uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Abolfazl Keshtkar

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.C.S.

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Swarm Intelligence-Based Image Segmentation

TITRE DE LA THÈSE / TITLE OF THESIS

Wail Gueaieb

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

A. El Saddik

T. White

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Swarm Intelligence-Based Image Segmentation

by

Abolfazl Keshtkar

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.A.Sc degree in
Computer Science

School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-34078-3
Our file Notre référence
ISBN: 978-0-494-34078-3

Abstract

One of the major difficulties met in *image segmentation* lies in the varying degrees of homogeneousness of the different regions in a given image. Hence, it is more efficient to adopt adaptive threshold type methodologies to identify the regions in the images. Throughout the last decade, many image processing tools and techniques have emerged based on the former technology which we called *conventional* and new technologies such as *intelligent-based* image processing techniques and algorithm. In some cases, a combination of both technologies is adapted to form a hybrid image processing technique.

Intelligent-based techniques are increasing nowadays. Due to the rapid growth of agent-based technology's environments which are adopting numerous agent-based applications, tools, models and softwares to enhance and improve the quality of the agent-based approach. In case of intelligent techniques to doing image processing; swarm intelligence techniques rarely have been used in term of image segmentation or boundary detection. However, there are many factors that make this task challenging. These factors include not only the limited such increasing number of agents in the environment, and the presence of techniques, but also how to efficiently find the right threshold in the image, develop a flexible design, and fully autonomous system that support different platform. A flexible architecture and tools need to be defined that overcomes these problems and permits a smooth and valuable image processing based on these new techniques in image processing. It would satisfy the needs of end users. This thesis illustrates the theoretical background, design, swarm based intelligent techniques and implementation of a fully agent-based model system that is called SIBIS (Swarm Intelligent Based Image Segmentation).

Acknowledgments

My sincere gratitude goes to my academic supervisor Dr. Wail Gueaieb, who helped, encouraged, and guided me toward my academic as well as my personal success. The thesis would not be successful without enormous help and advice from him. He provided me a substantial amount of help and suggestions throughout my work. Special gratitude also goes to fellow students of the Signal Processing Oriented Technologies (SPOT) and Multimedia Communication Research Laboratory (MCRLab) team for many helpful discussions and their constant encouragement and support.

I would like to give my special thanks to Dr. A. El Saddik, Dr. Tony White, and Dr. Oliver Yang for attending my Thesis defence and their great advices. Also special thanks go to Dr. Uri Wilensky for providing Netlogo software in the Center for Connected Learning (CCL) and Computer-Based Modeling at Northwestern University, Evanston, Chicago, IL.

I would like to give my sincere thanks to my beloved parents for their support throughout my graduate studies. I am also grateful to all my friends and family who helped me either with suggestions or with positive affirmation.

Contents

1 Introduction **1**
1.1 Image Processing . . . 1
1.2 Thesis Objectives . . . 2
1.3 Thesis Contribution . . . 2
1.4 Thesis Organization . . . 3

2 Overview on Image Segmentation **4**
2.1 Introduction . . . 4
2.2 Conventional Image Segmentation Methods . . . 5
2.2.1 Pixel-based Segmentation . . . 5
2.2.2 Edge-based Segmentation . . . 5
2.2.3 Region-based Segmentation . . . 6
2.2.4 Image Segmentation by Curve Evolution on Natural Images . . . 7
2.2.5 Watershed Segmentation . . . 7
2.3 Fuzzy Logic-Based Image Segmentation . . . 8
2.3.1 Multiresolution Threshold Selection Method Based on Training . 9
2.3.2 Image Segmentation by Finite Mixture and Spatial Information . 9
2.3.3 Image Segmentation by Robust Clustering Algorithm . . . 10
2.4 Agent-Based Image Segmentation . . . 12
2.4.1 Collaborative Multi-Agent Image Segmentation . . . 13
2.4.2 Distributed Image Segmentation System by a Multi-Agents Approach 13
2.4.3 A Multi-Agent Based System for Parallel Image Processing . . . 14
2.4.4 Adaptive Image Segmentation with Behavior-Based Agents . . . 14
2.4.5 An Algorithm for Swarm-Based Color Image Segmentation . . . 15
2.5 Conclusion . . . 15

3 Swarm Intelligence and Cellular Automata **16**
3.1 Introduction . . . 16

3.2 Cellular Automata 18
3.2.1 Mathematical Model 19
3.2.2 Local Transition Function and Rule 19
3.2.3 Two Dimensional Cellular Automata: 2D lattice 22
3.2.4 Applying Cellular Automata in an Agent-Based Algorithm 22
3.3 Conclusion 24

4 Swarm Intelligence Based Image Segmentation (SIBIS) 25
4.1 Introduction 25
4.2 Proposed Approach 26
4.2.1 Finding the Threshold 26
4.2.2 Agents Local Behavior 27
4.2.3 Marking Contours of the Labeled Pixels 32
4.2.4 Segmentation/Boundary-Detection Part 32
4.3 Conclusion 33

5 Results and Evaluation 34
5.1 Introduction 34
5.2 Evaluation Methodology 36
5.2.1 An Example for Precision-Recall and F-measure 38
5.3 Segmentation of Natural Images 39
5.3.1 Comparison of proposed algorithm with Human Segmentation . . 43
5.4 Segmentation of Radiographic Images 43
5.5 Segmentation of Brain Images 49
5.6 Conclusion 50

6 Conclusion and Future Work 55
6.1 Conclusion 55
6.2 Future Work 55

A SIBIS Implementation 57
A.1 Introduction 57
A.2 Netlogo Environment 58
A.3 Clear/Reset: Environment, Agents and Variables 60
A.4 Import Images 61
A.5 Setting the System 61
A.6 Segmentation Part 63

A.7 Boundary Detection Part 64

List of Tables

5.1 Comparison of human images. 46
5.2 Testing SIBIS with the three radiographs 49
5.3 Testing SIBIS with two brain images 50

List of Figures

2.1 Curve Evolution 8
2.2 (a) Original Image 'hand'.(b) Segmentation obtained 11
2.3 Segmentation on Lena classic image. 12

3.1 Segmentation of image 20
3.2 Segmentation of image 21
3.3 Segmentation of image 21
3.4 Segmentation of image 23

4.1 Block diagram of the proposed agent-based algorithm (SIBIS) 25
4.2 5 × 5 original sample image. 27
4.3 Finding the threshold for a sample image. 28
4.4 Local neighboring region of an agent 29
4.5 Sample image after Searching and Labeling step. 32
4.6 Sample image after marking letter. 33

5.1 Segmentation of image 35
5.2 Segmentation of image 36
5.3 Computing the F-measure using a Precision-Recall curve. 39
5.4 Segmentation of a two horse-image. 40
5.5 Segmentation of image 41
5.6 Segmentation of various images 42
5.7 Segmentation of image 44
5.8 Segmentation of Radigraphic image 45
5.9 Segmentation of radiographic image 47
5.10 Segmentation of radiographic image 48
5.11 Segmentation of image 51
5.12 Segmentation of image 52
5.13 Segmentation of brain images 53

5.14 Segmentation of image 54

A.1 SIBIS system with default values before processing the image 59
A.2 SIBIS system with default values after importing 59
A.3 SIBIS system with default values after processing 60

Glossary of Terms

ACO: Ant Colony Optimization.

ANFIS: Adaptive-Network-based Fuzzy Inference Systems.

BPP: Back Propagation Algorithm.

CA: Cellular Automata.

C: Component.

E-Health: Electronic Health.

EM: Expectation Maximization.

FSS: Fuzzy Supervising System.

Max: Maximum.

MCRLab: Multimedia Communication Laboratory.

Min: Minimum

P: Precision.

PCM: Possibilistic Component means.

PET: Position Emission Tomography.

PR: Precision-Recall.

PSO: Particle Swarm Optimization.

R: Recall.

RGCS: Robust Gaussian Clustering/System

ROC: Receiver Operating Characteristic.

SIBIS: Swarm Intelligent Based Image Segmentation.

SPOT: Signal Processing Oriented Technology.

Std: Standard Deviation.

Thr: Threshold.

Chapter 1

Introduction

1.1 Image Processing

Image processing is a term used to describe operations carried out on images with the aim of accomplishing a particular purpose [33]. This may be a very general definition and certainly too vague to answer the question 'what is image processing'? However, the lack of precision helps to highlight the difficulty of rigorously defining a set of applications with fuzzy boundaries. Image processing, like many other disciplines, is not an exclusive set. Awcock and Thomas [9] argue that the term image processing itself has become firmly associated with the limited objective of modifying images such that they are either: (a) corrected for errors introduced during acquisition or transmission (restoration); or (b) enhanced to overcome the weakness of the human visual system (enhancement). As such, the discipline of pure image processing may be succinctly summarized as being concerned with a process which takes an image input and generates a modified output image. Essential topics of image processing include image compression, image restoration, image enhancement, and image segmentation.

Image segmentation has a rich history in image processing research. Many different techniques that work on various image features such as brightness, color, and texture have been proposed. One aspect that makes image segmentation complexed ill-posed is its domain and application dependence. For natural images, a combination of gray scale, color and textural features are credited to provide a satisfactory performance. However, for aerial and remotely sensed imagery, textural features often lead to a good performance [34]. For other domains, such as medical images, or images of man-made objects, gray scale intensities are usually sufficient.

1.2 Thesis Objectives

The main objective of this thesis is to derive a swarm intelligent technique for the automatic segmentation of a large variety of images regardless of their nature. The main challenge faced here is that in many cases, the different segments in an image tend to be more or less homogeneous and their boundaries may be quiet fuzzy. In addition, the problem of image segmentation is subjective by nature. Different people might have different views of how an image should be segmented. Moreover, some regions might be too small within an image, which makes it difficult to decide on how to deal with them. Swarm intelligence is an emergency research field in the area of soft computing. It offers several advantages, probably the most important of which, is a distinguished intelligence framework that enables a swarm of particles to interact with a common environment and share their own knowledge while maintaining a certain degree of autonomy.

1.3 Thesis Contribution

The main contribution of this thesis is the design and development of a Swarm Intelligence based Image Segmentation (SIBIS) algorithm. In addition to segmenting and detecting the boundaries of an image, SIBIS adopts a cellular automata technique as a novel methodology to lead the swarm of agents to navigate through the image and operate on their pixels and local regions.

The three features such as swarm intelligence, agent-based modeling, and cellular automata are integrated together to make SIBIS more efficient. Those features are designed and developed in a fully agent-based environment that is called "Netlogo". The specialty of the SIBIS system is that it can find the image segmentation threshold automatically. SIBIS is designed in such a way that it finds the boundaries with a high accuracy without changing the background or the texture of the image. This is an advantage of SIBIS that is not found in many other image segmentation techniques. It should be mentioned here that this work is an extension of a similar graduate course project [22, 38]. However, in that project, a Markov model technique is used to control the behavior of the agent, whereas we are using cellular automata in this work. Also, the majority of the experiments we ran were conducted specifically for this thesis.

1.4 Thesis Organization

The rest of the thesis is organized as follows: In Chapter 2, the background of the thesis and related work about the development of conventional and agent-based image segmentation systems are introduced. In Chapter 3, the theoretical background of swarm intelligence, cellular automata, and agent-based model are detailed. In Chapter 4, the proposed swarm intelligence based image segmentation (SIBIS) algorithm and system design are presented. Chapter 5 provides the measurement data that we have found along with a comparison of features with other tools mentioned in Chapters 3 and 4. The thesis is brought to closure with some concluding remarks and potential future research directions.

Chapter 2

Overview on Image Segmentation

2.1 Introduction

In general, the problem of image segmentation mostly deals with detecting discontinuities, identifying boundary contours, and generating morphological. Image segmentation has a rich history in image processing research and it has been garnering increasing attention from image processing researchers due to its vital role in a number of applications, such as in the medical and biomedical sectors. Gray image segmentation is usually accomplished by applying certain types of transformations to the pixel's intensity. Color image segmentation algorithms, on the other hand, act on the pixel's intensity and color. Alternative techniques have been proposed to operate on different features of the image, such as brightness, color, and texture, to name a few. Image segmentation, in general, is characterized by its high complexity and application-dependency. Due to these reasons and to overcome such problems, several researchers based their image segmentation techniques on intelligent and heuristic algorithms. For instance, a combination of gray scale, color, and texture, has been shown to provide a satisfactory performance on natural images. However, aerial and remotely sensed images tend to be better segmented using textural features [34]. Over the years, researchers proposed several types of techniques to solve the automatic image segmentation problem. Although an extensive survey of the field is beyond the scope of this work, we present the following brief overview of the main classes of image segmentation algorithms proposed in the literature.

In image processing it is necessary to check each individual pixel to see whether it belongs to an object of interest or not. This operation is called *segmentation* and produces a *binary image* [19]. In this case a pixel has a value of one if it belongs to an object, otherwise its value is zero. Also, segmentation lies at the boundary of *low-level image* *processing* and *image analysis*. So, after segmentation is done we can distinguish the boundaries between the regions.

2.2 Conventional Image Segmentation Methods

In this section we introduce some of the traditional methods and algorithms that have been recently used in image processing and especially image segmentation. Some algorithms that introduce better solutions have rigorous computations. As we know Image processing itself has many internal computations. Hence, if algorithms have increased computations, obviously those methods become useless in real time or learning systems. It is evident that traditional image segmentation methods became out of date in image processing due to new techniques such as fuzzy and neural network methods which we introduce in the next section. In the following sections we will mention and present some of the conventional image segmentation algorithms that have been introduced by researchers in recent years.

2.2.1 Pixel-based Segmentation

The Pixel-based approach is one of the simplest methods that can be applied to image segmentation. The rationale behind this technique is to simply solve the problem from the root. In this method, usually a fixed threshold is used to find the intensity of the pixel based on the fixed threshold. In Pixel-based segmentation, if both the background and the object are uniform we can find an optimum threshold since it shows the uniform gray values. On the other hand, if the background and the object are not uniform, finding an optimum threshold is pointless. This threshold can not operate well to find the segment or objects in the image, due to variety in the background and the object.

2.2.2 Edge-based Segmentation

The problem with pixel-based segmentation appears even with perfect illumination. Pixel-based segmentation results affect the size of the segmentation objects when the objects show variation in their gray values. Darker objects become too small while brighter objects become too large [19]. So the edge-based methods are only possible if all objects show the same gray values or if they apply different thresholds for each object.

Edge-based segmentation uses various segmentation techniques. Some of the techniques such as a non-uniform background and object brightness are as follows:

- Bias by Uneven Illumination: in this technique an edge is blurred by a point spread function $h(x)$ such that;

$$\int_{-\infty}^{\infty} h(x) = 1, \tag{2.1}$$

with x being the edge values in the image.

- Edge Tracking: Edge-based Segmentation is a sequential method. So, each step depends on its predecessor. The algorithm's general form is as follows:

  1. Image scanned to find maximum brightness values
  2. A maximum is encountered
  3. Applying an algorithm which tries to follow the maximum value in part 2 around the object
  4. Then next search begins for next maximum

2.2.3 Region-based Segmentation

In this type of segmentation a small area of pixels can be classified as object pixels. In this approach, instead of the original image, it is the features of the image that are processed. These features represent not only a single pixel but also a small neighborhood, depending on the mask size of operations that are used. The correct procedure is to limit the mask size at the edge to the point of either object or the background. To improve the result in region-based segmentation algorithm, we can repeat the computation and segmentation until the procedure holds a stable result.

The *Pyramid Algorithm* is a popular region-based segmentation method. It includes the following steps:

- Computing the Gaussian Pyramid
- Segmentation by Pyramid linking
- Average of Linked-Pixels

Basic Formulation

Let R represent the image region. We may view segmentation as a process that partitions R into n subregions, $R_1, R_2, ..., R_n$ such that

(a) $\bigcup_{i=1}^{n} = R$

(b) R_i is a connected region where,$i = 1, 2, ..., n$.

(c) $R_i \bigcap R_j = \oslash$ for all i and $j, i \neq j$
(d) $P(R_i) = \ TRUE\ for\ i = \ 1, 2, ..., n.$
(e) $P(R_i \bigcup R_j) = \ FALSE\ for\ i \neq j.$
Here, $P(R_i)$ is a logical predicate defined over the points in set R_i and $\oslash$ is the null set. Condition (a) indicates that the segmentation must be complete (every pixel must be in a region). Condition (b) requires that the points in a region must be connected in some predefined sense. Condition (c) indicates that the regions must be disjoint. Condition (d) deals with the properties that must be satisfied by pixels in the segmentation region, and finally Condition (e) indicates that regions R_i and R_j are different in the sense of predicate P [16].

Edge and region-based approaches use closed contours to separate the different objects in the image from their backgrounds [11, 12]. This type of technique relies on mathematical formulas to define the gray level crossing thresholds between the different regions of the image. Most of such techniques have to be preceded by a prefiltering of the image, which may result in a loss of important information. In addition, such methods are also known to be susceptible to noise, and hence, can easily lead to poor quality contours. Another class of image segmentation techniques is based on Markov random fields [15, 17]. In this case, a Bayesian model is used to maximize a posteriori probability of the segmentation. Although this type of approach has been credited for its high performance in many cases, it still suffers from several limitations. Determining the optimal parameters associated with strength of the spacial interactions and the high computational complexity remain among the top challenges of these methods.

2.2.4 Image Segmentation by Curve Evolution on Natural Images

Flow-Field based segmentation methods have also been proposed within the last decade. The Edgeflow has been shown to work better on many real images [34]. It directly generates edges from the image, as follows: An estimate is measured for the best direction to the closest boundary at a specified spatial scale. It is selected as direction of the vectors. Then the directional gradient of the smoothed image is used as the vector magnitudes. Figure 2.1 shows the result on the cell by this algorithm.

2.2.5 Watershed Segmentation

Another method that has been quite popular within the last years is Watershed segmentation [28, 35]. This method uses a morphological region growing over time to partition

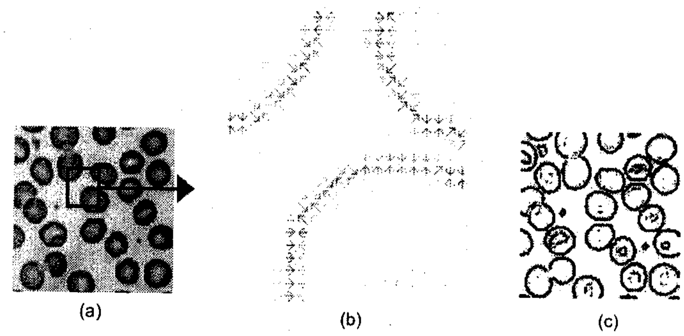


Figure 2.1: Curve Evolution
(a) Image of blood cells. (b) Edgeflow vector field corresponding to the rectangle on the image. (c) Edge function of the image generated from edgeflow vectors [34].

an image. The algorithm considers gradient information $g(x, y)$ about an image. The boundaries are found as the watershed lines if $g(x, y)$ is considered as mountain landscape and if it is flooded at certain markers. The main advantage for *watershed* methods have been their simplicity and computation efficiency. On the other hand selecting the markers' location and the number of markers present challenges. If the markers are chosen as the local minimum of the $g(x, y)$, then an over-segmentation is inevitable. Region merging or scale methods usually create a better segmentation. Watershed growing is not as flexible as curve evolution, since growth in watershed is only one-dimensional. Another problem in the watershed method is the difficulty to enforce constraints such as boundary smoothness. Most segmentation results by the watershed method show considerable noise in boundaries.

2.3 Fuzzy Logic-Based Image Segmentation

In this section we describe some fuzzy logic-based methods, algorithms, systems, and applications that have been recently used in image processing and especially image segmentation. During our research we discovered where image processing techniques stand and how much improvement they underwent during the last decade. As we mentioned in Section 2.2, the fuzzy and neural network methods became more powerful than conventional methods. The reason might be clear, since fuzzy and neural networks methods implement intelligent techniques to operate on the image . In the following we present a number of these algorithms.

2.3.1 Multiresolution Threshold Selection Method Based on Training

Martinez and Ollero [13] have presented a new training-based threshold selection method for gray level images. It is one of the hardships that they would like to overcome and improve this algorithm. They proposed a new multiresolution decomposition of image histogram using fuzzy system. In this method the adapted application extracts knowledge of a specific application from a set of training images. The system utilizes neuron-fuzzy training mechanism known as ANFIS (Adaptive-Network-based Fuzzy Inference Systems) [20]. The main contribution of this method referring to existing systems is that: It relies on object/background separability criteria learned for a specific application, while existing techniques rely on generic criteria. The method's performance exhibits considerable robustness to illumination conditions and noise in the image. Lack of capacity of adaptation in specific vision application is one of the main limitations of the proposed method.

The main part of this system is the FSS (Fuzzy Supervising System). FSS training is used to extract knowledge from a set of training images of a vision application and to introduce FSS by means of a training process. FSS can be divided into a pair set generation and fuzzy identification. A disadvantage of this system is that, the investigators have used fuzzy identification to approximate input and output pairs that contain knowledge of specific application as it is the computationally expensive.

2.3.2 Image Segmentation by Finite Mixture and Spatial Information

The finite mixture is a flexible and powerful modeling tool. In this model Xiang et al. [43] used the spatial information as a prior knowledge of the number of components. Spatial knowledge provides some indirect information about the value, instead of the value of the number of components. An expectation maximization (EM) algorithm is used to estimate mixture density using the indirect information.

Xiang et al. [43] claim that their algorithm is able to estimate the number of components without using any model selection criteria. They show by results that the algorithm has better performance in generating a small region in comparison with the same model. In the EM-based algorithm Xiang et al. supposed an unknown number of components, so they estimated the value of C (Component). The finite mixture design has the form:

$$f(x, \theta) = \sum_{j=1}^{c} \pi_j f_j(x, \psi_j). \tag{2.2}$$

Where p_i is the probability for component j, and f_j is finite mixture. When the number of components is unknown, but some indirect information about the component is provided, and two data points originate from the same component. In the EM-based method, the algorithm of density estimate for Gaussian mixtures is considered as follows:

(1) Choose a large initial value for C.

(2) Initialize means for all components by using the k-means algorithm.

(3) Use all the data points and their corresponding peers to update mixture parameters.

(4) Check the mixing probabilistic.

(5) Go to step 3 until convergence.

When the algorithm uses the peers of each pixel:

(1) Find out the edge of the input image.

(2) The neighbors of each pixel are considered its peers. If they are located at the boundary, then they are not considered in the mixture density estimation. (a) One of the advantages of this algorithm, is its ability to be applied in color image segmentation as well as gray level images; and it finds the smooth regions as well (See Figure 2.2). One important disadvantage of this algorithm is that a large value of C often leads to a slow convergence rate. Thus for images that are more complex and have many components, this algorithm may not be suitable.

2.3.3 Image Segmentation by Robust Clustering Algorithm

Wang et al [36] presented a novel clustering-based image segmentation method. They used Fuzzy Clustering to overcome the sensitivity to noise. They introduced the Gaussian estimator to clustering analysis as a weight or membership function. The Robust Gaussian Clustering/System (RGCS) algorithm represents more reasonable pixel classification and noise performance when applied to Image Analysis. They claim that by using a uniform resolution parameters scheme, this method avoids producing coincident clusters in clustering segmentation. It also reduces the influence of a large residual on the estimate. By using this method in image segmentation they reduce coincident clusters problems. This problem usually arises in PCM (Possibilistic C Means) based segmentation. The RGCS consists of the following parts:

(1) Robust Statistics Features

This part characterizes some feature of the algorithm.

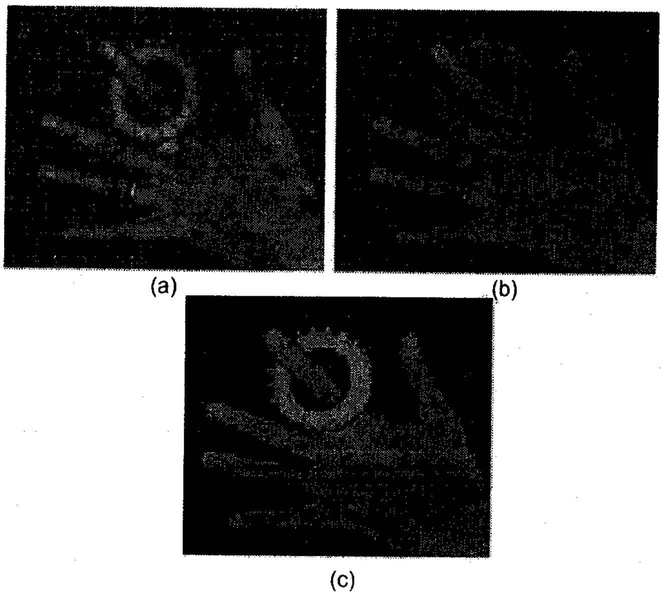


Figure 2.2: (a) Original Image 'hand'.(b) Segmentation obtained by Figueiredo's Algorithm. with estimated value of C=11. c) Segmentation obtained by Yang et al. Algorithm with estimated value of C=8.

(2) Robust Clustering based on M-estimator.
This part provides a relation between robust statistics and fuzzy clustering. The M-estimator is defined by the following function:

$$J(\theta) = \sum_{j=1}^{N} \rho(r_j), \tag{2.3}$$

where θ is a parameter of estimation, N is the number of observations, r_j is residual, (i.e $r_j = x_j - \theta$) and ρ is M-estimator of r.
(3) Robust Gaussian Clustering/Segmentation
To estimate C-Clusters, it may be implemented by a collection of C independent M-estimators that work simultaneously.
The problem is how to choose the M-estimator to achieve counter estimation. To overcome this problem Wang et al. used a Gaussian estimator in clustering analysis (see Figure 2.3).

In comparison with Xiang et al. [43], the algorithm presented by Wang et al. [36] is more sensitive to noise, but Xiang et al. algorithm [43] makes segmentation of the image smoother than Wang et al. algorithm [36].

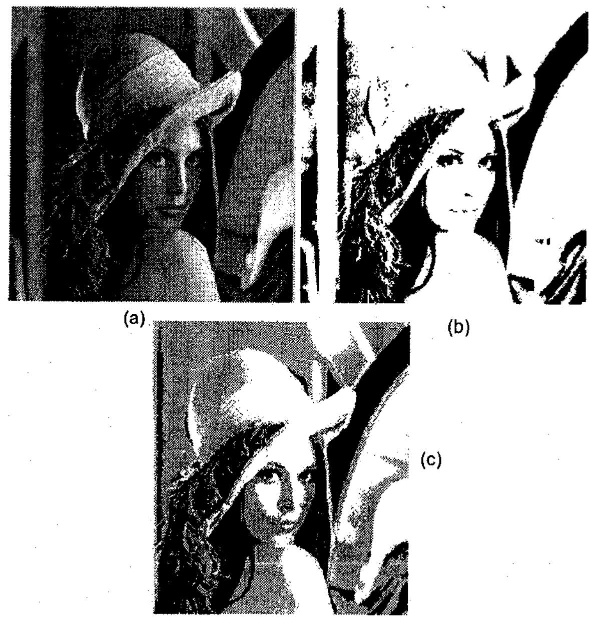


Figure 2.3: Segmentation on Lena classic image.
(a) Lena Image. (b) Result by PCM. (c) Result by RGCS [36].

2.4 Agent-Based Image Segmentation

Biologically inspired techniques have recently drawn the attention of a number of researchers thanks to their credibility in solving complex ill-defined problems. They are mainly inspired by the way humans and animals tend to analyze complex and ambiguous real-life situations without the need for precise apriori known mathematical models. Liu et al. [24] presented an autonomous swarm intelligence-based image segmentation approach. The proposed methodology models a digital image as a two-dimensional cellular environment. A swarm of autonomous agents is then used to scan the environment (image) and label homogeneous segments. In doing so, the agents rely on some reactive behaviors such as breeding and diffusion. The agents that are successful in finding the pixels of a specific neighboring region produce offspring agents to fine scan that particular region to find more homogeneous-segment pixels. In the mean time, the life of unsuccessful agents is put to an end. This approach has been applied to gray scale images of a brain scan and found to be reliable, easy to represent and implement, and adaptive to the local distribution of agents. White et al. [39] introduced an agent-based model for color image segmentation. The algorithm uses image pixel and a corresponding segment map to form a context in which segmentation can occur. The algorithm aims at finding spacial interactions between segments of similar pixels. A novel agent-based algorithm was proposed in [22]. The study explores new swarm intelligence techniques that mimic ant colony behavior in using both scouts and workers to segment digital images. The task of scouts is to search and label similar pixels in each neighborhood, while worker ants group segments with the same labels.

2.4.1 Collaborative Multi-Agent Image Segmentation

In this study, researchers [7] describe a multi-agent segmentation system in which asynchronous agents combine low-level image processing with high-level reasoning agents. Each does image processing and communicates and collaborates with other agents to resolve conflicts with a mutual goal of coming to a consistent overall interpretation of images. In this system, agents adjust their behavior and control the image processing depending on image content. In this model they designed and implemented an image processing method. The agent knowledge model contains over 300 rules. These rules consist of: how to do image processing (139 Rules), how to communicate (33 rules), how to resolve conflicts (100 rules), and general utilities and problem solving (26 rules). Most of the work is concerned with how agents communicate and how to avoid blocking in their communication while they perform image processing tasks.

2.4.2 Distributed Image Segmentation System by a Multi-Agents Approach

This work proposed an image segmentation system using multiple-agents [1]. A new segmentation system is obtained by the co-operate of robust edge detector of canny type and region growing methods. A multi-agent approach is provided to increase flexibility within the system. Each agent is responsible for a precise task and its interaction with other agents. The rules of agents in this system are as follows:

(*a*) Consists of 16 agents.

(*b*) Each agent is an executable file.

(*c*) Agents execute on several machines and process independently.

(*d*) Agents communicate with each other by passing messages through the network.

There are some issues in this model, *i.e.*; (1) agents only use agent-based concepts to keep communication between agents and tasks; (2) deadlock and other emerging problems in the system; (3) agents are not able to perform segmentation and processing on images.

2.4.3 A Multi-Agent Based System for Parallel Image Processing

In [25], researchers use the concept of agents to create cooperation and for using the agents ability to learn and allow long term improvement of its planning. They showed how image analysis tasks can be modeled with respect to their parallelization. In this model they used two basic module operators for time sharing, planning and performing of tasks. They did not utilize agent-based concepts to perform image segmentation. Instead they applied agent-based methods in order to decrease complexity and used time sharing to obtain parallelization.

2.4.4 Adaptive Image Segmentation with Behavior-Based Agents

In this work an autonomous agent-based image segmentation approach [24] is presented. In this approach, a digital image is considered as a two-dimensional cellular environment which the agents inhabit in order to label homogeneous pixels. To do so, the agents rely on some reactive behaviors such as breeding and diffusion. The agents that are successful in finding the pixels of a specific homogeneous segment will breed offspring agents inside their neighboring regions. Another behavior is introduced for agents and their offspring, hence, the offspring agents will become likely to find more homogeneous-segment pixels. When agents become unsuccessful they die or become inactive. In their algorithm the investigators define some local stimuli for agents, such as: *Contrast-criterion, mean-criterion, and std-criterion.*

After defining local stimuli, the researchers introduce important roles for agents such as: *Breeding and Diffusion and Labeling the pixels.* In breeding, while a specific agent finds a homogeneous pixel in its region, it breeds more offspring agents depending on the number of homogeneous pixels in the region. One of the main problems of this algorithm is the breeding of many offspring during the labeling of homogeneous pixels. This serves to increase the complexity of the algorithm. Additionally it is not developed in an agent-based environment. The agents do not have the ability of sensing their neighbors, so they have to search to find homogeneous pixels in their neighbors. However, the agent-based approach has been used for image segmentation.

2.4.5 An Algorithm for Swarm-Based Color Image Segmentation

Segmentation in a color image is one of the most difficult tasks in digital image processing. In this research, investigators propose a Swarm-based concept for image segmentation [39]. They applied a swarm intelligence algorithm to image pixel data and a corresponding segment map. After emerging the two approaches they connected similar pixels found in the image. They used a data structure to present their work, i.e. two matrices of which one belongs to image and environment that refer to a specific location of with image matrix. The matrix represents the segment map. They defined the swarm to consist of two specialized types of agents referred to as *scouts* and *workers*. This approach was inspired by colonies of ants that act as a group of workers. The role of *scouts* in this algorithm is to find pixels of similar color and mark their location in the segment map. The second agent that they have used in their algorithm is *workers*. The function of workers is to consolidate pixels of similar color, as identified by scouts. Workers move randomly to find pixels corresponding to the segment map.

2.5 Conclusion

In this chapter, we enumerated a number of image classification techniques in three classes: Conventional, fuzzy-logic-based, and agent-based algorithms. We also identified their main advantages and disadvantages in digital image segmentation. Among the major difficulties encountered in the automatic segmentation of a given image is that regions may be quite homogeneous. Thus, setting a unique predefined threshold for the whole image may not be an effective strategy. To overcome this challenge, we propose applying different operators to different regions, catering to the specific changes in the average region intensity gradients. This is accomplished through an intelligent multi-agent-based approach in which each agent is triggered only by a desirable homogeneous segment of the image. The swarm intelligence algorithm also tries to find similar pixels based on a sensor function, which is then used by swarm agents to determine the next appreciate pixel in the region/segment area. We also introduce a cellular automata-based dynamic flow algorithm to guide swarm agents to choose the best possible advancing direction to avoid traffic jams and inconsistency. The proposed image segmentation algorithm is tested on a set of dental radiographs.

Chapter 3

Swarm Intelligence and Cellular Automata

3.1 Introduction

Swarm intelligence is a class of biologically inspired computational models that exploit the processing power of a swarm of entities, where each entity has a specific set of behaviors. A swarm has been defined as a set of agents that are able to communicate directly or indirectly with each other by acting on their local environment. These swarms are engaged in distributed problem solving based on an infinitely complicated web of semitic interaction patterns. Agent-based algorithms are generally composed of three main components: the agents, the environment, and the rules [14]. Agents are the individual entities acting according to certain internal states of behavioral rules. Some states are fixed for the agent's life while others change through the interaction with other agents or with the external environment. The environment is the medium over which agents operate and interact. It can also be a more abstract structure, such as a communication network or digital image, for instance. The rules represent the law governing the agents' action and behavior within the environment. A possible rule can be for instance 'Look around as far as possible, find the homogeneous pixels, and label the pixels'. In real-world applications, autonomous agents are often used to solve specific tasks in a distributed fashion [30]. The agents interact with their environments in the course of problem-solving. Agents can select and exhibit different behavioral patterns. In their environments agents can have different behavioral patterns.

Swarm intelligence techniques draw their strength from several sources: they are computationally inexpensive, decentralized, adaptive, and are not based on an explicit model of the environment. These advantages led the image processing community to show an increasing interest in adopting different types of agent-based paradigms in digital image segmentation [1,7,21,22,24,25,27,32,39]. Among the most appealing features of Swarm Intelligence are:

- Distributed functionality, no control or data source
- No explicit model of environment
- Sentience of environment
- Ability to easily change environment
- Full autonomy
- Strong reaction and adaptation

Two of the most successful swarm intelligence techniques currently in existence are Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). ACO is a meta heuristic optimization algorithm that can be used to find approximate solutions to complex combinatorial optimization problems. In ACO paradigms, artificial ants are used to build solutions by moving on the problem graph, mimicking real ants in depositing artificial pheromone on the graph in such a way that future artificial ants can build better solutions. ACO-based algorithms have been successfully applied to an impressive number of optimization problems. Some ACO-based image segmentation algorithms are inspired by the roles of the different types of ants such as scouts and workers, in an ant colony in searching for food. The role of scouts is set to find pixels of similar color and mark their location in the segment map. Then, workers consolidate pixels of similar color, which were previously identified by scouts [22,39]. PSO is a global minimization technique for dealing with problems in which a best solution can be represented as a point or surface in an n-dimensional space [40]. Over time, particles are accelerated toward those points within their communication grouping which have better fitness values. The main advantage of such an approach over other global minimization strategies, such as simulated annealing, is that the large number of swarm particles makes this technique impressively resilient to the premature convergence problem that may rise from the fact of getting trapped in a local minimum, for instance. Swarm intelligence provides a basis with which it is possible to explore collective (or distributed) problem solving without the need for a centralized controller or the provision of a global model [37]. Several algorithms that are based on mimicking social colonies have proven to be quite efficient in solving many computationally complex problems, such as the non-threatening placement problem [18] and the traveling salesman problem [6]. Among the most successful approaches in swarm intelligence modeling is that inspired by social insects despite their limited cognitive ability and simple behaviors.

3.2 Cellular Automata

Cellular Automata microsimulation is an emerging technique for simulating traffic flow and modeling agents' movements in digital environments [2–5,24]. It allows incorporating human expertise in solving a particular problem intuitively using a knowledge base to govern the agent's behavior within certain regions in the environment in the form of simple if-then rules. Here, a Cellular Automaton is used to control the agent's inferencing mechanism to decide on the directions of their following steps.

A Cellular Automaton is a discrete mathematical model. Cellular Automata is used in various areas of science such as computational theory, theoretical biology, and mathematics. In a Cellular Automaton each cell can be considered as a state. Hence, it is obvious that each cell in CA has several neighbors. So, at time t, for each cell or state there is a finite function that can be assigned for its neighbor at time $t\pm$. In CA each cell has its own rule, and at each time in whole lattice of CA, a new procedure can be generated.

For example a sheet of paper such as a black and white grid can be considered as a Cellular Automaton. Each square as a cell has two possible states, and each cell has eight neighbors. So it can be concluded that for each cell and its neighbors there are 512 possible patterns. A table can be used to represent the behavioral rules for the Cellular Automaton. For all of the 512 possible patterns, the table can show whether the cell can be black or white based on function of time. This type of CA is called two dimensional Cellular Automaton [40,42].

A Cellular Automaton is usually considered as a finite grid and mostly two dimensional. The problem that arises in finite cellular automata is: how it is possible to solve the cells on boundaries. To solve and avoid this issue there are some solutions as follow.

- One possible method is to allow the values in those cells to remain constant.
- Another method is to define neighborhoods differently for these cells. One could say that they have fewer neighbors, but then one would also have to define new rules for the cells located on the edges.

In cellular automata, cells are changing their position to a different direction i.e: top, bottom, left, and right. To mathematically solve the boundary problem in one-dimensional approach: generated by:$\{x_{i-1}^{t-1}, x_i^{t-1}, x_{i+1}^{t-1}\}$, where x_it is a cell, t is the time, and i is the index [29].

3.2.1 Mathematical Model

Mathematically, a Cellular Automaton is represented by the four tuple $\{Z, S, N, f\}$ where:

- Z is the finite or infinite lattice.
- S is a finite set of cell states or values.
- N is the finite neighborhood, and
- f is the local transition function defined by the transition table or the rule.

The lattice is a finite or infinite discrete space of finite number of dimensions that is filled with cells. Each cell is defined by its discrete position by an integer number and by its discrete value with a finite set of integers. In this model time is also discrete. The future state of a cell at time $t+1$ is a function of the present state at time t and a finite number of cells defining the neighborhood.

Lattice, cell and configuration

The finite global state is a finite configuration $C \in S^Z$, where Z is a finite lattice, and S is a finite set $\{0, 1, 2, \dots, N-1\}$ of N integers.

$$C = c_0 c_1 \dots c_{x-1} c_x c_{x+1} \dots c_{N-2} c_{N-1} \tag{3.1}$$

Configurations and their parts can be written as strings by small Greek letters of the alphabet. Strings can be compactly written as numbers. Figure 3.1 shows the neighborhood size and position for each cell.

3.2.2 Local Transition Function and Rule

The cells in the neighborhood n_x of the cell c_x have indexes $x + I_i$, where $I_i \in I$ are values from the set of neighbors $I = \{-rL, -rL+1, ..., -1, 0, 1, ..., rR-1, rR\}$. rL and rR are neighborhood sizes to the left and to the right. Usually the neighborhood is a single radius r that can be used. The size of the neighborhood is $|I| = m = rL + 1 + rR = 2r + 1$.

$$n_x = c_{x-r_L} c_{x-r_L+1} \cdots c_{x+r_R} \tag{3.2}$$

The value n_x is a single integer defined as a number for the neighborhood k digits based on S.

$$n_x = \sum_{i=0}^{k-1} c_{x+N_i} |S|^{k-1-i} = \sum_{i=0}^{k-1} c_{x-k_0+i} |S|^{k-1-i} = c_{x-k_0}|S|^{k-1} + c_{x-k_0+1}|S|^{k-2} + \cdots + c_{x-k_0+k-1}|S|^0 \tag{3.3}$$

The neighborhood cell indexing and the local transition function of a neighborhood cell on function can be defined as:

$$f : S^N \longmapsto S \tag{3.4}$$

Equation 3.4 calculates the value of a single future cell c_x from the neighborhood of the cell. Figure 3.2 and Equation 3.5 show the neighborhood cell indexing and the local transition function.

$$c_x^{t+1} = f(c_{x-r_L}^t, c_{x-r_L+1}^t, \ldots, c_{x+r_R}^t) \tag{3.5}$$

The transition table defines the local transition function by listing the output value for each input value such as: $n \longrightarrow f(n)$. The rule f shows the local transition function. It is an integer that is defined as a number of k digits.

$$f = \sum_{j=1}^{|S|^k-1} f(j)|S|^j = f(|S|^k - 1)|S|^{|S|^k-1} + \cdots + f(1)|S|^1 + f(0)|S|^0 \tag{3.6}$$

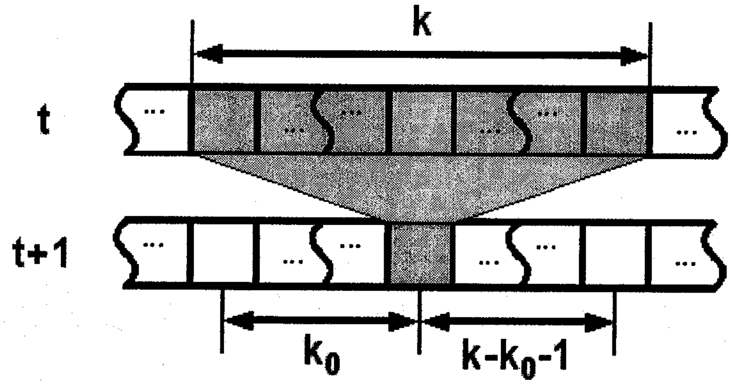


Figure 3.1: The neighborhood size and position [40]

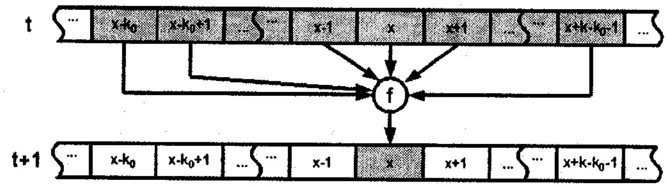


Figure 3.2: Indexing the neighborhood cell and show the local transition function [40]

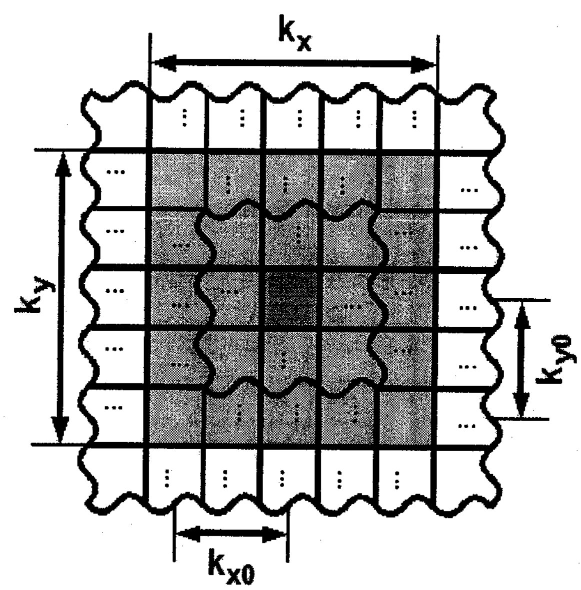


Figure 3.3: Neighborhood in a two dimension Cellular automata [40]

3.2.3 Two Dimensional Cellular Automata: 2D lattice

The definition of *n-dimensional* CA is similar to *one-dimensional* CA, the lattice becomes *n-dimensional*: k and k_0, become vectors of length n in (3.3) and (3.6).

There are different ways that 2D lattice can be covered in cells as follows.

- square grid
- triangular grid or hexagonal grid
- periodic grid [29].

Two dimensional cellular automata are used in different areas of scientific research. Second order local transition functions are frequently used to create backward rules [40, 42] (Figure 3.3).

3.2.4 Applying Cellular Automata in an Agent-Based Algorithm

Here, a Cellular Automaton is used to control the agent's inferencing mechanism to decide on the directions of their following steps [2–5, 24, 40, 42]. The rules used in the adopted Cellular Automata model are as follows:
A set of 4-direction rules based on Cellular Automata are extended here to apply in Agent-based model for the proposed algorithm. In each time step t these CA rules are applied parallel into the system to update and control the agents' movement. The rule set consists of two main independent steps:

- Changing Lanes and
- Moving Forward.

In terms of lane changing, agents' movements are random-based, so they do not follow the lanes usually. One of the best approaches to allow agents to follow the lanes is CA. Each agent is taken in its local neighborhood without anticipating what other agents behavior is. The agents are able to move only after all agents on the environment(here the environment is the image) have been visited the pixel and then they have been permitted to move.

Also there is a 4-directional rule set to show for unidirectional and bi-directional movement. The rules can be obtained by Forward Rules and are as follows:

- Avoiding Conflicts between agents

- Diagonal exchanges within agents
- Cross Movement within agents
- Lateral Movement within agents

Figure 3.4 shows the 4-directional Agent rule set in which $L(i,j)$ is the Lattice cell (i,j) for agent number a_i where i is the forward direction and j is the backward direction of agent a_i. A maximum speed could be applied for each agent a_i as v_max and moves with velocity $v(a_i)$ and determined for each agent in each time step. In this proposed algorithm the velocity is considered to be *one* while agents keep going and move (see Figure 3.4).

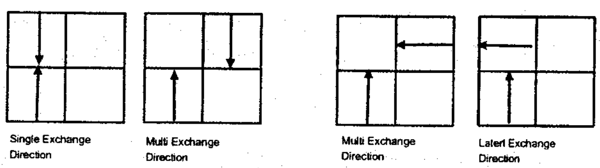


Figure 3.4: Movements with conflict

Changing the Current Position

1. Avoiding conflicts: Agents that are moving in lateral cells or adjacent cell may not conflict with one another. Here an empty cell is considered between two agents by probabilities p in which p is mostly considered 50/50. If *lane* is not available, *gap* is set to $\oslash$ or *null* (See Figure 3.4).
2. Applying lane and gaps to agents: same lane or adjacent (left or right) lane is chosen for forward movement according to $(left, center, right)$ up to v_{a_i}(the velocity of agents which here is 1).
   (a) If an agent from the opposing direction assigned by $gap = 0$ it will step out of the lane.
   (b) If *gap* in same direction is equal to zero the agent will step behind an agent moving in the same direction.
   (c) If there are maximum gaps ahead:
      i. adjacent changing lanes by considering: random50/50
      ii. Changing between current lane and single adjacent lane: stay in lane
3. each agent a_i is able to move by 0, +1 to $cell : L(i, j \pm [0,1])$.

Moving Forward

1. Solving Conflicts in cross-directional: for any pixel/cell that crosses to two cross-directional by agents a_i and agent a_j, cell $L(i + gap_{a_i}, j)$:
   IF probability $\prec p$ THEN
   agent a_i gets cell $L(i + gap_{a_i}, j)$ and agent a_j gets cell at $L(i + gap_{a_j} - 1, j)$
   ELSE
   agent a_j occupies cell $L(i + gap_B$, j) and agent a_i gets the cell at $L(i + gap_{a_i} - 1, j)$.
   IF a conflict at the new cell $L(i + gap_i - 1, j)$, repeat the procedure.

2. Solving Cross-forward in adjacent neighborhood:
   IF *gap* is equal to zero and $L(i + 1, j)$ occupied by a forward agent (forward cell occupied by an agent) THEN
   with probability p_i change the direction of current agent
   ELSE: stay in the current cell.

3. Moving the agents: each agent a_i is moved to the appropriate cell $L(i + 1, j[0, 1])$ in the environment(Image).

3.3 Conclusion

In this chapter we presented the theoretical background of Swarm Intelligence and Cellular Automata in general. The chapter is divided into the following sections

- In Section 3.1 Swarm Intelligence and its techniques were defined. We described what is Swarm Intelligence and its applications. Also pros and cons of Swarm Intelligent were explained. Meanwhile, we indicated how Swarm Intelligence can be used toward our image processing goal.

- In Section 3.2 we introduced what is *cellular automata* and different types of cellular automata and its application. Also we established rules of CA in term of leading agents in agent-based environment.

- Finally, we described the agent-based model, application, and their rules. We explored in which environment it can be used to be considered as agent-based model. Also, we examined how swarm intelligence and cellular automata techniques can be used in agent-based environments and their collaboration for image processing.

Chapter 4

Swarm Intelligence Based Image Segmentation (SIBIS)

4.1 Introduction

In this chapter we explore our approach for image segmentation based on swarm intelligence techniques. Here, a Swarm Intelligence Based Image Segmentation (SIBIS) algorithm is used to achieve our goal. The proposed algorithm is composed of five components (Figure 4.1): Computing local measure, Firing If-Then Rule, Searching, Labeling, and Segmentation/Boundary-detection. In the following sections we describe each of those components.

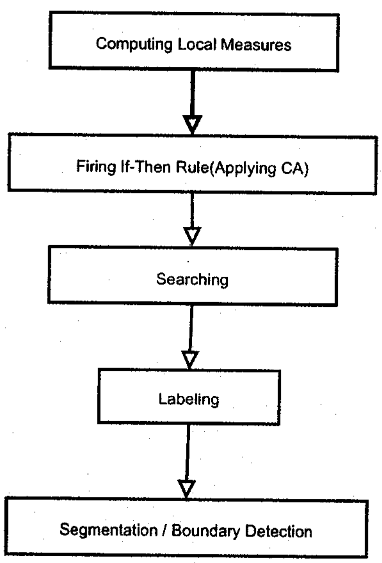


Figure 4.1: Block diagram of the proposed agent-based algorithm (SIBIS)

4.2 Proposed Approach

4.2.1 Finding the Threshold

In order to do segmentation and boundary detection, finding the *threshold* is one of the crucial parts of SIBIS. There are several methods that may apply to finding such a threshold [16]. In SIBIS agents have a strong sense for the environment and pixels. This gave us a higher ability to find a threshold by applying the threshold algorithm. In SIBIS we mixed both the mean value method and histograms approach to find a more accurate threshold.

The method that we used in SIBIS to find the sufficient threshold is as follows:

1. We consider an input image as $g(x)$ where x is the pixel intensity.

2. Initiate a default threshold to start as T.
   To find T we determine $Min(g(x))$ and $Max(g(x))$ of Image $g(x)$. Then, T is computed as
   $$T = (Max(g(x)) + Min(g(x)))/2 \tag{4.1}$$

3. In the second step we divide the image into two sub-images $g_1(x)$ and $g_2(x)$:
   (a) $g_1(x)$ is a set of pixels where $Intensity(I(x))$ of pixels are greater than T. There is, $g_1(x) = \{g(x)\ such\ that\ x > T\}$.
   (b) Likewise $g_2(x)$ is defined as: $g_2(x) = \{g(x)\ such\ that\ x \leq T\}$.

4. Given $g_1(x)$ and $g_2(x)$, their respective mean values ($Mean_1$ and $Mean_2$) are then calculated:
   $Mean_1 = Mean\ value\ of\ g_1(x)$
   $Mean_2 = Mean\ value\ of\ g_2(x)$

5. Then the new threshold is calculated by testing the $Mean_1$ and $Mean_2$. Mathematically,
   $T = (Mean_1 + Mean_2)/2$

6. The iteration starts from step 2 until the appropriate threshold is achieved.

Although the above algorithm gave us a close threshold for our algorithm, the calculated value of T is not very efficient. To achieve the valuable threshold in SIBIS we used the following approach. In the above algorithm, to assign the best threshold T in each iteration, we eliminate the sharp noise from the sub images $g_1(x)$ and $g_2(x)$. This algorithm removes the high and low intensity($I(x)$) of sub images $g_1(x)$ and $g_2(x)$.

To compute the sufficient threshold we used the following technique in SIBIS.

- In each iteration do :
  - *Calculate* the number of pixels that are higher and lower than threshold T. Meanwhile, we eliminate sharp noise from the sub images.
  - ***If*** the percentage of part a is very low ***Then***
    Remove those pixels from sub images $g_1(x)$ and $g_2(x)$

To express the proposed SIBIS algorithm, we illustrate and explain SIBIS with an example image for each step. The sample image is shown in Figure 4.2. This is only a 5×5 matrix where its cells show the intensity of pixels.

Applying the threshold-calculation algorithm described above, leads to virtually dividing the original image into two sub-images as depicted in Figure 4.3. Bold and regular digits represent the pixels whose intensities are higher and lower than the calculation threshold, respectively.

3	24	10	13	32
15	6	20	31	1
2	21	5	28	9
25	22	7	30	27
8	29	12	23	4

Figure 4.2: 5×5 original sample image.

4.2.2 Agents Local Behavior

In this work agents are to operate on a digital image, which is regarded as a two-dimensional environment, i.e., Image $A = I(x, y)$, where $I(x, y)$ denotes the intensity of pixel (x, y). In the proposed agent-based approach, agents are designed in such a way as to operate directly on the individual pixels of a digital image by continuously sensing their neighboring regions and checking the homogeneity criteria of relative contrast,

3	**24**	10	13	**32**
15	6	**20**	**31**	1
2	**21**	5	**28**	9
25	**22**	7	**30**	**27**
8	**29**	12	**23**	4

Figure 4.3: Finding the threshold for a sample image.

regional mean, and regional standard deviation. Based on the sensitivity and visibility of the agents swarm from their neighboring regions, the agents select and execute the homogeneous pixel from the region. Each agent may stay at its current pixel, move to an adjacent one, or perish, in which case the agent is considered to be reactive, by its local environment.

In this approach, the image is clustered according to several regional criteria to identify the different segments in an image. Four different measures can be used for this purpose. These measures are, the regional mean, regional standard deviation, regional maximum, and regional minimum, of the gray-level intensities. The neighboring region of an agent at location (i, j) is a rectangular region centered at location (i, j) with a radius of r pixels. The pixels falling inside this region are called the neighbors of that agent. In this study, two configurations are considered with r being set to 1. The first configuration excludes the pixels that are diagonally across from the agent's current pixel from the neighboring region, while the second configuration includes them. This is depicted in Fig. 4.4. For a local region $R(i, j)$ centered at pixel (i, j) of the image, the regional mean, regional standard deviation, regional maximum, and regional minimum, of the gray-level intensities are defined in (4.2), (4.3), (4.4), and (4.5), respectively, where

N is the number of pixels within $R(i,j)$.

$$\text{Mean}_{R(i,j)} = \frac{1}{N} \sum_{(x,y)\in R(i,j)} I(x,y) \tag{4.2}$$

$$\text{Std}_{R(i,j)} = \sqrt{\frac{1}{N} \sum_{(x,y)\in R(i,j)} (I(x,y) - \text{Mean}_{R(i,j)})^2} \tag{4.3}$$

$$\text{Max}_{R(i,j)} = \max_{(x,y)\in R(i,j)} I(x,y) \tag{4.4}$$

$$\text{Min}_{R(i,j)} = \min_{(x,y)\in R(i,j)} I(x,y) \tag{4.5}$$

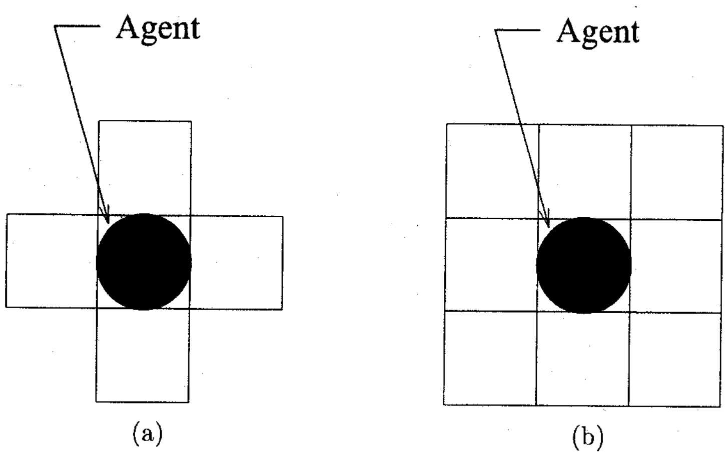


Figure 4.4: Local neighboring region of an agent: (a) a 4-neighbor region, (b) an 8-neighbor region

The goal of each agent is to localize homogeneous pixels within an image region which form a segment. This is accomplished through the following steps: first, one of the statistical measures (4.2), (4.3), (4.4), and (4.5), is computed for the agent's neighboring region. Then, the Cellular Automata rules are fired to determine the agent's next step.

Algorithm 1: The Proposed Swarm-based Image Segmentation Algorithm

```
Input: An i × j digital image.
Output: Segmented image.
begin
    Calculate local parameters of each agent's neighbor: Mean_R(i,j), Std_R(i,j), Max_R(i,j),
    and Min_R(i,j)
    Compute percentage of homogeneous pixels in image
    Sprout agents to environment based on homogeneous pixels
    while clocktime ≤ stoptime do
        Search Routine
        begin
            if homogeneous pixels found then
                Label pixel (i, j)
            else
                Fire the Cellular Automata rules to determine the direction of agents
                Move from current State S_i to next State S_i+1
            if agent is out of its region then
                Kill that agent
        end
        if Labeling the pixels are finished in environment then
            Do Segmentation
end
```

Computing the Local Measures of the Regions

In this block the statistical measure of the agent's neighboring region is computed. On the other hand equations 4.2, 4.3, 4.4 and 4.5 are calculated based on their measurement of the local behavior of agents for each neighboring pixels. Each agent determines whether or not the pixel it is operating on is homogeneous or not depending on the adopted regional criteria ($\text{Mean}_{R(i,j)}$, $\text{Std}_{R(i,j)}$, $\text{Max}_{R(i,j)}$, or $\text{Min}_{R(i,j)}$) and the pixel's intensity level. These two parameters form the agent's local stimulus. In other words, each agent considers its current pixel to be homogeneous within its neighboring region if it satisfies its local stimulus condition.

Firing the Swarm Techniques and CA Rules

The Cellular Automata rules that were described in Section 3.2.4 are fired to determine the agent's next step. The algorithm starts with randomly distributed agents over the image (the environment). After a number of behavioral evolution steps, the active agents belonging to the homogeneous segment being searched gradually vanish by the time all the homogeneous segments are identified. This iterative algorithm takes several behavioral evolution steps before it eventually converges when no new segments are detected. In this phase most of the homogeneous pixels found are labeled. So in the segmentation phase agents separate different labeled pixels to different region as segment parts.

Searching and Labeling the Pixels

In this step, the environment/region is searched for homogeneous pixels which are eventually labeled at a later stage. A block diagram of this approach is illustrated in Figure 4.1. The algorithm starts with randomly distributed agents over the image.

1. SEARCHING:
   Once the Swarm Techniques and CA are built, agents become able to move based on the information provided by the Swarm Techniques and CA. We may describe this behavior by the following expression:
   If agent is in current cell C_k and wishes to move to next cell C_{k+1} ***THEN***
   $agent_{(i,j)}(C_k) \Rightarrow C_{k+1,\theta}$
   where (i, j) is current location of agent, C_k is current cell in which the agent is located. $C_{l,\theta}$ is the next cell, and θ is the direction of the next cell.

2. LABELING:
   In this phase the pixels that are found by the Search algorithm must be labeled. After that, agents find homogeneous pixels and agents label the pixels in its neighboring region. To keep the labeled pixel alive SIBIS used an internal variable as a tag. Labeling does not have any effect on the image. The tags hold the labeled pixels during the process. We use a labeled tag in marking part which is explained in the next section. We formulate this behavior by the following expression:
   If local stimuli meet the criteria of agents ***THEN***
   $agents(R)_{(i,j)}(C_k) \Rightarrow L(i, j)$
   Where R is region of agent neighbor and $L(i, j)$: Pixels that are labeled by agents.

After a number of behavioral evolution steps, the active agents belonging to the homogeneous segment being searched gradually vanish by the time all the homogeneous segments are identified. This iterative algorithm takes several behavioral evolution steps before it eventually converges when no new segments are detected.

Now, we try to show how Figure 4.3 is changed after the Searching and Labeling step. The result for this step is shown in Figure 4.5. To explain this part we just choose one of the agent's neighborhoods to illustrate as an example. The area is shown by a different color in a nine-cell neighborhood. The agent follows all the rules described in Section 4.2.2 and this section to find the homogeneous pixel. After finding the appreciate pixels the agent labels them with a label L as shown in Figure 4.5.

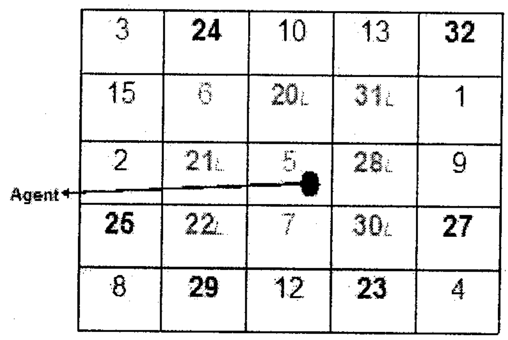


Figure 4.5: Sample image after Searching and Labeling step.

4.2.3 Marking Contours of the Labeled Pixels

After the Agent Local Behavior is executed properly, the labeled pixels are marked. By marking the labeled pixels the region and boundaries that have been segmented can be viewed in the environment. In SIBIS, agents label homogeneous pixels while they are searching and processing the image. Therefore SIBIS uses the Labeling ability that is used in the previous section for marking those pixels. Since SIBIS is a fully agent-based environment, SIBIS is able to recognize labeled pixels and mark them.

By above feature of marking task we can describe the marking behavior by the following expression:

If process is done ***And*** pixel $p(i,j)$ is *labeled* ***Then***

$P_{(i,j)}(l) \Rightarrow mark(P_{(i,j)}(l))$

Where $P(i,j)(l)$: is the labeled pixels by tag l in the image.

If we apply the Marking step on the image that is created in Section 4.2.2, the result can be viewed in Figure 4.6.

4.2.4 Segmentation/Boundary-Detection Part

In this part SIBIS segments the image based on segmentation methods that we described in implementation part in Appendix A. Here, we briefly explain the segmentation parts.

3	**24**	10	13	**32**
15	6	mark	mark	1
2	mark	5	mark	9
25	mark	7	mark	**27**
8	**29**	12	**23**	4

Figure 4.6: Sample image after marking letter.

1. Agent a_i calls the appropriate part of algorithms 1.
2. If pixel p_i is marked, then p_i is chosen as segmentation area.
3. Repeat and start from step one until system *stoptime* is achieved.

After this algorithm is completed, SIBIS is able to recognize the marked pixels as segmentation part and it can determine the boundary of each segment.

4.3 Conclusion

In this chapter we presented the design and architecture of SIBIS from two different points of view. First of all, we illustrated the proposed approach in Section 4.2. We described the techniques that are used by SIBIS to find a threshold for the whole image. Then, the Local behavior of Agent is introduced. In this section we explained the methods and techniques that are used by agents for searching, labeling and marking the regions and boundaries in the image. Implementation and software design of SIBIS are presented in Appendix A. The implementation part also describes the techniques and methods that are used for segmentation and boundary detection based on the SIBIS approach.

Chapter 5

Results and Evaluation

5.1 Introduction

To evaluate the performance of the proposed Swarm Intelligence Based Image Segmentation (SIBIS) algorithm, we tested it with two categories of image databases:-*biomedical* image database, and -*natural* image database. The biomedical images are taken from different applications: dental radiographic images and brain images.

Radiographic images are produced from an x-ray film. As the x-ray beam passes through an object it is reduced as a result of absorption. The film in dental radiography records the pattern produced by the remnant beam giving information about the object the beam passed through. After some chemical attraction the image is produced from negative and positive of charged ions. This kind of image possesses several characteristics such as: radiographic density, contrast, speed, noise and blurring. The feature we are most interested in here is the radiographic density which is given by

$$\textit{optical density} = \log_{10}(I_0/I_t) \tag{5.1}$$

Where I_0 is the *intensity of the incident light* and I_t is the *intensity of the transmitted light.* Another important parameter of the radiographic images is *radiographic contrast.* This is the difference between light and dark areas in the radiographic image.

The natural image database used here were collected by Martin et al. [26]. This database contains around 300 images, some of them are shown in Figures 5.1. Several humans were instructed to segment test images in about 20 disjoint regions. Some of the humans' segmentation images are shown in Figures 5.2. A thicker line in the image is one that was drawn by more than one person. These lines are considered as ground-truth in this case.

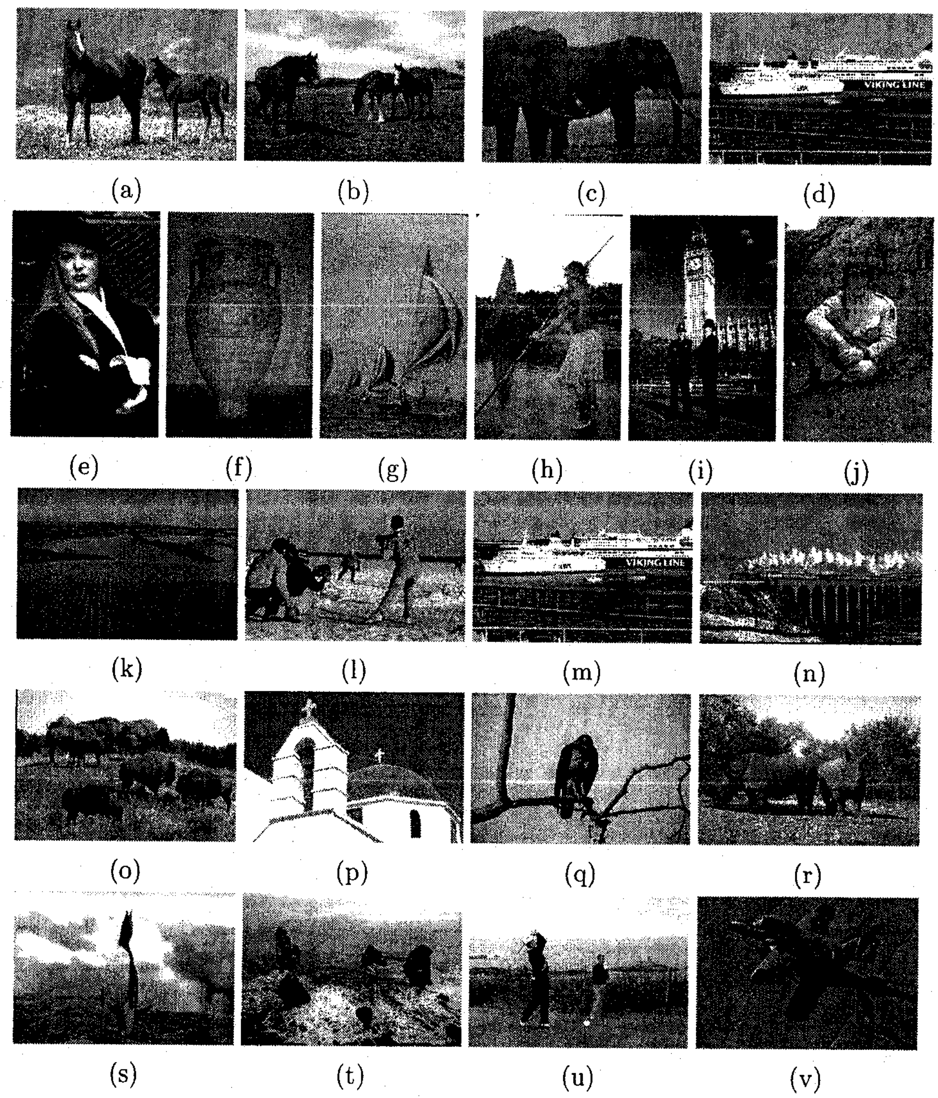


Figure 5.1: Outdoor images from the Berkeley Segmentation Database [26].

For the purpose of this study, we used 30 images and their associated segmentations as the test data for our proposed algorithm.

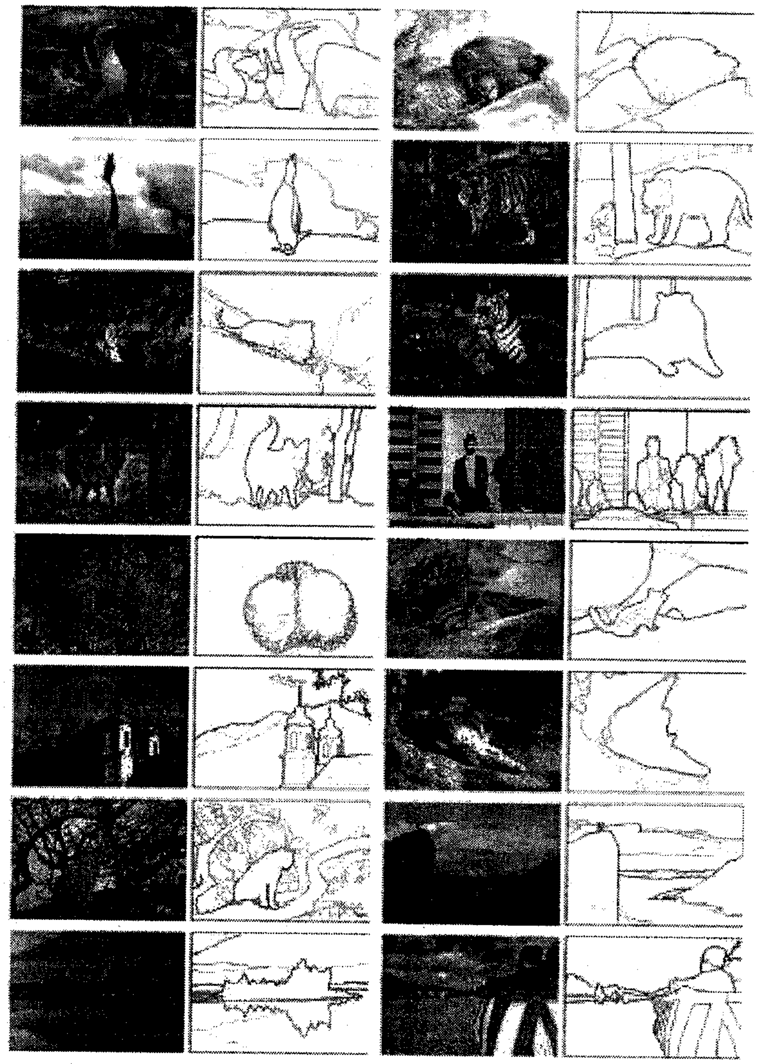


Figure 5.2: Example images and human-marked segment boundaries. Each image shows multiple human segmentations. The pixels are darker where more humans marked that boundary [26].

5.2 Evaluation Methodology

We formulate boundary-detection as a categorization problem to recognize differences such as non-boundary pixels from boundary pixels. The accuracy measure adopted in this study is the *precision-recall* measure. In order to optimize the parameters of the proposed algorithm and compare it to other techniques, we need a methodology for judging the quality of the detected boundaries.

The precision-recall curve is a parametric curve that displays the difference between the real value and the non-actual value as the detector threshold. Precision is the fraction of true positive to false positive detections. Recall is the fraction of true positive detections to missed values during the detection process. In probabilistic views, precision is the probability that the detector decision is true, and recall is the probability that the ground truth data is detected.

The precision-recall curves represent a standard evaluation technique in information retrieval [31]. A similar approach was proposed by Bowyer et al. [8] for boundary detector evaluation with Receiver Operating Characteristic (ROC) method. The parameters for an ROC curve are fallout and recall. In this technique, recall is the same as above. However, fallout is the probability that a true negative is detected as a false positive.

The reasons for which we prefer to use precision-recall curves instead of ROC curves are as follows: ROC curves show the same quality shown by precision-recall curves. However, ROC curves are not good enough to recognize boundary detection, as it is dependent on pixel values. If we reduce the radius of the pixels by a factor of n, the number of pixels grows by a factor of n^2, and the number of true negatives then grows quadratically in n while the number of true positives grows only linearly in n. Since boundaries are one-dimension of entities, the number of false positives is usually linear in n, and so the fallout is reduced by $1/n$. Precision, however, does not have this problem, because it is normalized by the number of true negatives instead of normalized by the number of false positives [26]. Another method of evaluating boundary detectors was used by Konishi et al. [23]. Although it showed its superiority in ranking algorithms, it did not demonstrate the same strength as an evaluation measure for boundary detection.

The precision and recall measures are meaningful in terms of boundary detection when we consider applications that use boundary detection and objects recognition. This way, R (recall) can show how many true values are recognized, and P (precision) can display how much noise can be detected. A specific application can be defined to show the relation cost α between P and R. The α value is a specific point on the *precision-recall* curve. So it can be formulated as the F-measure [31], and is defined as

$$F = \frac{PR}{\alpha R + (1-\alpha)P} \tag{5.2}$$

The location of the maximum F-measure on the curve provides the optimal detector threshold for a particular value of α. In this study, α is set to 0.5. With $\alpha = 0.5$, (5.2) can be re-written as

$$F = \frac{2PR}{R + P} \cdot \tag{5.3}$$

The computation of the precision and recall measures requires the differentiation between true positive and false negative detections. Each point on the precision-recall curve is computed from the detector output at a specific threshold t. In addition, we use a binary boundary as ground truth from the human subjects. Now, we would like to explain how to compute the precision and recall chosen as a single threshold. One of them can consider boundary pixels easily and recognize all unmatched pixels as either false positives or missed pixels. From Figures 5.5 to 5.10, it is clear that the boundary pixels to ground truth boundaries must have different errors since the ground truth data also has boundary errors. On the other hand, it is important to compute the considered multiple detections on the images. Single detection is one of the three goals of boundary detection formalized by Canny [10]. The other two being a high detection rate and good localization method.

Our goal in this approach is to estimate the precision and recall values to measure them as close together as possible based on output values.

5.2.1 An Example for Precision-Recall and F-measure

Here, we are trying to illustrate the meaning of precision-recall curve through an example. We use the area under the precision-recall curve to show the results. For each measure, we present both the sample mean, and standard deviation of the sample mean. It is not obvious how to compute the F-measure for the area under the precision-recall curve. The purpose of this example is to explain how this computation for the F-measure, P, and R, are done. Suppose, there are i points on the curve and let p_i be the probability that is assigned to point i. Suppose the true-positive points show the R from 0 to 1. Figure 5.3 is an example that shows precision-recall curve that estimate F-measure. We consider the x-axis to be precision and the y-axis to be recall. In any case the area under the curve does not change.

As mentioned above we are using set of thresholds to estimate the precision-recall curve. And finally we draw the P and R for each threshold. They connect all the points that are created by P and R with a line. Hence, The precision-recall curve can be shown as a set of trapezoids that can be seen in Figure 5.3. It is obvious that the area of each trapezoid is an estimation of the area under the precision-recall curve. Having more thresholds leads to a better approximation.

The precision-recall curve is a measure to describe the performance. When a single

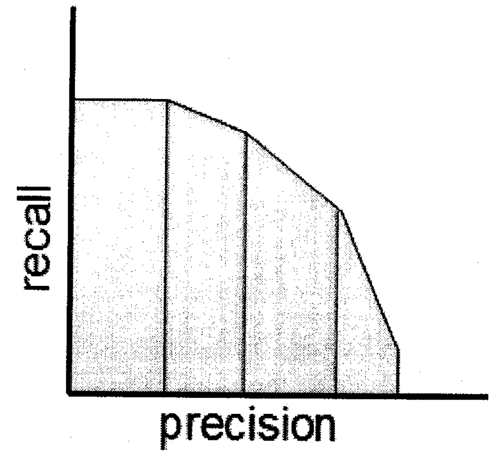


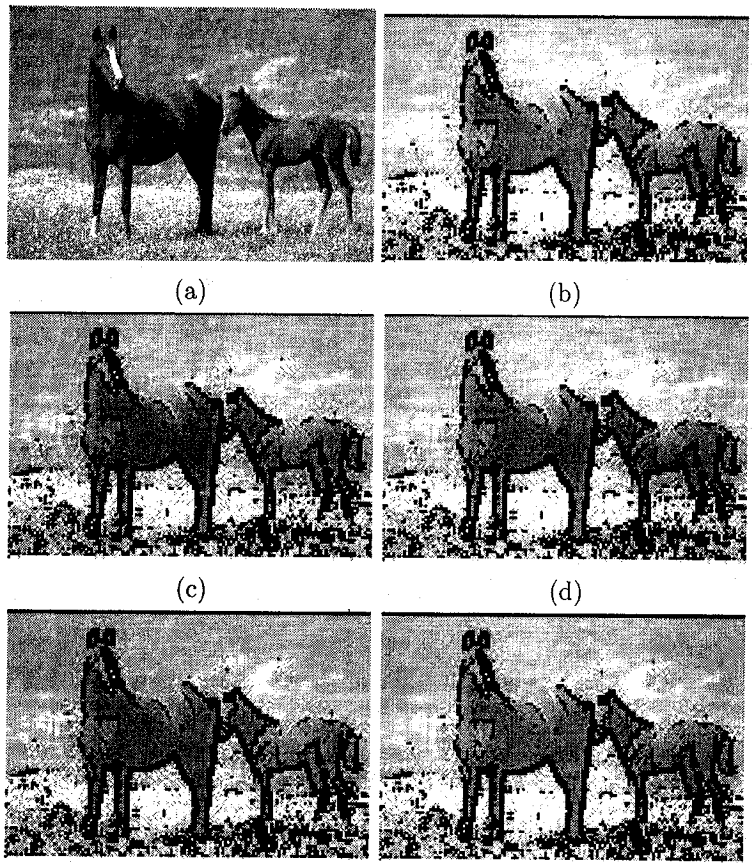
Figure 5.3: Computing the F-measure using a Precision-Recall curve.

performance measure is required or is sufficient, precision and recall can be combined with the F-measure. The F-measure curve is usually multi-values. So the maximal F-measure can be reported as a result of the detector's performance. We now go back to applying this evaluation methodology to optimizing our boundary detector, and comparing our approach to the standard methods.

5.3 Segmentation of Natural Images

There is a large number of features that exist in outdoor images. To test SIBIS with these types of features, 30 outdoor images from the Berkeley database were selected in different weather and sky conditions [26]. These images are shown in Figures 5.1. In almost all of these outdoor images, the sky was identified.

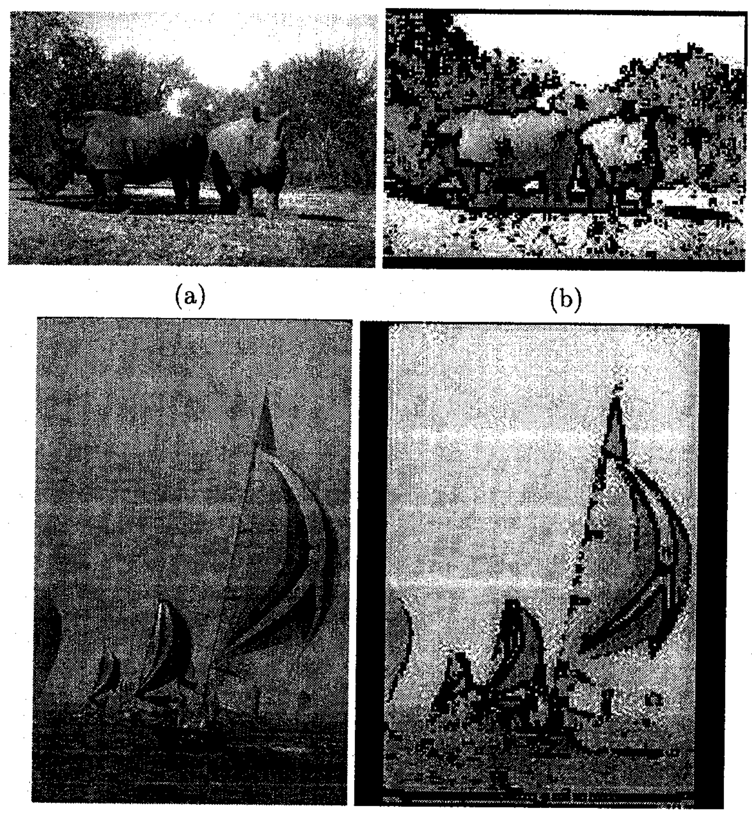
One of the major problems in image segmentation is to distinguish an object from a highly textured background. This is true especially in images of outdoor scenes, such as the images of Figures 5.4, this is a particular example of a highly textured image with two horses in the foreground and a dense vegetation in the background. The segmentations obtained with SIBIS using different thresholds are shown in Figure 5.4. It is clear that this segmentation has a lot of different regions, such as the white flowers and the grass. If we consider the two horses as objects; the shape of the two horses is distinguished from the background very well. The white part of the foot of the horse on the right is detected in the segmentation. The last extracted feature reveals one of the advantages of the proposed algorithm as it is rarely extracted in other well known segmentation algorithms [23, 26].



(a) (b) (c) (d) (e) (f)

Figure 5.4: Segmentation of a two horse-image.
(a) Image of two horses in a highly textured background. (b) Segmentation by normal threshold. (c) Segmentation after the texture is described in proposed algorithm. (d) Fine segmentation, with increasing threshold by 1. (e) Medium segmentation regions. (f) Coarse segmentation with decreasing threshold by 1.

In another experiment we test two different images. One is a hippo image with sky in the background and dense vegetation in the foreground. Another image is of boats of different sizes with flying sail, the sea and the sky in background that are displayed in Figure 5.5. The images are tested with SIBIS in normal parameters to detect the boundaries of the images. The segmentations obtained with SIBIS using one threshold and the results are shown in Figure 5.5b and 5.5d. This segmentation has a lot of different regions. If we consider the two hippos and three boats as objects to be distinguished; the shape of the two hippos is recognized from the background significantly. Also feet, bodies, ears, eyes, and shadows of the hippos are clear. Similarly, boats, sails, and rigging are quite clear in boat segmentation images and are recognized clearly. Another feature that is interesting and is worth mentioning is that the background such as the sky, trees, and land ground in the hippo image, and the sea and sky in the boat image do not have any effect on the segmentation. If we notice in gray-white hippos it is obvious that, the eye is recognized clearly, since without that is hard to notice the object.



(a) (b) (c) (d)

Figure 5.5: Segmentation of image: (a) Original image of two hippos with various background texture, (b) Segmented images, (c) Original image of boats in the sea with sky background and, (d) Segmentation image.

In the final experiments SIBIS was tested with different natural images. In these experiments we evaluated SIBIS' strength in finding boundaries with various types of images that contain different objects. We tested five images as shown in Figure 5.6. Each image is under different lighting conditions. As illustrated in Figure 5.6, the objects in all images are clearly distinguished. The backgrounds in all images remain unchanged, which is a significant advantage of SIBIS. The segments are obtained with the proposed algorithm by the same thresholds used for Figure 5.6.

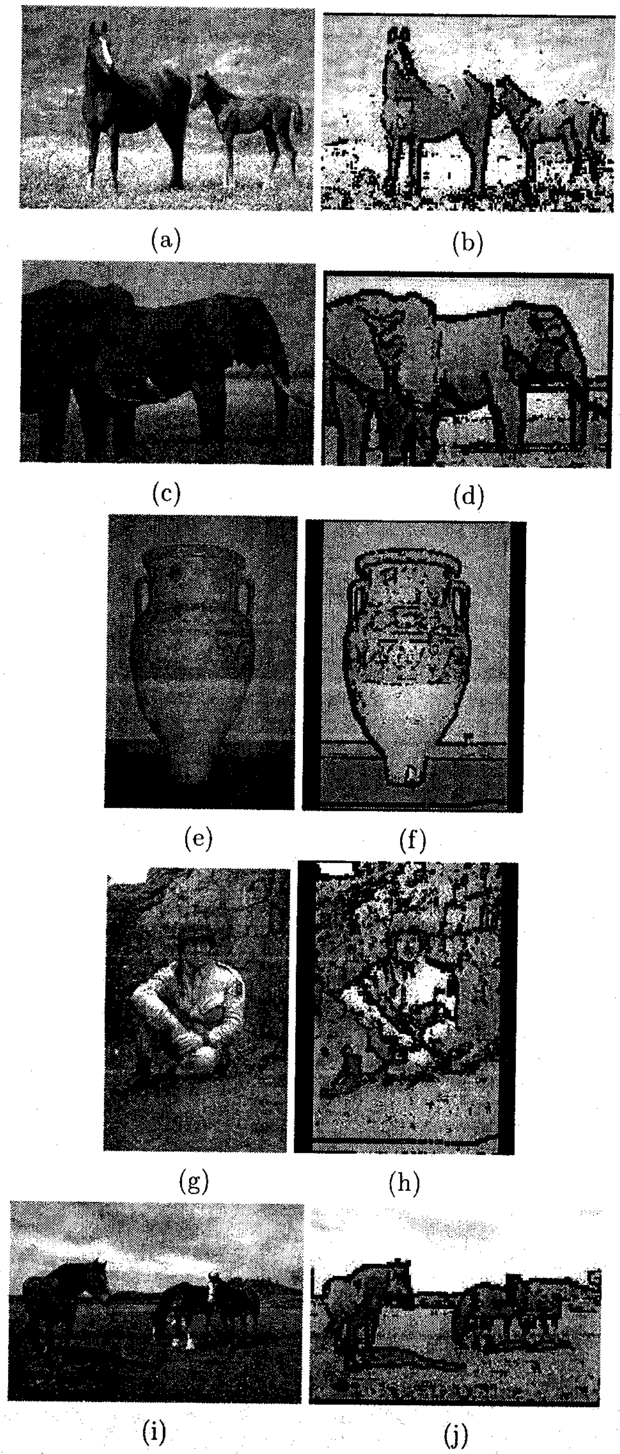


Figure 5.6: Segmentation of various images with different light condition.

5.3.1 Comparison of proposed algorithm with Human Segmentation

Human segmentation was compared with the proposed segmentation with *spot* value of threshold in the proposed algorithm. The *spot* value is a threshold which automatically is determined and measured by SIBIS. Table 5.1 shows a comparison between the proposed algorithm and the human segmentation. For every image, the database provides five different human segmentations, here referred to as U1, U2, U3, U4, and U5. These segmentations are compared with those provided by the proposed approach with different values of threshold T: 0.50, 1, 1.50, and 2 and spot which is the threshold automatically detected by SIBIS. ($Thr0.5$, $Thr1$, $Thr1.5, Thr2, Spot$).

As illustrated in Figure 5.7, humans considering segment, properly extract objects and identify them from the background. In the animal images and especially in high density vegetation, the proposed algorithm is not able to exactly differentiation between the animal bodies and the surrounding grass but it still does well in most cases. In the two boat images, the proposed algorithm is able to extract correctly the boats from the sea, just like humans do. A similar performance is noticed for the church but not as accurate as the human segmentation. Also in the tower bell image, humans are able to extract the edge between the tower and the wall, but our proposed algorithm is not perfectly able to do that. However, the background and the edges are properly extracted. In the eagle image both humans and SIBIS are able to identify the branches, but the latter even out-performed humans in this case.

5.4 Segmentation of Radiographic Images

To determine another aspect of SIBIS, the algorithm was tested with another category of images. Ten radiographic images have been chosen to be tested by SIBIS and they are shown in Figure 5.8. To test SIBIS with various features the dental images are chosen for different tooth shape.

In the first experiment, three radiographic images were chosen to determine the performance of the proposed segmentation algorithm. The three original images with their corresponding results are illustrated in Figure 5.9. Another feature needs to be pointed out in Figure 5.9, it illustrates how the segmentation works around the tooth area in radiographic images. The proposed algorithm properly shows all the texture in the background and correctly extracts the tooth boundaries. SIBIS is able to extract a shape from a background if the background texture has a fine scale. One of the advantages

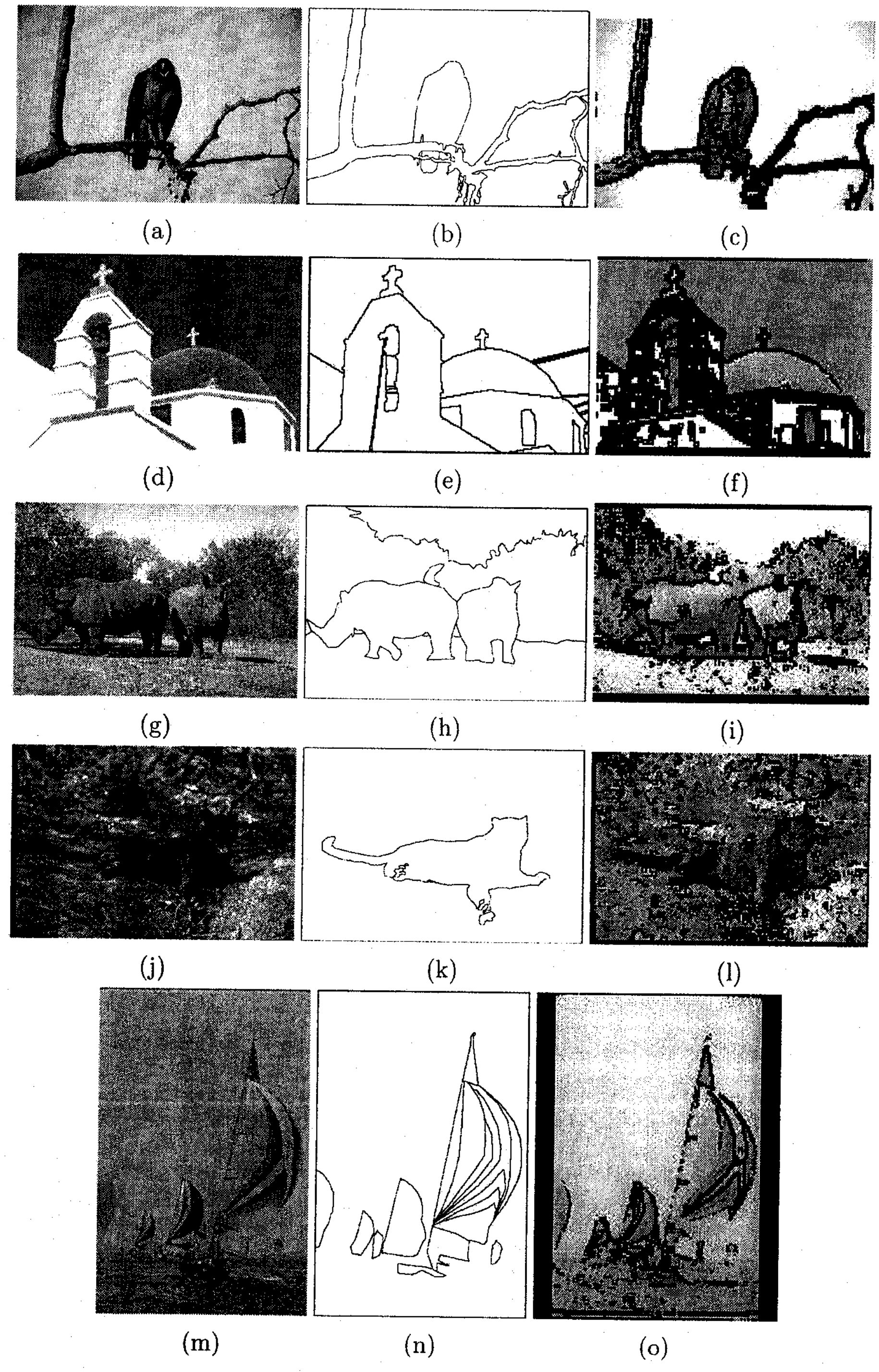


Figure 5.7: Comparison between the proposed algorithm and human segmentations. In the left column are five original images from the Berkeley Segmentation Database [26]. In the middle column reproduced in white are five human segmentations (U_1, U_2, U_3, U_4 and U_5); thicker lines indicate that more than one human observer identified that edge. The right column shown SIBIS segmentation.

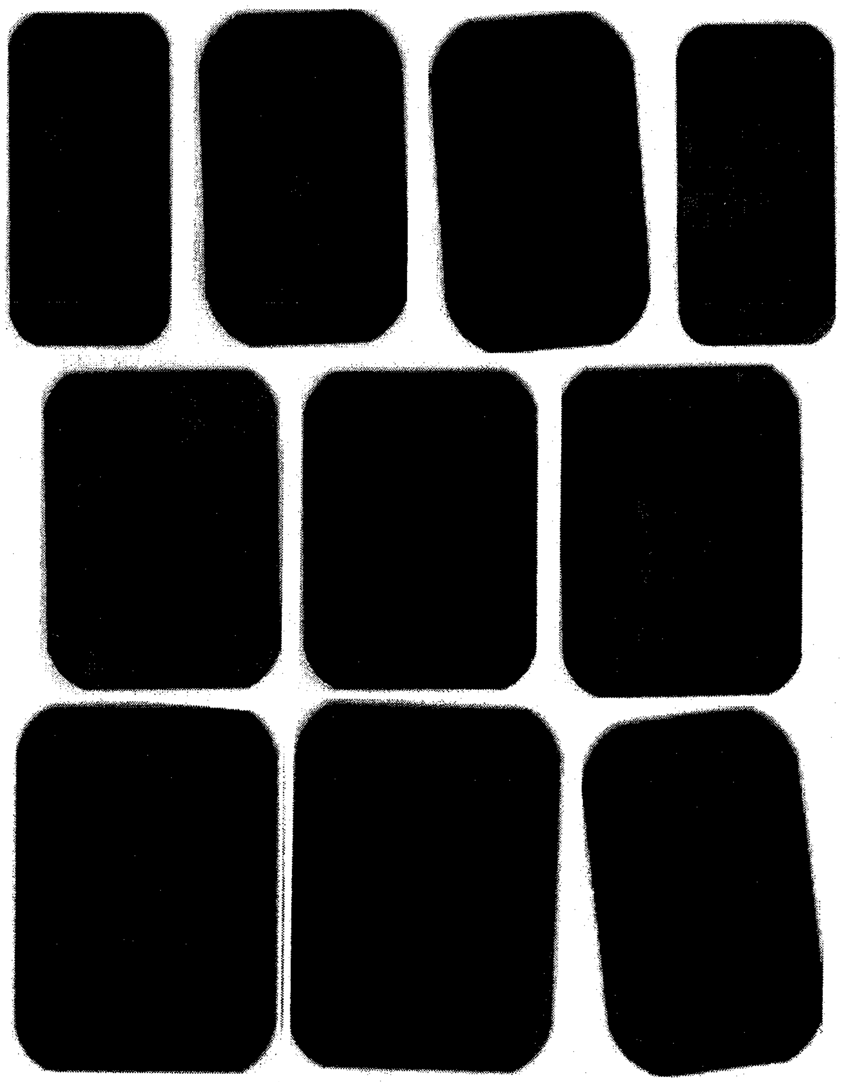


Figure 5.8: Ten Radiographic images tested by SIBIS.

Image 42049 'eagle'															
	U1			U2			U3			U4			U5		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Thr1{.5}	.89	.84	.80/75	.87	.87	.85/75	.82	.80	.86/77	.87	.87	.85/67	.82	.80	.86/63
Tr2{1.0}	.81	.88	.83/78	.85	.86	.82/77	.86	.89	.85/75	.89	.89	.87/71	.81	.83	.80/66
Thr3{1.5}	.89	.83	.84/82	.86	.84	.85/81	.82	.80	.86/87	.88	.86	.84/78	.89	.84	.83/70
Thr4{2.0}	.83	.82	.81/85	.84	.86	.83/85	.85	.86	.84/85	.87	.89	.85/86	.86	.83	.82/73
Spot	.91	.94	.80/86	.92	.88	.89/85	.82	.88	.86/86	.87	.87	.86/85	.88	.86	.88/74
Image 118035 'Church'															
Thr1{.5}	.83	.82	.81/51	.84	.86	.83/53	.85	.86	.84/71	.87	.89	.85/56	.86	.83	.82/55
Tr2{1.0}	.89	.83	.84/50	.86	.84	.85/55	.82	.80	.86/73	.88	.86	.84/58	.89	.84	.83/57
Thr3{1.5}	.87	.84	.83/54	.87	.87	.85/61	.82	.80	.84/79	.84	.85	.86/63	.82	.80	.86/63
Thr4{2.0}	.83	.82	.81/55	.84	.86	.83/62	.85	.86	.84/78	.87	.89	.85/64	.86	.83	.82/65
Spot	.90	.87	.88/56	.87	.87	.86/64	.86	.89	.87/80	.87	.87	.85/66	.88	.87	.86/67
Image 172032 'boats'															
Thr1{.5}	.89	.84	.83/43	.87	.87	.85/35	.82	.80	.84/45	.87	.85	.87/69	.82	.80	.86/47
Tr2{1.0}	.85	.87	.87/52	.84	.8	.85/43	.82	.80	.86/54	.85	.83	.85/72	.84	.83	.85/54
Thr3{1.5}	.87	.84	.85/53	.87	.87	.85/42	.82	.80	.86/65	.87	.87	.87/67	.82	.80	.81/54
Thr4{2.0}	.89	.83	.84/64	.87	.85	.86/48	.83	.84	.86/64	.82	.84	.85/69	.82	.80	.86/62
Spot	.92	.88	.88/63	.89	.88	.87/48	.86	.88	.89/64	.87	.87	.86/71	.88	.86	.88/61
Image 112082 'rhinos'															
Thr1{.5}	.87	.84	.84/33	.87	.87	.85/31	.82	.80	.86/30	.87	.87	.85/29	.82	.80	.86/30
Tr2{1.0}	.84	.83	.88/39	.87	.87	.85/37	.82	.80	.86/36	.87	.87	.85/34	.82	.80	.86/36
Thr3{1.5}	.88	.87	.83/45	.87	.87	.85/41	.82	.80	.86/43	.87	.87	.85/38	.82	.80	.86/41
Thr4{2.0}	.87	.84	.83/48	.87	.87	.85/44	.82	.80	.84/43	.84	.85	.86/40	.82	.80	.86/44
Spot	.89	.84	.80/56	.87	.87	.85/45	.82	.80	.86/45	.87	.87	.85/41	.82	.80	.86/45
Image 304034 'wild cat'															
Thr1{.5}	.56	.67	.62/13	.58	.66	.61/15	.67	.68	.63/20	.59	.67	.62/12	.52	.68	.65/15
Tr2{1.0}	.53	.58	.53/17	.58	.63	.63/19	.52	.66	.58/24	.55	.58	.55/15	.56	.60	.56/19
Thr3{1.5}	.57	.64	.62/21	.59	.67	.66/22	.62	.64	.65/28	.58	.67	.64/18	.62	.63	.62/22
Thr4{2.0}	.59	.67	.65/22	.57	.64	.62/28	.67	.60	.65/35	.67	.66	.67/28	.67	.70	.68/28
Spot	.59	.64	.63/31	.68	.67	.66/29	.66	.69	.67/36	.65	.67	.65/29	.68	.67	.66/28

Table 5.1: Comparison of human images.

The proposed algorithm for segmentation of human images; data from the Berkeley Segmentation dataset [26] P, R and F defined in equations 5.2, and 5.3 were computed in different conditions.

of the proposed algorithm is that if the image and the background have the same scale pixel and the same average gray value, the proposed algorithm will be able to distinguish correctly the background from the foreground.

To determine the performance of SIBIS, we measure the precision-recall curve with F-measure, with respective parameters P, R, and F in (5.2). In this experiment the SIBIS algorithm was tested with two different dental images shown in Figure 5.10. Their respective precision-recall values, and F-measures are illustrated in Figure 5.11.

Table 5.4 illustrates a valuable evaluation of the obtained segmentation through the precision-recall measure. The segmentation was established by using different threshold values. With this setting for both parameters, P is equal to 0.89 and R is equal to 0.84

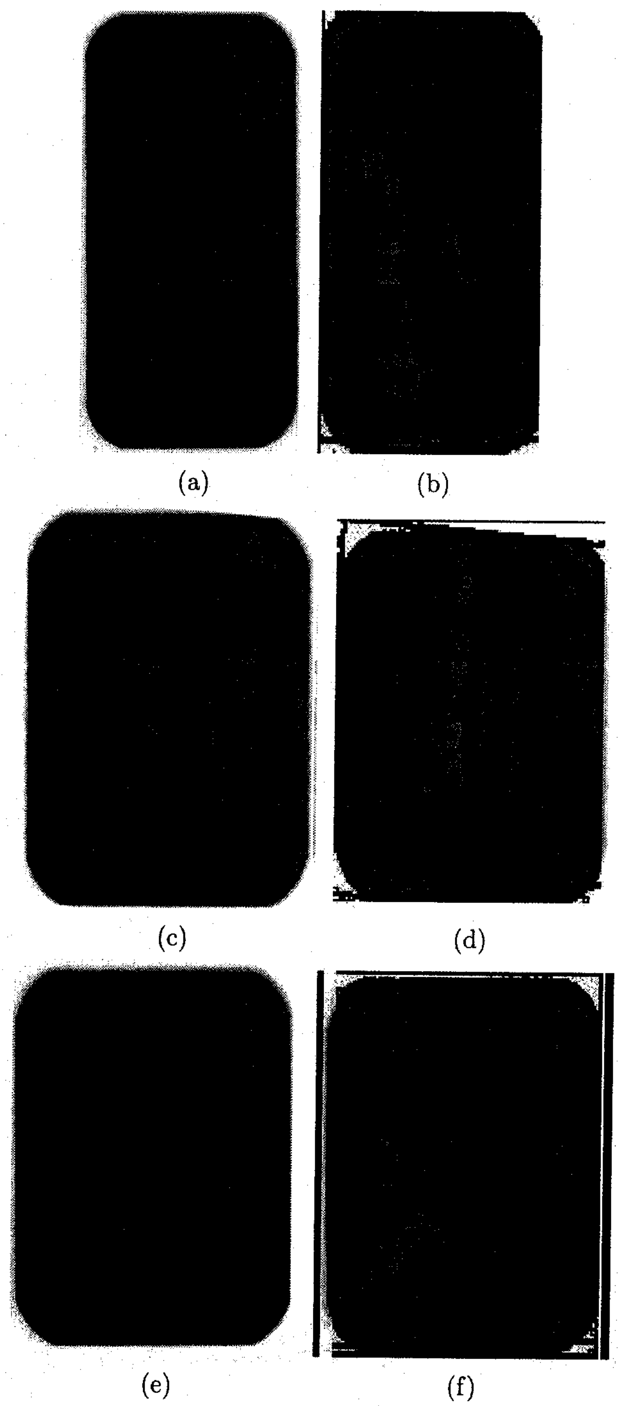


Figure 5.9: Detecting the filled and cavity area of three dental images.

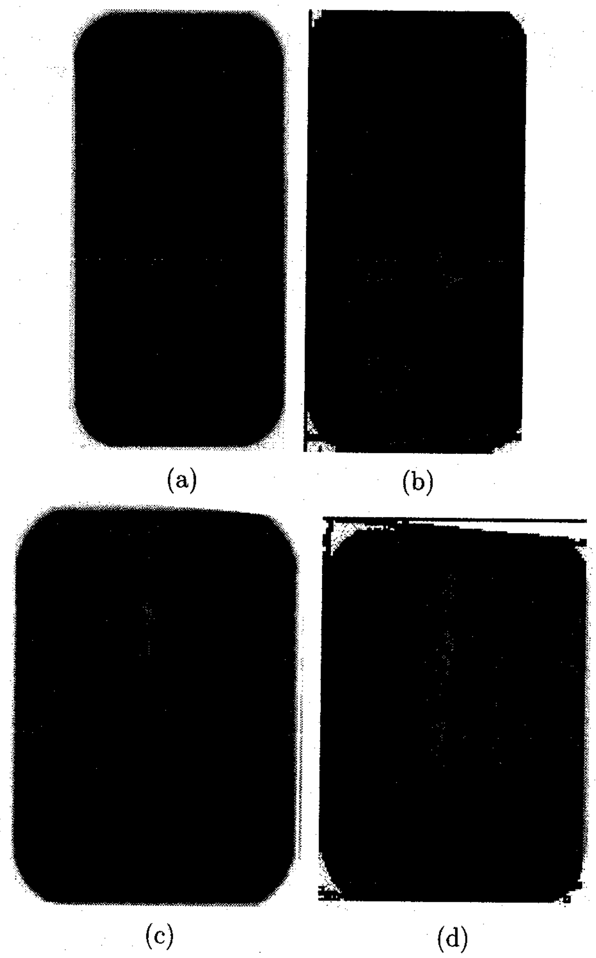


Figure 5.10: Boundary detection in radiographic image: (a) and (c) Original image of four and three teeth, (b) and (d) Detecting the filled and cavity area of the tooth.

for the results displayed in Figure 5.10. The value of the threshold varies between -2 and 2 for the test images. If the comparison is limited to Gray and White, the value of the threshold increases to 3 and to 4 (see Figure 5.11). To have optimized segmentation, a value of 0.86 is considered.

three radiographic images	Image:5.9 (a)			Image:5.9 (c)			Image:5.9 (e)		
	P	R	F	P	R	F	P	R	F
Thr1{.5}	0.88	0.84	0.85	0.87	0.84	0.85	0.82	0.80	0.80
Thr2{1.0}	0.89	0.86	0.87	0.89	0.87	0.87	0.84	0.81	0.82
Thr3{1.5}	0.87	0.85	0.86	0.86	0.85	0.86	0.87	0.83	0.84
Thr4{2}	0.85	0.88	0.86	0.88	0.88	0.88	0.85	0.82	0.83
Spot	0.86	0.81	0.83	0.86	0.81	0.83	0.86	0.80	0.82

Table 5.2: Testing SIBIS with the three radiographs
of Figure 5.9. Quantities P, R and F were computed for different conditions.

5.5 Segmentation of Brain Images

To apply the SIBIS algorithm on brain images we tested with two different types of brain images shown in Figure 5.12 along with the obtained results.

Figure 5.13 illustrates how the boundary detection works in the gray area of the images. The proposed algorithm properly shows all the gray texture in the background and correctly extracts the boundaries. The SIBIS algorithm is able to extract shapes and dark areas from the gray parts of the brain with fine scale of background textures. SIBIS clearly extracts the boundaries while the threshold is increasing and nicely separates background from textures. This last result is one of the significant advantages of the proposed algorithm.

Table 5.5 illustrates a valuable evaluation of the obtained segmentation that we are using the precision and recall values. Segmentation was established by using different threshold values. With this setting for both parameters P and R are taken to be 0.83 and 0.80, respectively.

Again, to measure the performance of SIBIS, we determine the precision-recall curve and F-measure. The results are shown in Figure 5.13 in Table 5.5 and Diagram 5.14 for the brain images of Figures 5.13a and 5.13b respectively.

Two Brain images	Image:5.12a (c)			Image:5.12b (d)		
	P	R	F	P	R	F
Thr1{.5}	0.83	0.80	0.81	0.82	0.80	0.81
Thr2{1.0}	0.87	0.83	0.84	0.84	0.81	0.82
Thr3{1.5}	0.86	0.85	0.85	0.87	0.83	0.85
Thr4{2}	0.88	0.85	0.86	0.85	0.82	0.84
Spot	0.86	0.82	0.83	0.86	0.84	0.85

Table 5.3: Testing SIBIS with two brain images of Figure 5.12.

5.6 Conclusion

The quality of the proposed segmentation is evaluated by measuring the quantities P and R introduced by Martin et al. [26]. These parameters provide a good measurement of the quality of segmentation and edge detection when ground truth data are available. Optimal segmentation has values of P, R and F equal to 1. The proposed algorithm has some adequate major advantages that can be considered: It is able to perform segmentation and boundary detection on different types of images. Indeed, it provides sufficient and adequate results. Another advantage that is very valuable is that it is lossless and that the threshold is obtained automatically in all image segmentations.

We can categorize the features of our proposed algorithm as follows:

- The major limitation of the proposed segmentation is the comparison with human segmentation especially in large human images.
- Humans extract objects from the background very well and our proposed algorithm performed in most of the images as well.
- The proposed segmentation, which does not use any a priori knowledge in images, in some cases is able to recognize and detect the boundaries around the objects in images sufficiently.
- The proposed algorithm is able to separate the objects from background. As is shown in Figure 5.7.

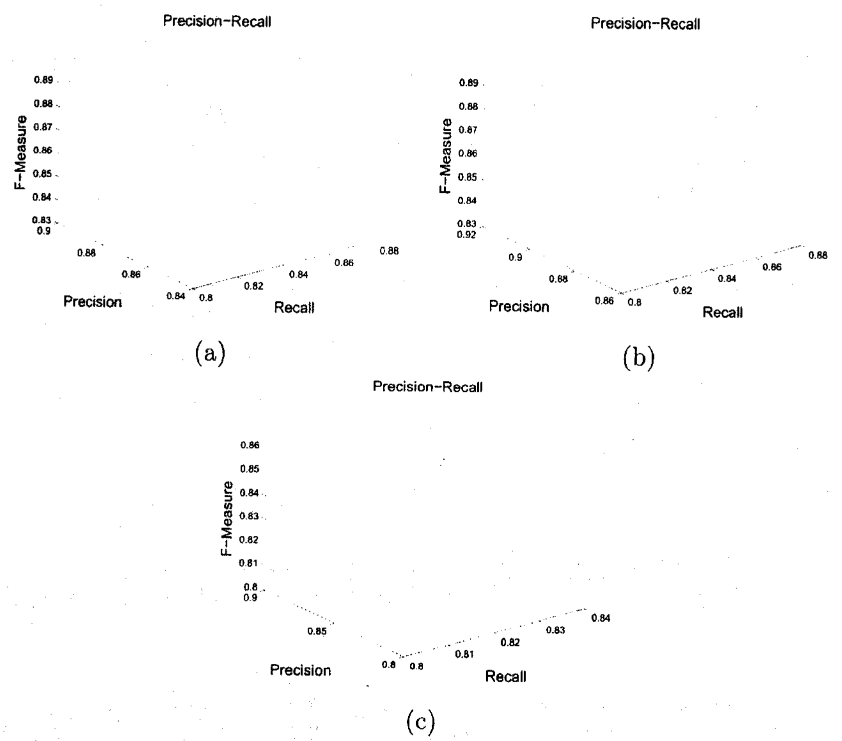


Figure 5.11: Diagram of result of F-Measure for Table 5.5 respectively

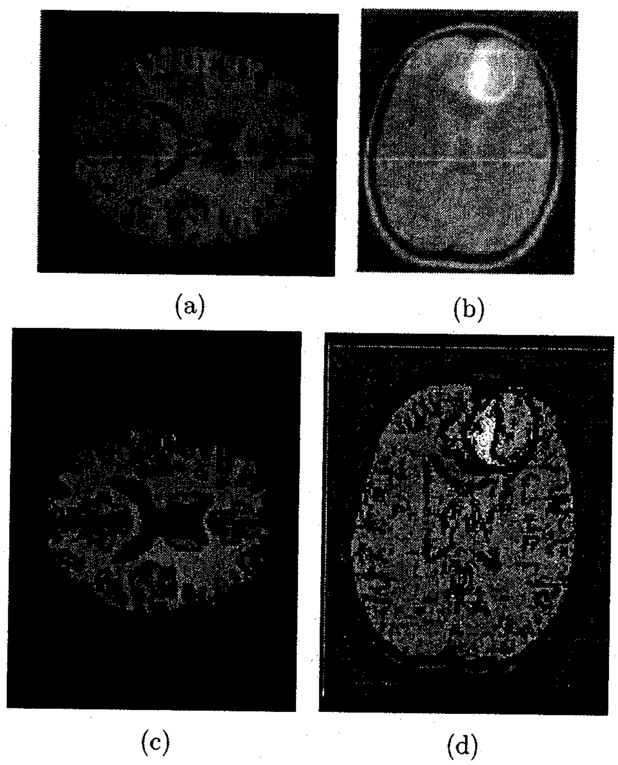
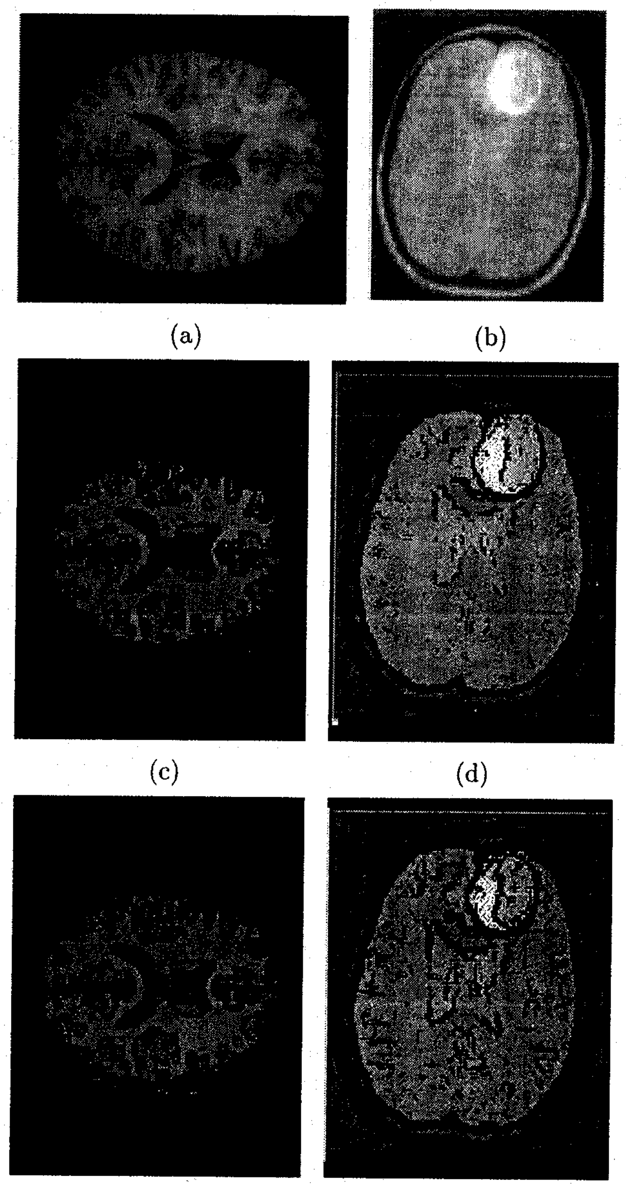
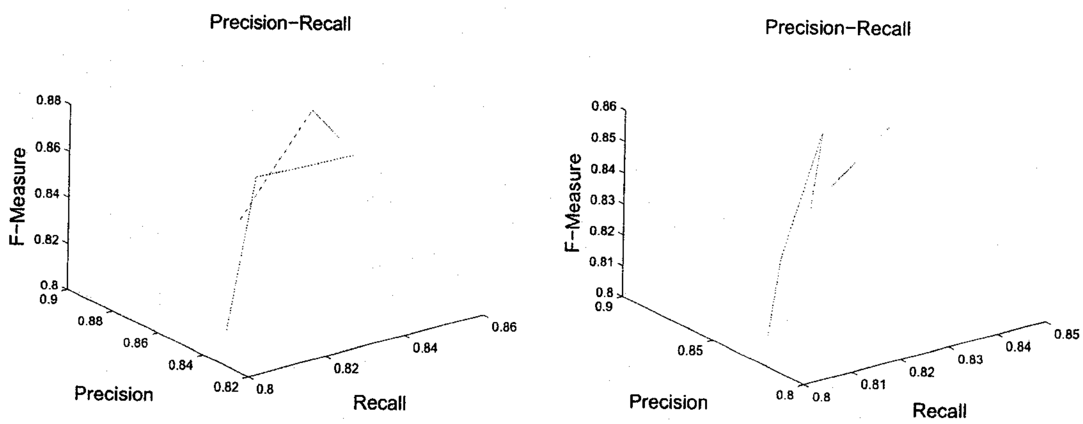


Figure 5.12: Segmentation of image: (a) and (b) Original of different brain images, (c) and (d) Detecting the boundaries with fine scale.



(a) (b) (c) (d) (e) (f)

Figure 5.13: (a) Original image of high gray texture of brain image. (c) and (e) Detecting the boundaries by testing with different thresholds. (b) Original image of another brain image. (d) and (f) Detecting the boundaries by testing with different thresholds.



(a) (b)

Figure 5.14: Diagram of measurement results for parameter P, R, and F in Table 5.5

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis we described the design and implementation of a Swarm Intelligence-Based Image Segmentation Algorithm (SIBIS). Here, we mention some of the advantages of SIBIS as the following:

- Better performance that a similar well recognized algorithm
- The proposed segmentation does not use any a priori knowledge at images
- No pre or post-processing of image unlike many of its kind do
- When the object and the background have a similar texture, our proposed algorithm still is able to detect the objects from background
- Distributivity and scalability which lead to sustaining the same performance in distributed systems.

6.2 Future Work

Important work still remains to be done. First, we need to work on dynamic group management in SIBIS. In the current version of SIBIS, we assume that the agents lead to segmentation and collaboration with environment and interact with other agents in terms of communication and segmentation. The cellular automata techniques are there to guide the agents in the environment.

Another possible future research avenue is to apply conventional or fuzzy image processing and segmentation approach by swarm and agent-based techniques. By doing this, we can have a better view in comparison with both approaches in terms of results and performances.

Appendix A

SIBIS Implementation

A.1 Introduction

There are several applications integrated with the current version of the Swarm-based system that we developed as described in previous sections and shown in Figures A.1, A.2 and A.3. Figure A.1 shows SIBIS at the beginning without loading any image in the environment and all variables set to default values. Figure A.2 illustrates the system after marking the area of segmentation or boundary detection and loading the image in the patch or environment. Figure A.3 displays the SIBIS after boundary detection of the loaded image.

Those applications mentioned above are Netlogo, Segmentation, Boundary Detection, Finding Direction, Finding Threshold, Setup the Environment Variable, Import the image, Clean/Reset the Environment and other features that we briefly explain in the following sections. This section provides the implementation of each of the applications mentioned above. During developing the Proposed Swarm Intelligence Based Image Segmentation (SIBIS) algorithm, we used Netlogo [41] (most recent version) released as follows:

- NetLogo 3.1beta1 (February 7, 2006)
  Java VM: 1.5.0.05 (Sun Microsystems Inc.; 1.5.0.05-b05)
  Operating System: Windows XP 5.1 (x86 processor)
  Java heap: used = 11 MB, free = 10 MB, max = 508 MB
  JOGL: (3D View)
  OpenGL Graphics: (3D View)

A.2 Netlogo Environment

NetLogo is an agent-based environment used for the following circumstances:

1. Programming
2. Modeling
3. Simulating
4. Multi-agent modeling
5. Simple to use

Netlogo is well developed to model complex systems. Netlogo is a fully agent-based environment. And based on this strong feature; developers are able to handle enormous individual and independent agents. Netlogo also has the following features in system platform, environment and language.

- System: Run in different platforms such as Windows. Linux and Mac.
- Language
  Fully programmable, simple structure, mobile Agent, link between agents and environment.
- Environment
  As an environment Netlogo has strong capabilities such as:
  1. Netlogo provides environment for 2D and 3D view for modeling and simulation
  2. Different and Potable shape for agents.
  3. Agent and Patches are able to have label.
  4. Good Interface such as: buttons, sliders, switches, choosers, monitors, text boxes, etc.
  5. Strong plotting tools.
- Web-based features
  In Netlogo all models can be saved as applet and users are able to run their system in the web [41].

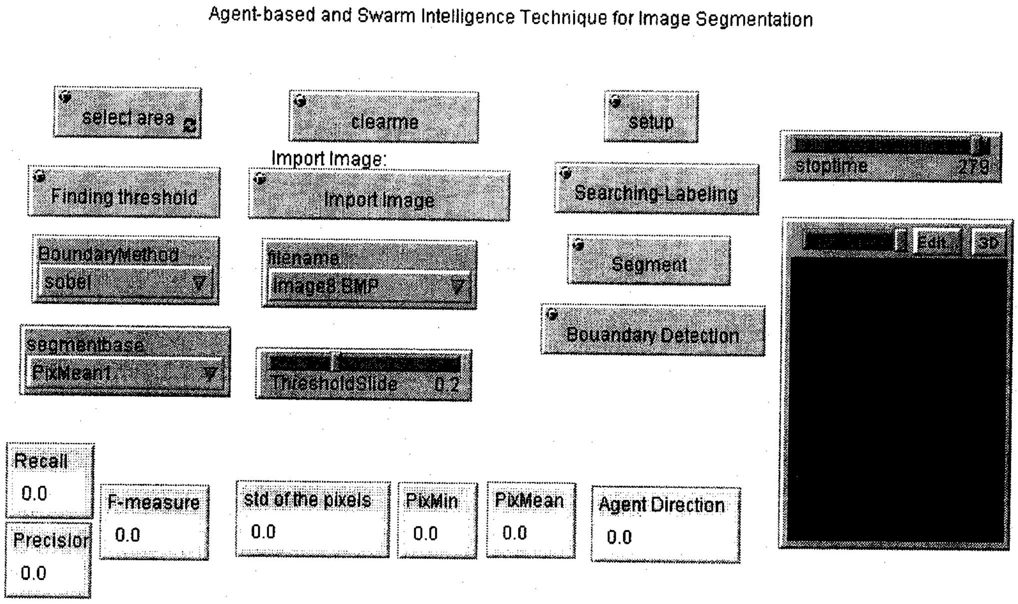


Figure A.1: SIBIS system with default values before processing the image

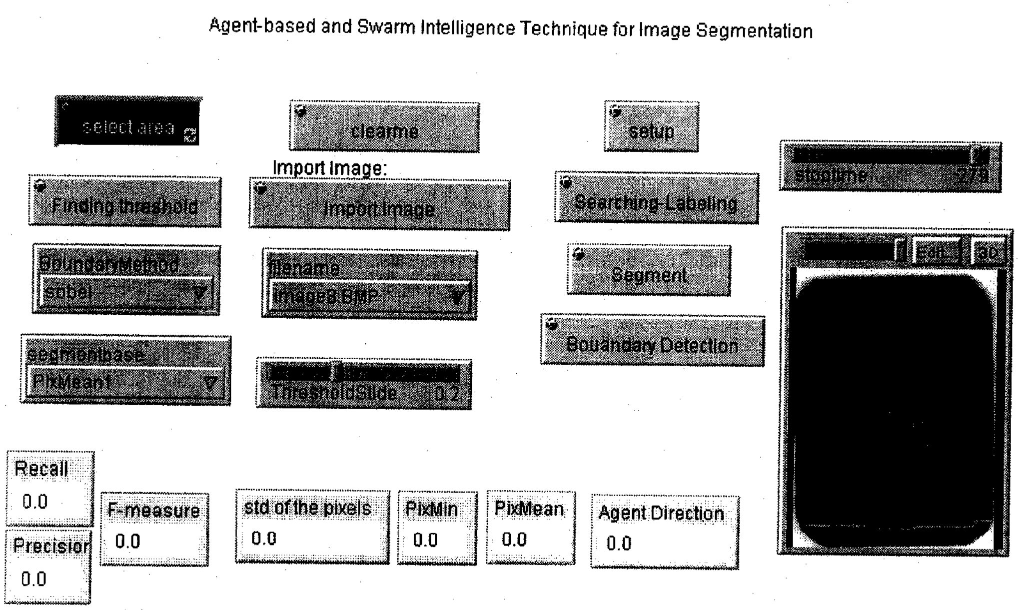


Figure A.2: SIBIS system with default values after importing the sample image into the environment

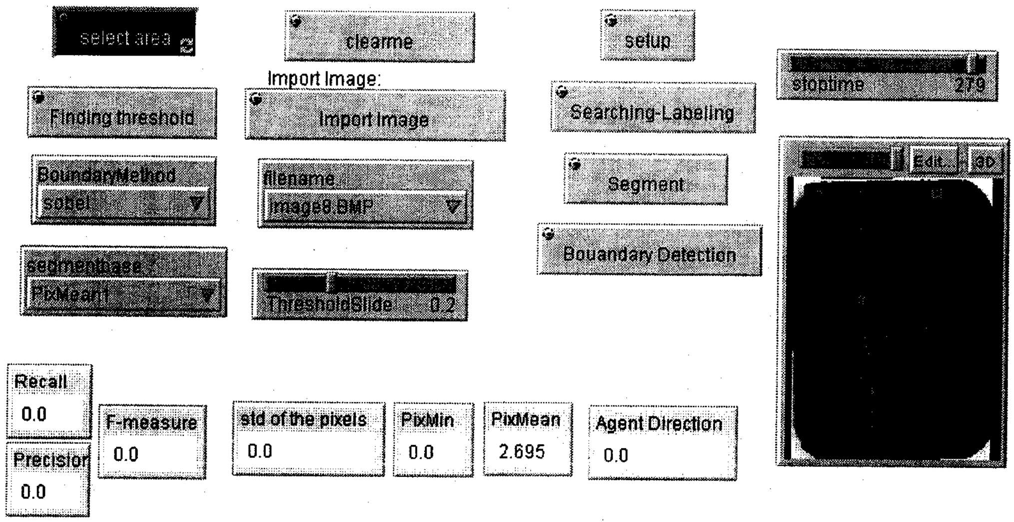


Figure A.3: SIBIS system with default values after processing the image and finding the boundaries in the sample image

A.3 Clear/Reset: Environment, Agents and Variables

In this function we clean and clear the environment and other variables for new simulation and testing. So this function allows the system to reset all following variables to zero. It resets all global variables to zero, and calls clear-turtles, clear-patches,

- All global variables set to zero
- All agents: by calling *clear-agents*; resets and clear all agent to default values.
- Clear-patch and environment: The important part is Environment or Patch to be cleaned by *clear-patches.*
- Clear-output: Reset and clear all the outputs variables such as text and other output variable area.
- Clear-drawing: Clear any shapes, stamps, labels or lines drawn by agents during simulation by agents.

A.4 Import Images

Netlogo is able to import two kinds of images into the environment as follows

1. Pcolor-image
   SIBIS allows users to scale the size of the environment or patches. So based on the size of environment users are able to import their image into the environment with the same ratio transfer. The command that imports the image into the patch is called *import-pcolor*. By using the *import-pcolor* each pixel of the image is considered as a patch in the environment. This strong ability in Netlogo allows agents to sense the pixel or even to sense the total image. Hence, agents have the ability to process, analyze and perform other operations on the image.

2. Drawing-image
   This property allows users to import the image as a *drawing*. The scale and size of the image is based on the scale and size of the environment such as *pcolor* image. In the drawing aspect agents are not able to sense the image as they can do by *pcolor* format.

In both *pcolor* and *drawing* format Netlogo supports the following image formats: BMP, JPG, GIF, and PNG.

A.5 Setting the System

In the *Setup* we set and initialize the SIBIS variables and other parameters to default values. Hence, the SIBIS initials and sets the following parameters.

- Define and set Global variables
  Define and set the following global variables: MycounterThr, F-measure, Recall, Precision1 Mainthreshold, Mouslistymin, Mmouslistymax, Mouslistxmax, Mouslistxmin, Mouslistxcor, PixMean, PixMax, PixMin, PixThresh, Testcolor, Stdpercentage, Agentdirection, Countagents, Std, Meanvalue1, Hmgpix, Localrandom, Localdistance, Mylistagent, Randomitem, Clockvar, and Randompic.

- Define and set own-patches or image variables
  During the image processing SIBIS defines and sets different variables for imported image and patch as an environment. These variables help SIBIS to compute and calculate the Boundary detection, Segmentation, and other Image parameters as well. When Image imports into the patch as *pcolor* format the SIBIS sets and

computes the following variables such as RGB values and original values of each pixel. Some significants ones are:

- *pix-x*: This variable displays the *x-Coordinate* of each pixel in the system.
- *pix-y*: This variable displays the *y-Coordinate* of each pixel in the system.
- *pix-r*: Set to original *red* values for each pixel in the image
- *pix-g* : Set to original *blue* values for each pixel in the image
- *pix-b*: Set to original *green* values for each pixel in the image
- *pix-c*: Set to original real values and color of each pixel in the image.
- *pix-l*: During the image processing SIBIS defines a *tag, and labels* for each pixel. So variable *pix-l* sets each pixel labeled and tagged during the processing.

- Define and set own-agents variables
  In SIBIS *agents* have their own variable such as patch and environment that are mentioned above. The most important variables that agents carry during the process are the following:

  - *Size*: This variable sets the size of each agent in the environment. It is important to mention agents are able to have different sizes.
  - *Color*: Sets the color of each agent. Each agent can be assigned a different color. This depends on the mission and behavior of each agent during processing in the environment.
  - *Direction*: Keep the direction of each agent in each step time. As SIBIS is a fully agent-based environment, so agents are ever moving in the environment randomly. Hence, it is clear that each agent has a different direction in each step time clock. To avoid collision and other difficulties, SIBIS needs to know the direction of each agent during process.
  - *Coordinates*: Keeps the coordinates of agents in each step time.
  - *Id*: This variable keeps the identification of agents. While agents sprout into the environment, each agent is assigned an individual and specific ID. And they carry this ID as long as they live. So SIBIS is able to control all agents during the process.
  - *HmgPix*: Keeps the number of homogeneous pixels that are found by each agent.

- *Speed*: Sets speed of agents. In environment each agent is assigned a different speed and move forward. So by this variable SIBIS has the capability to maintain, increase, or decrease the speed of each agent.

- Sprout agent into the environment
  Distributing agents in the environment are one of the necessary parts of SIBIS. Broadcasting the agent in the environment is important somehow based on the following reasons:

  - To avoid collisions between agents
  - Easier movement forward through the image, environment and patches
  - Efficient number of agents in the patches
  - Performance and Speed of system during processing the image

  In order to achieve the above parameter, SIBIS uses the efficient randomize distribution to sprout the agent into the environment. In Netlogo different distribution statistics are provided. These include *exponential, gamma-alpha-lambda), normal mean standard-deviation, and poisson* distribution. SIBIS uses *random mean-standard-deviation* distribution to sprout the agent into the environment. To do so, we use
  random-normal mean standard-deviation:
  To create number of agents for imported-image into the patch. We found that the *mean* and *standard-deviation* are needed to make computations. Then we sprout the agent randomly into the environment as below:
  sprout-agents number [commands]
  By running this function the agents are able to be assigned immediately: size, color, heading and other properties of each agent. During running the command part, all of the agents were able to run and operate. After all agents initialized properly.

A.6 Segmentation Part

In this part SIBIS segments the image based on segmentation methods that we choose from $Segmentation - method - slider$. To do segmentation, SIBIS calls the Algorithm 1 that is described in 4.2 briefly. Here we propose what is done during the implementation part. Also the agents rules and their operations are mentioned.

1. Ask agent a_i to do Segmentation

2. Apply Cellular Automata Rules for agent a_i
3. Agent a_i calls Segmentation algorithms 1
4. If pixel p_i belongs to one of the segmentation methods that is chosen from *Segmentation-slider* Then
   Label the current pixel p_i
5. Repeat and start from step one until system *stoptime* is achieved.

After this algorithm is completed, the SIBIS is able to recognize the labeled pixels that were found by agents. Then easily labeled pixels are selected and displayed as a segmentation part.

A.7 Boundary Detection Part

In this part SIBIS determines the Boundary detection of the image based on Boundary Detection methods chosen from $Boundary - method - Slider$. To do so SIBIS calls the algorithm 1 that is described in 4.2. Here we describe the implementation part of Boundary Detection and rules of agents during boundary detection.

1. Ask agent a_i to do Boundary Detection
2. Apply Cellular Automata Rules for agent a_i
3. Agent a_i calls Boundary Detection algorithms 1
4. If pixel p_i belongs to one of the Boundary Detection methods that is chosen in *Boundary-slider* Then
   Label the current pixel p_i
5. Repeat and start from step one until system *stoptime* is achieved.

References

[1] Yacine Akbir and A. Relhadj-Aissa. Distributed image segmentation system by a multi-agent approach. In *Proceedings of the Euro PVM/MPI*, pages 138–147, 2002.

[2] V.J. Blue and J.L. Adler . Modeling four-directional pedestrian movements. *Transportation Research Record*, 1:1, 2000.

[3] V.J. Blue and J. L. Adler. Emergent fundamental pedestrian flows from cellular automata microsimulation. *Transportation Research Record*, 1644:29–36, 1998.

[4] V.J. Blue and J.L. Adler. Cellular automata microsimulation of bi-directional pedestrian flows. transportation research record,. *Journal of the Transportation Research Board*, 1678:135–141, 2000.

[5] V.J. Blue and J.L. Adler. Cellular automata model of emergent collective bi-directional pedestrian dynamics. *The Seventh International Conference on the Simulation and Synthesis of Living Systems Directional Pedestrian Dynamics*, 2:1, 2000.

[6] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press, 1999.

[7] E.G.P Bovenkamp, J. Disjkstra, J.G. Bosch, and J.H.C. Reilber. Collaborative multi-agent ivus image segmentation. In *Proceedings of the MICCAI*, pages 1185–1186, 2001.

[8] K. Bowyer, C. Kranenburg, and S. Dougherty. Edge detector evaluation using empirical roc curves. *Proc. of Conf. Computer Vision and Pattern Recognition*, 1:20–31, 1999.

[9] Thomas Braunl. *Parallel Image Processing.* Springer, http://www.springer.de, 2002.

[10] J. Canny. A computational approach to edge detection. *IEEETrans. Pattern Analysis and Machine Intelligence*, 8:679–698, 1986.

[11] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. In *Proc. IEEE Int. Conf. Computer Vision*, pages 694–699, 1995.

[12] V. Caselles, R. Kimmel, G. Sapiro, and C. Sbert. Minimal surfaces based object segmentation. *IEEE Trans. Pattern Anal. Mach. Intel.*, 1:694–699, 1995.

[13] J.R Mortinez de Dois and A. Ollero. A multiresolution threshold selection method based on training. In *Proceeding of the ICIAR*, pages 90–97, 2004.

[14] Joshua M. Epstein and Probert Axtell. *Growing Artificial Societies.* MIT Press, 1998.

[15] A. F. Goldszal, C. Davatzikos, D. L. Pham, M. X. H. Yan, R. N. Bryan, and S. M. Resnick. An image processing system for qualitative and quantitative volumetric analysis of brain images. *J. Comput. Assist. Tomog.*, 1:827–837, 1998.

[16] Rafael Gonzalez and Richard Woods. *Digital Image Processing.* Printice Hall, www.prenhall.com, 2002.

[17] K. Held. Markov random field segmentation of brain mr images. *IEEE Trans. of Medical Image*, 1:878–886, 1997.

[18] X. Hu, R. C. Eberhart, and Y. Shi. Swarm intelligence for permutation optimization: a case study on n-queens problem. In *IEEE Swarm Intelligence Symposium 2003, Indianapolis, IN, USA*, 2003.

[19] Bernd Jahne. *Digital Image Processing.* Springer, http://www.springer.de, 5th edition, 2002.

[20] J.R. Jang. Anfis: Adaptive-network-based fuzzy inference systems. *IEEE Transactionson Systems, Man and Cybernetics*, 23(3):665–685, 1993.

[21] Fazel Keshtkar and Wail Gueaieb. Conventional and agent-based image segmentation algorithms: A comparative study. In *14th Iranian Conference on Electrical Engineering ICEE'2006, Tehran, Iran*, 16-18 May 2006.

[22] Fazel Keshtkar, Wail Gueaieb, and Anthony White. An agent-based model for image segmentation. In *13th Multi-disciplinary Iranian Researchers Conference in Europe (IRCE2005), Leeds, United Kingdom*, July 2005.

[23] S. Konishi, A.L. Yuille, J. Coughlan, and S.C. Zhu. Fundamental bounds on edge detection: An information theoretic evaluation of different edge cues. *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, 1:573–579, 1999.

[24] Jiming Liu and Yuan Y. Tang. Adaptive image segmentation with distributed behavior-based agents. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1999.

[25] M. Luckenhaus and W. Eckstein. A multi-agent based system for parallel image segmentation. In *Parallel image segmentation.* Munich University of Technology, 1997.

[26] S. D. R. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):530–539, June 2004.

[27] I. Noda. Segmentation of environments using hidden markov modeling of other agents. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems*, July 2002.

[28] J. Park and J.M Keller. Snake on the watershed. In *IEEE Transaction on Pattern Recognition Analysis, and machine Vision*, pages 1201–1205, 2001.

[29] Roger Penrose. *The Emperor's New Mind.* ISBN 0-198-51973-7, 1989.

[30] DV. Ramos, F. Muge, and P. Pina. Self-organized data and image retrieval as a consequence of inter-dynamic synergistic relationships in artificial ant colonies. In *Proceedings of 2nd Int. Conf. on Hybrid Intelligent Systems*, volume 87, pages 500–509. IOS Press, 2002.

[31] C. J. Rojsbergen. *Information Retireval Butterworths.* Dept. of Computer Science, Univ. of Glasgow, 1979.

[32] Rui Shi, Huamin Feng, Tat-Seng Chua, and Chin-Hui Lee. An adaptive image content representation and segmentation approach to autmatic image annotation. In *Proceeding of CIVR*, pages 545–554, springer, 2004.

[33] Sonka, M. Hlavac, and V. Boyle. *Image Processing and Analysis and Machine Vision.* Pacific Grove, second edition, 1999.

[34] Baris Sumengen. *Variational Image Segmentation and Curve Evolution on Natural Images.* UC, Santa Barbara, September 2004.

[35] I. Vahamel and H. Sahli. Multi scale gradient watershed of color images. In *IEEE Transaction on Image Processing*, pages 617–626, 2003.

[36] Xlei Wang, Hongbing Ji, and Xinbo Gao. Image segmentation by a robust clustering algorithm using gaussian estimator. In *International Conference on Image Analysis and Recognition*, pages 74–81, Springer, 2004. Springer-Verlag Berlin Heidelberg.

[37] Anthony White. *Swarm Intelligence: A Gentle Introduction with Applications.* http://www.sce.carleton.ca/netmanage/tony/swarm.html.

[38] Anthony White. *Swarm Intelligence Course, Carleton University.* http://www.scs.carleton.ca/arpwhite/courses/reports/fazel-winter-2005.pdf, 2005.

[39] Charles E. White, Gene A. Tagliarini, and Sridhar Narayan. An algorithm for swarm-based color image segmentation. In *Proceedings of the Euro PVM/MPI*, 2004.

[40] The Encyclopedia Wikipedia. *Swarm Intelligence.* http://www.wikipedia.com.

[41] Uri Wilensky. *NetLogo. http://ccl.northwestern.edu/netlogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL*, 1999.

[42] Stephen Wolfram. *A New Kind of Science.* Wolfram Media, 2002.

[43] Xiangyu Yang and Shankar Krishnan. Image segmentation using mixture and spatial information. In *Proceeding of Image and Vision Computing*, pages 735–745. Elsevier, 2004.