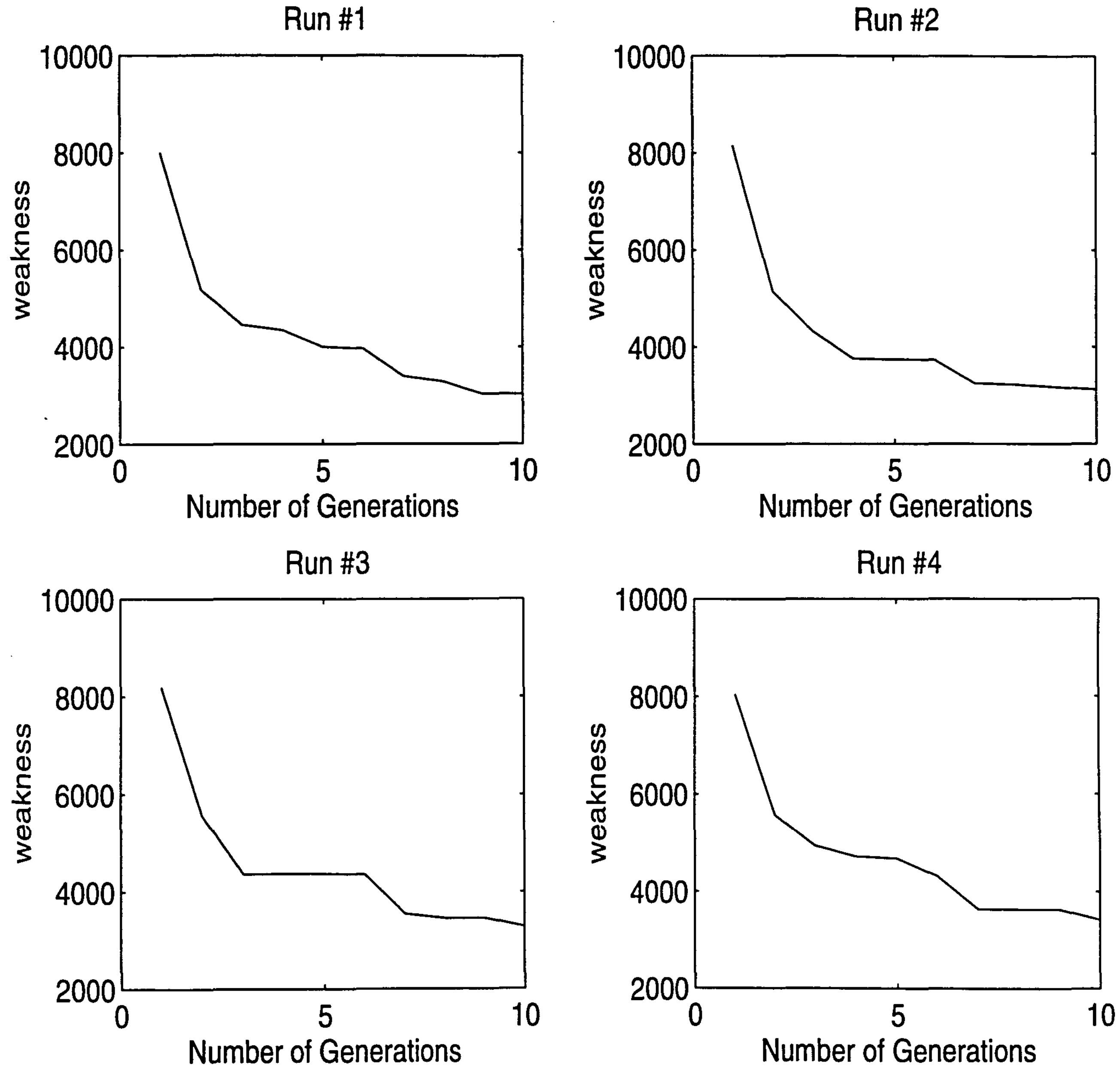


Figure 4.17: The plot of the "weakness" versus the number of generations for the four trials

The stopping criteria was therefore changed and the algorithm was stopped after 40 generations. Figure 4.15 shows the plot of the average "weakness" versus number of generations. The final segmented image was the same as in Figure 4.13. The plot shows decrease in "weakness" as expected. For the sake of curiosity, the number of generations was decreased to 10. Figure 4.16 shows the results obtained in this case in different trials of the algorithm with different random intial population. The noisy input image is the same as in Figure 4.13. However, in each of the runs, a different set of initial population was generated randomly. Figure 4.17 shows corresponding plots of "weakness" versus number of generations in the different runs. It can be observed from Figure 4.16 that the "solution" is reached in 10 generations in three of the four trials. But, the one spurious case suggests that the number of generations should be more than 10. However it should be noted at this stage that the the global "solution" can be reached in 10 generations too but with some probability. Also if the starting population is kept the same in the different runs, the same solution is obtained after 10 generations.

Performance on Low Contrast Images

The image under consideration has a vertical boundary. Here, the object intensity (Ro) was decreased from 150 to 125 and the background intensity (Rb) was increased from 50 to 75. A random gaussian noise of variance $\sigma^2 = 2500$ was added to the 8 X 8 image to get a noisy image with $SNR = 1$. The initial population was generated randomly and the size of the initial population was kept 30. The algorithm was stopped after 50 generations and the results are presented in the Figure 4.18. The plot of the average "weakness" versus number of generations is shown in Figure 4.19.

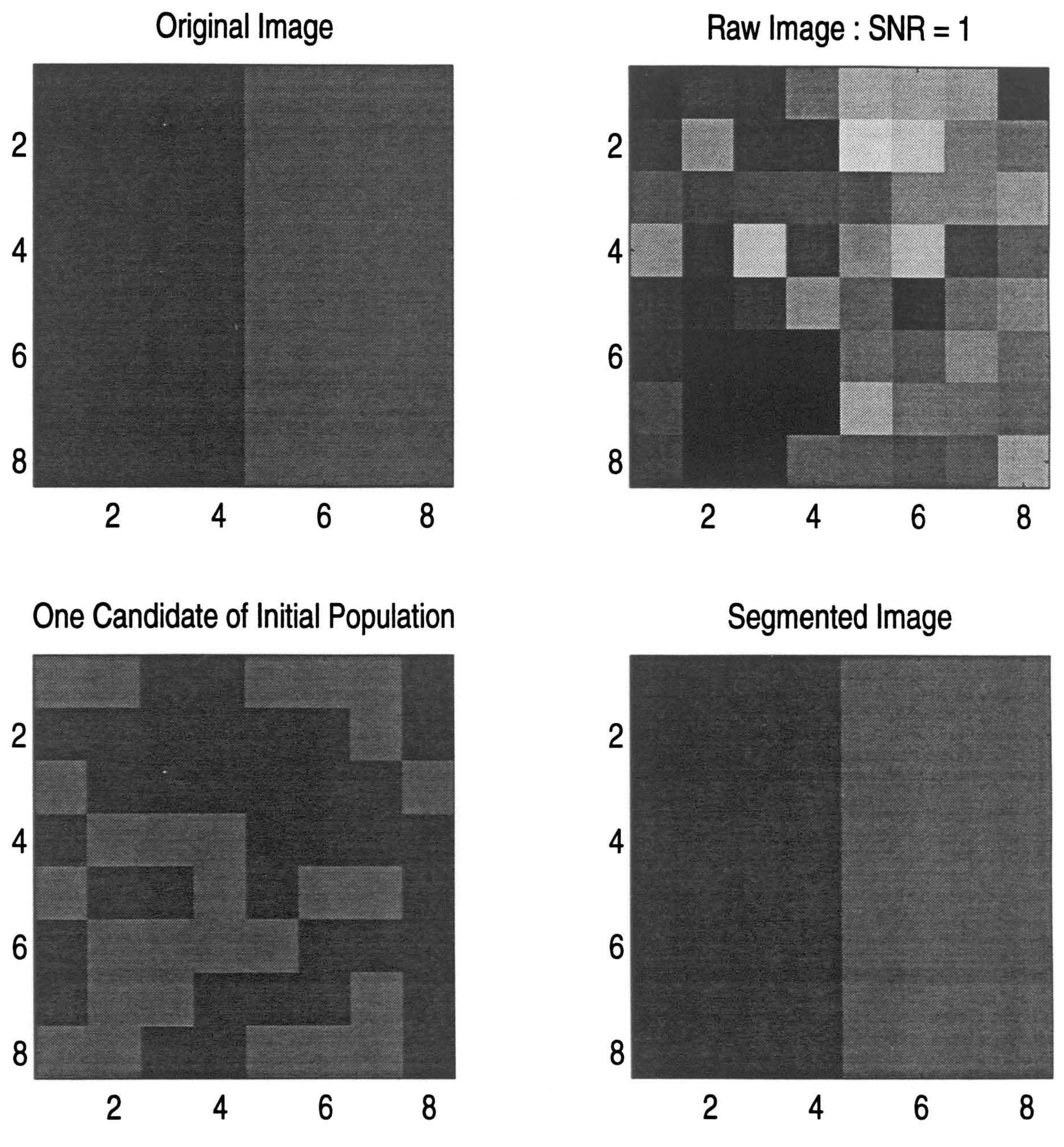


Figure 4.18: Initial results for 8 X 8 low contrast subimage

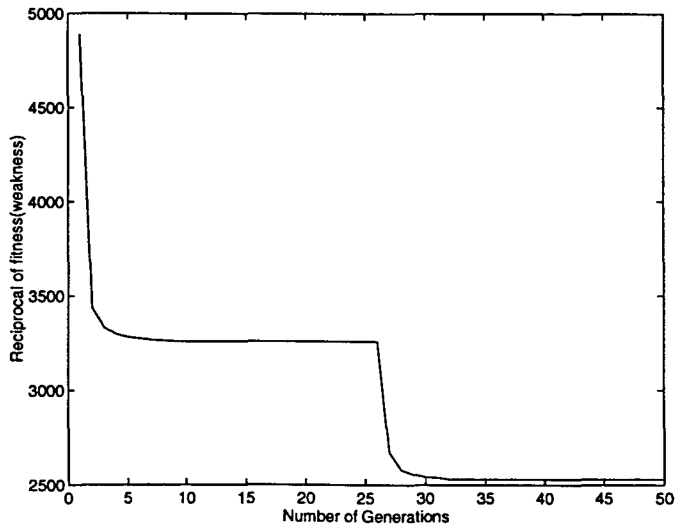


Figure 4.19: The plot of "weakness" versus the number of generations (50)

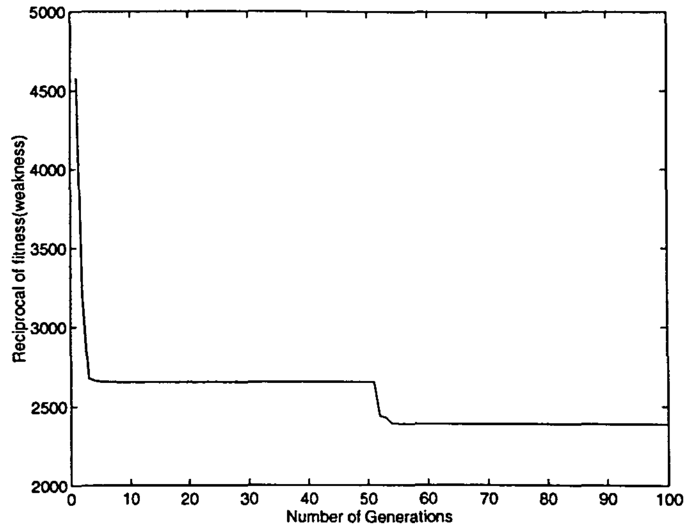


Figure 4.20: The plot of "weakness" versus the number of generations (100) for Figure 4.21

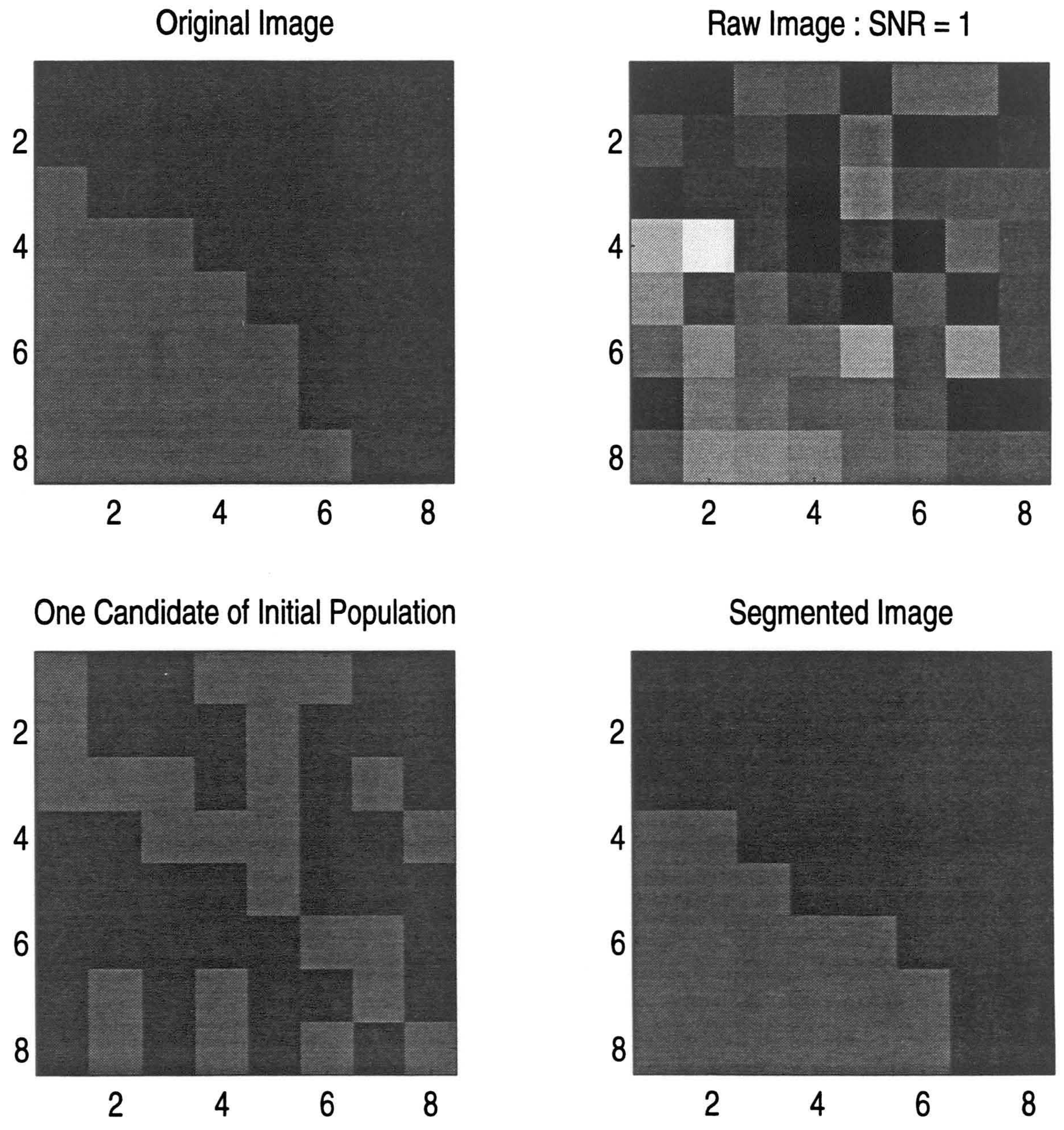


Figure 4.21: Results for 8 X 8 low contrast image with curved object edge

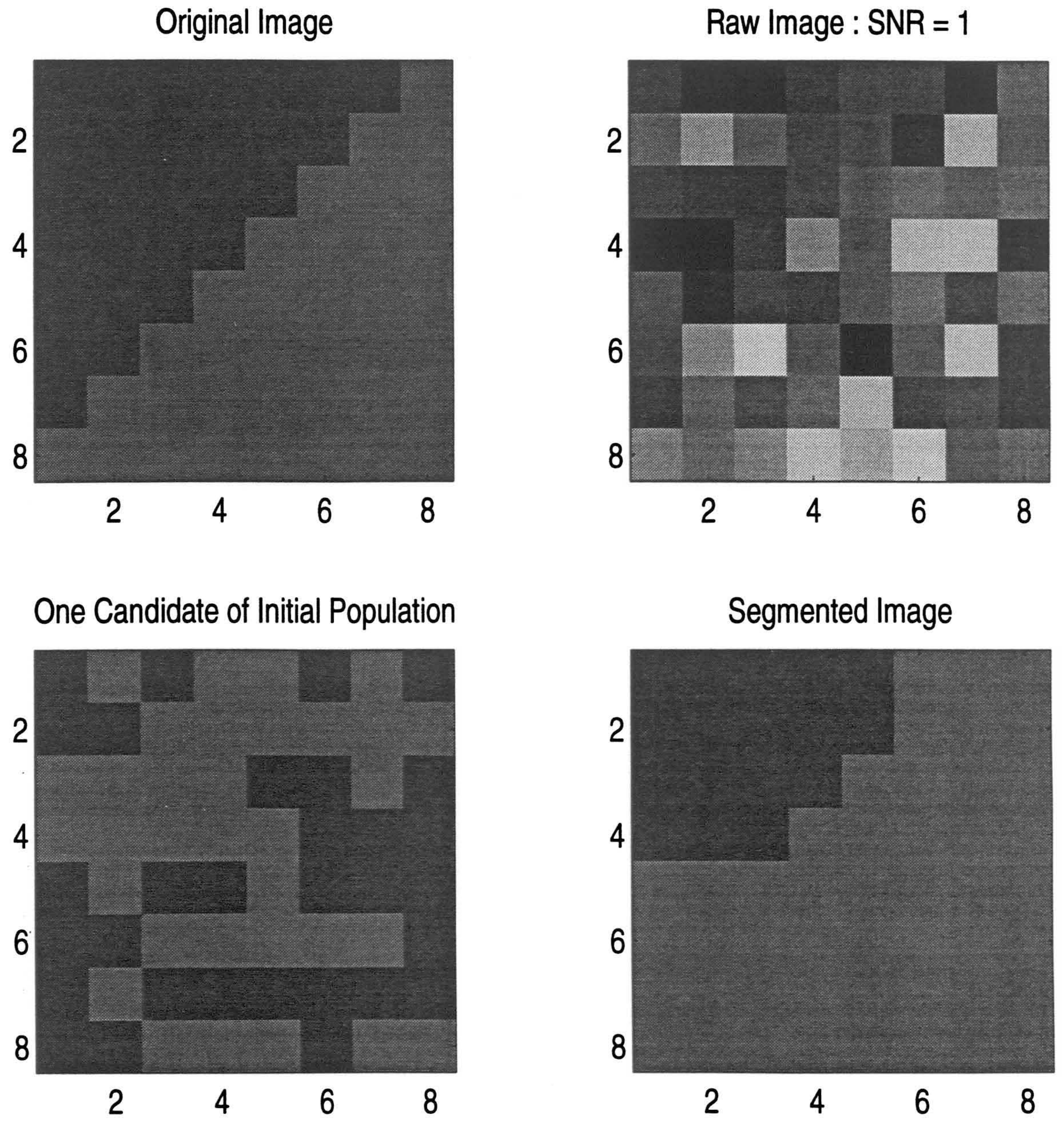


Figure 4.22: Results for 8 X 8 low contrast image with jagged object edge

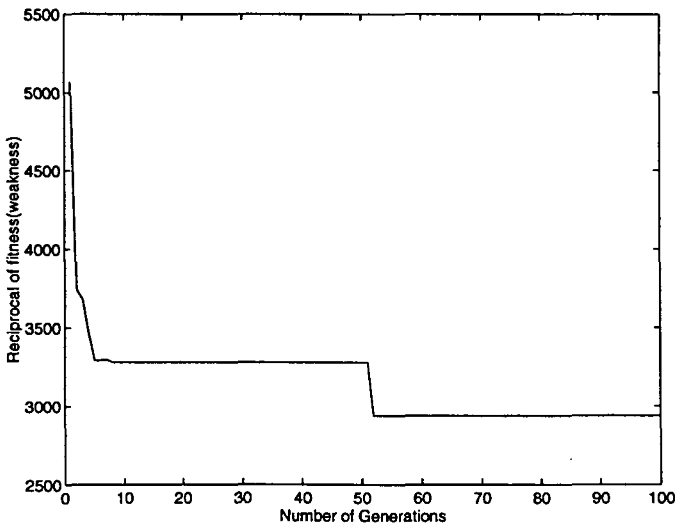


Figure 4.23: The plot of "weakness" versus the number of generations (100) for Figure 4.22

The algorithm was tried a number of times with varying initial population. The segmented image converged to the true image in majority of the trials indicating high probability of convergence to the global minima even at a low SNR of 1. Howver, since the signal to noise ratio was decreased, the number of generations was increased to 100.

Figures 4.21 and 4.22 show the results on object boundaries that are curved and jagged object edges. Figures 4.20 and 4.23 show the corresponding plots of average "weakness" versus number of generations. In these images, the specifications were the same, i.e. Ro = 125, Rb = 75, and $SNR = 1$. The population size was 30 and the number of generations for which the algorithm was made to run was 100.

Stopping Criteria

In the experiments conducted so far, the stopping criterion used for the termination of the algorithm was a fixed number of generations. In this section an alternative stopping criteria is described and its effect on the segmentation results same are presented. The algorithm is terminated whe the fitness function converges to a steady value, i.e. the difference between the average fitness and the maximum fitness becomes less than a threshold ϕ.

$$\frac{\sum_{i=1}^{N} f(Y_k^i)}{N} - Max(f) < \phi \qquad where \ \ \phi = Ro - Rb \tag{4.1}$$

In order that the algorithm does not get stuck in a local extrema, the algorithm is forced to terminate after 100 generations even if the convergence is not reached. In the previous experiments, the number of generations played an important role in reaching a good solution, and the number of generations was obtained by trial and error. Such a procedure can lead to severe problems in a practical application. One possible approach is to choose the number of generations sufficiently large.However this would be very inefficient and computationally wasteful. Figures 4.24, 4.25 and 4.26 present the results obtained when the the stopping criterion described above is used. The images used have Ro = 150, Rb = 50, SNR = 1.58 and the initial population size is 50. The number of generations after which the algorithm converged is 13, 8 and 10 respectively. Figures 4.27, 4.28 and 4.29 show the plot of the average "weakness" versus the number of generations for the the results shown in Figures 4.24, 4.25 and 4.26 respectively.

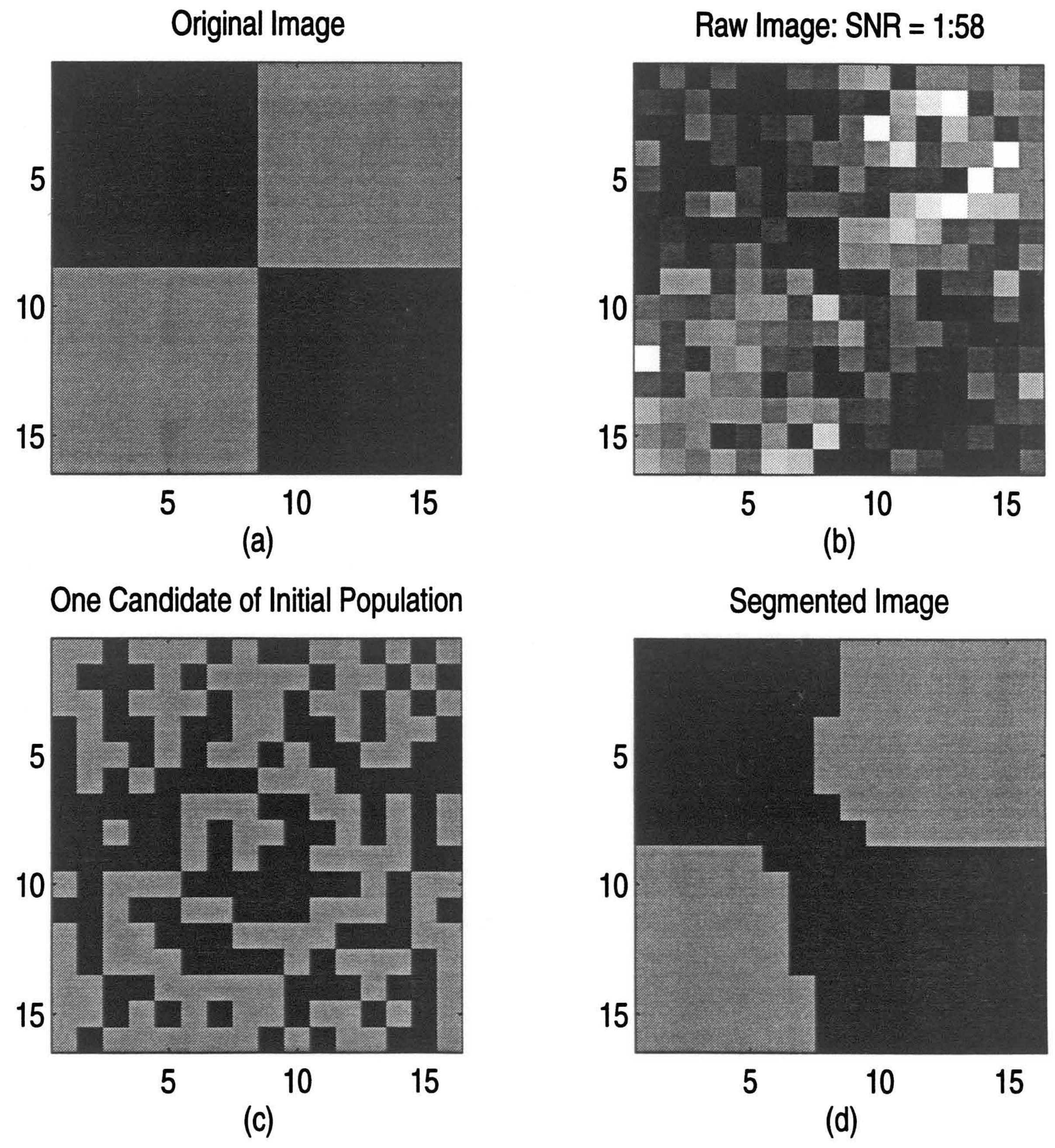


Figure 4.24: New stopping criterion used: horizontal and vertical boundary

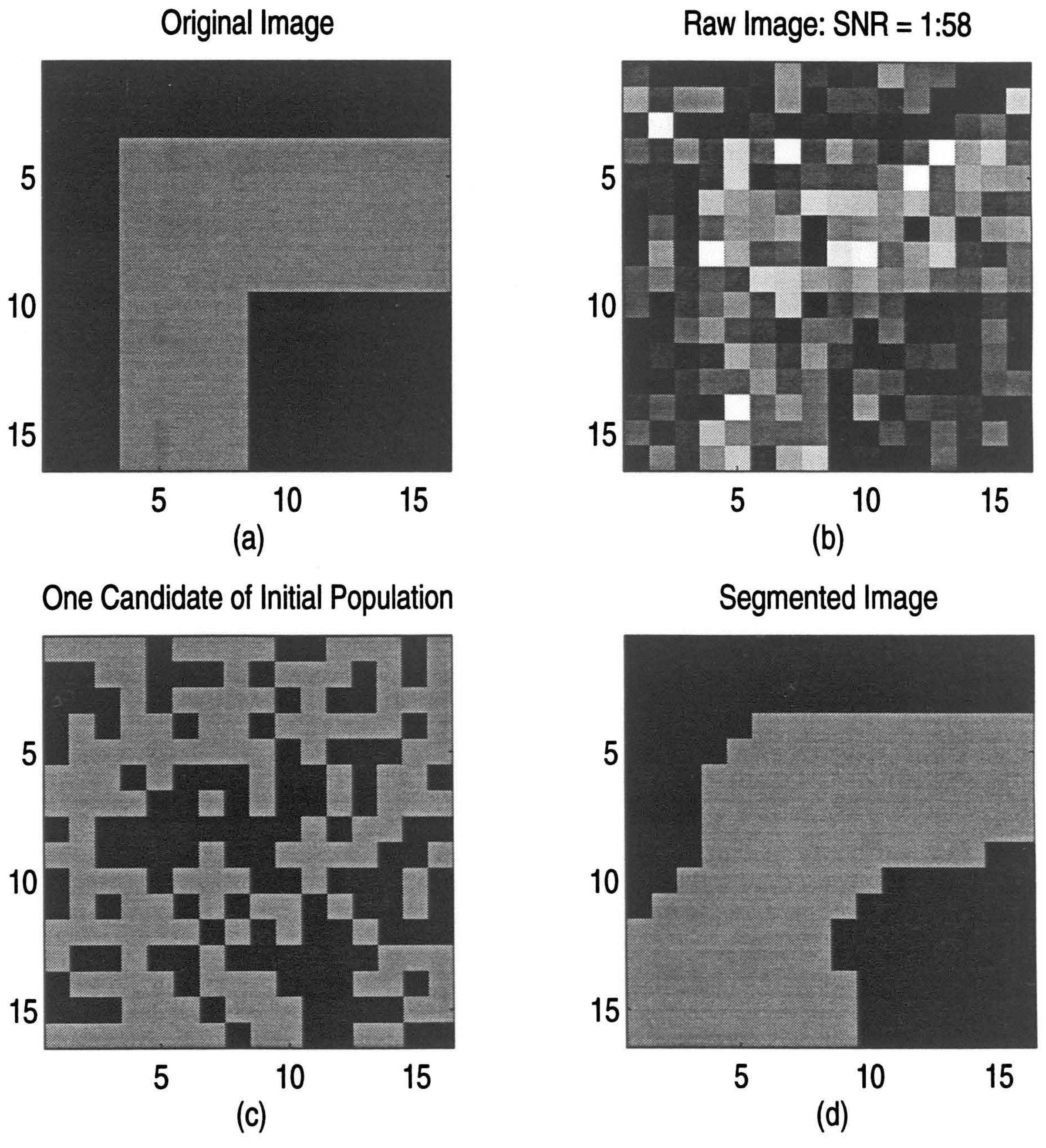


Figure 4.25: New stopping criterion used: sharp edges

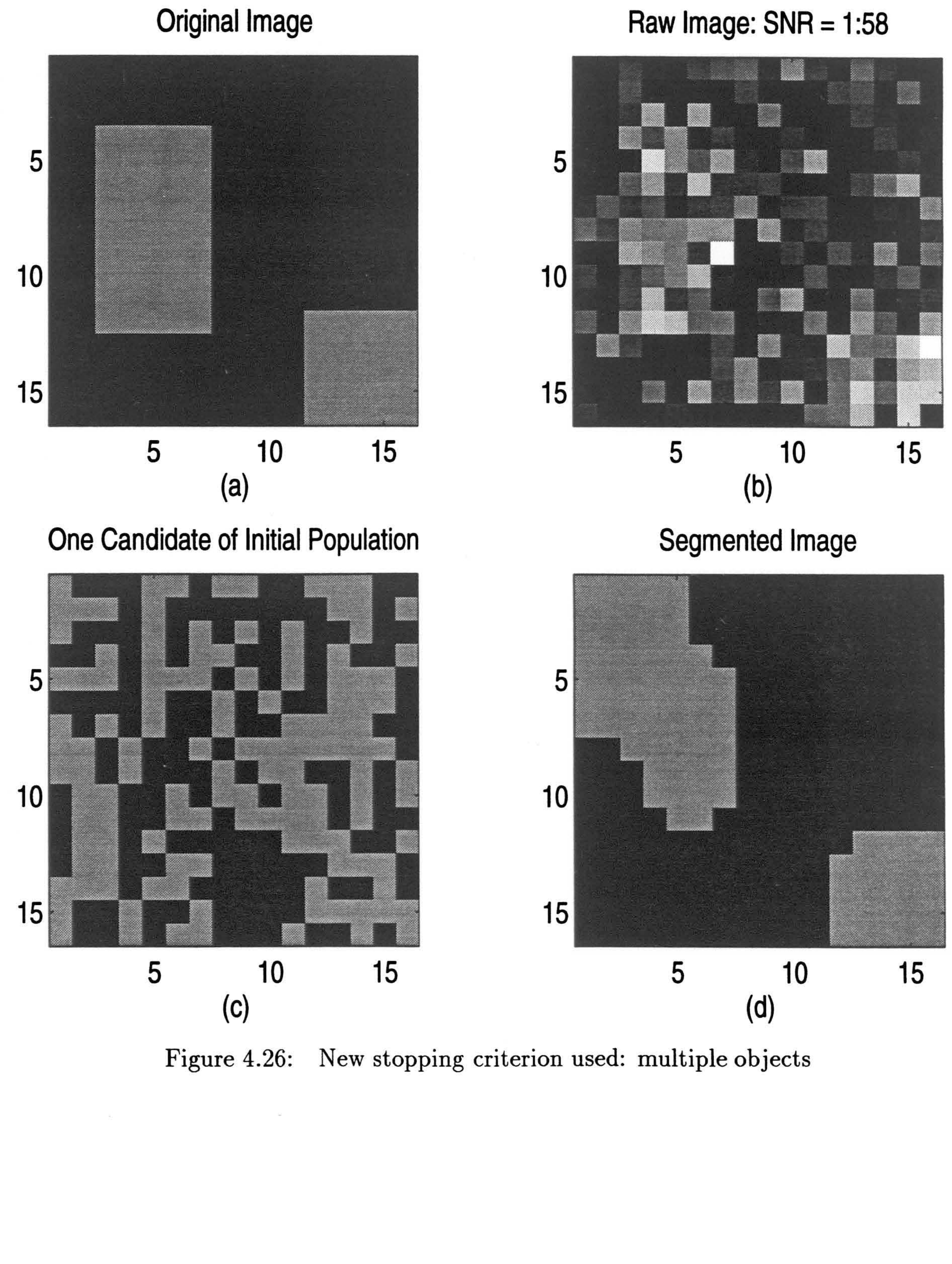


Figure 4.26: New stopping criterion used: multiple objects

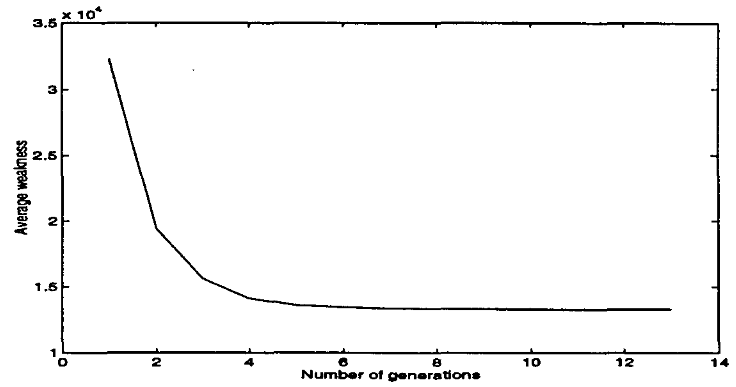


Figure 4.27: Average weakness plot for Figures 4.24

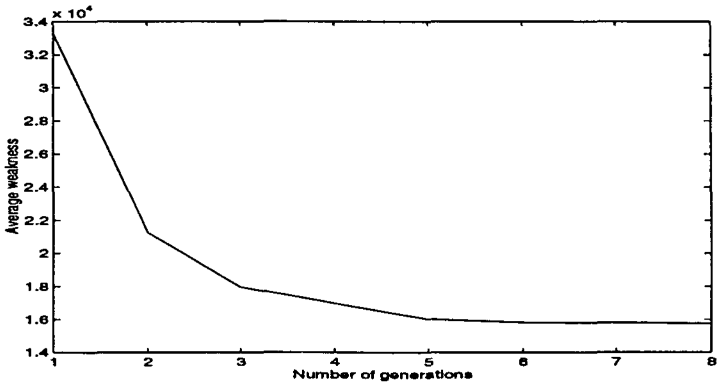


Figure 4.28: Average weakness plot for Figure 4.25

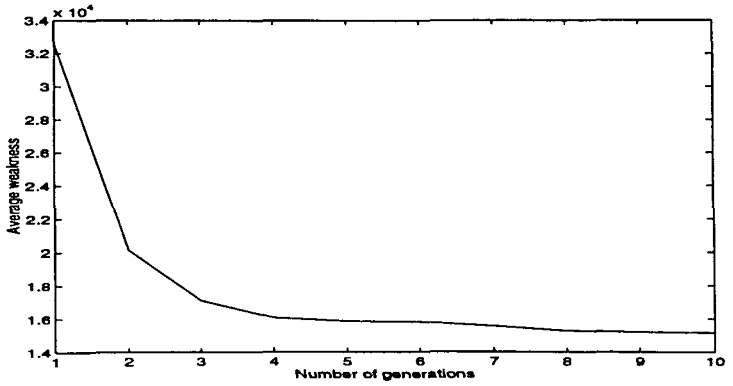


Figure 4.29: Average weakness plot for Figure 4.26

Effect of the Fitness Function Parameters

The fitness function comprises of a similarity measure ($E(Y_k)$), a reciprocal of the transition count ($T(Y_k)$), and a weighting function α The effect of modifying these parameters on the segmentation procedure is studied next.

1. **Similarity function** ($E(Y_k)$)

   In this experiment the fitness function was defined by only ($E(Y_k)$) Figures 4.30, 4.31 and 4.32 show the results obtained for different images with Ro = 150, Rb = 50, SNR = 1.58, and a population size of 50. The number of generations required in each of the three cases is 7, 9 and 7 respectively. As can be observed the fitness function with only the similarity measure gives reasonably good results by itself. The second term is added only to give improved results.

2. **The role of** ($T(Y_k)$)

   The parameter ($T(Y_k)$) is dependent on the number of transition (or discontinuities) in the candidate under consideration. It does not in any way depend on the raw image, which is the input to the system. This parameter is introduced only for fine tuning the fitness function and its function is to weight a homogeneous individual in the population over one that is inhomogeneous. In fact, images in which small objects are to be detected, should preferably not have the term ($T(Y_k)$). However, it can be adjusted with the α parameter, the role of which is studied in the next section.

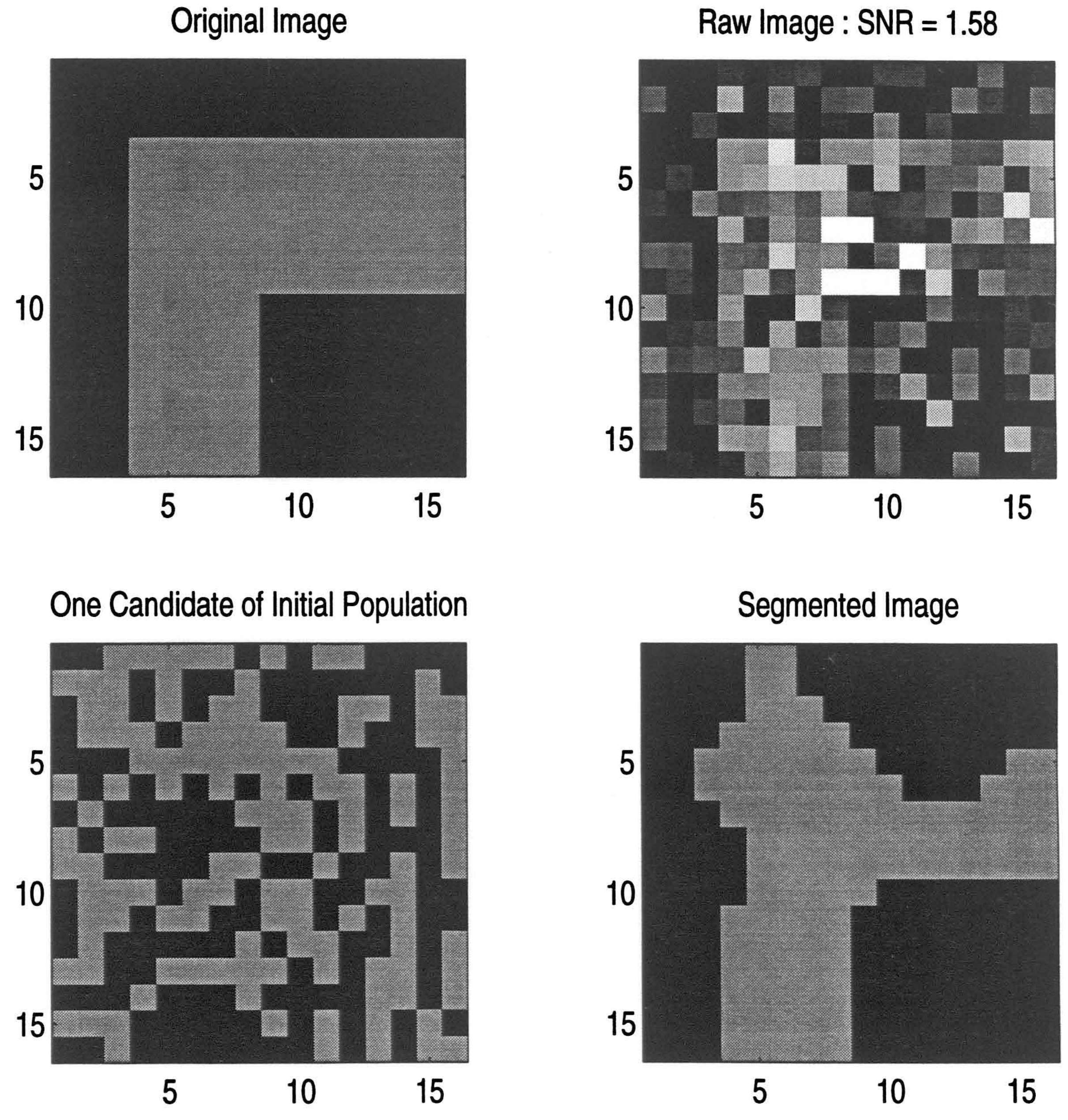


Figure 4.30: Results for 16 X 16 image with sharp edges: fitness function = $E(Y^k)$

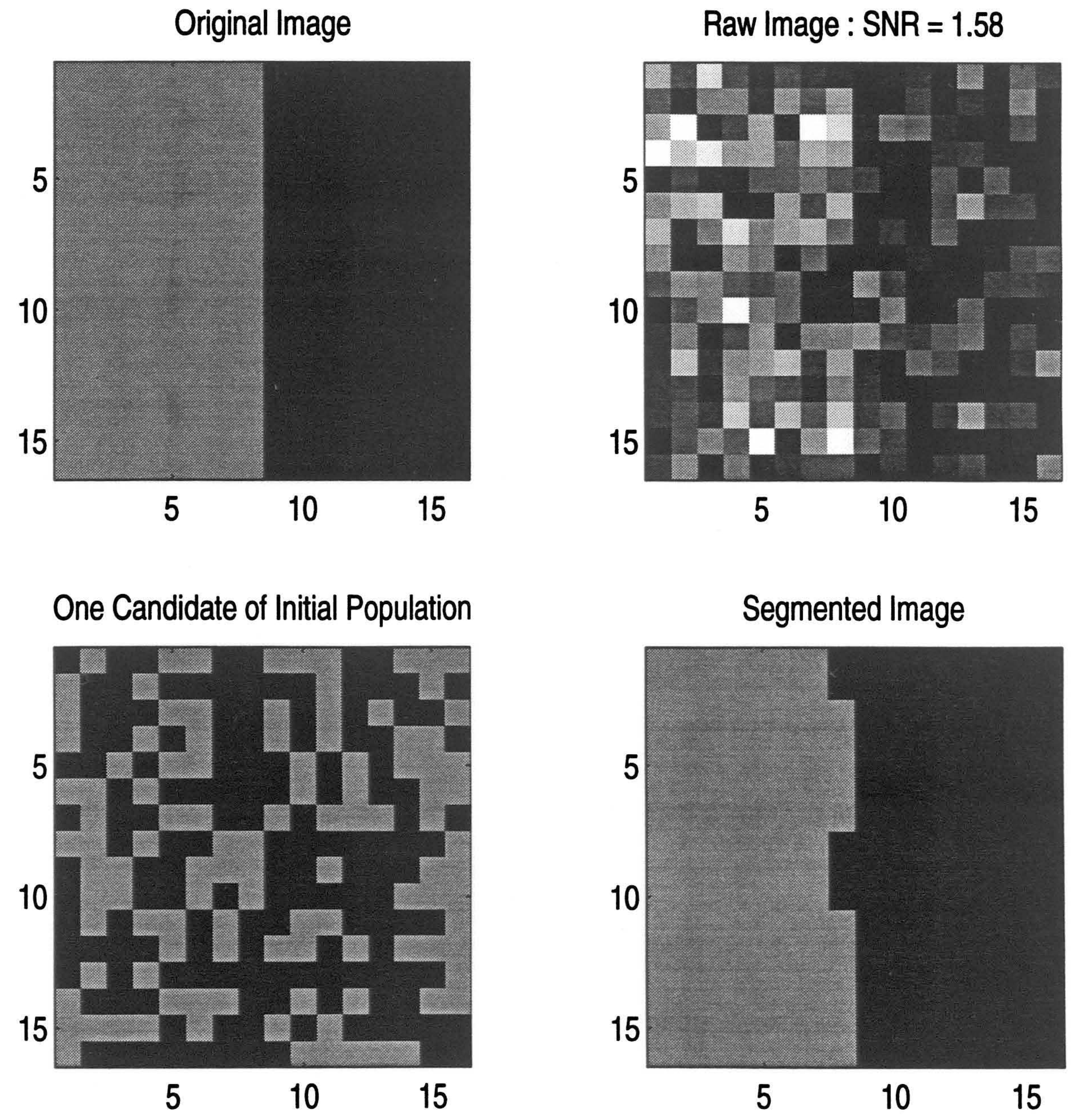


Figure 4.31: Results for 16 X 16 image with vertical boundary: fitness function = $E(Y^k)$

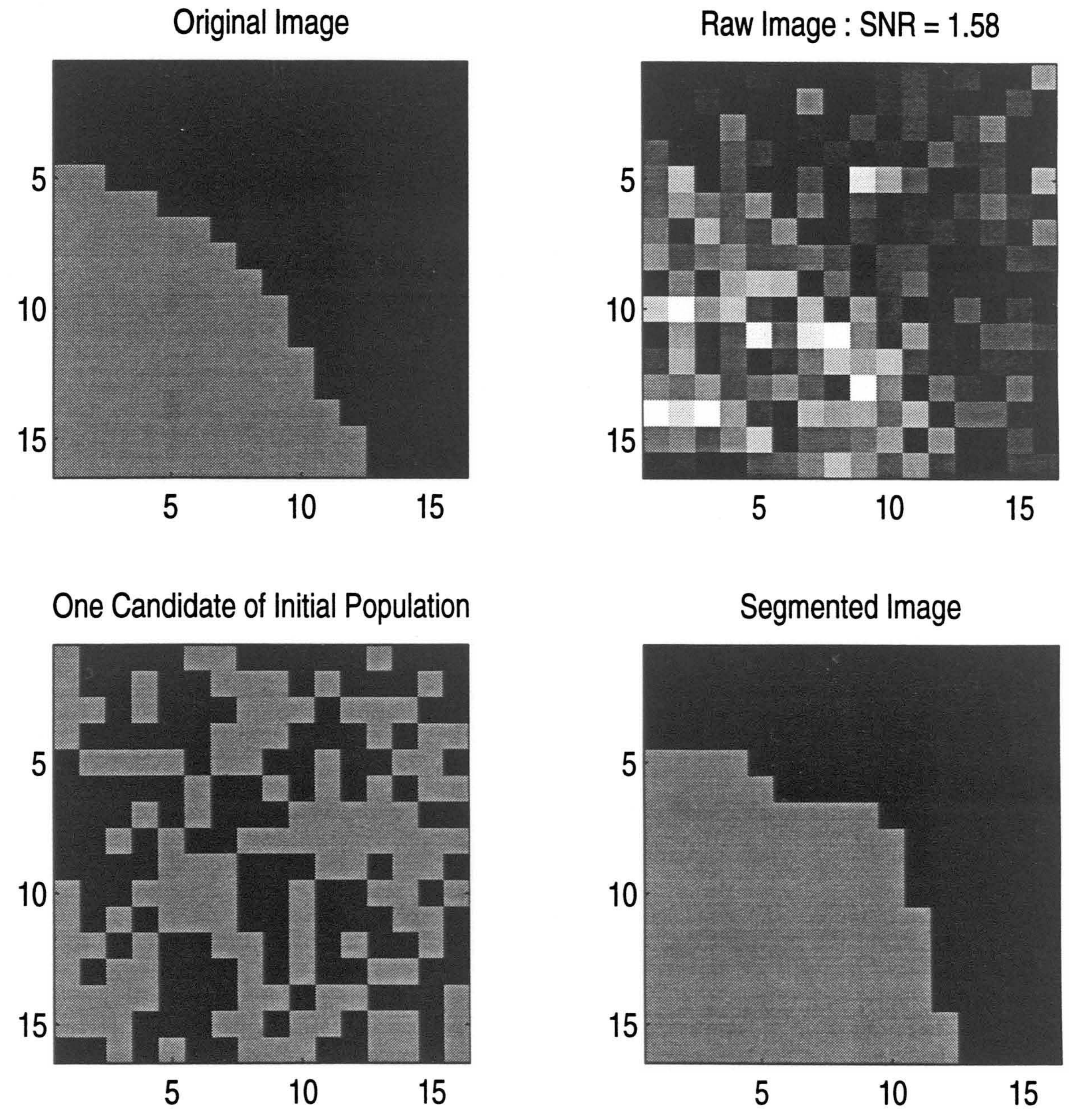


Figure 4.32: Results for 16 X 16 image with smooth curve: fitness function = $E(Y^k)$

3. **The effect of using different values for α** In all the experiments conducted so far, alpha was kept as 0.3. The values of $E(Y^k)$ and $T(Y^k)$ were observed and it was found that 0.3 was a reasonable factor to normalize the two quantities. The raw images shown in Figure 4.34 was generated by introducing noise in images shown in Figure 4.33. Figures 4.35, 4.36, 4.37 and 4.38 show the results with $\alpha = 0.15$, $\alpha = 0.5$, $\alpha = 1$, and $\alpha = 10$ respectively. As can be observed, with very a very high value of α the $T(Y^k)$ is overemphasised. As discussed before, $T(Y^k)$ does not take the image data into consideration. So the results obtained with high α are not good. The value of α should therefore be less than 0.5 and was chosen to be 0.3 after trial and error.

Reproduction

Crossover and the Mutation are the two operators used for reproduction. Once the potential candidates are selected from the current generation, they are placed in a mating pool. All the experiments conducted so far, use the crossover technique described in Chapter 3. An alternate crossover technique is studied here.

The Mutation operator used so far considers an 8-pixel neighborhood of each bit in the chromosome. In this section we study the effect of a 4-neighborhood mutation operator.

Crossover

The crossover applied here can be explained as follows. The individuals in the mating pool are selected two at a time.

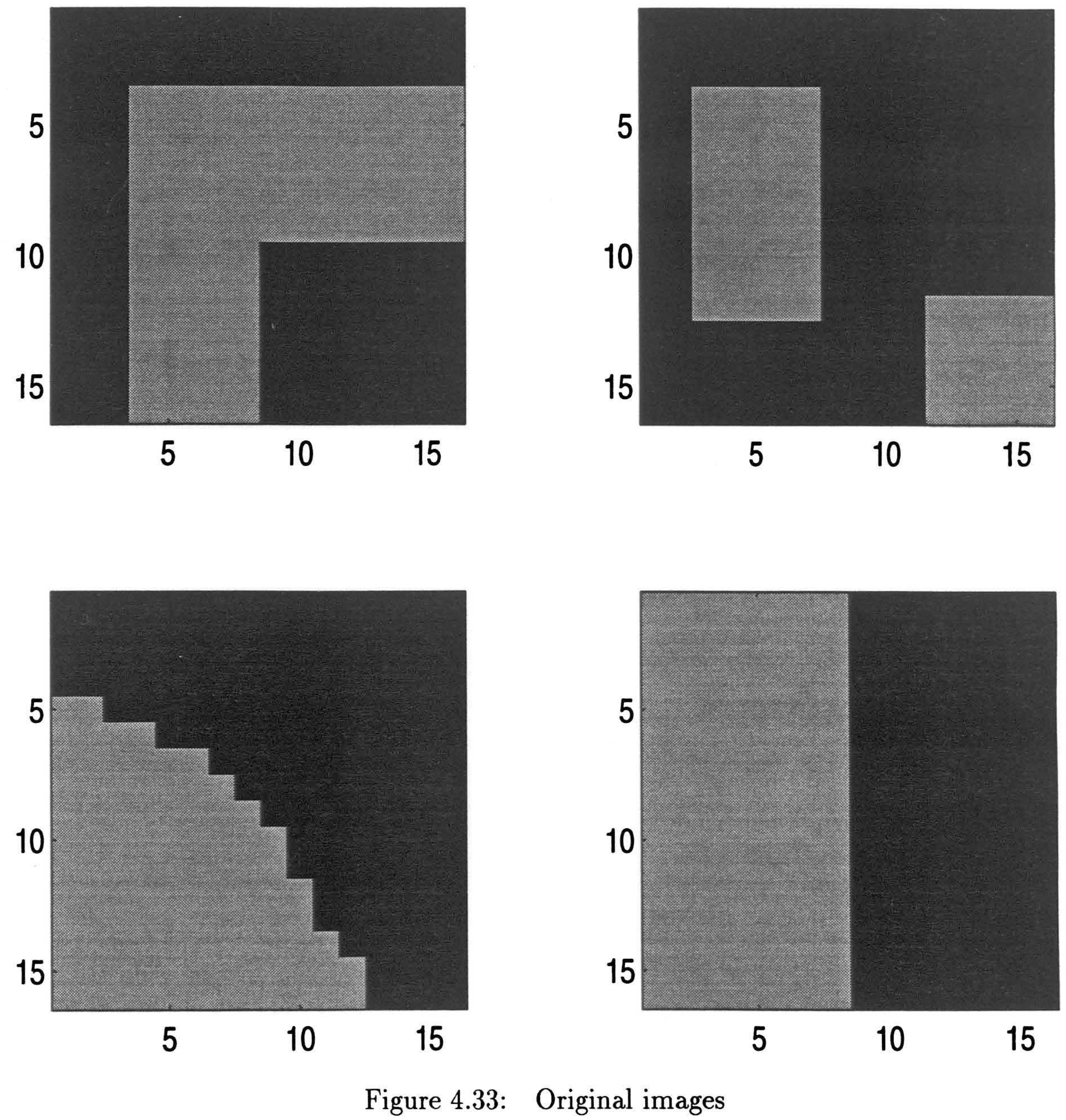


Figure 4.33: Original images

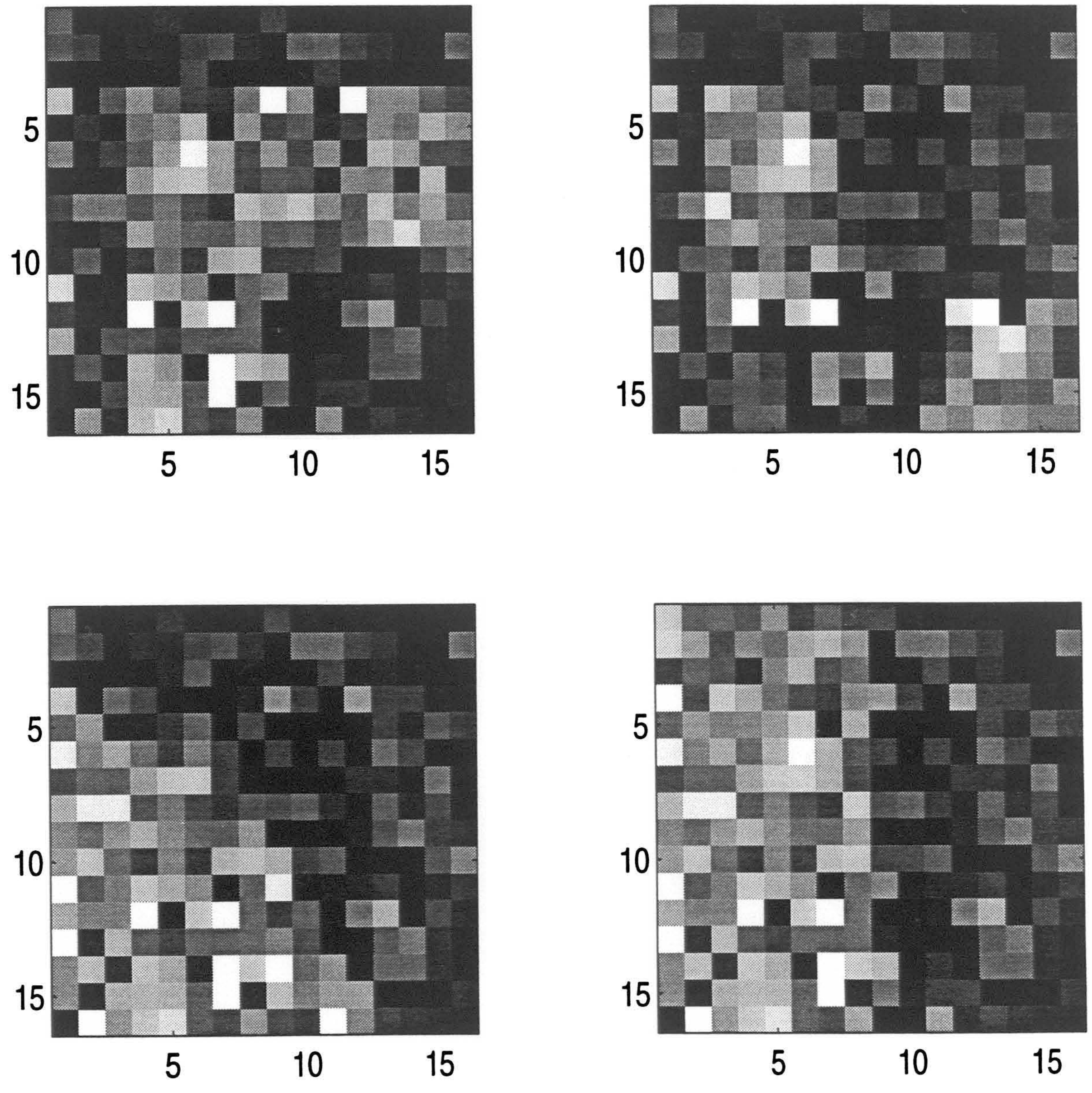


Figure 4.34: Raw images

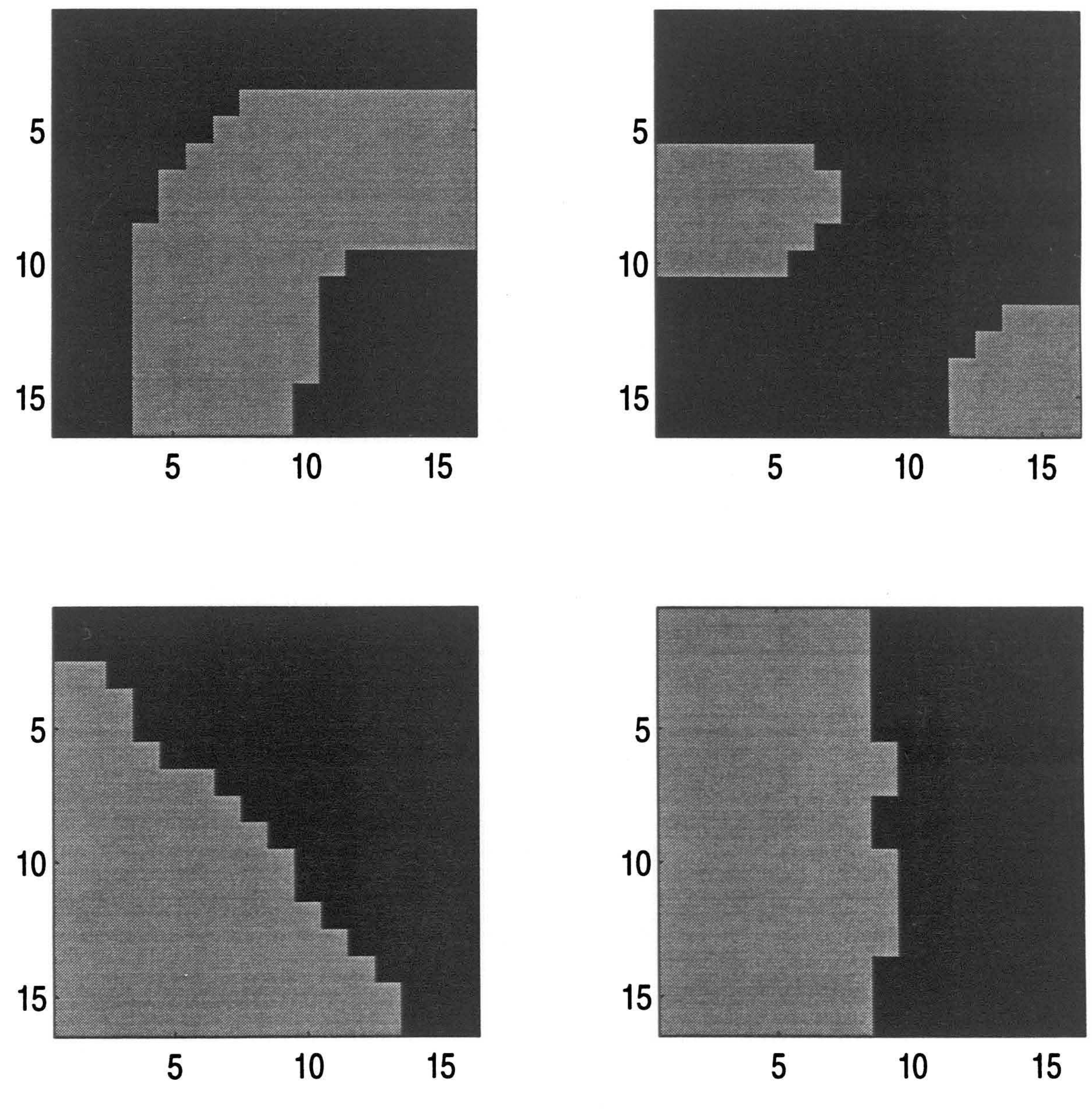


Figure 4.35: Results for 16 X 16 image: $\alpha = 0.15$

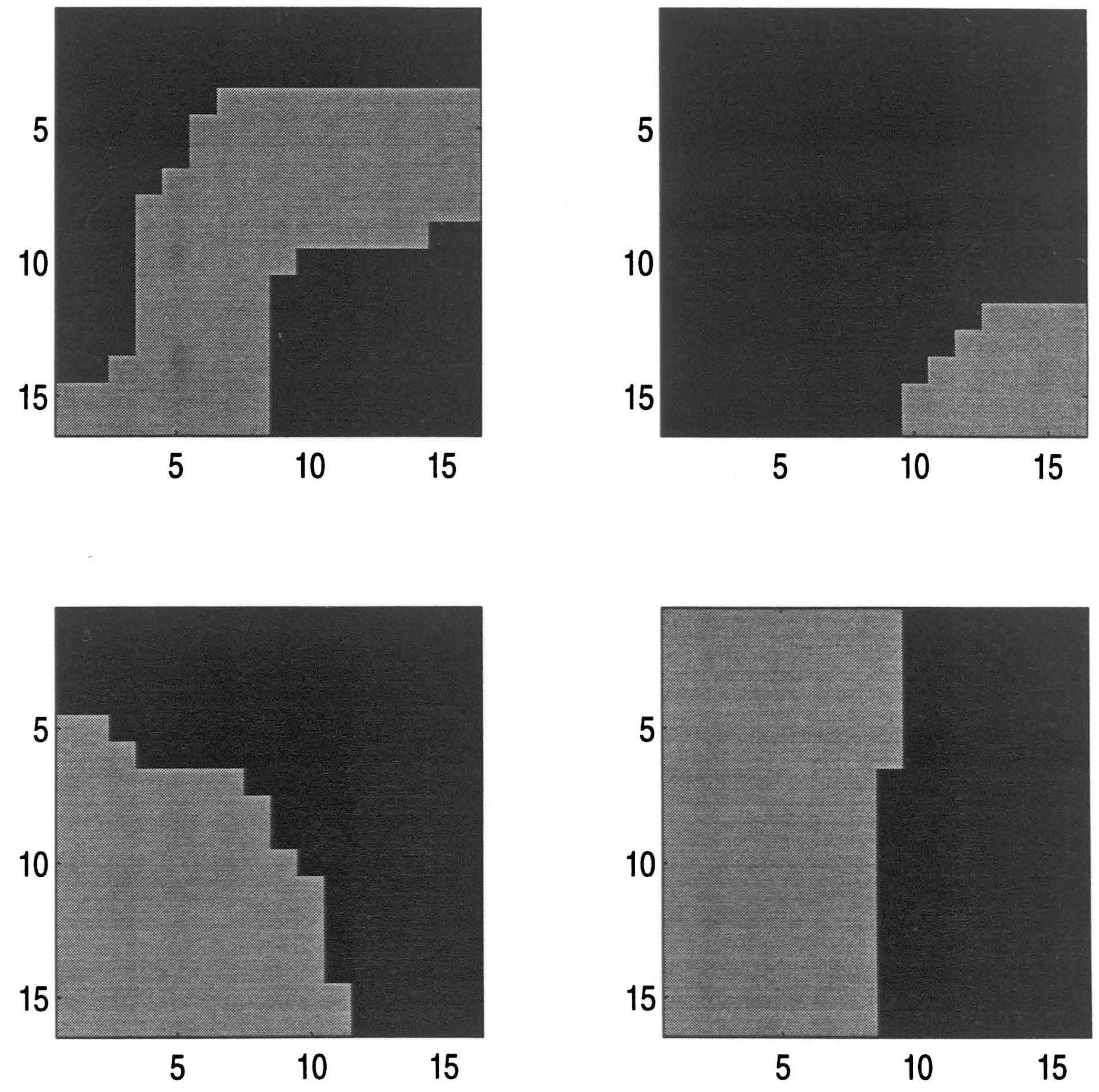


Figure 4.36: Results for 16 X 16 image: $\alpha = 0.5$

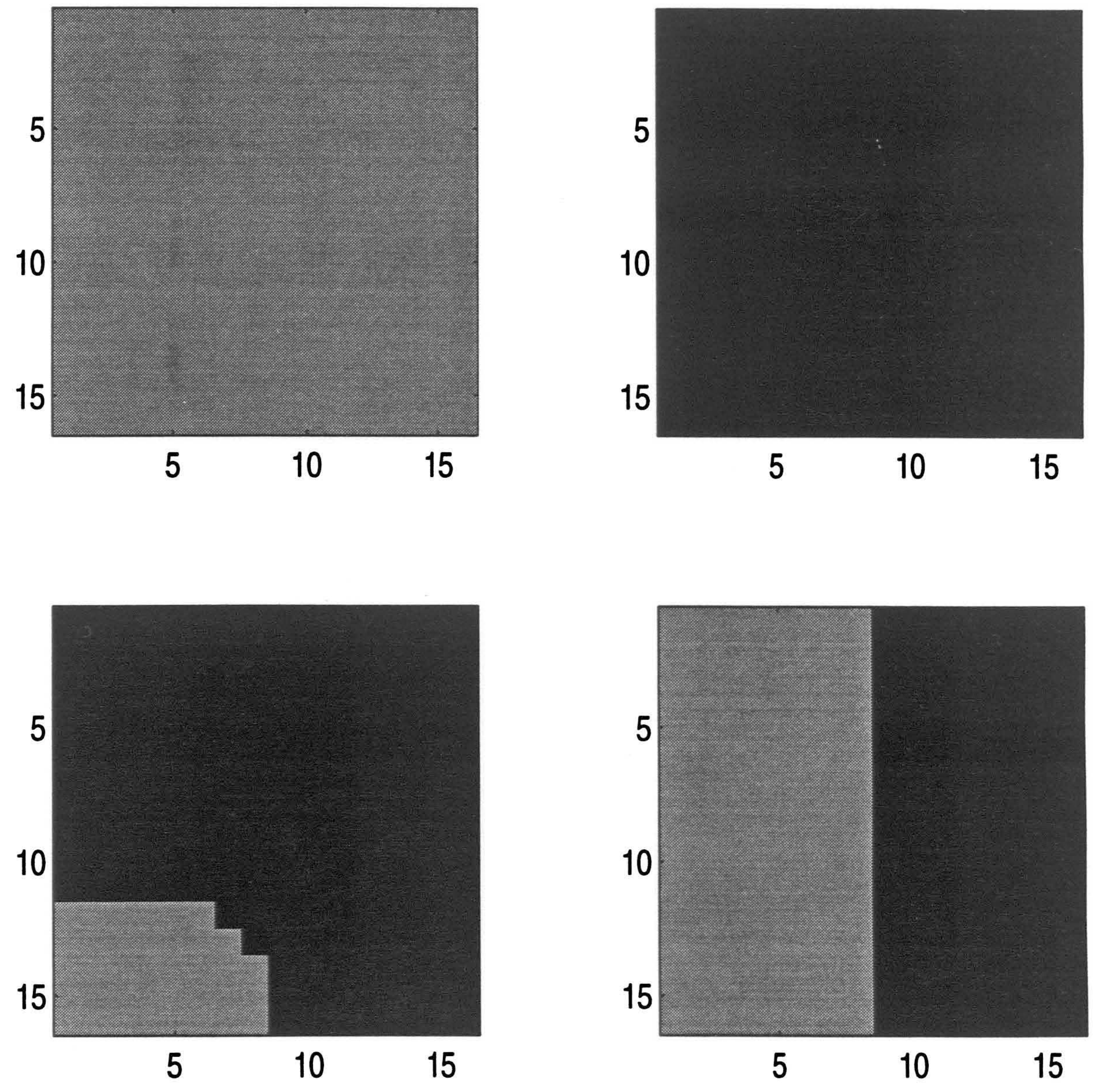


Figure 4.37: Results for 16 X 16 image: $\alpha = 1$

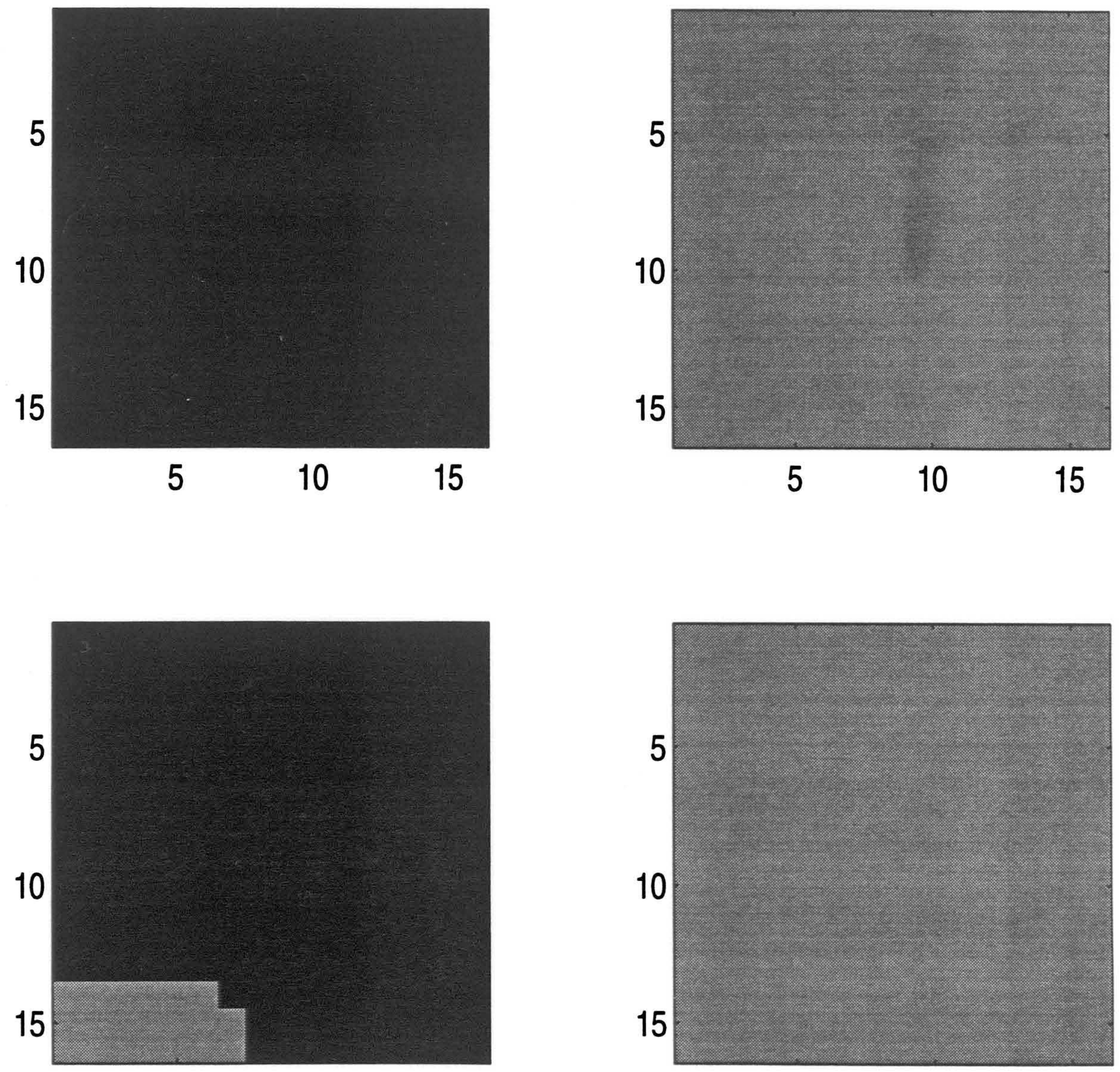


Figure 4.38: Results for 16 X 16 image: $\alpha = 10$

These two form the parents and a random crossover point is selected and the two parents produce two offsprings just as in a conventional crossover. The population size is fixed as 200. In each crossover, the number of times the reproduction operator is applied is chosen randomly.

For example, if the random number generated is 23 (the random number is set to be between 1 and size of the population), then a pair of parents is selected for reproduction 23 times (a different pair everytime) to produce 46 offsprings.

The images used here are similar to previous images, wherein Ro = 150, Rb=50 SNR = 1.58 and the stopping criteria used is the convergence of fitness function as shown in equation 4.1. α is chosen to be 0.2 in these experiments. The Figures 4.39, 4.40, and 4.41 show the results obtained with the random crossover operator.

Mutation

If the object is very small, the mutation operator has the effect of deleting very small objects that are treated as noise. Hence the mutation operator using an 8-pixel neighborhood, will not be successful in detecting it. However, a 4-pixel neighborhood mutation operator applied with the crossover mentioned above is successful in detection objects of size as small as 3 X 3.

These results are presented in Figures 4.42 and 4.43. The images have R0 = 150, Rb = 50, SNR = 1.58, α = 0.2 and the population size is chosen as 200. The initial population is generated randomly. As can be seen, this mutation operator performs well only in case of presence of small objects. In images with continuous objects, 8-neighborhood mutation operator performs better.

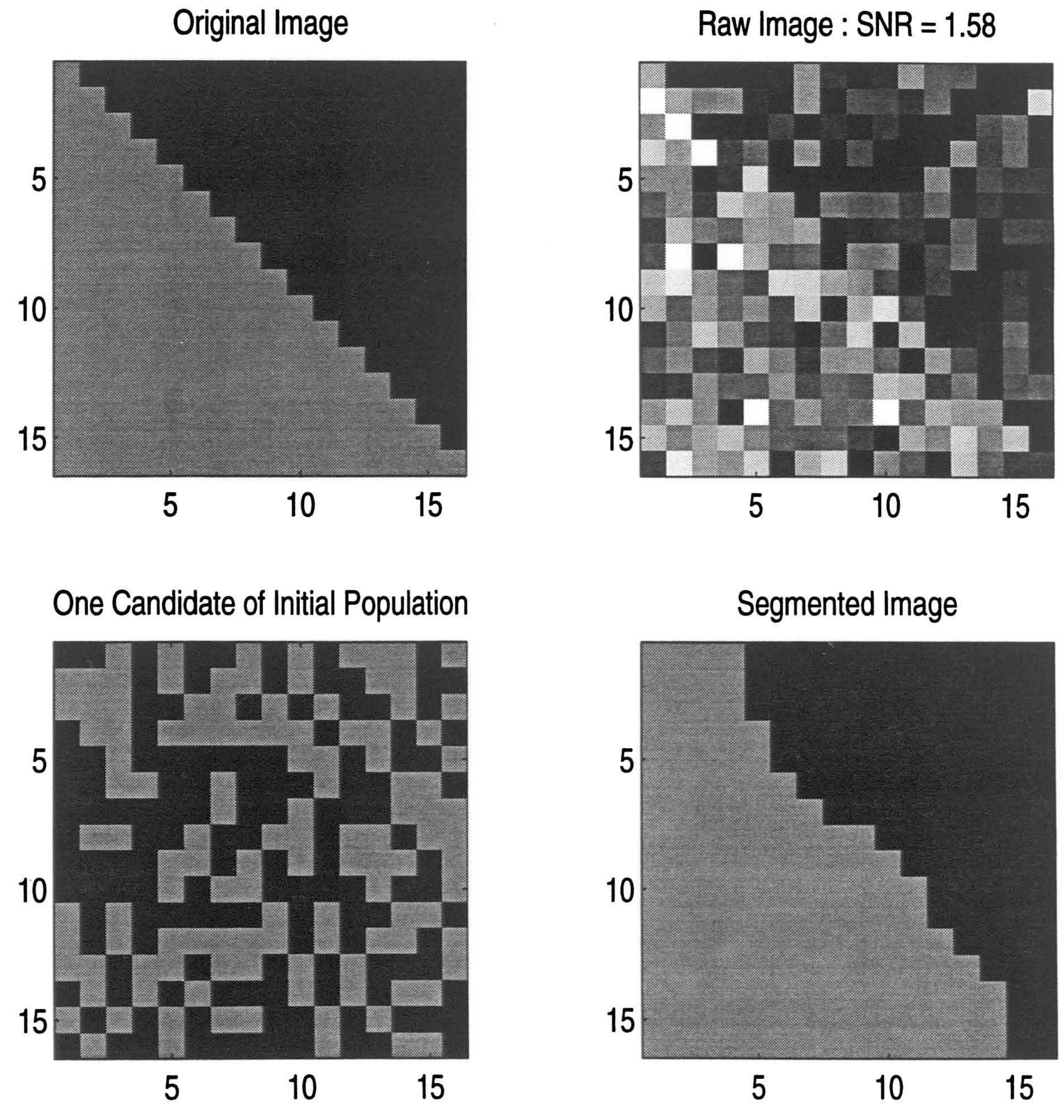


Figure 4.39: Results for 16 X 16 image with jagged edges: new crossover technique

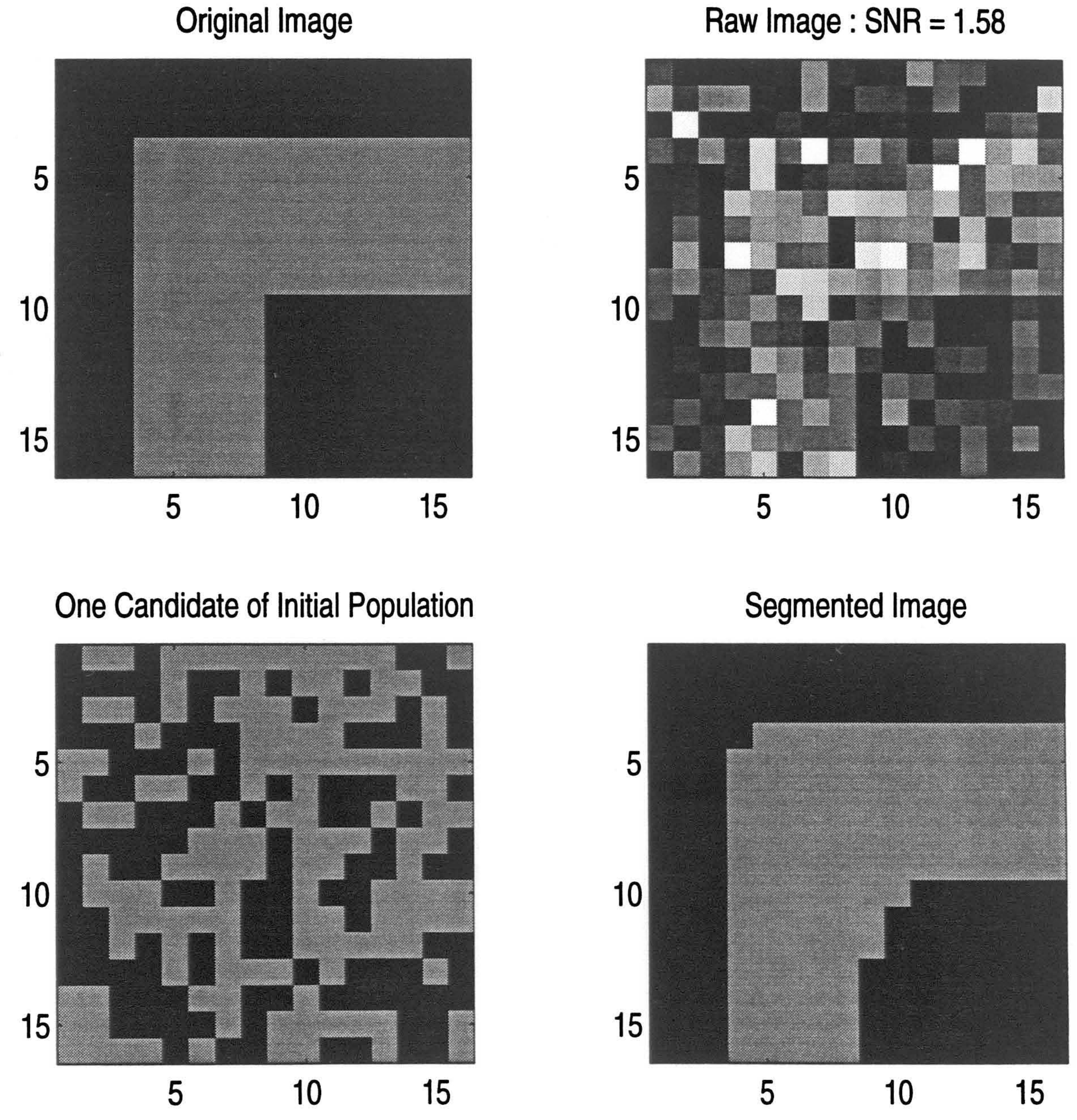


Figure 4.40: Results for 16 X 16 image with sharp edges: new crossover technique

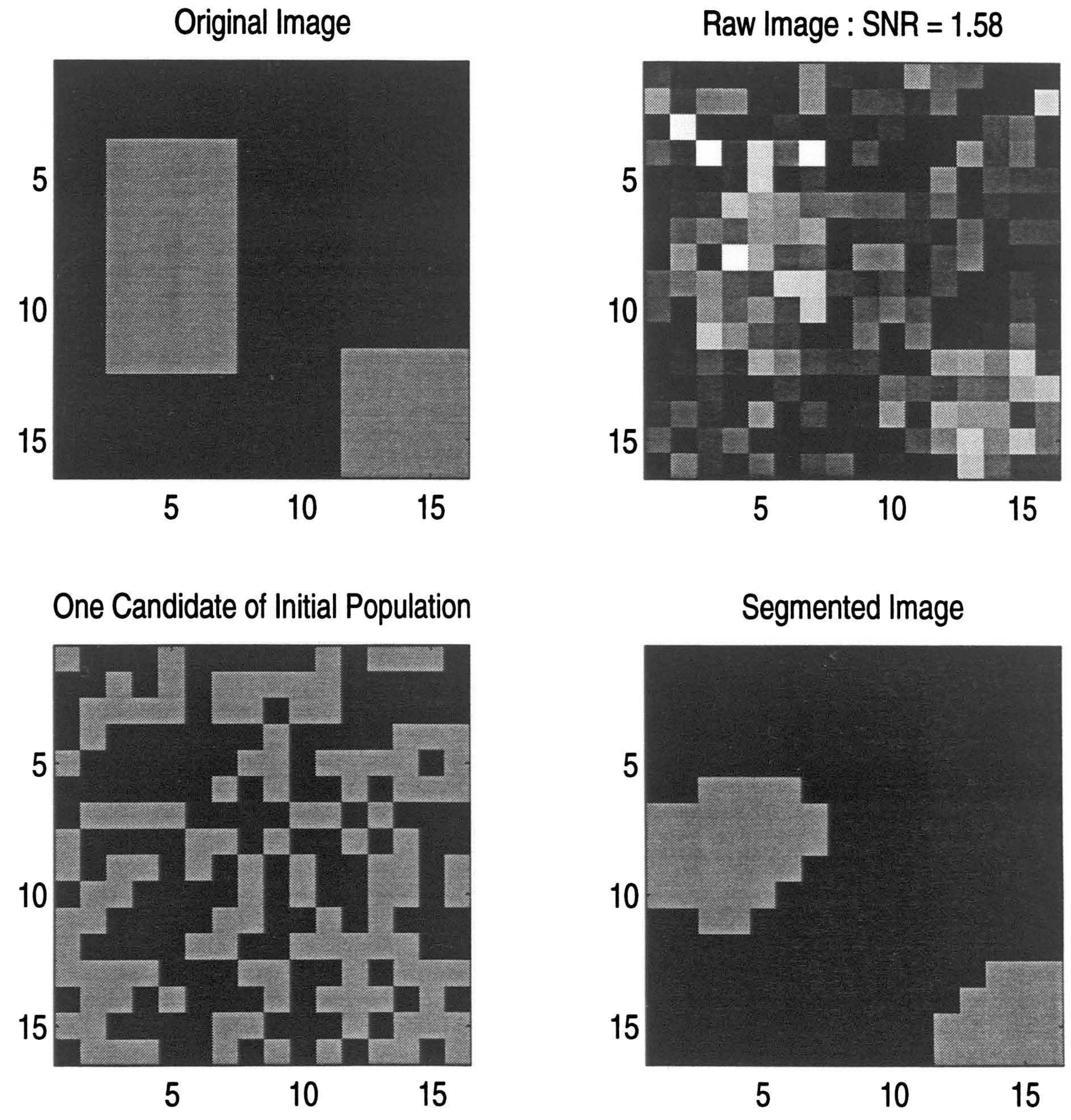


Figure 4.41: Results for 16 X 16 image with multiple objects: new crossover technique

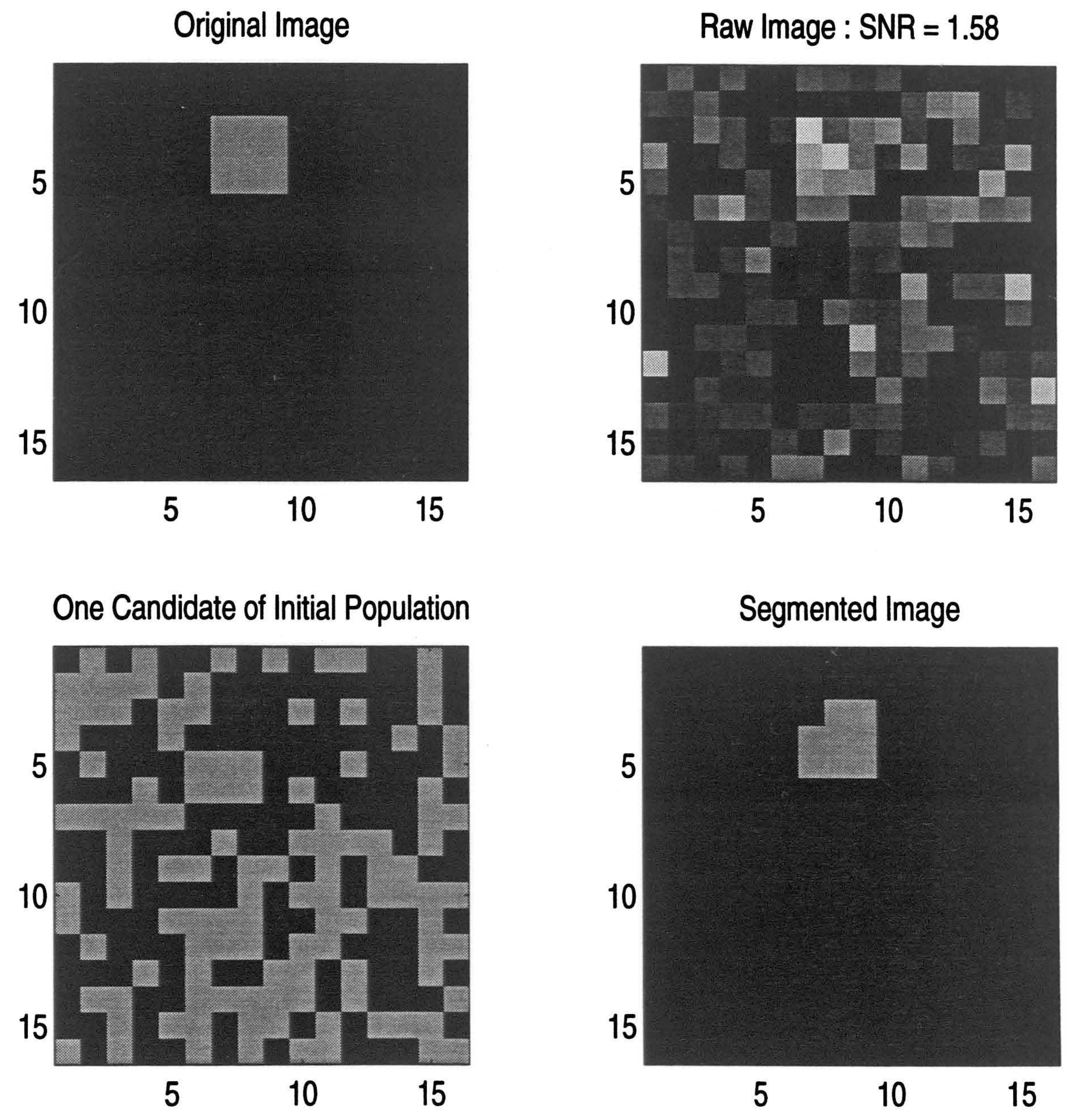


Figure 4.42: Results for 16 X 16 image with small object: 4-pixel neighborhood mutation

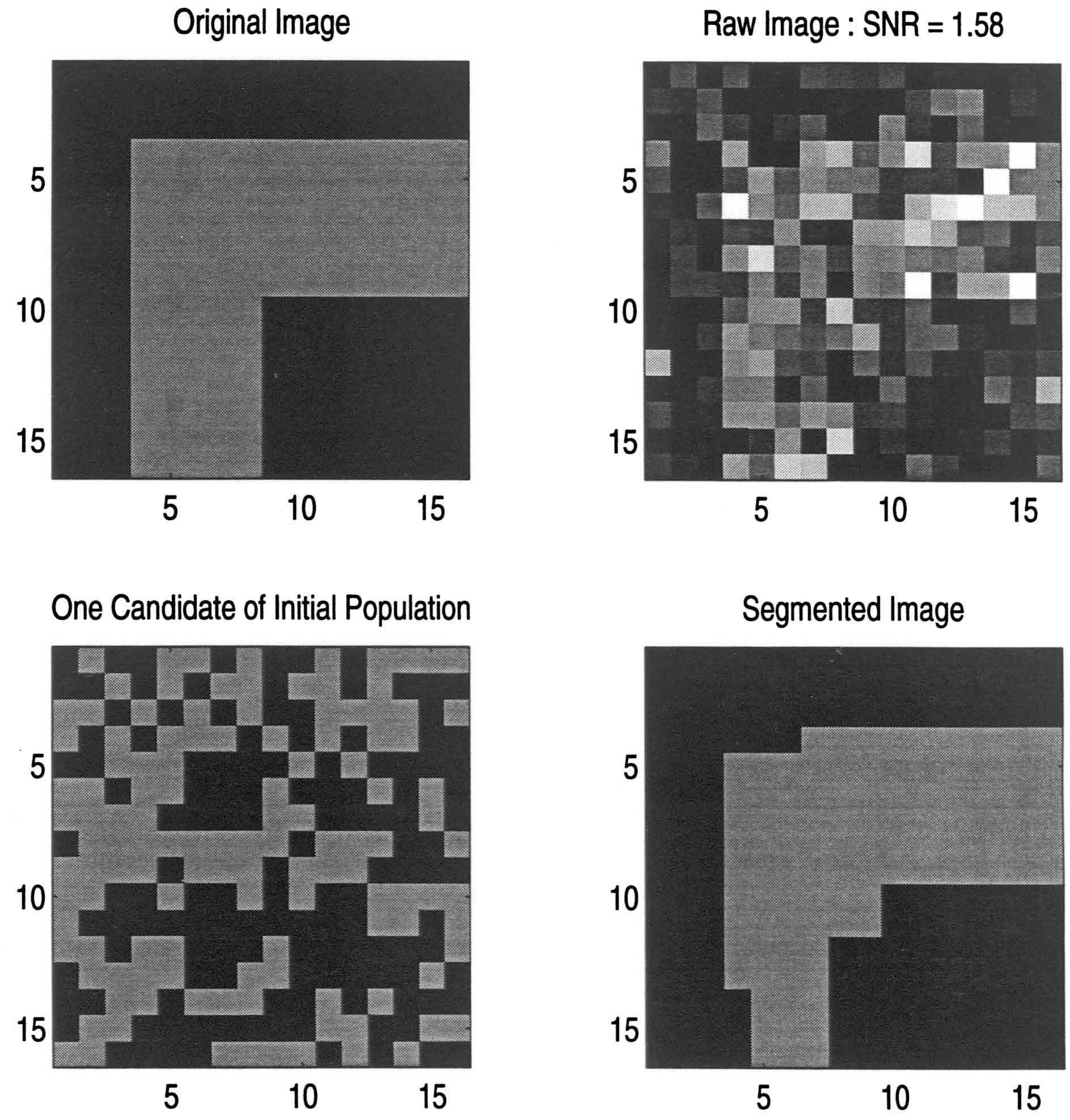


Figure 4.43: Results for 16 X 16 image with sharp edges: 4-pixel neighborhood mutation

Application to Bigger Images

As has been explained before in the proposed algorithm, the image is subdivided into subimages and the segmentation algorithm is performed independently on each subimage. The resulting segmented images are combined to give the final image. Figures 4.44 and 4.45 show experiments carried out with images of size 64 X 64. If the image size is not a factor of 16 it can be padded with zeroes so that the image dimensions are then a factor of 16. In Figure 4.44 the specifications are Ro = 150, Rb = 50, SNR = 1.58, size of the initial population = 100, $\alpha = 0.2$, initial population is generated randomly, crossover used is with randomly selected parents and the mutation operator uses the 8-pixel neighborhood. In Figure 4.45 the Ro is 125, Rb = 75 and SNR = 1, while the other specifications are the same as above. The results show that the algorithm is successful in detecting multiple objects of different sizes and shapes in low contrast images with signal to noise ratio as low as 1.0.

Application to X-ray Images

The results presented so far were obtained using simulated images. The algorithm was implemented on experimental X-ray images and the results obtained are presented in Figure 4.46. The 128 X 256 size image that was input to the algorithm has various features as indicated by the varying gray levels. At the center of the image, there is a circular flaw which is to be identified. The object intensity and the background intensity found using histogram analysis were 184 and 63 respectively.

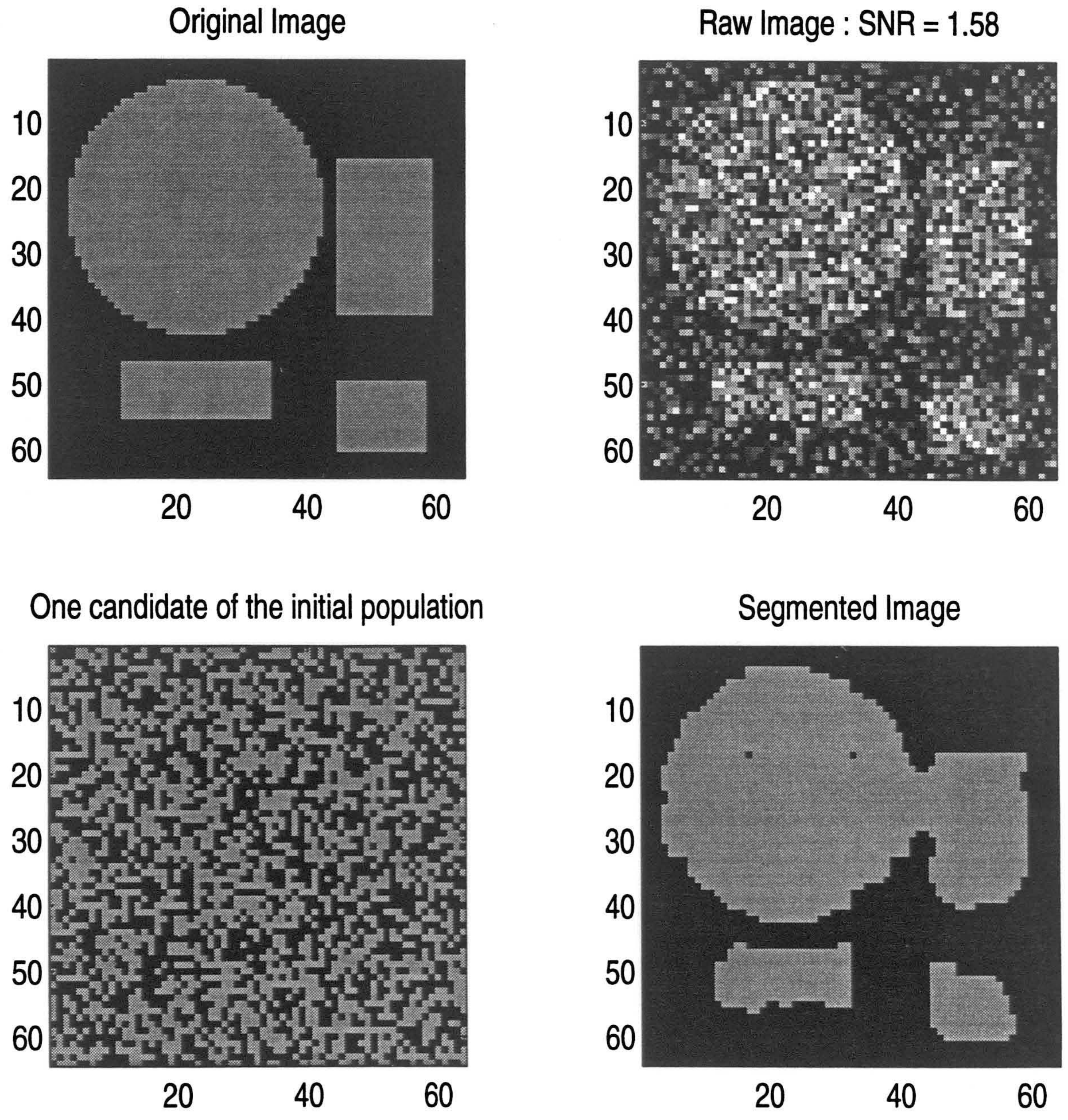


Figure 4.44: Results for 64 X 64 image: SNR = 1.58

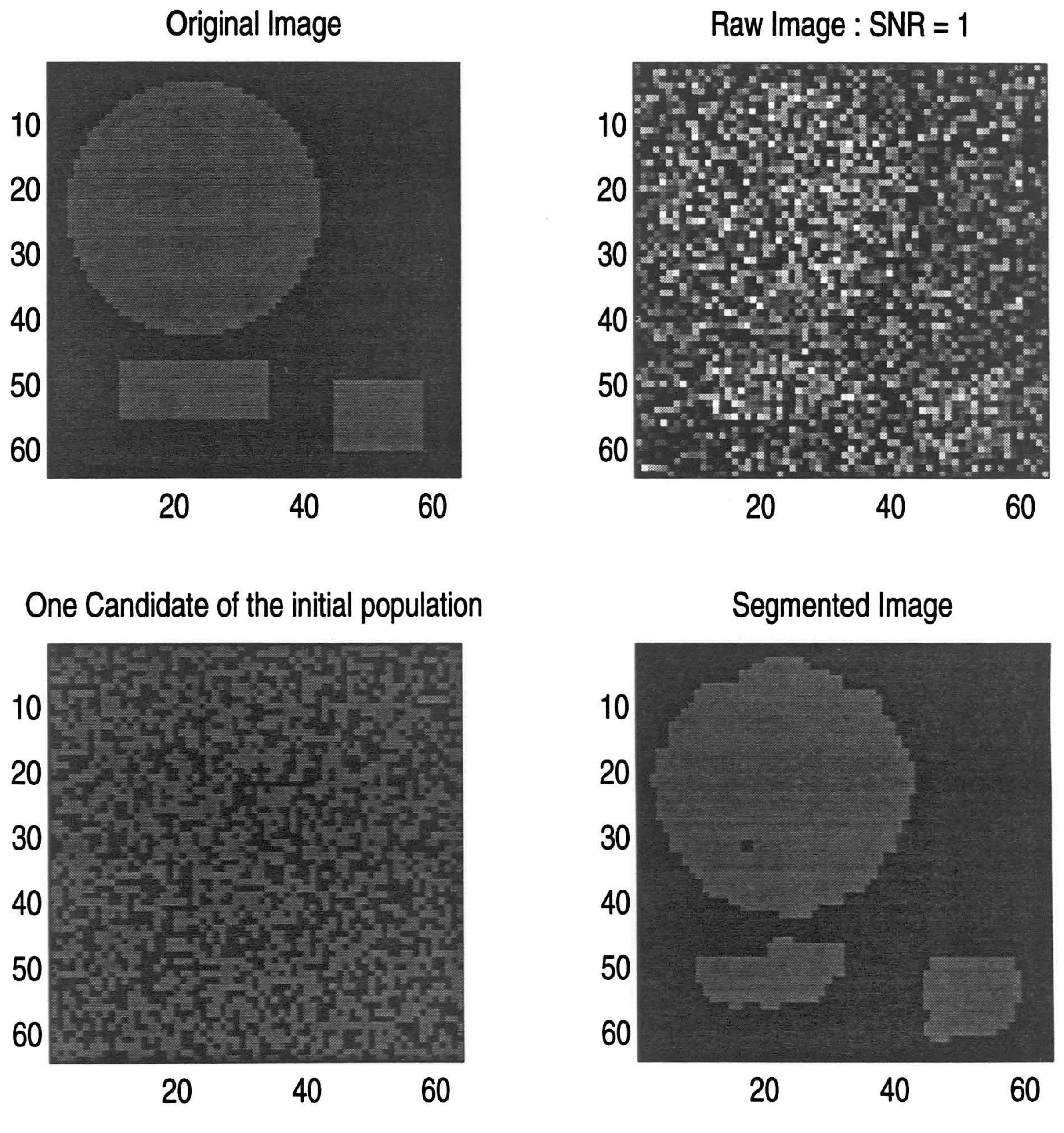
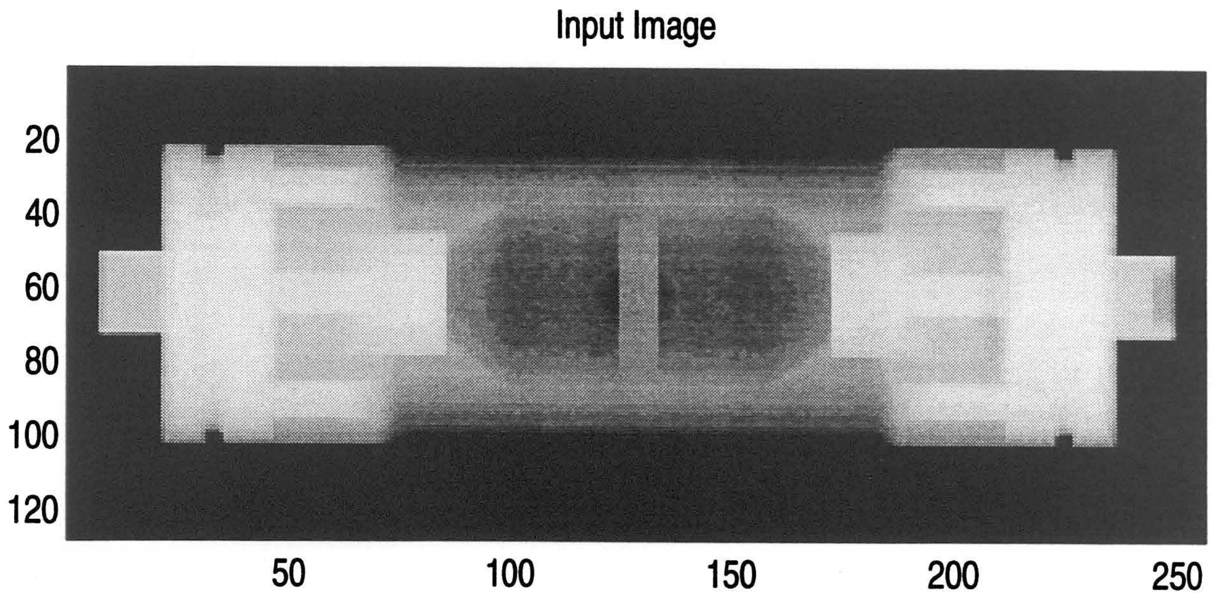


Figure 4.45: Results for 64 X 64 low contrast image: SNR = 1



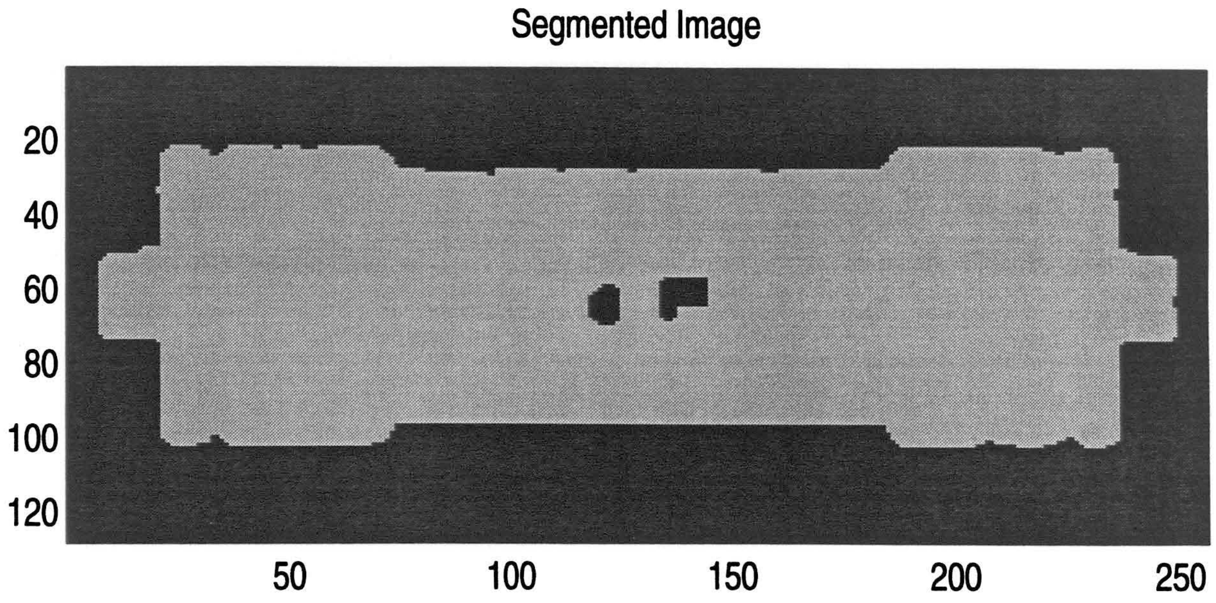


Figure 4.46: Flaw detection in an X-ray image

However, 184 corresponds to the brightest region in the image while 63 corresponds to the dark background. The flaw to be identified however is in the center, embedded in a very low contrast background and is barely visible to the human eye. The GA was executed with population size maintained at 100 while α was chosen to be 0.2. The initial population was generated randomly and the alternative method of crossover, in which a pair of parents are selected randomly from the mating pool to generate offsprings, was applied. The 8-pixel neighborhood mutation operator was employed. The segmented image obtained is a binary image with $Ro = 184$ and $Rb = 63$. The results, clearly demonstrate that the algorithm is successful in detecting the flaw, inspite of the low contrast.

CHAPTER 5. CONCLUSION AND FUTURE WORK

Summary

The goal of this thesis was to develop a novel Image Segmentation procedure using the genetic algorithm approach. Image Segmentation is first formulated as an optimization problem and the optimization is carried out using genetic algorithms. Genetic algorithms are based on the evolution theory which advocates the "survival of the fittest" principle. In the genetic algorithm approach, an initial population of individuals is required along with a measure of fitness to evaluate each individual. Only the fitter candidates survive and are able to produce offsprings. The offsprings so created tend to inherit the "better" features of their parents. The over all fitness of the entire population shows improvement over the generations until a point is reached when any one individual in the population is as good as another.

This approach has been successfully applied to the Image Segmentation application where the typical input images are noisy and the pixel intensity varies from 0 to 255. There is no restriction on the size of the image since the image is divided into subimages and each subimage is processed independently. Each subimage is represented as a string and the range of values are (0 - 255).

Two methods of generating the initial population of size N are discussed. Each candidate in the population is a string of the same size as the subimage, however it is a binary string having either object intensity *Ro* or background intensity *Rb*. In the first method, the individuals (strings) are made up of *Ro* and *Rb* randomly. The second approach uses a heuristic method based on the knowledge of noise variance in the image to generate the initial population. There are tradeoffs associated in using one method over another as discussed in chapter 3. In most of the literature, random methods of generating initial population is recommended and the experiments performed assert the same.

The fitness function describe in equation 3.2 is used to evaluate the merit of each candidate. It is composed of a similarity measure and a transition count. This kind of a fitness, weights a candidate with a homogeneous object that has a lower hamming distance with the subimage under consideration, over a candidate with a discontinuous object.

A criteria for selecting eligible individuals capable of reproduction needs to be defined. All candidates with fitness greater than the threshold θ, given by equation 3.5, are put together in a mating pool. The most critical step in the GA is the reproduction because it is responsible for evolution. Two methods of crossover are presented in this thesis. In one of the methods every individual is forced to mate with all the remaining individuals in the pool. In the other method a pair of parents is selected randomly to produce offsprings. The number of pairs of parents selected is again random and this method of reproduction resembles the way individuals find their partners in nature. In order to add some diversity to the population, the mutation operator is applied. A neighborhood is defined and the value of the pixel is flipped depending on the neighborhood. Mutation operators with 4-pixel and 8-pixel neighborhood are presented.

The next generation comprises of the best N candidates from the previous generations and their offsprings. In this way evolution takes place from one generation to another. The algorithm is terminated after a certain fixed number of generations or is made to run until convergence of the average fitness function is achieved.

Conclusion

An exhaustive study of the Image Segmentation procedure using genetic algorithm was performed. A number of parameters affect the performance of the algorithm, and a detailed study of these parameters was conducted. The following observations were concluded:

1. The algorithm proved to be successful for a subimage size of 8 X 8, but a subimage size of 16 X 16 subimage was optimal in terms of computational efficiency and accuracy of the results.

2. The random method of generating initial population performed well, and no substantial improvement was obtained either in terms of accuracy or convergence speed by using the heuristic approach.

3. The algorithm was terminated after a varying number of generations and the results show that the algorithm did not show significant improvement after 200 generations.

4. The algorithm performs more efficiently if the the algorithm is terminated on convergence rather than after a fixed number of generations.

5. The value of the weighting factor α in the fitness function given by equation 3.2 must be 0.2 or 0.3 typically, so that the solution is based more on the image data.

6. The optimal population size in each generation was found to be 100.

7. The method of crossover in which a pair of parents is selected randomly for reproduction has more appeal than the one in which every individual in the mating pool is made to reproduce with every other individual in the pool.

8. The mutation operator can be used with either 4-pixel or 8-pixel neighborhood. If images under consideration have very small objects to be identified, a 4-pixel neighborhood mutation operator gives better results.

9. The algorithm performs extremely well with a low signal to noise ratio of 1.

10. The performance of the algorithm on low contrast images is exceptionally good.

In short, the results have shown that the performance of the algorithm compare favorably with related techniques.

Future Work

Future work comprises of experimenting with:

1. different methods to generate initial population.

2. other fitness functions.

3. different selection criteria to select eligible individuals for the mating pool.

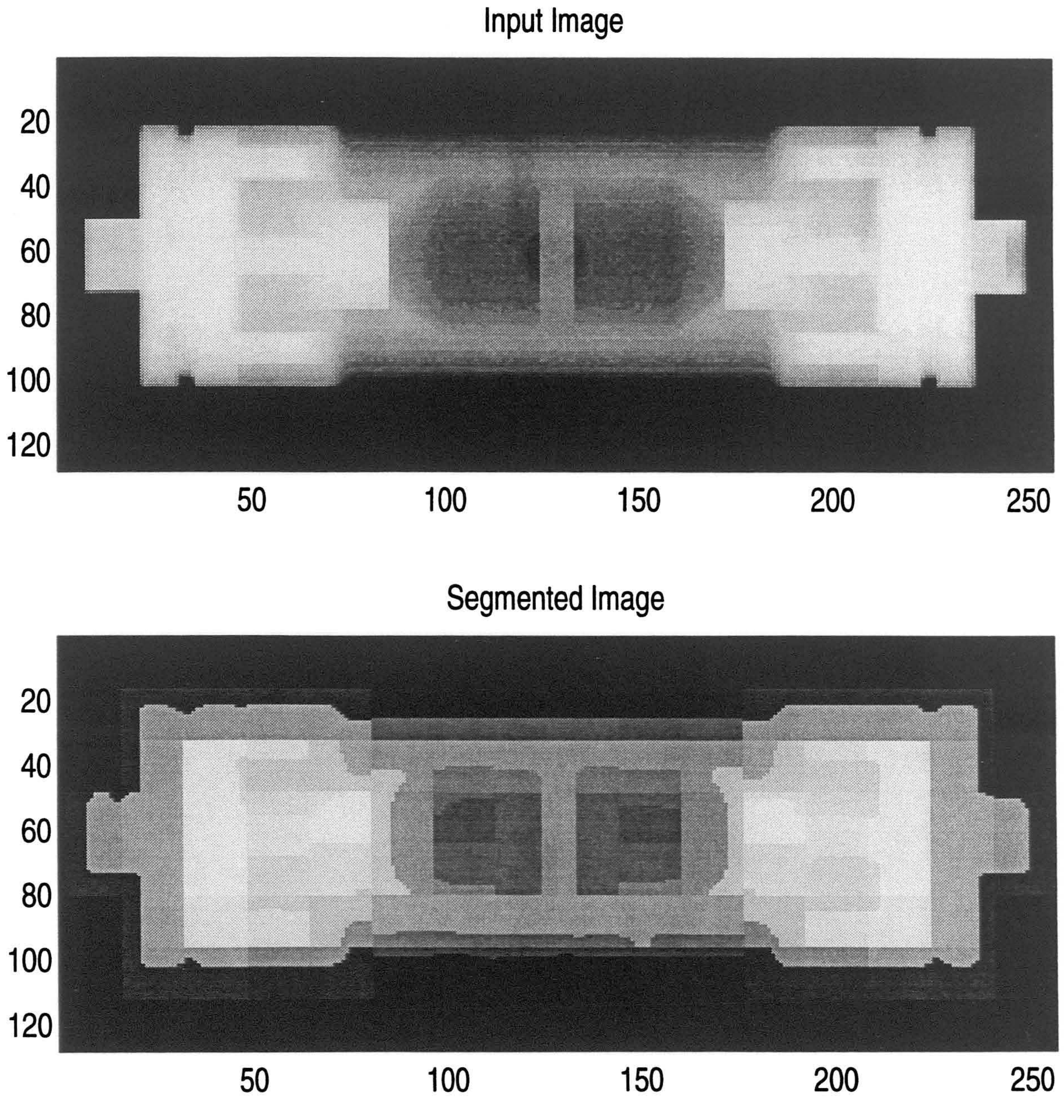


Figure 5.1: Results for multiple objects with different intensities

4. different crossover operations (e.g. multi-point crossover).

5. different neighborhoods for mutation operators.

6. different kinds of mutation operator.

7. a varying population size instead of a fixed one.

Future work in this area would be to apply the algorithm to practical images. As seen in chapter 4, the results of the X-ray image (Figure 4.46) show that the segmented image is binary. The original image has different features which cannot be seen in the output image. The reason for this is that the histogram of the entire image is considered to obtain the object intensity and the background intensity. The image can be interpreted as having multiple objects with different intensities. For every subimage, the histogram can be used to calculate the object and the background intensity. The *Ro* and *Rb* will be different for each subimage. The genetic algorithm is then applied to each subimage and the resulting segmented subimages are combined to get the entire image. Preliminary work for detecting multiple objects with different intensities has been performed. Figure 5.1 illustrates the results. As can be observed the algorithm does extremely well in detecting the various features of the specimen under inspection.

The automation of the entire procedure is of interest to many industries. The current implementation requires the user to provide the parameters like Ro and Rb. Methods of automatic estimation of these parameter have been studied [7]. The image segmentation procedure based on genetic algorithm can be linked with this method to make it fully automatic.

Finally, the intrinsic parallelism of the algorithm can be explored to make the implementation more effective. A parallel implementation of this algorithm can be done to increase the efficiency. Results obtained to date are very promising and further work studying the various aspects of the algorithm must be done.

BIBLIOGRAPHY

[1] Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press.

[2] Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, MA: Addison Wesley.

[3] Daviss, L. (1991). *Handbook of Genetic Algorithms.* NewYork: Van Nostrand Reinhold Publications.

[4] Buckles, B.P. and Petry, F.E. (1994). *Genetic Algorithms* Los Alamitos, CA: IEEE Computer Society Press.

[5] Grefenstette, J.J. (1987). Genetic Algorithms and their Applications. *Proceedings of the Second Internation Conference on Genetic Algorithms*: 89-92. Massachusetts Institute of Technology, Cambridge, MA.

[6] Forrest, S. (1993). *Genetic Algorithms: Principles of Natural Selection Applied to Computation.* Science, vol. 261: 104-108.

[7] Vyas, R.J. (1993). *A Fully Unsupervised Boundary Estimation Algorithm for Noisy Multiple Object Images* M.S. Thesis: Iowa State University, Ames, IA.