

**SOFT COMPUTING APPLIED TO OPTIMIZATION,
COMPUTER VISION AND MEDICINE**

DISSERTATION ZUR ERLANGUNG DES GRADES EINER DOKTORIN DER
NATURWISSENSCHAFTEN (DR. RER. NAT.) AM FACHBEREICH MATHEMATIK
UND INFORMATIK DER FREIEN UNIVERSITÄT BERLIN

VON

MARGARITA ARIMATEA DIAZ CORTES

AUS MEXIKO

BERLIN, 2019

SOFT COMPUTING APPLIED TO OPTIMIZATION, COMPUTER VISION AND MEDICINE

DISSERTATION

Autorin: Margarita Arimatea Diaz Cortes

Betreuer:

Prof. Dr. Raúl Rojas
Dahlem Center for Machine Learning and Robotics
Institut für Mathematik und Informatik
Freie Universität Berlin

Erstgutachter:

Prof. Dr. Raúl Rojas
Dahlem Center for Machine Learning and Robotics
Institut für Mathematik und Informatik
Freie Universität Berlin

Zweitgutachter:

Dr. Erik Cuevas
Departamento de Computación
Universidad de Guadalajara

Tag de Disputation: 05.07.2019

Selbständigkeitserklärung

Hiermit versichere ich, dass ich alle Hilfsmittel und Hilfe angegeben habe und auf dieser Grundlage die Arbeit selbstständig verfasst habe.

Ich erkläre weiterhin, dass die Arbeit nicht schon einmal in einem früheren Promotionsverfahren eingereicht wurde.

Berlin, den 21. Januar 2019

Margarita Arimatea Diaz Cortes

Acknowledgements

I want to thank to the members of the work group in the Freie Universität Berlin, they have contributed immensely to my personal and professional time here. The group has been a source of friendships as well as good advice and collaboration. Particularly I would like to thank Daniel and Tobias for their support and advises on my dissertation.

My time in Berlin was made enjoyable in large part due to the many friends that became a part of my life, and now they are my family here. I am grateful for the time spent with my Mexican-Berlin family, Mariana, José, Ricardo, Omar, Fernando, Salvador, Sebastián, and Karina, thank you all for the memorable times together, and all the help you gave me during these last years, without you this would have not been possible.

I would like to thank my mom for all their love and encouragement, you are always my rock and my support, to my brothers who supported me in all my pursuits. Mostly, I want to thank Jost; you became part of my life and my family during this adventure too, thanks to your patience, encouragement and love, life has a new meaning.

Lastly, I gratefully acknowledge the funding sources that made my Ph.D. work possible. My work was supported by the Consejo Nacional de Ciencia y Tecnología and the Heinrich Böll Foundation.

Thank you.

Summary

Artificial intelligence has permeated almost every area of life in modern society, and its significance continues to grow. As a result, in recent years, Soft Computing has emerged as a powerful set of methodologies that propose innovative and robust solutions to a variety of complex problems. Soft Computing methods, because of their broad range of application, have the potential to significantly improve human living conditions. The motivation for the present research emerged from this background and possibility.

This research aims to accomplish two main objectives: On the one hand, it endeavors to bridge the gap between Soft Computing techniques and their application to intricate problems. On the other hand, it explores the hypothetical benefits of Soft Computing methodologies as novel effective tools for such problems.

This thesis synthesizes the results of extensive research on Soft Computing methods and their applications to optimization, Computer Vision, and medicine. This work is composed of several individual projects, which employ classical and new optimization algorithms.

The manuscript presented here intends to provide an overview of the different aspects of Soft Computing methods in order to enable the reader to reach a global understanding of the field. Therefore, this document is assembled as a monograph that summarizes the outcomes of these projects across 12 chapters. The chapters are structured so that they can be read independently. The key focus of this work is the application and design of Soft Computing approaches for solving problems in the following: Block Matching, Pattern Detection, Thresholding, Corner Detection, Template Matching, Circle Detection, Color Segmentation, Leukocyte Detection, and Breast Thermogram Analysis.

One of the outcomes presented in this thesis involves the development of two evolutionary approaches for global optimization. These were tested over complex benchmark datasets and showed promising results, thus opening the debate for future applications. Moreover, the applications for Computer Vision and medicine presented in this work have highlighted the utility of different Soft Computing methodologies in the solution of problems in such subjects. A milestone in this area is the translation of the Computer Vision and medical issues into optimization problems.

Additionally, this work also strives to provide tools for combating public health issues by expanding the concepts to automated detection and diagnosis aid for pathologies such as Leukemia and breast cancer. The application of Soft Computing techniques in this field has attracted great interest worldwide due to the exponential growth of these diseases.

Lastly, the use of Fuzzy Logic, Artificial Neural Networks, and Expert Systems in many everyday domestic appliances, such as washing machines, cookers, and refrigerators is now a reality. Many other industrial and commercial applications of Soft Computing have also been integrated into everyday use, and this is expected to increase within the next decade. Therefore, the research conducted here contributes an important piece for expanding these developments. The applications presented in this work are intended to serve as technological tools that can then be used in the development of new devices.

Zusammenfassung

Künstliche Intelligenz hat nahezu jeden Lebensbereich der modernen Gesellschaft durchdrungen, und ihre Bedeutung nimmt stetig zu. In diesem Sinne hat sich auch Soft-Computing in den letzten Jahren zu einem leistungsfähigen Methodenset entwickelt, das innovative und robuste Lösungen für eine Vielzahl komplexer Probleme bietet. Soft-Computing-Methoden haben aufgrund ihrer vielfältigen Anwendungsmöglichkeiten das Potenzial, die menschlichen Lebensbedingungen maßgeblich zu verbessern. Hieraus entstand die Motivation für die vorliegende Forschung.

Mit dieser Forschungsarbeit sollen zwei Hauptziele erreicht werden: Auf der einen Seite wird versucht, die Lücke zwischen Soft Computing-Techniken und deren Anwendung bei komplexen Problemen zu schließen. Auf der anderen Seite werden die hypothetischen Vorteile von Soft Computing-Methoden als neuartige und effektive Werkzeuge für solche Probleme untersucht.

Diese Dissertation fasst die Ergebnisse umfangreicher Forschungen zu Soft-Computing-Methoden und deren Anwendungen für Optimierungsprobleme, Bildverarbeitung und Medizin zusammen. Die Arbeit besteht aus mehreren Einzelprojekten, in denen klassische und neue Optimierungsalgorithmen verwendet werden.

Das hier vorgestellte Manuskript soll einen Überblick über die verschiedenen Aspekte des Soft-Computings geben, um dem Leser ein generelles Verständnis dieses Fachgebiets zu vermitteln. Daher ist dieses Dokument als Monographie zusammengestellt, welche die Ergebnisse dieser Projekte in zwölf Kapiteln bündelt. Die Kapitel sind so aufgebaut, dass sie unabhängig voneinander gelesen werden können. Der Schwerpunkt liegt auf der Anwendung und Gestaltung von Soft-Computing-Ansätzen zur Lösung folgender Probleme: Block Matching, Mustererkennung, Schwellenwert-verfahren, Eckenerkennung, Template Matching, Kreiserkennung, Farbsegmentierung, Leukozytenerkennung und Brustthermogrammanalyse.

Eines der Ergebnisse, die in dieser Arbeit vorgestellt werden, beinhaltet die Entwicklung von zwei evolutionären Ansätzen für die globale Optimierung. Diese wurden an komplexen Benchmark-Datensätzen getestet und zeigten vielversprechende Ergebnisse, was die Diskussion über zukünftige Anwendungen eröffnet. Darüber hinaus haben die hier dargestellten Anwendungen für Bildverarbeitung und Medizin den Nutzen verschiedener Soft-Computing-Methoden bei der Lösung von Problemen in solchen Gebieten hervorgehoben. Ein Meilenstein in diesem Bereich ist die Überführung von Bildverarbeitung und medizinischen Fragestellungen in Optimierungsprobleme.

Zudem zielt diese Arbeit darauf ab, Instrumente zur Bewältigung von Problemen der öffentlichen Gesundheitsfürsorge bereitzustellen, indem die Konzepte auf automatisierte Erkennungs- und Diagnosehilfen für Krankheiten wie Leukämie und Brustkrebs ausgeweitet werden. Die Anwendung von Soft Computing-Techniken in diesem Bereich hat aufgrund der exponentiellen Zunahme dieser Krankheiten weltweit großes Interesse geweckt.

Abschließend ist zu erwähnen, dass der Einsatz von Fuzzy-Logik, künstlichen neuronalen Netzen und Expertensystemen in vielen Haushaltsgeräten, wie Waschmaschinen, Küchenherden und Kühlschränken, heutzutage bereits Realität ist. Auch viele andere industrielle und kommerzielle Anwendungen von Soft Computing wurden in den Alltag integriert, was sich im kommenden Jahrzehnt noch verstärken wird. Die hier durchgeführte Forschung leistet daher einen wichtigen Beitrag zur Steigerung dieser Entwicklungen. Die vorgestellten Anwendungen sollen als technologische Werkzeuge dienen, die bei der Entwicklung neuer Geräte eingesetzt werden können.

Contents

OPTIMIZATION

1. INTRODUCTION.....	11
1.1. MOTIVATION.....	11
1.2. SOFT COMPUTING	15
1.2.1. Fuzzy logic	16
1.2.2. Neural Networks.....	17
1.2.3. Evolutionary computation.....	17
1.3. OPTIMIZATION CONCEPTS.....	18
1.3.1. Definition of an optimization problem	18
1.3.2. Classical optimization.....	19
1.3.3. Optimization with Evolutionary computation	21
1.4. RESEARCH OBJECTIVES.....	22
1.5. THESIS OVERVIEW	23
2. GLOBAL OPTIMIZATION USING OPPOSITION-BASED ELECTROMAGNETISM-LIKE ALGORITHM.....	27
2.1. INTRODUCTION	27
2.2. ELECTROMAGNETISM - LIKE OPTIMIZATION ALGORITHM (EMO)	29
2.2.1. Initialization	30
2.2.2. Local Search.....	30
2.2.3. Total force vector computation	30
2.2.4. Movement	32
2.3. OPPOSITION-BASED LEARNING (OBL)	32
2.3.1. Opposite Number	32
2.3.2. Opposite Point.....	33
2.3.3. Opposite-based Optimization.....	33
2.4. OPPOSITION-BASED ELECTROMAGNETISM-LIKE OPTIMIZATION ALGORITHM	34
2.4.1. Opposition-based Population Initialization	35
2.4.2. Opposition-based production for new generation.....	35
2.5. EXPERIMENTAL RESULTS	36
2.5.1. Test problems	36
2.5.2. Parameter settings for the involved EMO algorithms.....	38
2.5.3. Results	38
2.6. CONCLUSIONS	42
3. A METAHEURISTIC OPTIMIZATION METHODOLOGY BASED ON FUZZY LOGIC.....	44
3.1. INTRODUCTION	44
3.2. FUZZY LOGIC AND REASONING MODELS	46
3.2.1. Fuzzy logic concepts	47
3.2.2. The Takagi-Sugeno (TS) fuzzy model	47
3.3. THE FUZZY-BASED METHODOLOGY	49
3.3.1. Optimization strategy	49
3.3.2. Computational procedure	54
3.4. DISCUSSION ABOUT THE PROPOSED METHODOLOGY	55
3.4.1. Optimization algorithm	55
3.4.2. Modeling characteristics.....	56
3.5. EXPERIMENTAL STUDY.....	56
3.5.1. Performance evaluation with regard to its own tuning parameters.....	57

3.5.2. Comparison with other optimization approaches	61
3.6. CONCLUSIONS	76
3.7. LIST OF BENCHMARK FUNCTIONS	76

COMPUTER VISION

4. COLOR SEGMENTATION USING LVQ NEURAL NETWORKS..... 79

4.1. INTRODUCTION	79
4.1.1. Histogram thresholding and color space clustering	80
4.1.2. Edge detection	80
4.1.3. Probabilistic methods.....	80
4.1.4. Soft-Computing techniques	81
4.1.5. Scheme	81
4.2. BACKGROUND ISSUES	82
4.2.1. RGB Space Color	82
4.2.2. Artificial Neural Networks	82
4.3. COMPETITIVE NETWORKS	83
4.4. LEARNING VECTORS QUANTIZATION VECTORS	84
4.5. ARCHITECTURE OF THE COLOR SEGMENTATION SYSTEM.....	85
4.6. IMPLEMENTATION	86
4.7. RESULTS AND DISCUSSION	87
4.8. CONCLUSIONS	89

5. MOTION ESTIMATION ALGORITHM USING BLOCK-MATCHING AND HARMONY SEARCH OPTIMIZATION..... 91

5.1. INTRODUCTION	91
5.2. HARMONY SEARCH ALGORITHM.....	94
5.2.1. The Harmony Search Algorithm.....	94
5.2.2. Computational procedure	96
5.3. FITNESS APPROXIMATION METHOD	96
5.3.1 Updating the individual database	97
5.3.2 Fitness calculation strategy	97
5.3.3. HS optimization method	100
5.4. MOTION ESTIMATION AND BLOCK-MATCHING	100
5.5. BLOCK-MATCHING ALGORITHM BASED ON HARMONY SEARCH WITH THE ESTIMATION STRATEGY	101
5.5.1. Initial population.....	102
5.5.2. Tuning of the HS algorithm.....	103
5.5.3. The HS-BM algorithm	104
5.5.4. Discussion on the accuracy of the fitness approximation strategy.....	105
5.6. EXPERIMENTAL RESULTS	107
5.6.1. HS-BM results	107
5.6.2. Results on H.264	111
5.7. CONCLUSIONS	113

6. MULTI-THRESHOLD SEGMENTATION USING LEARNING AUTOMATA 114

6.1. INTRODUCTION	114
6.2. GAUSSIAN APPROXIMATION	116
6.3. LEARNING AUTOMATA (LA).....	117
6.3.1. CARLA Algorithm	119
6.4. IMPLEMENTATION	120
6.5. EXPERIMENTAL RESULTS	122
6.5.1. LA algorithm performance in image segmentation	122
6.5.2. Comparing the LA algorithm vs. the EM and LM methods.....	126

6.6. CONCLUSIONS	130
7. FUZZY-BASED SYSTEM FOR CORNER DETECTION.....	131
7.1. INTRODUCTION	131
7.2. FUZZY RULE-BASED SYSTEM	133
7.2.1. <i>Fuzzy System</i>	133
7.2.2. <i>Robustness</i>	137
7.2.3. <i>Corner Selection</i>	138
7.3. EXPERIMENTAL RESULTS	139
7.4. PERFORMANCE COMPARISON.....	141
7.4.1. <i>Stability Criterion</i>	141
7.4.2. <i>Noise Immunity</i>	142
7.4.3. <i>Computational Effort</i>	142
7.4.4. <i>Comparison Results</i>	142
7.5. CONCLUSIONS	143
8. CLONAL SELECTION ALGORITHM APPLIED TO CIRCLE DETECTION	145
8.1. INTRODUCTION	145
8.2. CLONAL SELECTION ALGORITHM	147
8.2.1. <i>Definitions</i>	148
8.2.2. <i>CSA Operators</i>	148
8.3. CIRCLE DETECTION USING CSA	151
8.3.1. <i>Individual representation</i>	152
8.3.2. <i>Objective function or matching function</i>	153
8.3.3. <i>Implementation of CSA</i>	154
8.4. EXPERIMENTAL RESULTS	155
8.4.1. <i>Parametric setup</i>	155
8.4.2. <i>Error score and success rate</i>	155
8.4.3. <i>Presentation of results</i>	156
8.5. CONCLUSIONS	159
9. STATES OF MATTER ALGORITHM APPLIED TO PATTERN DETECTION	161
9.1. INTRODUCTION	161
9.2. GAUSSIAN APPROXIMATION	163
9.3. STATES OF MATTER SEARCH (SMS)	164
9.3.1. <i>Definition of Operators</i>	164
9.3.2. <i>SMS algorithm</i>	167
9.3. FITNESS APPROXIMATION METHOD	170
9.3.1. <i>Updating individual database</i>	171
9.3.2. <i>Fitness calculation strategy</i>	171
9.3.3. <i>Proposed optimization SMS method</i>	173
9.4. PATTERN DETECTION PROCESS	173
9.5. PD ALGORITHM BASED ON SMS WITH THE ESTIMATION STRATEGY	175
9.5.2. <i>The SMS-PD algorithm</i>	176
9.6. EXPERIMENTAL RESULTS	177
9.7. CONCLUSIONS	181
10. ARTIFICIAL BEE COLONY ALGORITHM APPLIED TO MULTI-THRESHOLD SEGMENTATION	182
10.1. INTRODUCTION	182
10.2. GAUSSIAN APPROXIMATION	184
10.3. ARTIFICIAL BEE COLONY (ABC) ALGORITHM	185
10.3.1. <i>Biological bee profile</i>	185
10.3.2. <i>Description of the ABC algorithm</i>	185

10.4. DETERMINATION OF THRESHOLDING VALUES	187
10.5. EXPERIMENTAL RESULTS	188
10.5.1. Comparing the ABC algorithm vs. the EM and LM methods	191
10.6. CONCLUSIONS	193

MEDICINE

11. LEUKOCYTE DETECTION THROUGH AN EVOLUTIONARY METHOD..... 195

11.1. INTRODUCTION	195
11.2. DIFFERENTIAL EVOLUTION ALGORITHM.....	197
11.3. ELLIPSE DETECTION USING DE.....	199
11.3.1 Data Preprocessing.....	199
11.3.2 Individual Representation	199
11.3.3 Objective Function	200
11.3.4 Implementation of de DE for ellipse detection	202
11.4. THE WHITE BLOOD CELL DETECTOR.....	203
11.4.1 Image Preprocessing.....	203
11.4.2 Ellipse Detection Approach	205
11.4.3. Numerical Example.....	205
11.5. EXPERIMENTAL RESULTS.....	206
11.6. COMPARISONS TO OTHER METHODS	208
11.6.1 Detection Comparison.....	208
11.6.2 Robustness Comparisson.....	209
11.6.3 Stability Comparison.....	210
11.7. CONCLUSIONS	212

12. A MULTI-LEVEL THRESHOLDING METHOD FOR BREAST THERMOGRAMS ANALYSIS USING DRAGONFLY ALGORITHM 214

12.1. INTRODUCTION	214
12.2. DRAGONFLY ALGORITHM	217
12.3. SEGMENTATION APPROACHES	219
12.3.1. Energy Curve	219
12.3.2. Otsu's between class variance	220
12.3.3. Kapur's entropy	221
12.4. DRAGONFLY ALGORITHM FOR THERMAL IMAGE THRESHOLDING	221
12.4.1. DA Implementation	222
12.4.2. Performance Evaluation	222
12.5. EXPERIMENTAL RESULTS.....	224
12.5.1. Otsu's results.....	226
12.5.2. Kapur results.....	231
12.6. DISCUSSION	236
12.7. CONCLUSIONS	237

CONCLUSION..... 239

REFERENCES..... 244

Chapter 1

Introduction

Artificial Intelligence (AI) has emerged as a powerful tool for information processing, decision-making, and knowledge management. Soft Computing, as a subfield of AI, has been successfully developed to solve problems in all areas of science and technology. Due to the strong capacities of Soft Computing methods, they can be used to elucidate intricate problems to help improve people's lives in several areas.

The central focus of this research is the application and design of Soft Computing approaches to the following problems in global optimization, Computer Vision, and medicine: Block Matching, Pattern Detection, Thresholding, Corner Detection, Template Matching, Circle Detection, Color Segmentation, Leukocyte Detection, and Breast Thermogram Analysis.

The aim of this work is to design technological tools that can be used in the development of new devices, as well as to highlight the utility of different Soft Computing methodologies as solutions for problems in a variety of fields. This research started from the premise that the approaches proposed here may help, even indirectly, to improve quality of life. The framework for these potential contributions is briefly presented in section 1.1.

This chapter also provides a conceptual overview of a selection of Soft Computing techniques and optimization approaches and describes their main characteristics. This introduction will present the considerations involved in using Soft Computing methods for solving complex problems. The study of optimization methods found in the later chapters clearly demonstrates the necessity of using intelligent optimization methods for the solution of problems in optimization, Computer Vision, and medicine.

The remainder of this chapter is organized as follows: Section 1.1 presents the motivation for this research project. Section 1.2 gives a brief description of Soft Computing, its main characteristics, and the properties of three of its important methodologies. Section 1.3 explains the key concepts and features of optimization. Section 1.4 enumerates the main objectives of this research. Finally, Section 1.5 provides an overview of this manuscript.

1.1. Motivation

The number of people living on less than \$1.25 per day has dramatically increased in the last three decades. To understand poverty, we must first define it. The Centers for Disease Control and Prevention (CDC) defines poverty simply as a condition in which “a person or group of people lack human needs because they cannot afford them” (“Definitions | Social Determinants of Health | NCHHSTP | CDC,” 2018).

The top five poorest countries in the world are India (with 33% of the world's poor), China (13%), Nigeria (7%), Bangladesh (6%) and the Democratic Republic of Congo (5%). If expanded to Indonesia, Pakistan, Tanzania, Ethiopia, Kenya, and Latin American, these collective regions would include almost 80% of the world's poor. About 22,000 children die each day due to conditions of

poverty. Approximately 1.2 billion people still live without access to electricity and healthcare. Sub-Saharan Africa accounts for more than one-third of the world's extreme poor. The combined results from 27 Sub-Saharan African countries show 54% of residents living in extreme poverty; this is the highest proportion among global regions worldwide. About 75% of the world's poor live in rural areas, depending on agriculture for their livelihood. In 2010, the average income of the extremely poor in the developing world was 87 cents per capita per day, up from 74 cents in 1981 (US Census Bureau Applications Development and Services Division, 2018).

Poverty around the globe has brought with it a large number of problems. Quality of life, measured in terms of access to basic services (e.g. water, energy, healthcare, food, among others) has only gotten worse and worse.

Fighting poverty is about more than slick posters and hashtags. It involves a tough look at all of the factors that create economic inequality, both locally and on the global scale. Because of the complexity of this issue, several specialists believe that the next round of progress against poverty will be driven by new technologies, especially in health, farming, and education.

Among the newest of these technologies, Artificial Intelligence is concerned with the use of human models as an inspiration for solving computational problems. Moreover, it has proved to be useful in several areas (Shahin, 2016) through making relevant contributions to optimization, pattern recognition, shape detection, and machine learning. It has also gained considerable research interest in the Computer Vision community, as AI algorithms have successfully contributed to solving challenging Computer Vision problems.

AI is generally defined as computational modelling of human behaviour or human thoughts. The exact meaning of AI is much debated. It is widely accepted that the concept of AI evolved closely with the development of computers that have the potential ability for a thought process similar to that of humans, which includes learning, reasoning, and self-correction (Londhe & Bhasin, 2018).

AI has evolved from the research efforts of scientists from a variety of disciplines, including mathematics, philosophy, economics, and neuroscience. We humans accelerate the future with our minds. This is both a strength and a weakness. Often, our predictions of the future are highly inaccurate. Based on forecasts from a book called *The World in 2010*, published in 1976, we should have been living above and below the surfaces of three planets as of five years ago. Predictions regarding the future of AI are equally likely to be off base (Gurkaynak, et al., 2016). Most of what we consider AI today is really our own intelligence re-formatted and re-cycled with the help of computers that lack any real skills for learning or a true consciousness of being.

Computer scientists have adapted Descartes' concept of 'tabula rasa' to describe the development of autonomous agents that have the capability to reason and plan towards their goal without any built-in knowledge base of their environment (Gurkaynak et al., 2016). While weak AI may outperform humans at a specific task, such as playing chess or solving equations, general AI would outperform humans at nearly every cognitive level.

AI is an important technology that supports daily social life and economic activity. It contributes greatly to sustainable economic growth and solves various social problems. In recent years, Soft Computing (SC) as a specific area of AI has attracted attention as a key for growth in developed and developing countries (Lu, et al., 2017).

In recent years, SC technologies have developed dramatically due to improvements in computer processing capacity and the accumulation of big data. However, the yields of current AI technologies remain limited to specific intellectual areas, such as image recognition, speech recognition,

and dialogue response (Lu et al., 2017). Currently, SC is a specialized type of AI acting intellectually in a so-called individual area. Therefore, research in this field is still an open problem.

Presently, we are faced with many complex engineering systems that need to be manipulated. As they are usually not fully theoretically tractable, it is not possible to use traditional deterministic methods. SC, as opposed to conventional deterministic methods, is a set of methodologies working synergistically rather than competitively that, in one form or another, exploits the tolerance for imprecision, uncertainty, and approximate reasoning in order to achieve tractability, robustness, low-cost solutions, and close resemblance to human-like decision making. Among the methodologies of SC, those most often employed in current research include Neural Networks, Evolutionary Computation, Fuzzy Logic, and Learning Automata. Recent years have witnessed tremendous success by these powerful methods in virtually all areas of science and technology, as evidenced by the large numbers of results published in a variety of journals, conferences, and books.

On the other hand, optimization is a rich source of challenges where each new approach that is developed by mathematicians and computer scientists is quickly identified, understood, and assimilated for application. Because these specific problems are often of a nature that can only be solved by an expert with proper training, SC has targeted this class by capturing the essence of human cognition at the highest level.

In recent years, there has been a growing interest in the use of SC in all science domains, and it has fueled many visions and hopes (Salehi & Burgueño, 2018). Adeli & Hung (1995) presented a multiparadigm learning technique, through which the authors demonstrated that performance could be notably enhanced by skillful integration of different SC branches, including neural networks, genetic algorithms, fuzzy sets, and parallel processing. An extensive study of evolutionary computation, a branch of SC, in the context of structural design was conducted by Kicinger et al. (2005). Liao et al. (2011) carried out a review of studies concerning the application of metaheuristics as optimization techniques to address issues faced in the lifetime of a construction or engineering project. A survey on different SC methods for civil engineering was conducted by Lu et al. (2012). Shahin et al. (Shahin, 2013) studied applications of SC in geotechnical engineering; and Saka et al. (2013) conducted a survey on mathematical and metaheuristic algorithms in design optimization of steel frame structures. Aldwaik et al. (2014) carried out a review on progress in the optimization of high-rise buildings; and a survey on the applications and methodologies of the fuzzy multiple criteria decision-making techniques was conducted by Mamdani et al. (2015).

Parallel developments in SC in the field of medicine have also had impacts on the field of healthcare. Furthermore, advancements in medical technology have allowed physicians to better diagnose and treat multiple patients. Thanks to the continuous development of technology in the medical field, countless lives have been saved, and the overall quality of life in poor continues could be improved over time.

The application of SC concepts to healthcare has led to an era of cognitive computing in which such models can digest a large amount of data to detect patterns they have previously been exposed to. For example, targeted therapies have been used in patients with cancer with limited success despite oncologists' attempt to define subsets of patients who might benefit from a specific treatment (Londhe & Bhasin, 2018). It is predictable that in the near future SC will play a more important role in the medical context.

Moreover, recent advances in digital imaging and computer hardware technology have led to an explosion in the use of these images in a variety of medical applications, particularly for the improvement and acceleration of illness diagnosis. A significant percentage of medical images are obtained by X-ray radiography, CT, and MRI. These provide essential information for prompt and

accurate diagnoses based on advanced computer vision techniques. These applications result from the interaction between fundamental scientific research on the one hand and the development of new and high-standard technology on the other.

It is important to note that medical images considered for diagnoses often have artifacts and noise produced by the physical method with which they are taken. Normally, classical image processing methods face great difficulty dealing with images containing noise and distortions. To address these conditions, SC approaches have been recently extended to these challenging real-world image processing problems (Shahin, 2016). Interest among researchers in this application to medical imaging is increasing day by day, as is indicated by the huge volume of research published in leading international journals and international conference proceedings.

An example of the aforementioned mentioned work in medicine is the unsupervised protein-protein interaction algorithms that led to novel therapeutic target discoveries (Theofilatos et al., 2015). Included in the virtual applications of SC are electronic medical records where specific algorithms are used to identify subjects with a family history of a hereditary disease or an augmented risk of a chronic disease (Hamet & Tremblay, 2017).

Much research has been conducted in the area of medical characteristic detection. In Wang & Chu (2009), a method based on boundary support vectors is proposed to identify characteristics. In this approach, the intensity of each pixel is used to construct feature vectors whereas a support vector machine (SVM) is used for classification and segmentation. By using a different approach, Wu et al. (2007) developed an iterative Otsu method based on the circular histogram for leukocyte segmentation. According to this technique, the smear images are processed in the Hue-Saturation-Intensity (HSI) space by considering that the Hue component contains most of the WBC information. One of the latest advances in medical characteristics research is the algorithm proposed by Shitong et al. (2007), which is based on the fuzzy cellular neural network (FCNN). Although such a method has proved successful in detecting several features in the image, it has not yet been tested over images containing complex conditions. Moreover, its performance commonly decays when the iteration number is not properly defined, itself yielding a challenging problem with no clear clues on how to make the best determination.

Since several medical anomalies can be approximated with a quasi-circular form, shape detector algorithms may be employed. The problem of detecting shape features holds paramount importance, particularly for medical image analysis (Karkavitsas & Rangoussi, 2005). The shape detection in digital images is commonly performed by the Hough transform (Muammar & Nixon, 1989). A typical Hough-based approach employs an edge detector whose information guides the inference for shape values. Peak detection is then attained by averaging, filtering, and histogramming the transform space. However, such an approach requires a large storage space, given the 3D cells needed to cover all parameters. It also implies a high computational complexity, yielding a low processing speed. Furthermore, the accuracy of the extracted parameters for the detected circle is poor, particularly in the presence of noise (Atherton & Kerbyson, 1993). For a digital image holding a significant width and height and a densely populated edge pixel map, the required processing time for circular Hough transform prohibits deployment in real time applications. In order to overcome such a problem, researchers have proposed new approaches based on the Hough transform, including the probabilistic Hough transform (Fischler & Bolles, 1981; Shaked, et al., 1996), the randomized Hough transform (RHT) (Xu, et al., 1990), and the fuzzy Hough transform (Han, et al., 1994). Alternative transformations have also been presented, such as the one proposed by Becker et al. (2002). Although these new approaches demonstrate better processing speeds in comparison to the original Hough transform, they are still very sensitive to noise.

As an alternative to Hough transform-based techniques, the shape detection problem has also been handled through evolutionary computation (EC) methods. These methods are derivative-free pro-

cedures that do not require that the objective function be neither two-times differentiable nor unimodal. Therefore, EC methods as global optimization algorithms can deal with non-convex, non-linear, and multimodal problems subject to linear or nonlinear constraints with continuous or discrete decision variables. In general, EC methods have been shown to deliver better results than those based on Hough Transform (HT), considering accuracy, speed, and robustness (Ayala-Ramirez, et al., 2006). Such approaches have produced several robust shape detectors using different optimization algorithms, such as genetic algorithms (GAs) (Ayala-Ramirez et al., 2006), harmony search (HS) (Cuevas, et al., 2012), differential evolution (DE) (Cuevas, et al., 2011), and the electromagnetism-like optimization algorithm (EMO) (Cuevas, et al., 2012). A detailed description of these and other evolutionary methods can be found in my previous collaborations with other researchers (Cuevas, et al., 2016).

Although detection algorithms based on the optimization approaches present several advantages in comparison to those based on the Hough transform, they have rarely been applied to medical characteristic detection. One exception is the work presented by Karkavitsas et al. (2005) that solves the shape detection problem through the use of Genetic Algorithms. However, since the evaluation function, which assesses the quality of each solution, considers the number of pixels contained inside of a circle with a fixed radius, the method is prone to produce misdetections, particularly for images that contain overlapped or irregular White Blood Cells (WBC).

Conversely, detection of medical characteristics in images plays a significant role in the diagnosis of different diseases. Although digital image processing techniques have successfully contributed to generating new methods for cell analysis, which, in turn, have led into more accurate and reliable systems for disease diagnosis, high variability of such characteristics and localization complicates the data extraction process. Moreover, medical images contain noise and distortions produced by unstable lighting conditions during the capturing process.

In this research, we strive to present new state-of-the-art techniques by applying recent Soft Computing strategies to challenging and significant problems, such as those described above, in optimization, Computer Vision, and medicine.

1.2. Soft Computing

Soft computing (SC) (Patnaik & Zhong, 2014) is a computer science area that tries to develop intelligent systems. Under soft computing, the objective is to produce computer elements which artificially operate based on intelligent processes typically extracted from natural or biological phenomena.

SC considers a set of methodologies that try to develop systems with tolerance to the imprecision and robustness to the uncertainty. SC techniques have demonstrated their capacities in the solution of several engineering applications. Different to classical methods, soft computing approaches emulate cognition elements in many important aspects: they can acquire knowledge from experience. Furthermore, SC techniques employ flexible operators that perform input/output mappings without the consideration of deterministic models. With such operators, it is possible to extend the use of SC techniques even to those applications in which the precise and accurate representations are not available.

SC involves a set of approaches extracted from many fields of the computational intelligence. SC considers three main elements (Chaturvedi, 2008): (A) fuzzy Systems, (B) neural networks and (C) evolutionary computation. Furthermore, other alternative approaches have been associated with SC. They include techniques such as machine learning (ML) and probabilistic reasoning

(PR), belief networks, and expert systems, etc. Fig. 1.1 shows a conceptual map that describe the different branches of SC.

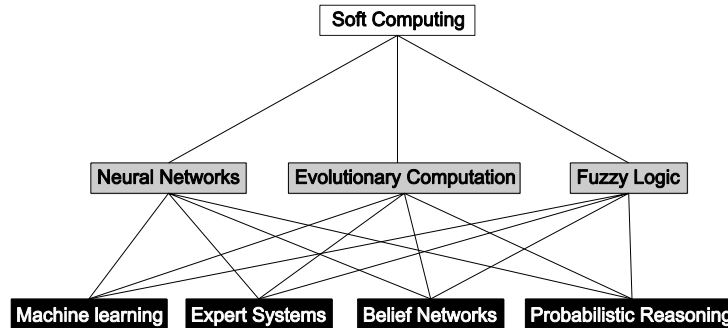


Fig. 1.1. Different branches of Soft Computing.

Soft Computing, in general, considers the inspiration from natural phenomena or biological systems. Neural networks emulate the animal brain connections; evolutionary algorithms base its operation on the characteristics of Darwinian evolution. Meanwhile fuzzy logic aims to simulate the highly imprecise nature of human reasoning.

1.2.1. Fuzzy logic

Humans handle imprecise and uncertain information in their day life routines. This fact can be observed in the structure of the language which includes various qualitative and subjective terms and expressions such as “quite expensive”, “very old”, or “pretty close”, “cheap”, etc. In human knowledge, approximate reasoning is employed and considered to provide several levels of imprecision and uncertainty in the concepts that are transmitted among persons.

Fuzzy logic (Ghosh, et al., 1998) represents a generalization of classical logic. It considers fuzzy sets which are an extension of crisp sets in classical logic theory. In classical logic, an element belongs to either a member of the set or not at all, in fuzzy logic, an element could belong at a certain level “degree of membership” to the set. Hence, in fuzzy logic, the membership function of an element varies in the range from 0–1, but in classical logic, the membership value is only 0 or 1.

A fuzzy system involves four different modules: a fuzzy rule base, a fuzzification module, an inference engine, and a defuzzification module. Such elements are shown in Fig. 1.2. The fuzzification module translates the input values into fuzzy values which are used by the fuzzy system. The inference engine considers the outputs of the fuzzification element and uses the fuzzy rules in the fuzzy rule base to produce an intermediate result. Finally, the defuzzification module uses this intermediate result to obtain the global output.

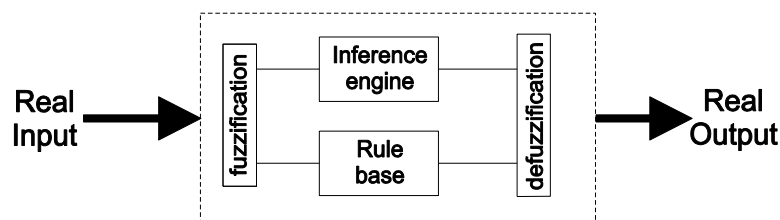


Fig. 1.2. Elements of a fuzzy system.

1.2.2. Neural Networks

Neural networks (NN) (G. P. Zhang, 2000), are defined as interconnected systems which parallel wise perform a general process. Such elements are undergone to a learning task which automatically modifies the network parameters as a consequence of the optimization of a certain criterion. Each unit of the neural network is considered as a highly simplified model of the biological neuron present in the animal brain.

The main characteristics of NN models can be summarized as follows:

- Parallelism – All neural network architectures maintain some level of parallelism in the computation of their numerous units;
- Units – The basic units of a NN correspond to the same elements which maintain the same characteristics and behavior;
- Information processing – the output value of each unit depends completely on its current state and the output values of other units to which it is associated; and
- Learning – NN parameters are undergone to changes, according to a learning scheme which minimizes a certain performance criterion.

NN architectures are classified in according to several criteria:

1. According to their connections, there are three major types of NN such as, a. recurrent network, b. Feed forward network and c. Competitive networks.
2. According to their structure: a. Static (fixed) structure and b. Dynamic structure.
3. According to their learning algorithms, a. Supervised learning and b. Unsupervised learning.
4. NN models can also be divided regarding the processes in which they are used such as a. Pattern recognition, b. Classification and c. Regression.

1.2.3. Evolutionary computation

Evolutionary computation (EC) (Dan, 2013) methods are derivative-free procedures, which do not require that the objective function must be neither two-times differentiable nor uni-modal. Therefore, EC methods as global optimization algorithms can deal with non-convex, nonlinear, and multimodal problems subject to linear or nonlinear constraints with continuous or discrete decision variables.

The field of EC has a rich history. With the development of computational devices and demands of industrial processes, the necessity to solve some optimization problems arose despite the fact that there was not sufficient prior knowledge (hypotheses) on the optimization problem for the application of a classical method. In fact, in the majority of image processing and pattern recognition cases, the problems are highly nonlinear, or characterized by a noisy fitness, or without an explicit analytical expression as the objective function might be the result of an experimental or simulation process. In this context, the EC methods have been proposed as optimization alternatives.

An EC technique is a general method for solving optimization problems. It uses an objective function in an abstract and efficient manner, typically without utilizing deeper insights into its mathematical properties. EC methods do not require hypotheses on the optimization problem nor any

kind of prior knowledge on the objective function. The treatment of objective functions as “black boxes” (Blum & Roli, 2003) is the most prominent and attractive feature of EC methods.

EC methods obtain knowledge about the structure of an optimization problem by utilizing information obtained from the possible solutions (i.e., candidate solutions) evaluated in the past. This knowledge is used to construct new candidate solutions which are likely to have a better quality.

Recently, several EC methods have been proposed with interesting results. Such approaches use as inspiration our scientific understanding of biological, natural or social systems, which at some level of abstraction can be represented as optimization processes (Nanda & Panda, 2014a). These methods include the social behavior of bird flocking and fish schooling such as the Particle Swarm Optimization (PSO) algorithm (Kennedy & Eberhart, 1995b), the cooperative behavior of bee colonies such as the Artificial Bee Colony (ABC) technique (Dervis Karaboga, 2005), the improvisation process that occurs when a musician searches for a better state of harmony such as the Harmony Search (HS) (Geem, et al., 2001), the emulation of the bat behavior such as the Bat Algorithm (BA) method (Yang, 2010a), the mating behavior of firefly insects such as the Firefly (FF) method (Yang, 2009), the social-spider behavior such as the Social Spider Optimization (SSO) (Cuevas, et al., 2013), the simulation of the animal behavior in a group such as the Collective Animal Behavior (Cuevas, et al., 2012), the emulation of immunological systems as the clonal selection algorithm (CSA) (Leandro N. De Castro & Von Zuben, 2002), the simulation of the electromagnetism phenomenon as the electromagnetism-like algorithm (EMO) (Birbil & Fang, 2003), and the emulation of the differential and conventional evolution in species such as the Differential Evolution (DE) (Storn & Price, 1995) and Genetic Algorithms (GA) (Goldberg & Holland, 1988), respectively.

1.3. Optimization Concepts

1.3.1. Definition of an optimization problem

The vast majority of image processing and pattern recognition algorithms use some form of optimization, as they intend to find some solution which is “best” according to some criterion. From a general perspective, an optimization problem is a situation that requires to decide for a choice from a set of possible alternatives in order to reach a predefined/required benefit at minimal costs (Akay & Karaboga, 2015).

Consider a public transportation system of a city, for example. Here the system has to find the “best” route to a destination location. In order to rate alternative solutions and eventually find out which solution is “best,” a suitable criterion has to be applied. A reasonable criterion could be the distance of the routes. We then would expect the optimization algorithm to select the route of shortest distance as a solution. Observe, however, that other criteria are possible, which might lead to different “optimal” solutions, e.g., number of transfers, ticket price or the time it takes to travel the route leading to the fastest route as a solution.

Mathematically speaking, optimization can be described as follows: Given a function $f : S \rightarrow \mathfrak{R}$ which is called the objective function, find the argument which minimizes f :

$$x^* = \arg \min_{x \in S} f(x) \quad (1.1)$$

S defines the so-called solution set, which is the set of all possible solutions for the optimization problem. Sometimes, the unknown(s) x are referred to design variables. The function f describes

the optimization criterion, i.e., enables us to calculate a quantity which indicates the “quality” of a particular x .

In our example, S is composed by the subway trajectories and bus lines, etc., stored in the database of the system, x is the route the system has to find, and the optimization criterion $f(x)$ (which measures the quality of a possible solution) could calculate the ticket price or distance to the destination (or a combination of both), depending on our preferences.

Sometimes there also exist one or more additional constraints which the solution x^* has to satisfy. In that case we talk about constrained optimization (opposed to unconstrained optimization if no such constraint exists). As a summary, an optimization problem has the following components:

- One or more design variables x for which a solution has to be found
- An objective function $f(x)$ describing the optimization criterion
- A solution set S specifying the set of possible solutions x
- (optional) One or more constraints on x

In order to be of practical use, an optimization algorithm has to find a solution in a reasonable amount of time with reasonable accuracy. Apart from the performance of the algorithm employed, this also depends on the problem at hand itself. If we can hope for a numerical solution, we say that the problem is well-posed. For assessing whether an optimization problem is well-posed, the following conditions must be fulfilled:

1. A solution exists.
2. There is only one solution to the problem, i.e., the solution is unique.
3. The relationship between the solution and the initial conditions is such that small perturbations of the initial conditions result in only small variations of x^* .

1.3.2. Classical optimization

Once a task has been transformed into an objective function minimization problem, the next step is to choose an appropriate optimizer. Optimization algorithms can be divided in two groups: derivative-based and derivative-free (X. S. Yang, 2010b).

In general, $f(x)$ may have a nonlinear form respect to the adjustable parameter x . Due to the complexity of $f(\cdot)$, in classical methods, it is often used an iterative algorithm to explore the input space effectively. In iterative descent methods, the next point x_{k+1} is determined by a step down from the current point x_k in a direction vector \mathbf{d} :

$$x_{k+1} = x_k + \alpha \mathbf{d}, \quad (1.2)$$

where α is a positive step size regulating to what extent to proceed in that direction. When the direction \mathbf{d} in Eq. 1.2 is determined on the basis of the gradient (\mathbf{g}) of the objective function $f(\cdot)$, such methods are known as gradient-based techniques.

The method of steepest descent is one of the oldest techniques for optimizing a given function. This technique represents the basis for many derivative-based methods. Under such a method, the Eq. 1.3 becomes the well-known gradient formula:

$$x_{k+1} = x_k - \alpha \mathbf{g}(f(x)), \quad (1.3)$$

However, classical derivative-based optimization can be effective as long the objective function fulfills two requirements:

- The objective function must be two-times differentiable.
- The objective function must be uni-modal, i.e., have a single minimum.

A simple example of a differentiable and uni-modal objective function is

$$f(x_1, x_2) = 10 - e^{-(x_1^2 + 3x_2^2)} \quad (1.4)$$

Figure 1.3 shows the function defined in Eq. 1.4.

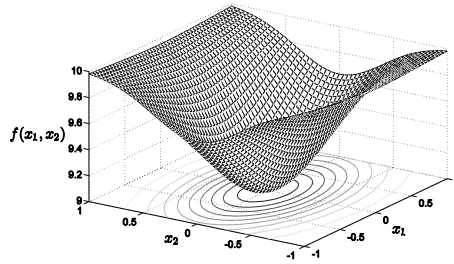


Fig. 1.3. Uni-modal objective function.

Unfortunately, under such circumstances, classical methods are only applicable for a few types of optimization problems. For combinatorial optimization, there is no definition of differentiation.

Furthermore, there are many reasons why an objective function might not be differentiable. For example, the “floor” operation in Eq. 1.5 quantizes the function in Eq. 1.4, transforming Fig. 1.3 into the stepped shape seen in Fig. 1.4. At each step’s edge, the objective function is non-differentiable:

$$f(x_1, x_2) = \text{floor}\left(10 - e^{-(x_1^2 + 3x_2^2)}\right) \quad (1.5)$$

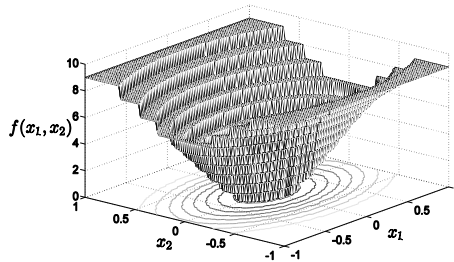


Fig. 1.4. A non-differentiable, quantized, uni-modal function

Even in differentiable objective functions, gradient-based methods might not work. Let us consider the minimization of the Griewank function as an example.

minimize	$f(x_1, x_2) = \frac{x_1^2 + x_2^2}{4000} - \cos(x_1) \cos\left(\frac{x_2}{\sqrt{2}}\right) + 1$	(1.6)
----------	--	-------

subject to	$-30 \leq x_1 \leq 30$ $-30 \leq x_2 \leq 30$
------------	---

From the optimization problem formulated in Eq. 1.6, it is quite easy to understand that the global optimal solution is $x_1 = x_2 = 0$. Figure 1.5 visualizes the function defined in Eq. 1.6. According to Fig. 1.5, the objective function has many local optimal solutions (multimodal) so that the gradient methods with a randomly generated initial solution will converge to one of them with a large probability.

Considering the limitations of gradient-based methods, image processing and pattern recognition problems make difficult their integration with classical optimization methods. Instead, some other techniques which do not make assumptions and which can be applied to wide range of problems are required (Sumit Ghosh et al., 1998).

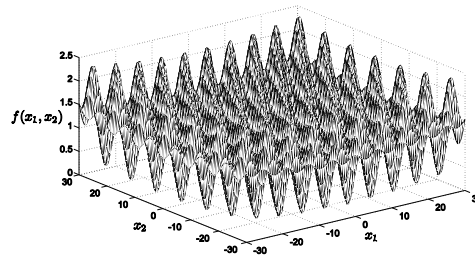


Fig. 1.5. The Griewank multi-modal function

1.3.3. Optimization with Evolutionary computation

From a conventional point of view, an EC method is an algorithm that simulates at some level of abstraction a biological, natural or social system. To be more specific, a standard EC algorithm includes:

1. One or more populations of candidate solutions are considered.
2. These populations change dynamically due to the production of new solutions.
3. A fitness function reflects the ability of a solution to survive and reproduce.
4. Several operators are employed in order to explore an exploit appropriately the space of solutions.

The EC methodology suggest that, on average, candidate solutions improve their fitness over generations (i. e., their capability of solving the optimization problem). A simulation of the evolution process based on a set of candidate solutions whose fitness is properly correlated to the objective function to optimize will, on average, lead to an improvement of their fitness and thus steer the simulated population towards the global solution.

Most of the optimization methods have been designed to solve the problem of finding a global solution of a nonlinear optimization problem with box constraints in the following form:

maximize	$f(\mathbf{x}), \quad \mathbf{x} = (x_1, \dots, x_d) \in \mathbb{R}^d$	(1.7)
subject to	$\mathbf{x} \in \mathbf{X}$	

where $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is a nonlinear function whereas $\mathbf{X} = \{\mathbf{x} \in \mathbb{R}^d \mid l_i \leq x_i \leq u_i, i = 1, \dots, d\}$ is a bounded feasible search space, constrained by the lower (l_i) and upper (u_i) limits.

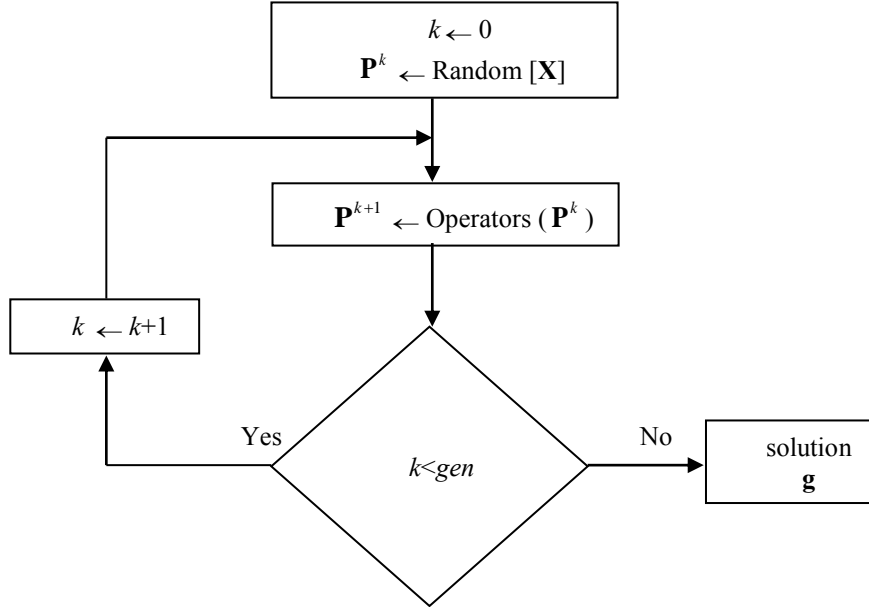


Fig. 1.6. The basic cycle of an EC method.

In order to solve the problem formulated in Eq. 1.6, in an evolutionary computation method, a population $\mathbf{P}^k (\{\mathbf{p}_1^k, \mathbf{p}_2^k, \dots, \mathbf{p}_N^k\})$ of N candidate solutions (individuals) evolves from the initial point ($k=0$) to a total gen number iterations ($k=gen$). In its initial point, the algorithm begins by initializing the set of N candidate solutions with values that are randomly and uniformly distributed between the pre-specified lower (l_i) and upper (u_i) limits. In each iteration, a set of evolutionary operators are applied over the population \mathbf{P}^k to build the new population \mathbf{P}^{k+1} . Each candidate solution \mathbf{p}_i^k ($i \in [1, \dots, N]$) represents a d -dimensional vector $\{p_{i,1}^k, p_{i,2}^k, \dots, p_{i,d}^k\}$ where each dimension corresponds to a decision variable of the optimization problem at hand. The quality of each candidate solution \mathbf{p}_i^k is evaluated by using an objective function $f(\mathbf{p}_i^k)$ whose final result represents the fitness value of \mathbf{p}_i^k . During the evolution process, the best candidate solution \mathbf{g} (g_1, g_2, \dots, g_d) seen so-far is preserved considering that it represents the best available solution. Fig. 1.6 presents a graphical representation of a basic cycle of an EC method.

1.4. Research Objectives

The central focus of this research is to address global problems in optimization, Computer Vision, and medicine through the development and application of Soft Computing approaches including Block Matching, Pattern Detection, Thresholding, Corner Detection, Template Matching, Circle Detection, Color Segmentation, Leukocyte Detection, and Breast Thermogram Analysis.

The aim of this work is to design technological tools, which can be used in the development of new devices, as well as to highlight the utility of different Soft Computing methodologies in solving problems in a variety of fields. This research started from the premise that the approaches proposed here may help to improve human quality of life, even if indirectly.

A particular goal of this research is to apply both well-known and new methodologies in SC to the solution of complex problems in the three aforementioned key fields in order to demonstrate the strong capabilities of such methodologies. This research was partially conducted as a collaboration between the Freie Universität Berlin and the University of Guadalajara. In this project, we strive to present solutions that can be potentially beneficial for the engineering and medical communities.

Another objective of this research is the translation of Computer Vision and medical problems into optimization problems. The main reason for this is to open the topic for new proposals that can also present new methodologies for solving such problems.

Moreover, in this Ph.D. research, the anomaly detection task in the field of medicine is approached as an optimization problem, and AI techniques are used to build the shape approximation. These methods are mainly Evolutionary Computation Techniques (ECTs), which are based on several biological or social phenomena. In the detection process, ECTs search the entire parameter space of the optimization problem through candidate solutions (individuals). An objective function is employed to evaluate the quality of the detection. Conducted as such by the values of objective function, the group of candidate solutions are modified through evolutionary operators so that the optimal detection can be found.

This research also strives to expand the above concepts to automated detection and diagnosis for pathologies such as Leukemia and breast cancer. It is my goal to have a significant impact on healthcare by bringing such advances to the clinical context.

Finally, a secondary goal of this research is the development of new evolutionary approaches for the optimization of complex problems. In this stage, general approaches to the construction of efficient solutions to optimization problems are produced. They can be translated into common control and data structures provided by most high-level languages. The temporal and spatial requirements of the algorithms that result will be precisely analyzed. The continuous growth in complexity of computer systems is making the task of implementation increasingly multifaceted.

In general, the problem of developing effective algorithms can be stated as finding the best trade-off between accuracy and speed. This optimal trade-off is dependent on the particular use of the implementation process. The process of translation and design of a generic optimization problem in engineering is described in section 1.3 of this chapter.

1.5. Thesis Overview

This manuscript is composed of 13 chapters and, it was assembled in a way, that each chapter can be read independently from the other. The purpose of this, was to create a collection of the applications and algorithms developed during this research. In this sense, each chapter holds its own conclusions, such construction also provides easiness in the reading and comprehension of the diverse topics treated along this manuscript. The organization of the following chapters was done, bearing in mind the three proposed research topics. Considering this, chapters 1 and 2, correspond merely to optimization approaches, chapters 3 to 10 are related to Computer Vision problems and, chapters 11 and 12 present medical applications, Finally, chapter 13 elucidates some conclusions of this research. The remainder of this section is dedicated to present a brief description of the contents of each chapter.

In *Chapter 2*, is presented an enhanced evolutionary approach known as Electromagnetism-Like (EMO) algorithm. The improvement considers the Opposition-Based Learning (OBL) approach to accelerate the global convergence speed. OBL is a machine intelligence strategy which considers the current candidate solution and its opposite value at the same time, achieving a faster explora-

tion of the search space. The presented method significantly reduces the required computational effort yet avoiding any detriment to the good search capabilities of the original EMO algorithm. Experiments are conducted over a comprehensive set of benchmark functions, showing that the presented method obtains promising performance for most of the discussed test problems.

Chapter 3 presents a methodology to implement human-knowledge-based optimization strategies. In the scheme, a Takagi-Sugeno Fuzzy inference system is used to reproduce a specific search strategy generated by a human expert. Therefore, the number of rules and its configuration only depend on the expert experience without considering any learning rule process. Under these conditions, each fuzzy rule represents an expert observation that models the conditions under which candidate solutions are modified in order to reach the optimal location. To exhibit the performance and robustness of the presented method, a comparison to other well-known optimization methods is conducted. The comparison considers several standard benchmark functions which are typically found in scientific literature. The results suggest a high performance of the Fuzzy-based methodology.

In *Chapter 4*, an image segmentator algorithm based on Learning Vector Quantization (LVQ) networks is presented and tested on a tracking application. In LVQ networks, neighboring neurons learn to recognize neighboring sections of the input space. Neighboring neurons would correspond to object regions illuminated in a different manner. The segmentator involves a LVQ network that operate directly on the image pixels and a decision function. This algorithm has been applied to color tracking, and have shown more robustness on illumination changes than other standard algorithms.

Chapter 5 presents a Block matching (BM) algorithm that combines an evolutionary algorithm (such Harmony Search) with a fitness approximation model is presented. The approach uses motion vectors belonging to the search window as potential solutions. A fitness function evaluates the matching quality of each motion vector candidate. In order to save computational time, the approach incorporates a fitness calculation strategy to decide which motion vectors can be only estimated or actually evaluated. Guided by the values of such fitness calculation strategy, the set of motion vectors is evolved through evolutionary operators until the best possible motion vector is identified. The presented method is also compared to other BM algorithms in terms of velocity and coding quality. Experimental results demonstrate that the presented algorithm exhibits the best balance between coding efficiency and computational complexity.

In *Chapter 6*, the use of the Learning Automata (LA) algorithm to compute threshold points for image segmentation is explored. Despite other techniques commonly seek through the parameter map, LA explores in the probability space providing better convergence properties and robustness. In the chapter, the segmentation task is therefore considered as an optimization problem and the LA is used to generate the image multi-threshold separation. In the approach, a 1-D histogram of a given image is approximated through a Gaussian mixture model whose parameters are calculated using the LA algorithm. Each Gaussian function approximating the histogram represents a pixel class and therefore a threshold point. The method shows fast convergence avoiding the typical sensitivity to initial conditions such as the Expectation-Maximization (EM) algorithm or the complex time-consuming computations commonly found in gradient methods. Experimental results demonstrate the algorithm's ability to perform automatic multi-threshold selection and show interesting advantages as it is compared to other algorithms solving the same task.

Chapter 7 presents an algorithm based on fuzzy reasoning to detect corners even under imprecise information. The robustness of the presented algorithm is compared to well-known conventional corner detectors and its performance is then tested on a number of benchmark images to illustrate the efficiency of the algorithm under uncertainty conditions.

Chapter 8 presents an algorithm for the automatic detection of circular shapes from complicated and noisy images with no consideration of the conventional Hough transform principles. The presented algorithm is based on an Artificial Immune Optimization (AIO) technique, known as the Clonal Selection algorithm (CSA). The CSA is an effective method for searching and optimizing following the Clonal Selection Principle (CSP) in the human immune system which generates a response according to the relationship between antigens (Ag), i.e. patterns to be recognized and antibodies (Ab) i.e. possible solutions. The algorithm uses the encoding of three points as candidate circles over the edge image. An objective function evaluates if such candidate circles (Ab) are actually present in the edge image (Ag). Guided by the values of this objective function, the set of encoded candidate circles are evolved using the CSA so that they can fit to the actual circles on the edge map of the image. Experimental results over several synthetic as well as natural images with varying range of complexity validate the efficiency of the presented technique with regard to accuracy, speed, and robustness.

In *Chapter 9*, an algorithm for detecting patterns in images is presented. The approach is based on an evolutionary algorithm known as the States of Matter. Under the presented method can be strongly reduced the number of search locations in the detection process. In the presented approach, individuals emulate molecules that experiment state transitions which represent different exploration–exploitation levels. In the algorithm, the computation of search locations is radically reduced by incorporating a fitness calculation strategy which indicates when it is feasible to calculate or only estimate the Normalized cross-correlation values for new search locations. Conducted simulations show that the presented method achieves the best balance over other detecting algorithms, in terms of estimation accuracy and computational cost.

Chapter 10 explores the use of the Artificial Bee Colony (ABC) algorithm to compute threshold selection for image segmentation. ABC is a heuristic algorithm motivated by the intelligent behavior of honey-bees which has been successfully employed to solve complex optimization problems. In this approach, an image 1-D histogram is approximated through a Gaussian mixture model whose parameters are calculated by the ABC algorithm. For the approximation scheme, each Gaussian function represents a pixel class and therefore a threshold. Unlike the Expectation-Maximization (EM) algorithm, the ABC-based method shows fast convergence and low sensitivity to initial conditions. Remarkably, it also improves complex time-consuming computations commonly required by gradient-based methods. Experimental results demonstrate the algorithm's ability to perform automatic multi-threshold selection yet showing interesting advantages by comparison to other well-known algorithms.

In *Chapter 11*, an algorithm for the automatic detection of blood cell images based on the DE algorithm is presented. The proposed method uses the encoding of five edge points as candidate ellipses in the edge map of the smear. An objective function allows to accurately measure the resemblance of a candidate ellipse with an actual WBC on the image. Guided by the values of such objective function, the set of encoded candidate ellipses are evolved using the DE algorithm so that they can fit into actual WBC on the image. The approach generates a sub-pixel detector, which can effectively identify leukocytes in real images. Experimental evidence shows the effectiveness of such method in detecting leukocytes despite complex conditions. Comparison to the state-of-the-art WBC detectors on multiple images demonstrates a better performance of the proposed method.

Furthermore, in *Chapter 12* is described a segmentation technique for thermographic images that consider the spatial information of the pixel contained in the image. This approach employs a novel optimization technique called the Dragonfly Algorithm to compute the best thresholds that segment the image. The experimental results exhibit a well-performance of the method in comparison to the other methods over a set of randomly selected thermograms retrieved from the Database for Research Mastology with Infrared Image. The presented approach could provide a highly reliable

clinical decision support, which aims to help clinicians in performing a diagnosis using thermography images.

Finally, *Chapter 13* elucidates the conclusions of this research and their contribution.

Chapter 2

Global optimization using Opposition-based Electromagnetism-like algorithm

Electromagnetism-like Optimization (EMO) is a global optimization algorithm which allows to solve complex multimodal optimization problems. EMO employs search agents that emulate a population of charged particles which interact to each other according to Electro-magnetism's laws (attraction and repulsion). Traditionally, EMO requires a large number of generations in its local search procedure. If this local search process is eliminated, it is severely damaged the overall convergence, exploration, population diversity and accuracy. This chapter presents an enhanced EMO algorithm called OBEMO, which uses the Opposition-based Learning (OBL) approach to accelerate the global convergence speed. OBL is a machine intelligence strategy which considers the current candidate solution and its opposite value at the same time, achieving a faster exploration of the search space. The presented OBEMO method significantly reduces the required computational effort without causing any detriment to the search capabilities of the original EMO algorithm. Experiments showed that OBEMO obtains promising performance for most of the discussed test problems.

2.1. Introduction

Global Optimization (GO) (Borji & Hamidi, 2008; S. Tan, Cheng, & Xu, 2007) has issued applications for many areas of science (Yang, et al., 2009), engineering (W. Gao & Ren, 2011), economics (Chang, 2009; Xu, et al., 2006) and others whose definition requires mathematical modeling (Borzabadi, et al., 2010; Takeuchi, 2008). In general, GO aims to find the global optimum for an objective function which has been defined over a given search space. The difficulties associated with the use of mathematical methods over GO problems have contributed to the development of alternative solutions. Linear programming and dynamic programming techniques, for example, often have failed in solving (or reaching local optimum at) NP-hard problems which feature a large number of variables and non-linear objective functions. In order to overcome such problems, researchers have proposed metaheuristic-based algorithms for searching near-optimum solutions.

Metaheuristic algorithms are stochastic search methods that mimic the metaphor of biological or physical phenomena. The core of such methods lies on the analysis of collective behavior of relatively simple agents working on decentralized systems. Such systems typically gather an agent's population that can communicate to each other while sharing a common environment. Despite a non-centralized control algorithm regulates the agent behavior, the agent can solve complex tasks by analyzing a given global model and harvesting cooperation to other elements. Therefore, a novel global behavior evolves from interaction among agents as it can be seen on typical examples that include ant colonies, animal herding, bird flocking, fish schooling, honey bees, bacteria, charged particles and many more. Some other metaheuristic optimization algorithms have been recently proposed to solve optimization problems, such as Genetic Algorithms (GA) (Holland, 1992), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995a), Ant Colony Optimization (ACO) (Dorigo et al., 1991), Differential Evolution (DE) (Price, et al., 2005), Artificial Immune Systems (AIS) (Fyfe & Jain, 2005), Artificial Bee Colony (Dervis Karaboga, 2005) and Gravitational Search Algorithm (GSA) (Rashedi, et al., 2011). Electromagnetism-like algorithm

(EMO) is a relatively new population-based metaheuristic algorithm which was firstly introduced by Birbil and Fang (2003) to solve continuous optimization models using bounded variables. The algorithm imitates the attraction–repulsion mechanism between charged particles in an electromagnetic field. Each particle represents a solution and carries a certain amount of charge which is proportional to the solution quality (objective function). In turn, solutions are defined by position vectors which give real positions for particles within a multi-dimensional space. Moreover, objective function values of particles are calculated considering such position vectors. Each particle exerts repulsion or attraction forces over other population members; the resultant force acting over a particle is used to update its position. Clearly, the idea behind the EMO methodology is to move particles towards the optimum solution by exerting attraction or repulsion forces. Unlike other traditional metaheuristics techniques such as GA, DE, ABC and AIS, whose population members exchange materials or information between each other, the EMO methodology assumes that each particle is influenced by all other particles in the population, mimicking other heuristics methods such as PSO and ACO. Although the EMO algorithm shares some characteristics with PSO and ACO, recent works have exhibited its better accuracy regarding optimal parameters (Rocha & Fernandes, 2009a, 2009b; Tsou & Kao, 2008; Wu, et al., 2004a), yet showing convergence (Birbil, et al., 2004). EMO has been successfully applied to solve different sorts of engineering problems such as flow-shop scheduling (Naderi, et al., 2010), communications (Hung & Huang, 2011), vehicle routing (Yurtkuran & Emel, 2010), array pattern optimization in circuits (Jhang & Lee, 2009), neural network training (Wu, et al., 2004b) control systems (Lee & Chang, 2010) and image processing (Cuevas, et al., 2012).

EMO algorithm employs four main phases: initialization, local search, calculation and movement. The local search procedure is a stochastic search in several directions over all coordinates of each particle. EMO's main drawback is its computational complexity resulting from the large number of iterations which are commonly required during the searching process. The issue becomes worst as the dimension of the optimization problem increases. Several approaches, which simplify the local search, have been proposed in the literature to reduce EMO's computational effort. In Alcalá-Fdez, et al. (2009) was proposed a discrete encoding for the particle set in order to reduce search directions at each dimension. In Rocha & Fernandes (2007) and Rocha & Fernandes (2011), authors include a new local search method which is based on a fixed search pattern and a shrinking strategy that aims to reduce the population size as the iterative process progresses. Additionally, in (Rocha & Fernandes, 2009a), a modified local search phase that employs the gradient descent method is adopted to enhance its computational complexity. Although all these approaches have improved the computational time which is required by the original EMO algorithm, recent works have demonstrated that reducing or simplifying EMO's local search processes also affects other important properties, such as convergence, exploration, population diversity and accuracy (Lee & Chang, 2010; Lee, et al., 2012).

On the other hand, the Opposition-based Learning (OBL), that has been initially proposed in Tizhoosh, (2005), is a machine intelligence strategy which considers the current estimate and its correspondent opposite value (i.e., guess and opposite guess) at the same time to achieve a fast approximation for a current candidate solution. It has been mathematically proved (Rahnamayan, et al., 2007, 2008; Wang, et al., 2011) that an opposite candidate solution holds a higher probability for approaching the global optimum solution than a given random candidate, yet quicker. Recently, the concept of opposition has been used to accelerate metaheuristic-based algorithms such as GA (Iqbal, et al., 2011), DE (Rahnamayan, et al., 2008), PSO (Wang, et al., 2011) and GSA (Shaw, et al., 2012)

In this chapter, an Opposition-Based EMO called OBEMO is presented, it was constructed by combining the opposition-based strategy and the standard EMO technique. The enhanced algorithm allows a significant reduction on the computational effort which required by the local search procedure yet avoiding any detriment to the good search capabilities and convergence speed of the

original EMO algorithm. The presented algorithm has been experimentally tested by means of a comprehensive set of complex benchmark functions. Comparisons to the original EMO and others state-of-the-art EMO-based algorithms (Takeuchi, 2008) demonstrate that the OBEMO technique is faster for all test functions, yet delivering a higher accuracy. Conclusions on the conducted experiments are supported by statistical validation that properly supports the results.

The rest of the chapter is organized as follows: Section 2.2 introduces the standard EMO algorithm. Section 2.3 gives a simple description of OBL and Section 2.4 explains the implementation of the proposed OBEMO algorithm. Section 2.5 presents a comparative study among OBEMO and other EMO variants over several benchmark problems. Finally, some conclusions are drawn in Section 2.6.

2.2. Electromagnetism - Like Optimization Algorithm (EMO)

EMO algorithm is a simple and direct search algorithm which has been inspired by the electromagnetism phenomenon. It is based on a given population and the optimization of global multimodal functions. In comparison to GA, it does not use crossover or mutation operators to explore feasible regions; instead it does implement a collective attraction–repulsion mechanism yielding a reduced computational cost with respect to memory allocation and execution time. Moreover, no gradient information is required as it employs a decimal system which clearly contrasts to GA. Few particles are required to reach converge as has been already demonstrated in (Dorigo et al., 1991).

EMO algorithm can effectively solve a special class of optimization problems with bounded variables in the form of:

$$\begin{aligned} \min f(x) \\ x \in [l, u] \end{aligned} \quad (2.1)$$

where $[l, u] = \{x \in \mathbb{R}^n \mid l_d \leq x_d \leq u_d, d = 1, 2, \dots, n\}$ and n being the dimension of the variable x , $[l, u] \subset \mathbb{R}^n$, a nonempty subset and a real-value function $f: [l, u] \rightarrow \mathbb{R}$. Hence, the following problem features are known:

- n : Dimensional size of the problem.
- u_d : The highest bound of the k^{th} dimension.
- l_d : The lowest bound of the k^{th} dimension.
- $f(x)$: The function to be minimized.

EMO algorithm has four phases (Chang, 2009): initialization, local search, computation of the total force vector and movement. A deeper discussion about each stage follows.

Initialization, a number of m particles is assembled as their highest (u) and lowest limit (l) are identified.

Local search, collects local information for a given point \mathbf{g}^p , where $p \in (1, \dots, m)$.

Calculation of the total force vector, charges and forces are calculated for every particle.

Movement, each particle is displaced accordingly, matching the corresponding force vector.

2.2.1. Initialization

First, the population of m solutions is randomly produced at an initial state. Each n -dimensional solution is regarded as a charged particle holding a uniform distribution between the highest (u) and the lowest (l) limits. The optimum particle (solution) is thus defined by the objective function to be optimized. The procedure ends when all the m samples are evaluated, choosing the sample (particle) that has gathered the best function value.

2.2.2. Local Search

The local search procedure is used to gather local information within the neighbourhood of a candidate solution. It allows obtaining a better exploration and population diversity for the algorithm.

Considering a pre-fixed number of iterations known as $ITER$ and a feasible neighbourhood search δ , the procedure iterates as follows: Point \mathbf{g}^p is assigned to a temporary point \mathbf{t} to store the initial information. Next, for a given coordinate d , a random number is selected (λ_1) and combined with δ as a step length, which in turn, moves the point \mathbf{t} along the direction d , with a randomly determined sign (λ_2). If point \mathbf{t} observes a better performance over the iteration number $ITER$, point \mathbf{g}^p is replaced by \mathbf{t} and the neighbourhood search for point \mathbf{g}^p finishes, otherwise \mathbf{g}^p is held. The pseudo-code is listed in Fig. 2.1.

In general, the local search for all particles can also reduce the risk of falling into a local solution but is time consuming. Nevertheless, recent works (Lee et al., 2012; Rocha & Fernandes, 2009a) have shown that eliminating, reducing or simplifying the local search process affects significantly the convergence, exploration, population diversity and accuracy of the EMO algorithm.

2.2.3. Total force vector computation

The total force vector computation is based on the *superposition principle* (Fig. 2.2) from the electro-magnetism theory which states: “the force exerted on a point via other points is inversely proportional to the distance between the points and directly proportional to the product of their charges” (Cowan, 1968). The particle moves following the resultant Coulomb’s force which has been produced among particles as a charge-like value. In the EMO implementation, the charge for each particle is determined by its fitness value as follows:

$$q^p = \exp \left(-n \frac{f(\mathbf{g}^p) - f(\mathbf{g}^{best})}{\sum_{h=1}^m (f(\mathbf{g}^h) - f(\mathbf{g}^{best}))} \right), \forall p \quad (2.2)$$

where n denotes the dimension of \mathbf{g}^p and m represents the population size. A higher dimensional problem usually requires a larger population. In Eq. (2.2), the particle showing the best fitness function value \mathbf{g}^{best} is called the “*best particle*”, getting the highest charge and attracting other particles holding high fitness values. The repulsion effect is applied to all other particles exhibiting lower fitness values. Both effects, attraction-repulsion are applied depending on the actual proxim-

ity between a given particle and the best-graded element. The overall resultant force between all particles determines the actual effect of the optimization process.

The final force vector for each particle is evaluated under the Coulomb's law and the superposition principle as follows:

$$\mathbf{F}^p = \sum_{h \neq p}^m \left\{ \begin{array}{l} (\mathbf{g}^h - \mathbf{g}^p) \frac{q^p q^h}{\|\mathbf{g}^h - \mathbf{g}^p\|^2} \quad \text{if } f(\mathbf{g}^h) < f(\mathbf{g}^p) \\ (\mathbf{g}^p - \mathbf{g}^h) \frac{q^p q^h}{\|\mathbf{g}^h - \mathbf{g}^p\|^2} \quad \text{if } f(\mathbf{g}^h) \geq f(\mathbf{g}^p) \end{array} \right\}, \forall p \quad (2.3)$$

where $f(\mathbf{g}^h) < f(\mathbf{g}^p)$ represents the attraction effect and $f(\mathbf{g}^h) \geq f(\mathbf{g}^p)$ represents the repulsion force (see Fig. 2.3). The resultant force of each particle is proportional to the product between charges and is inversely proportional to the distance between particles. In order to keep feasibility, the vector in expression (2.3) should be normalized as follows:

$$\hat{\mathbf{F}}^p = \frac{\mathbf{F}^p}{\|\mathbf{F}^p\|}, \quad \forall p. \quad (2.4)$$

1:	$Counter \leftarrow 1$	12:	$\mathbf{t}_d \leftarrow \mathbf{t}_d - \lambda_2 (Lenght)$
2:	$Lenght \leftarrow \delta(\max\{u_d - l_d\})$	13:	end if
3:	for $p=1$ to m do	14:	if $f(\mathbf{t}) < f(\mathbf{g}^p)$ then
4:	for $p=1$ to n do	15:	$\mathbf{g}^p \leftarrow \mathbf{t}$
5:	$\lambda_1 \leftarrow U(0,1)$	16:	$Counter \leftarrow ITER - 1$
6:	while $Counter < ITER$ do	17:	end if
7:	$\mathbf{t} \leftarrow \mathbf{g}^p$	18:	$Counter \leftarrow Counter + 1$
8:	$\lambda_2 \leftarrow U(0,1)$	19:	end while
9:	if $\lambda_1 > 0.5$ then	20:	end for
10:	$\mathbf{t}_d \leftarrow \mathbf{t}_d + \lambda_2 (Lenght)$	21:	end for
11:	Else	22:	$\mathbf{g}^{best} \leftarrow \operatorname{argmin}\{f(\mathbf{g}^p), \forall p\}$

Fig. 2.1. Pseudo-code list for the local search algorithm

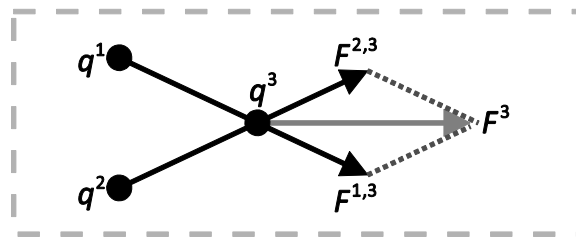


Fig. 2.2. The superposition principle

2.2.4. Movement

The change of the d -coordinate for each particle p is computed with respect to the resultant force as follows:

$$g_d^p = \begin{cases} g_d^p + \lambda \cdot \hat{F}_d^p \cdot (u_d - g_d^p) & \text{if } \hat{F}_d^p > 0 \\ g_d^p + \lambda \cdot \hat{F}_d^p \cdot (g_d^p - l_d) & \text{if } \hat{F}_d^p \leq 0 \end{cases}, \forall p \neq best, \forall d \quad (2.5)$$

In Eq. (2.5), λ is a random step length that is uniformly distributed between zero and one. u_d and l_d represent the upper and lower boundary for the d -coordinate, respectively. \hat{F}_d^p represents the d element of $\hat{\mathbf{F}}^p$. If the resultant force is positive, then the particle moves towards the highest boundary by a random step length. Otherwise it moves toward the lowest boundary. The best particle does not move at all, because it holds the absolute attraction, pulling or repelling all others in the population.

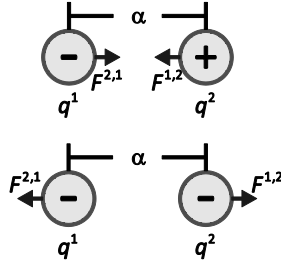


Fig. 2.3. Coulomb law: α represents the distance between charged particles, q^1, q^2 are the charges, and F is the exerted force as has been generated by the charge interaction.

The process is halted when a maximum iteration number is reached or when the value $f(\mathbf{g}^{best})$ is near to zero or to the required optimal value.

2.3. Opposition-based Learning (OBL)

Opposition-based Learning (Tizhoosh, 2005) is a new concept in computational intelligence that has been employed to effectively enhance several soft computing algorithms (Mahootchi, et al., 2007; Shokri, et al., 2006). The approach simultaneously evaluates a solution x and its opposite solution \bar{x} for a given problem, providing a renewed chance to find a candidate solution lying closer to the global optimum (Rahnamayan et al., 2007).

2.3.1. Opposite Number

Let $x \in [l, u]$ be a real number, where l and u are the lowest and highest bound respectively. The opposite of x is defined by:

$$\bar{x} = u + l - x \quad (2.6)$$

2.3.2. Opposite Point

Similarly, the opposite number definition is generalized to higher dimensions as follows: Let $\mathbf{x} = (x_1, x_2, \dots, x_n)$ be a point within a n -dimensional space, where $x_1, x_2, \dots, x_n \in \mathbb{R}$ and $x_i \in [l_i, u_i]$, $i \in 1, 2, \dots, n$. The opposite point $\bar{\mathbf{x}} = (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)$ is defined by:

$$\bar{x}_i = u_i + l_i - x_i \quad (2.7)$$

2.3.3. Opposite-based Optimization

Metaheuristic methods start by considering some initial solutions (initial population) and trying to improve them toward some optimal solution(s). The process of searching ends when some predefined criteria are satisfied. In the absence of a priori information about the solution, random guesses are usually considered. The computation time, among others algorithm characteristics, is related to the distance of these initial guesses taken from the optimal solution. The chance of starting with a closer (fitter) solution can be enhanced by simultaneously checking the opposite solution. By doing so, the fitter one (guess or opposite guess) can be chosen as an initial solution following the fact that, according to probability theory, 50% of the time a guess is further from the solution than its opposite guess (Rahnamayan et al., 2008). Therefore, starting with the closer of the two guesses (as judged by their fitness values) has the potential to accelerate convergence. The same approach can be applied not only to initial solutions but also to each solution in the current population.

By applying the definition of an opposite point, the opposition-based optimization can be defined as follows: Let \mathbf{x} be a point in a n -dimensional space (i.e. a candidate solution). Assume $f(\mathbf{x})$ is a fitness function which evaluates the quality of such candidate solution. According to the definition of opposite point, $\bar{\mathbf{x}}$ is the opposite of \mathbf{x} . If $f(\bar{\mathbf{x}})$ is better than $f(\mathbf{x})$, then \mathbf{x} is updated with $\bar{\mathbf{x}}$, otherwise current point \mathbf{x} is kept. Hence, the best point (\mathbf{x} or $\bar{\mathbf{x}}$) is modified using known operators from the population-based algorithm.

Figure 2.4 shows the opposition-based optimization procedure. In the example, Fig. 2.4(a) and 2.4(b) represent the function to be optimized and its corresponding contour plot, respectively. By applying the OBL principles to the current population P (see Fig. 2.4(b)), the three particles \mathbf{x}_1 , \mathbf{x}_2 and \mathbf{x}_3 produce a new population OP , gathering particles $\bar{\mathbf{x}}_1$, $\bar{\mathbf{x}}_2$ and $\bar{\mathbf{x}}_3$. The three fittest particles from P and OP are selected as the new population P' . It can be seen from Fig. 2.4(b) that \mathbf{x}_1 , $\bar{\mathbf{x}}_2$ and $\bar{\mathbf{x}}_3$ are three new members in P' . In this case, the transformation conducted on \mathbf{x}_1 did not provide a best chance of finding a candidate solution closer to the global optimum. Considering the OBL selection mechanism, $\bar{\mathbf{x}}_1$ is eliminated from the next generation.

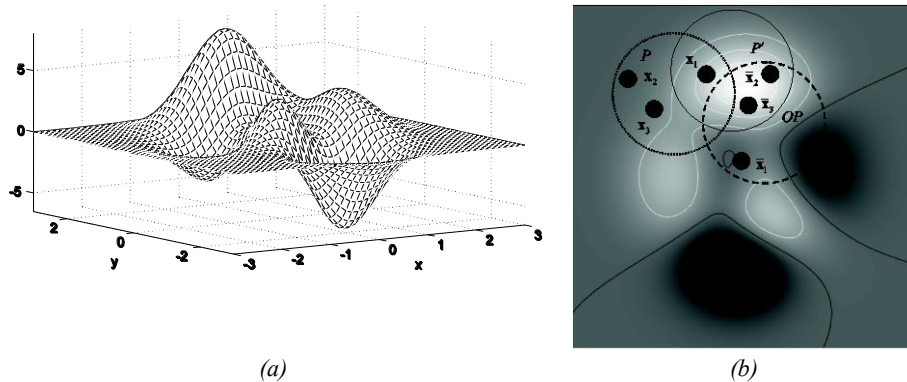


Fig. 2.4. The opposition-based optimization procedure: (a) Function to be optimized and (b) its contour plot. The current population P includes particles \mathbf{x}_1 , \mathbf{x}_2 and \mathbf{x}_3 . The corresponding opposite population OP is represented by $\bar{\mathbf{x}}_1$, $\bar{\mathbf{x}}_2$ and $\bar{\mathbf{x}}_3$. The final population P' is obtained by the OBL selection mechanism yielding particles \mathbf{x}_1 , $\bar{\mathbf{x}}_2$ and $\bar{\mathbf{x}}_3$.

2.4. Opposition-based Electromagnetism-like Optimization Algorithm

Similarly, to all metaheuristic-based optimization algorithms, two steps are fundamental for the EMO algorithm: the population initialization and the production of new generations by evolutionary operators. In the approach, the OBL scheme is incorporated to enhance both steps. However, the original EMO is considered as the main algorithm while the opposition procedures are embedded into EMO aiming to accelerate its convergence speed. Figure 2.5 shows a data flow comparison between the EMO and the OBEMO algorithm. The novel extended opposition procedures are explained in the following subsections.

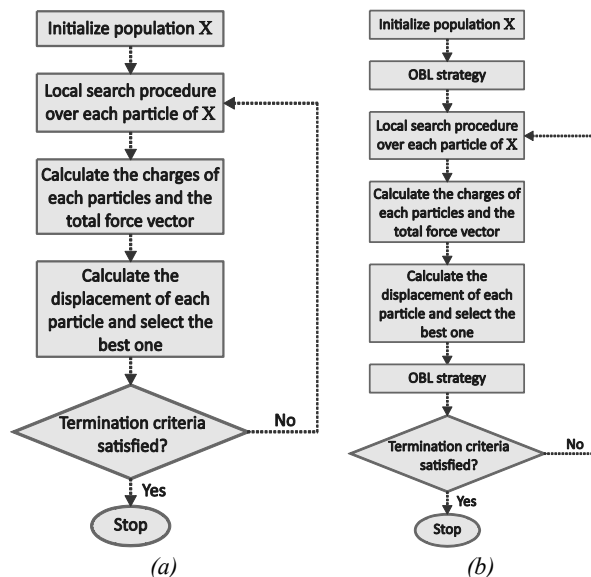


Fig. 2.5. Dataflow for: (a) the EMO method and (b) the OBEMO algorithm.

2.4.1. Opposition-based Population Initialization

In population-based meta-heuristic techniques, the random number generation is the common choice to create an initial population in absence of a priori knowledge. Therefore, as mentioned in Section 2.3, it is possible to obtain fitter starting candidate solutions by utilizing OBL despite no a-priori knowledge about the solution(s) is available. The following steps explain the overall procedure.

- 1) Initialize the population \mathbf{X} with N_p representing the number of particles.
- 2) Calculate the opposite population by

$$\bar{x}_i^j = u_i + l_i - x_i^j \quad (2.8)$$

$$i = 1, 2, \dots, n; j = 1, 2, \dots, N_p$$

where x_i^j and \bar{x}_i^j denote the i th parameter of the j th particle of the population and its corresponding opposite particle.

- 3) Select the N_p fittest elements from $\{\mathbf{X} \cup \bar{\mathbf{X}}\}$ as initial population.

2.4.2. Opposition-based production for new generation

Starting from the current population, the OBL strategy can be used again to produce new populations. In this procedure, the opposite population is calculated, and the fittest individuals are selected from the union of the current population and the opposite population. The following steps summarize the OBEMO implementation as follows:

Step 1	Generate N_p initial random particles \mathbf{x}^h to create the particle vector \mathbf{X} , with $h \in 1, 2, \dots, N_p$.
Step 2	Apply the OBL strategy by considering N_p particles from vector \mathbf{X} and generating the opposite vector $\bar{\mathbf{X}}$ through Eq. (2.7).
Step 3	Select the N_p fittest particles from $\mathbf{X} \cup \bar{\mathbf{X}}$ according to $f(\cdot)$. These particles build the initial population \mathbf{X}_0 .
Step 4	Calculate the local search procedure for each particle of \mathbf{X}_0 as follows: For a given dimension d , the particle \mathbf{x}^h is assigned to a temporary point y to store the initial information. Next, a random number is selected and combined with δ to yield the step length. Therefore, the point y is moved along that direction. The sign is determined randomly. If $f(\mathbf{x}^h)$ is minimized, the particle \mathbf{x}^h is replaced by y , ending the neighborhood-wide search for a particle h . The result is stored into the population vector \mathbf{X}_{Local} .
Step 5	Determine the best particle \mathbf{x}^{best} of the population vector \mathbf{X}_{Local} (with $\mathbf{x}^{best} \leftarrow \arg \min \{f(\mathbf{x}^h), \forall h\}$).
Step 6	Calculate the charge among particles using expression (2.2) and the vector force through Eq. (2.3). The particle showing the better objective function value holds a bigger charge and therefore a bigger attraction force.
Step 7	Change particle positions according to their force magnitude. The new particle's po-

	sition is calculated by expression (2.5). \mathbf{x}^{best} is not moved because it has the biggest force and attracts others particles to itself. The result is stored into the population vector \mathbf{X}_{Mov} .
Step 8	Apply the OBL strategy over the m particles of the population vector \mathbf{X}_{Mov} , the opposite vector $\bar{\mathbf{X}}_{Mov}$ can be calculated through Eq. (2.7).
Step 9	Select the m fittest particles from $\mathbf{X}_{Mov} \cup \bar{\mathbf{X}}_{Mov}$ according to $f(\cdot)$. Such particles represent the population \mathbf{X}_0 .
Step 10	Increase the <i>Iteration</i> index. If <i>iteration</i> = <i>MAXITER</i> or the value of $f(X)$ is smaller than the pre-defined threshold value, then the algorithm is stopped and the flow jumps to step 11. Otherwise, it jumps to step 4.
Step 11	The best particle \mathbf{x}^{best} is selected from the last iteration as it is considered as the solution.

2.5. Experimental Results

In order to test the algorithm's performance, the proposed OBEMO is compared to the standard EMO and others state-of-the-art EMO-based algorithms. In this section, the experimental results are discussed in the following subsections:

(2.5.1) Test problems

(2.5.2) Parameter settings for the involved EMO algorithms

(2.5.3) Results and discussions

2.5.1. Test problems

A comprehensive set of benchmark problems, that includes 14 different global optimization tests, has been chosen for the experimental study. According to their use in the performance analysis, the functions are divided in two different sets: original test functions ($f_1 - f_9$) and multidimensional functions ($f_{10} - f_{14}$). Every function at this paper is considered as a minimization problem itself.

The original test functions, which are shown in Table 2.1, agree to the set of numerical benchmark functions presented by the original EMO paper (Birbil & Fang, 2003). Considering that such function set is also employed by a vast majority of EMO-based new approaches, its use in our experimental study facilitates its comparison to similar works. More details can be found in (Hirche, 1979).

Function	Search domain	Global minima
$f_1(x_1, x_2) = (x_2 - \frac{5}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$	$-5 \leq x_1 \leq 10$ $0 \leq x_2 \leq 15$	0.397887
$f_2(x_1, x_2) = -\frac{-x_1^2 + 4.5x_2^2 + 2}{e^{2x_2^2}}$	$-2 \leq x_1, x_2 \leq 2$	-1.031
$f_3(x_1, x_2) = 1 + (x_1 + x_2 + 1)^2 \times (19 - 14x_1 + 13x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \times (30 + 2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 - 48x_2 - 36x_1x_2 + 27x_2^2)$	$-2 \leq x_1, x_2 \leq 2$	3.0

$f_4(\mathbf{x}) = -\sum_{i=1}^4 \alpha_i \exp\left[-\sum_{j=1}^3 A_{ij}(x_j - P_{ij})^2\right] \text{ (3-dimensional)}$ $\boldsymbol{\alpha} = [1, 1.2, 3, 3.2], \mathbf{A} = \begin{bmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 35 \end{bmatrix},$ $\mathbf{P} = 10^{-4} \begin{bmatrix} 6890 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{bmatrix}$	$0 \leq x_i \leq 1$ $i = 1, 2, 3$	-3.8627
$f_5(\mathbf{x}) = -\sum_{i=1}^4 \alpha_i \exp\left[-\sum_{j=1}^6 B_{ij}(x_j - Q_{ij})^2\right] \text{ (6-dimensional)}$ $\boldsymbol{\alpha} = [1, 1.2, 3, 3.2], \mathbf{B} = \begin{bmatrix} 10 & 3 & 17 & 3.05 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{bmatrix},$ $\mathbf{Q} = 10^{-4} \begin{bmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{bmatrix}$	$0 \leq x_i \leq 1$ $i = 1, 2, 3, \dots, 6$	-3.8623
$S_m(\mathbf{x}) = -\sum_{j=1}^m \left[\sum_{i=1}^4 (x_i - C_{ij})^2 + \beta_j \right]^{-1} \text{ (4-dimensional)}$ $\boldsymbol{\beta} = [1, 2, 2, 4, 4, 6, 3, 7, 5, 5]^T,$ $\mathbf{C} = \begin{bmatrix} 4.0 & 1.0 & 8.0 & 6.0 & 3.0 & 2.0 & 5.0 & 8.0 & 6.0 & 7.0 \\ 4.0 & 1.0 & 8.0 & 6.0 & 7.0 & 9.0 & 5.0 & 1.0 & 2.0 & 3.6 \\ 4.0 & 1.0 & 8.0 & 6.0 & 3.0 & 2.0 & 3.0 & 8.0 & 6.0 & 7.0 \\ 4.0 & 1.0 & 8.0 & 6.0 & 7.0 & 9.0 & 3.0 & 1.0 & 2.0 & 3.6 \end{bmatrix}$	$0 \leq x_i \leq 1$ $i = 1, 2, 3, 4$	
$f_6(\mathbf{x}) = S_5(\mathbf{x})$		-10.1532
$f_7(\mathbf{x}) = S_7(\mathbf{x})$		-10.4029
$f_8(\mathbf{x}) = S_{10}(\mathbf{x})$		-10.5364
$f_9(x_1, x_2) = \left(\sum_{i=1}^5 i \cos((i+1)x_1 + i) \right) \left(\sum_{i=1}^5 i \cos((i+1)x_2 + i) \right)$	$-10 \leq x_1, x_2 \leq 10$	-186.73

Table 2.1. Optimization test functions corresponding to the original test set ($f_1 - f_9$).

The major challenge of an EMO-based approach is to avoid the computational complexity that arises from the large number of iterations which are required during the local search process. Since the computational complexity depends on the dimension of the optimization problem, one set of multidimensional functions (see Table 2.2) is used in order to assess the convergence and accuracy for each algorithm. Multidimensional functions include a set of five different functions whose dimension has been fixed to 30.

Function	Search	Global minima
----------	--------	---------------

	domain	
$f_{10}(\mathbf{x}) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^{30}$	0
$f_{11}(\mathbf{x}) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i)\right) + 20$	$[-32, 32]^{30}$	0
$f_{12}(\mathbf{x}) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^{30}$	0
$f_{13}(\mathbf{x}) = \frac{\pi}{n} \left\{ 10 \sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\}$ $+ \sum_{i=1}^n u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50, 50]^{30}$	0
$f_{14}(\mathbf{x}) = \sin^2(3\pi x_1) + \sum_{i=1}^n (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)]$ $+ (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] + \sum_{i=1}^n u(x_i, 5, 100, 4)$	$[-50, 50]^{30}$	0

Table 2.2. Multidimensional test function set ($f_{10} - f_{14}$).

2.5.2. Parameter settings for the involved EMO algorithms

The experimental set aims to compare four EMO-based algorithms including the proposed OBEMO. All algorithms face 14 benchmark problems. The algorithms are listed below:

- Standard EMO algorithm (Birbil & Fang, 2003);
- Hybridizing EMO with descent search (HEMO) (Rocha & Fernandes, 2009b);
- EMO with fixed search pattern (FEMO) (Rocha & Fernandes, 2007);
- The proposed approach OBEMO.

For the original EMO algorithm described in (§. I. Birbil & Fang, 2003) and the proposed OBEMO, the parameter set is configured considering: $\delta = 0.001$ and $LISTER=4$. For the HEMO, the following experimental parameters are considered: $LsIt_{\max} = 10$, $\varepsilon_r = 0.001$ and $\gamma = 0.00001$. Such values can be assumed as the best configuration set according to (Rocha & Fernandes, 2009a). Diverging from the standard EMO and the OBEMO algorithm, the HEMO method reduces the local search phase by only processing the best found particle \mathbf{x}^{best} . The parameter set for the FEMO approach is defined by considering the following values: $N_{fe}^{\max} = 100$, $N_{ls}^{\max} = 10$, $\delta = 0.001$, $\delta^{\min} = 1 \times 10^{-8}$ and $\varepsilon_\delta = 0.1$. All aforementioned EMO-based algorithms use the same population size of $m = 50$.

2.5.3. Results

Original test functions set

On this test set, the performance of the OBEMO algorithm is compared to standard EMO, HEMO and FEMO, considering the original test functions set. Such functions, presented in Table 2.1, hold different dimensions and one known global minimum. The performance is analysed by consider-

ing 35 different executions for each algorithm. The case of no significant changes in the solution being registered (i.e. smaller than 10^{-4}) is considered as stopping criterion.

The results, shown by Table 2.3, are evaluated assuming the averaged best value $f(x)$ and the average number of executed iterations (*MAXITER*). Figure 2.6 shows the optimization process for the function f_3 and f_6 . Such function values correspond to the best case for each approach that is obtained after 35 executions.

In order to statistically analyse the results in Table 2.3, a non-parametric significance proof known as the Wilcoxon's rank test (García, et al., 2008; Santamaría, et al., 2009; Wilcoxon, 1945) has been conducted. Such proof allows assessing result differences among two related methods. The analysis is performed considering a 5% significance level over the "averaged best value of $f(x)$ " and the "averaged number of executed iterations of *MAXITER*" data.

Function		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
Dimension		2	2	2	3	6	4	4	4	2
EMO	Averaged best values $f(x)$	0.3980	-1.015	3.0123	-3.7156	-3.6322	-10.07	-10.23	-10.47	-186.71
	Averaged <i>MAXITER</i>	103	128	197	1.59E+03	1.08E+03	30	31	29	44
OBEMO	Averaged best values $f(x)$	0.3980	-1.027	3.0130	-3.7821	-3.8121	-10.11	-10.22	-10.50	-186.65
	Averaged <i>MAXITER</i>	61	83	101	1.12E+03	826	18	19	17	21
HEMO	Averaged best values $f(x)$	0.5151	-0.872	3.413	-3.1187	-3.0632	-9.041	-9.22	-9.1068	-184.31
	Averaged <i>MAXITER</i>	58	79	105	1.10E+03	805	17	18	15	22
FEMO	Averaged best values $f(x)$	0.4189	-0.913	3.337	-3.3995	-3.2276	-9.229	-9.88	-10.18	-183.88
	Averaged <i>MAXITER</i>	63	88	98	1.11E+03	841	21	22	19	25

Table 2.3. Comparative results for the EMO, the OBEMO, the HEMO and the FEMO algorithms considering the original test functions set (Table 2.1).

Table 2.4 and Table 2.5 reports the p -values produced by Wilcoxon's test for the pair-wise comparison of the "averaged best value" and the "averaged number of executed iterations" respectively, considering three groups. Such groups are formed by OBEMO vs. EMO, OBEMO vs. HEMO and OBEMO vs. FEMO. As a null hypothesis, it is assumed that there is no difference between the values of the two algorithms. The alternative hypothesis considers an actual difference between values from both approaches. The results obtained by the Wilcoxon test indicate that data cannot be assumed as occurring by coincidence (i.e. due to the normal noise contained in the process).

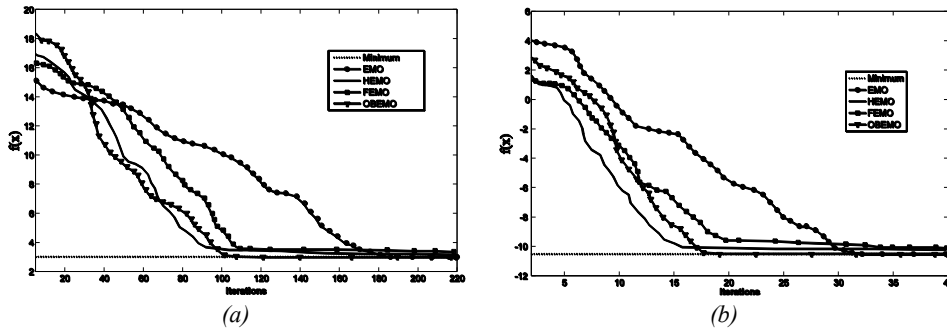


Fig. 2.6. Comparison of the optimization process for two original test functions: (a) f_3 and (b) f_6 .

Table 2.4 considers the Wilcoxon analysis with respect to the “averaged best value” of $f(x)$. The p -values for the case of OBEMO vs EMO are larger than 0.05 (5% significance level), which is a strong evidence supporting the null hypothesis which indicates that there is no significant difference between both methods.

Function	p -Values OBEMO vs.		
	EMO	HEMO	FEMO
f_1	0.3521	1.21E-04	1.02E-04
f_2	0.4237	1.05E-04	0.88E-04
f_3	0.2189	4.84E-05	3.12E-05
f_4	0.4321	1.35E-05	1.09E-05
f_5	0.5281	2.73E-04	2.21E-04
f_6	0.4219	1.07E-04	0.77E-04
f_7	0.3281	3.12E-05	2.45E-05
f_8	0.4209	4.01E-05	3.62E-05
f_9	0.2135	1.86E-05	1.29E-05

Table 2.4. Results from Wilcoxon’s ranking test considering the “averaged best value of $f(x)$ ”.

On the other hand, in cases for the p -values corresponding to the OBEMO vs HEMO and OBEMO vs FEMO, they are less than 0.05 (5% significance level), which accounts for a significant difference between the “averaged best value” data among methods. Table 2.5 considers the Wilcoxon analysis with respect to the “averaged number of executed iterations” values. Applying the same criteria, it is evident that there is a significant difference between the OBEMO vs. EMO case, despite the OBEMO vs HEMO and OBEMO vs FEMO cases offering similar results.

Multidimensional functions

In contrast to the original functions, Multidimensional functions exhibit many local minima/maxima which are, in general, more difficult to optimize. In this section the performance of the OBEMO algorithm is compared to the EMO, the HEMO and the FEMO algorithms, considering functions in Table 2.2. This comparison reflects the algorithm’s ability to escape from poor local optima and to locate a near-global optimum, consuming the least number of iterations. The dimension of such functions is set to 30. The results (Table 2.6) are averaged over 35 runs reporting the “averaged best value” and the “averaged number of executed iterations” as performance indexes.

Function	<i>p</i> -Values OBEMO vs.		
	EMO	HEMO	FEMO
f_1	2.97E-04	0.2122	0.2877
f_2	3.39E-04	0.1802	0.2298
f_3	8.64E-09	0.1222	0.1567
f_4	7.54E-05	0.2183	0.1988
f_5	1.70E-04	0.3712	0.3319
f_6	5.40E-13	0.4129	0.3831
f_7	7.56E-04	0.3211	0.3565
f_8	1.97E-04	0.2997	0.2586
f_9	1.34E-05	0.3521	0.4011

Table 2.5. Results from Wilcoxon's ranking test considering the "averaged number of executed iterations".

Function		f_{10}	f_{11}	f_{12}	f_{13}	f_{14}
Dimension		30	30	30	30	30
EMO	Averaged best values $f(x)$	2.12E-05	1.21E-06	1.87E-05	1.97E-05	2.11E-06
	Averaged <i>MAXITER</i>	622	789	754	802	833
OBEMO	Averaged best values $f(x)$	3.76E-05	5.88E-06	3.31E-05	4.63E-05	3.331E-06
	Averaged <i>MAXITER</i>	222	321	279	321	342
HEMO	Averaged best values $f(x)$	2.47E-02	1.05E-02	2.77E-02	3.08E-02	1.88E-2
	Averaged <i>MAXITER</i>	210	309	263	307	328
FEMO	Averaged best values $f(x)$	1.36E-02	2.62E-02	1.93E-02	2.75E-02	2.33E-02
	Averaged <i>MAXITER</i>	241	361	294	318	353

Table 2.6. Comparative results for the EMO, OBEMO, HEMO and the FEMO algorithms being applied to the multidimensional test functions (Table 2.2).

The Wilcoxon rank test results, presented in Table 2.7, shows that the *p*-values (regarding to the "averaged best value" values of Table 2.6) for the case of OBEMO vs EMO, indicating that there is no significant difference between both methods. *p*-values corresponding to the OBEMO vs HEMO and OBEMO vs FEMO show that there is a significant difference between the "averaged best" values among the methods. Figure 2.7 shows the optimization process for the function f_{12} and f_{14} . Such function values correspond to the best case, for each approach, obtained after 35 executions.

Table 2.8 considers the Wilcoxon analysis with respect to the "averaged number of executed iterations" values of Table 2.6. As it is observed, the outcome is similar to the results from last test on the original functions.

Function	<i>p</i> -Values OBEMO vs.		
	EMO	HEMO	FEMO
f_{10}	0.2132	3.21E-05	3.14E-05
f_{11}	0.3161	2.39E-05	2.77E-05
f_{12}	0.4192	5.11E-05	1.23E-05
f_{13}	0.3328	3.33E-05	3.21E-05
f_{14}	0.4210	4.61E-05	1.88E-05

Table 2.7. Results from Wilcoxon's ranking test considering the "best averaged values".

Function	<i>p</i> -Values OBEMO vs.		
	EMO	HEMO	FEMO
f_{10}	3.78E-05	0.1322	0.2356
f_{11}	2.55E-05	0.2461	0.1492
f_{12}	6.72E-05	0.3351	0.3147
f_{13}	4.27E-05	0.2792	0.2735
f_{14}	3.45E-05	0.3248	0.3811

Table 2.8. Results from Wilcoxon's ranking test considering the "averaged number of executed iterations".

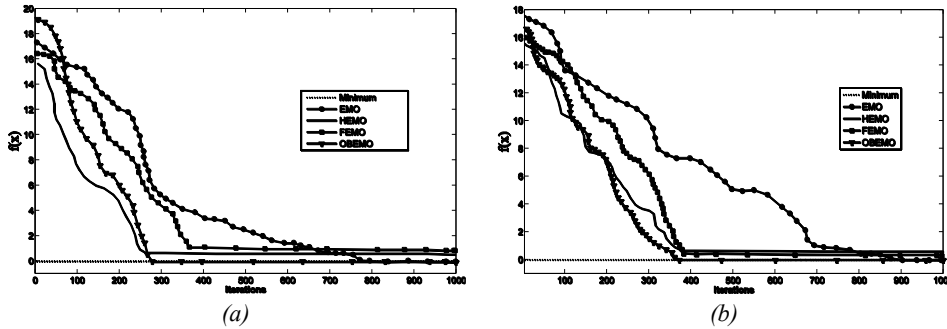


Fig. 2.7. Optimization process comparison for two multidimensional test functions: (a) f_{12} and (b) f_{14} .

2.6. Conclusions

In this chapter, an Opposition-Based EMO, named as OBEMO, has been presented by combining the opposition-based learning (OBL) strategy and the standard EMO technique. The OBL is a machine intelligence strategy which considers, at the same time, a current estimate and its opposite value to achieve a fast approximation for a given candidate solution. The standard EMO is enhanced by using two OBL steps: the population initialization and the production of new generations. The enhanced algorithm significantly reduces the required computational effort yet avoiding any detriment to the good search capabilities of the original EMO algorithm.

A set of 14 benchmark test functions has been employed for experimental study. Results are supported by a statistically significant framework, the Wilcoxon test (García et al., 2008; Santamaría et al., 2009; Wilcoxon, 1945), to demonstrate that the OBEMO is as accurate as the standard EMO yet requiring a shorter number of iterations. Likewise, it is as fast as others state-of-the-art EMO-

based algorithms such as HEMO (Takeuchi, 2008) and FEMO (Rocha & Fernandes, 2007), still keeping the original accuracy.

Although the results offer evidence to demonstrate that the Opposition-Based EMO method can yield good results on complicated optimization problems, the chapter's aim is to show that the Opposition-based Electromagnetism-like method can effectively be considered as an attractive alternative for solving global optimization problems.

Chapter 3

A Metaheuristic Optimization methodology based on Fuzzy Logic

Many processes are too complex to be manipulated quantitatively; however, humans succeed by using simple rules of thumb that are extracted from their experiences. Fuzzy Logic emulates the human reasoning in the use of imprecise information to generate decisions. Unlike traditional approaches, which require a mathematical understanding of the system, Fuzzy Logic comprises an alternative way of processing, which permits modeling complex systems through the use of human knowledge. On the other hand, several new metaheuristic algorithms have recently been proposed with interesting results. Most of them use operators based on metaphors of natural or social elements to evolve candidate solutions. In this chapter, a methodology to implement human-knowledge-based optimization strategies is presented. In the scheme, a Takagi-Sugeno Fuzzy inference system is used to reproduce a specific search strategy generated by a human expert. Therefore, the number of rules and its configuration only depend on the expert experience without considering any learning rule process. Under these conditions, each fuzzy rule represents an expert observation that models the conditions under which candidate solutions are modified in order to reach the optimal location. To exhibit the performance and robustness of the presented method, a comparison to other well-known optimization methods is conducted. The comparison considers several standard benchmark functions which are typically found in scientific literature. The results suggest a high performance of the proposed methodology.

3.1. Introduction

There are processes that humans can do much better than deterministic systems or computers, such as obstacle avoidance while driving or planning a strategy. This may be due to our unique reasoning capabilities and complex cognitive processing. Although processes can be complex, humans undertake them by using simple rules of thumb extracted from their experiences.

Fuzzy Logic (Zadeh, 1965) is a practical alternative for a variety of challenging applications since it provides a convenient method for constructing systems via the use of heuristic information. The heuristic information may come from a system-operator who has directly interacted with the process. In the Fuzzy Logic design methodology, this operator is asked to write down a set of rules on how to manipulate the process. We then incorporate these into a Fuzzy system that emulates the decision-making process of the operator (He, et al., 2015) For this reason, the partitioning of the system behavior into regions is an important characteristic of a Fuzzy system (Taur & Tao, 1997). In each region, the characteristics of the system can be simply modeled using a rule that associates the region under which certain actions are performed (Ali & Shabir, 2014). Typically, a Fuzzy model consists of a rule base, where the information available is transparent and easily readable. The Fuzzy modeling methodology has been largely exploited in several fields such as Pattern Recognition (Novák, et al., 2015; Papakostas, et al., 2013), Control (Castillo & Melin, 2014; Wang, et al., 2015) and Image Processing (Raju & Nair, 2014; Zareiforoush, et al., 2015).

Recently, several optimization algorithms based on random principles have been proposed with interesting results. Such approaches are inspired by our scientific understanding of biological or so-

cial systems, which at some abstraction level can be represented as optimization processes (Nanda & Panda, 2014a). These methods mimic the social behavior of bird flocking and fish schooling in the Particle Swarm Optimization (PSO) method (Kennedy & Eberhart, 1995b), the cooperative behavior of bee colonies in the Artificial Bee Colony (ABC) technique (Dervis Karaboga, 2005), the improvisation process that occurs when a musician searches for a better state of harmony in the Harmony Search (HS) (Geem et al., 2001), the attributes of bat behavior in the Bat Algorithm (BAT) method (Yang, 2010a), the mating behavior of firefly insects in the Firefly (FF) method (Yang, 2009), the social behaviors of spiders in the Social Spider Optimization (SSO) (Cuevas et al., 2013), the characteristics of animal behavior in a group in the Collective Animal Behavior (CAB) (Cuevas, et al., 2012) and the emulation of the differential and conventional evolution in species in the Differential Evolution (DE) (Storn & Price, 1995) and Genetic Algorithms (GA) (Goldberg, 1989), respectively.

On the other hand, the combination of Fuzzy systems with metaheuristic algorithms has recently attracted the attention in the Computational Intelligence community. As a result of this integration, a new class of systems known as Evolutionary Fuzzy Systems (EFSs) (Fernández, et al., 2015; Herrera, 2008) has emerged. These approaches basically consider the automatic generation and tuning of fuzzy systems through a learning process based on a metaheuristic method. The EFSs approaches reported in the literature can be divided into two classes (Fernández, et al., 2015; Herrera, 2008): tuning and learning.

In a tuning approach, a metaheuristic algorithm is applied to modify the parameters of an existent Fuzzy system, without changing its rule base. Some examples of tuning in EFSs include the calibration of Fuzzy controllers (Caraveo, et al., 2016; Castillo, et al., 2015), the adaptation of type-2 Fuzzy models (Olivas, et al., 2016) and the improvement of accuracy in Fuzzy models (Castillo, et al., 2016; Guerrero, et al., 2015). In learning, the rule base of a fuzzy system is generated by a metaheuristic algorithm, so that the final Fuzzy system has the capacity to accurately reproduce the modeled system. There are several examples of learning in EFSs, which consider different types of problems such as the selection of Fuzzy rules with membership functions (Alcalá-Fdez et al., 2009; Alcalá, et al., 2011), rule generation (Alcalá-Fdez, et al., 2011; Alcalá, et al., 2007) and determination of the entire Fuzzy structure (Carmona, et al., 2011; Cordon, 2011; Cruz-Ramírez, et al., 2014).

The proposed method cannot be considered an EFSs approach, since the Fuzzy system, used as optimizer, is not automatically generated or tuned by a learning procedure. On the contrary, its design is based on expert observations extracted from the optimization process. Therefore, the number of rules and its configuration are fixed, remaining static during its operation. Moreover, in a typical EFSs scheme, a metaheuristic algorithm is used to find an optimal base rule for a Fuzzy system with regard to an evaluation function. Different to such approaches, in the presented method here, a Fuzzy system is employed to obtain the optimum value of an optimization problem. Hence, the produced Fuzzy system directly acts as any other metaheuristic algorithm conducting the optimization strategy implemented in its rules.

A metaheuristic algorithm is conceived as a high-level problem-independent methodology that consists of a set of guidelines and operations to develop an optimization strategy. In this chapter, it is described how the Fuzzy Logic design methodology can be used to construct algorithms for optimization tasks. As opposed to “conventional” metaheuristic approaches where the focus is on the design of optimization operators that emulate a natural or social process, in this presented approach it is focused on gaining an intuitive understanding of how to conduct an efficient search strategy to model it directly into a Fuzzy system.

Although sometimes unnoticed, it is well understood that human heuristics play an important role in optimization methods. It must be acknowledged that metaheuristic approaches use human heu-

ristics to tune their corresponding parameters or to select the appropriate algorithm for a certain problem (Lessmann, et al., 2011). Under such circumstances, it is important to ask the following questions: How much of the success may be assigned to the use of a certain metaheuristic approach? How much should be attributed to its clever heuristic tuning or selection? Also, if we exploit the use of human heuristic information throughout the entire design process, can we obtain higher performance optimization algorithms?

The use of Fuzzy Logic for the construction of optimization methods presents several advantages. (A) Generation. “Conventional” metaheuristic approaches reproduce complex natural or social phenomena. Such a reproduction involves the numerical modeling of partially-known behaviors and non-characterized operations, which are sometimes even unknown (Sørensen, 2015). Therefore, it is notably complicated to correctly model even very simple metaphors. On the other hand, Fuzzy Logic provides a simple and well-known method for constructing systems via the use of human knowledge (Omid et al., 2010). (B) Transparency. The metaphors used by metaheuristic approaches lead to algorithms that are difficult to understand from an optimization perspective. Therefore, the metaphor cannot be directly interpreted as a consistent search strategy (Sørensen, 2015). On the other hand, Fuzzy Logic generates fully interpretable models whose content expresses the search strategy as humans can conduct it (Fullér, et al., 2012). (C) Improvement. Once designed, metaheuristic methods maintain the same procedure to produce candidate solutions. Incorporating changes to improve the quality of candidate solutions is very complicated and severely damages the conception of the original metaphor (Sørensen, 2015). As human experts interact with an optimization process, they obtain a better understanding of the correct search strategies that allow finding the optimal solution. As a result, new rules are obtained so that their inclusion in the existing rule base improves the quality of the original search strategy. Under the Fuzzy Logic methodology, new rules can be easily incorporated to an already existent system. The addition of such rules allows the capacities of the original system to be extended (Cordón & Herrera, 1997).

In this chapter, a methodology to implement human-knowledge-based optimization strategies is presented. In the scheme, a Takagi-Sugeno Fuzzy inference system (Takagi & Sugeno, 1985) is used to reproduce a specific search strategy generated by a human expert. Therefore, the number of rules and its configuration only depend on the expert experience without considering any learning rule process. Under these conditions, each Fuzzy rule represents an expert observation that models the conditions under which candidate solutions are modified in order to reach the optimal location. To exhibit the performance and robustness of the proposed method, a comparison to other well-known optimization methods is conducted. The comparison considers several standard benchmark functions which are typically found in the literature of metaheuristic optimization. The results suggest a high performance of the proposed methodology in comparison to existing optimization strategies.

This chapter is organized as follows: In Section 3.2, the basic aspects of fuzzy logic and the different reasoning models are introduced. In Section 3.3, the proposed methodology is exposed. Section 3.4 discusses the characteristics of the proposed methodology. In Section 3.5 the experimental results and the comparative study is presented. In Section 3.6, conclusions are drawn. Additionally, in Section 3.7 are exhibited the benchmark functions employed.

3.2. Fuzzy logic and reasoning models

This section presents an introduction to the main Fuzzy Logic concepts. The discussion particularly considers the Takagi-Sugeno Fuzzy inference model (Takagi & Sugeno, 1985).

3.2.1. Fuzzy logic concepts

A Fuzzy set (A) (Zadeh, 1965) is a generalization of a Crisp or Boolean set, which is defined in a universe of discourse X . A is a linguistic label which defines the Fuzzy set through the word A . Such a word defines how a human expert perceives the variable X in relationship to A . The Fuzzy set (A) is characterized by a membership function $\mu_A(x)$ which provides a measure of degree of similarity of an element x from X to the Fuzzy set A . It takes values in the interval $[0,1]$, that is:

$$\mu_A(x) : X \rightarrow [0,1] \quad (3.1)$$

Therefore, a generic variable x_c can be represented using multiple Fuzzy sets $\{A_1^c, A_2^c, \dots, A_m^c\}$, each one modeled by a membership function $\{\mu_{A_1^c}(x_c), \mu_{A_2^c}(x_c), \dots, \mu_{A_m^c}(x_c)\}$.

A Fuzzy system is a computing model based on the concepts of Fuzzy Logic. It includes three conceptual elements: a rule base, which contains a selection of Fuzzy rules; a database, which defines the membership functions used by the Fuzzy rules; and a reasoning mechanism, which performs the inference procedure. There are two different inference Fuzzy systems: Mamdani (Mamdani & Assilian, 1999) and Takagi-Sugeno (TS) (Takagi & Sugeno, 1985).

The central difference between the two inference models is in the consequent section of the Fuzzy systems. In the Mamdani model, all the structure of the Fuzzy system has linguistic variables and Fuzzy sets. However, the consequent section of the TS model consists of mathematical functions. Different to the Mamdani structure, the TS model provides computational efficiency and mathematical simplicity in the rules (Bagis & Konar, 2016). Therefore, in order to obtain higher modelling accuracy with fewer rules, the TS Fuzzy model is a good candidate that obtains better models when the rules are described as functional associations defined in several local behaviors (Bagis & Konar, 2016; Guney & Sarikaya, 2009). Since the available knowledge for the design of the Fuzzy system conceived in our approach includes functional, local behaviors, the TS inference model has been used in this work for the system modeling.

3.2.2. The Takagi-Sugeno (TS) fuzzy model

TS Fuzzy systems allow us to describe complicated nonlinear systems by decomposing the input space into several local behaviors, each of which is represented by a simple regression model (Taur & Tao, 1997). The main component of a TS Fuzzy system is the set of its K Fuzzy rules. They code the human knowledge that explains the performance of the actual process. Each rule denoted by R^i relates the input variables to a consequence of its occurrence. A typical TS Fuzzy rule is divided in two parts: Antecedent (I) and consequent (II), which are described as follows:

$$R^i : \overbrace{\text{IF } x_1 \text{ is } A_p^1 \text{ and } x_2 \text{ is } A_q^2, \dots, \text{ and } x_n \text{ is } A_r^n}^{\text{I}} \text{ Then } \underbrace{y_i = g_i(\mathbf{x})}_{\text{II}}, \quad i = 1, 2, \dots, K \quad (3.2)$$

where $\mathbf{x} = [x_1, x_2, \dots, x_n]^T$ is the n -dimensional input variable and y_i represents the output rule. $g(\mathbf{x})$ is a function which can be modeled by any function as long as it can appropriately describe the behavior of the system within the Fuzzy region specified by the antecedent of rule i . In Eq.(3.2), p , q and r symbolizes one Fuzzy set which models the behavior of variables x_1 , x_2 and x_n , respectively.

3.2.2.1. Antecedent (I)

The antecedent is a logical combination of simple prepositions of the form “ x_e is A_d^e ”. Such a preposition, modeled by the membership function $\mu_{A_d^e}(x_e)$, provides a measure of degree of similarity between x_e and the Fuzzy set A_d^e . Since the antecedent is concatenated by using the “and” connector, the degree of fulfilment of the antecedent $\beta_i(\mathbf{x})$ is calculated using a t-norm operator such as the minimum:

$$\beta_i(\mathbf{x}) = \min\left(\mu_{A_1^e}(x_1), \mu_{A_2^e}(x_2), \dots, \mu_{A_n^e}(x_n)\right) \quad (3.3)$$

3.2.2.2. Consequent (II)

$g_i(\mathbf{x})$ is a function which can be modeled by any function as long as it can appropriately describe the behavior of the system within the Fuzzy region specified by the antecedent of rule i .

3.2.2.2. Inference in the TS model

The global output y of a TS Fuzzy system is composed as the concatenation of the local behaviors, and can be seen as the weighted mean of the consequents:

$$y = \frac{\sum_{i=1}^K \beta_i(\mathbf{x}) \cdot y_i}{\sum_{i=1}^K \beta_i(\mathbf{x})} \quad (3.4)$$

where $\beta_i(\mathbf{x})$ is the degree of fulfilment of the i th rule’s antecedent and y_i is the output of the consequent model of that rule. Fig. 3.1 shows the Fuzzy reasoning procedure for a TS Fuzzy system with two rules. The example considers two variables (x_1, x_2) and only two membership functions (I and II) for each variable. Now, it should be clear that the spirit of Fuzzy Logic systems resembles that of “divide and conquer”. Therefore, the antecedent of a Fuzzy rule defines a local Fuzzy region, while the consequent describes the behavior within the region

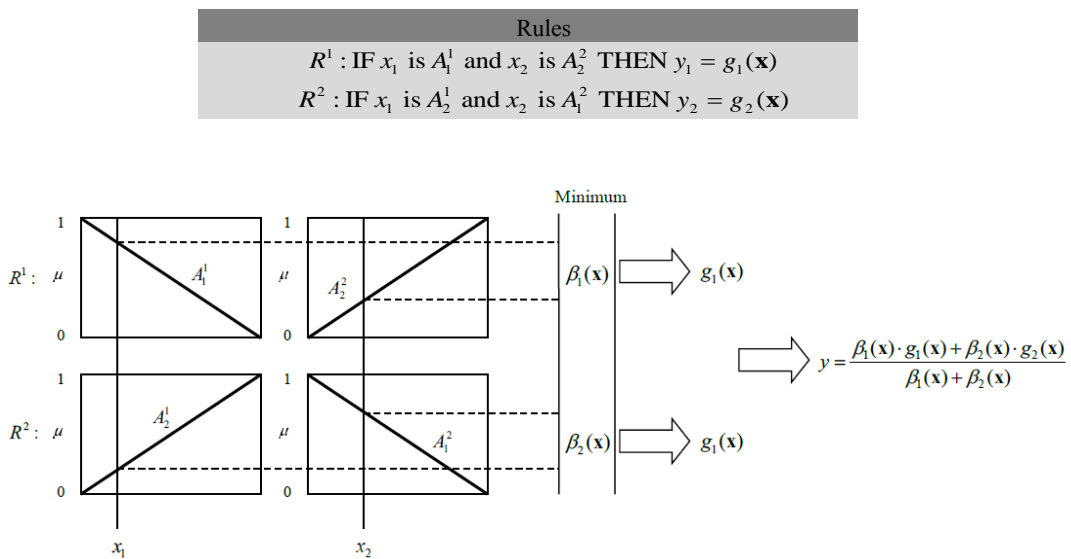


Fig. 3.1. TS Fuzzy model

3.3. The Fuzzy-based methodology

Since there is no specific solution for several kinds of complex problems, human experts often follow a trial-and-error approach to solve them. Under this process, humans obtain experience as the knowledge gained through the interaction with the problem. In general, a Fuzzy system is a model that emulates the decisions and behavior of a human that has specialized knowledge and experience in a particular field. Therefore, a Fuzzy system is then presumed to be capable of reproducing the behavior of a target system. For example, if the target system is a human operator in charge of a chemical reaction process, then the Fuzzy system becomes a Fuzzy controller that can regulate the chemical process. Similarly, if the target system is a person who is familiar with optimization strategies and decision-making processes, then the Fuzzy inference becomes a Fuzzy expert system that can find the optimal solution to a certain optimization problem, as if the search strategy were conducted by the human expert.

In this chapter we present a methodology for emulating human search strategies in an algorithmic structure. In this section, the Fuzzy optimization approach is explained in detail. First, each component of the Fuzzy system is described; then, the complete computational procedure is presented.

Under a given set of circumstances, an expert provides a description of how to conduct an optimization strategy for finding the optimal solution to a generic problem using natural language. Then, the objective is to take this “linguistic” description and model it into a Fuzzy system. The linguistic representation given by the expert is divided into two parts: (A) linguistic variables and (B) rule base formulation.

(A) Linguistic variables describe the way in which a human expert perceives the circumstances of a certain variable in terms of its relative values. One example is the velocity that could be identified as low, moderate and high. (B) Rule base formulation captures the construction process of a set of IF-THEN associations. Each association (rule) expresses the conditions under which certain actions are performed. Typically, a Fuzzy model consists of a rule base that maps Fuzzy regions to actions. In this context, the contribution of each rule to the behavior of the Fuzzy system will be different depending on the operating region.

3.3.1. Optimization strategy

Most of the optimization methods have been designed to solve the problem of finding a global solution to a nonlinear optimization problem with box constraints in the following form (Baldick, 2006):

$$\begin{aligned} & \text{maximize} && f(\mathbf{x}), && \mathbf{x} = (x_1, \dots, x_n) \in \mathbb{R}^n \\ & \text{subject to} && \mathbf{x} \in \mathbf{X} \end{aligned} \quad (3.5)$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a nonlinear function whereas $\mathbf{X} = \{\mathbf{x} \in \mathbb{R}^n \mid l_i \leq x_i \leq u_i, i = 1, \dots, n\}$ is a bounded feasible search space, constrained by the lower (l_i) and upper (u_i) limits.

To solve the optimization problem presented in Eq. (3.5), from a population-based perspective (Dan, 2013), a set \mathbf{P}^k ($\{\mathbf{p}_1^k, \mathbf{p}_2^k, \dots, \mathbf{p}_N^k\}$) of N candidate solutions (individuals) evolves from an initial state ($k=0$) to a maximum number of generations ($k=Maxgen$). In the first step, the algorithm initiates producing the set of N candidate solutions with values that are uniformly distributed be-

tween the pre-specified lower (l_i) and upper (u_i) limits. In each generation, a group of evolutionary operations are applied over the population \mathbf{P}^k to generate the new population \mathbf{P}^{k+1} . In the population, an individual \mathbf{p}_i^k ($i \in [1, \dots, N]$) corresponds to a n -dimensional vector $\{p_{i,1}^k, p_{i,2}^k, \dots, p_{i,n}^k\}$ where the dimensions represent the decision variables of the optimization problem to be solved. The quality of a candidate solution \mathbf{p}_i^k is measured through an objective function $f(\mathbf{p}_i^k)$ whose value corresponds to the fitness value of \mathbf{p}_i^k . As the optimization process evolves, the best individual \mathbf{g} (g_1, g_2, \dots, g_n) seen so-far is conserved, since it represents the current best available solution.

In the presented approach, an optimization human-strategy is modelled in the rule base of a TS Fuzzy inference system, so that the implemented Fuzzy rules express the conditions under which candidate solutions from \mathbf{P}^k are evolved to new positions \mathbf{P}^{k+1} .

3.3.1.1. Linguistic variables characterization (A)

To design a fuzzy system from expert knowledge, it is necessary the characterization of the linguistic variables and the definition of a rule base. A linguistic variable is modeled through the use of membership functions. They represent functions which assign a numerical value to a subjective perception of the variable. The number and the shape of the membership functions that model a certain linguistic variable depend on the application context (Wong, et al., 2015). Therefore, in order to maintain the design of the Fuzzy system as simple as possible, we characterize each linguistic variable by using only two membership functions (Yap, Wong, & Tiong, 2013). One example is the variable velocity V that could be defined by the membership functions: low (μ_L) and high (μ_H). Such membership function are mutually exclusive or disjoint. Therefore, if $\mu_L = 0.7$, then $\mu_H = 0.3$. Assuming that the linguistic variable velocity V has a numerical value inside the interval from 0 to 100 revolutions per minute (rpm), μ_L and μ_H are characterized according to the membership functions shown in Figure 3.2.

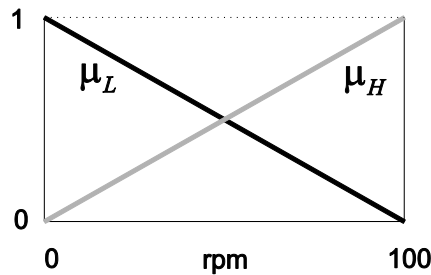


Fig. 3.2. Example of membership functions that characterize a linguistic variable.

3.3.1.2. Rule base formulation (B)

Several optimization strategies can be formulated by using human knowledge. In this section, a simple search strategy is formulated considering basic observations of the optimization process. Therefore, the simplest search strategy is to move candidate solutions to search regions of the space where it is expected to find the optimal solution. Since the values of the objective function are only known in the positions determined by the candidate solutions, the locations with the highest probabilities of representing potential solutions are those located near the best candidate solution in terms of its fitness value.

Taking this into consideration, a simple search strategy could be formulated by the following four rules:

1. IF the distance from \mathbf{p}_i^k to \mathbf{g} is short AND $f(\mathbf{p}_i^k)$ is good THEN \mathbf{p}_i^k is moved towards (Attraction) \mathbf{g} .
This rule represents the situation where the candidate solution \mathbf{p}_i^k is moved to the best candidate solution seen so-far \mathbf{g} in order to improve its fitness quality. Since the fitness values of \mathbf{p}_i^k and \mathbf{g} are good in comparison to other members of \mathbf{P}^k , the region between \mathbf{p}_i^k and \mathbf{g} maintains promising solutions that could improve \mathbf{g} . Therefore, with this movement, it is expected to explore the unknown region between \mathbf{p}_i^k and \mathbf{g} . In order to show how each rule performs. Fig. 3 shows a simple example which expresses the conditions under which action rules are executed. In the example, a population \mathbf{P}^k of five candidate solutions is considered (see Fig. 3.3(a)). In the case of rule 1, as it is exhibited in Fig. 3.3(b), the candidate solution \mathbf{p}_5^k that fulfills the rule requirements is attracted to \mathbf{g} .
2. IF the distance from \mathbf{p}_i^k to \mathbf{g} is short AND $f(\mathbf{p}_i^k)$ is bad THEN \mathbf{p}_i^k is moved away from (Repulsion) \mathbf{g} .
In this rule, although the distance between \mathbf{p}_i^k and \mathbf{g} is short, the evidence shows that there are no good solutions between them. Therefore, the improvement of \mathbf{p}_i^k is searched in the opposite direction of \mathbf{g} . A visual example of this behavior is presented in Fig. 3.3(c).
3. IF the distance from \mathbf{p}_i^k to \mathbf{g} is large AND $f(\mathbf{p}_i^k)$ is good THEN \mathbf{p}_i^k is refined.
Under this rule, a good candidate solution \mathbf{p}_i^k that is far from \mathbf{g} is refined by searching within its neighborhood. The idea is to improve the quality of competitive candidate solutions which have already been found (exploitation). Such a scenario is presented in Figure 3.3(d) where the original candidate solution \mathbf{p}_2^k is substituted by a new position \mathbf{p}_2^{k+1} which is randomly produced within the neighborhood of \mathbf{p}_2^k .
4. IF the distance from \mathbf{p}_i^k to \mathbf{g} is large AND $f(\mathbf{p}_i^k)$ is bad THEN a new position is randomly chosen.
This rule represents the situation in Figure 3.3(e) where the candidate solution \mathbf{p}_4^k is so bad and so far from \mathbf{g} that is better to replace it by other solution (\mathbf{p}_4^{k+1}) randomly produced within the search space \mathbf{X} .

Each of the four rules listed above is a “linguistic rule” which contains only linguistic information. Since linguistic expressions are not well-defined descriptions of the values that they represent, linguistic rules are not accurate. They represent only conceptual ideas about how to achieve a good optimization strategy according to the human perspective. Under such conditions, it is necessary to define the meaning of their linguistic descriptions from a computational point of view.

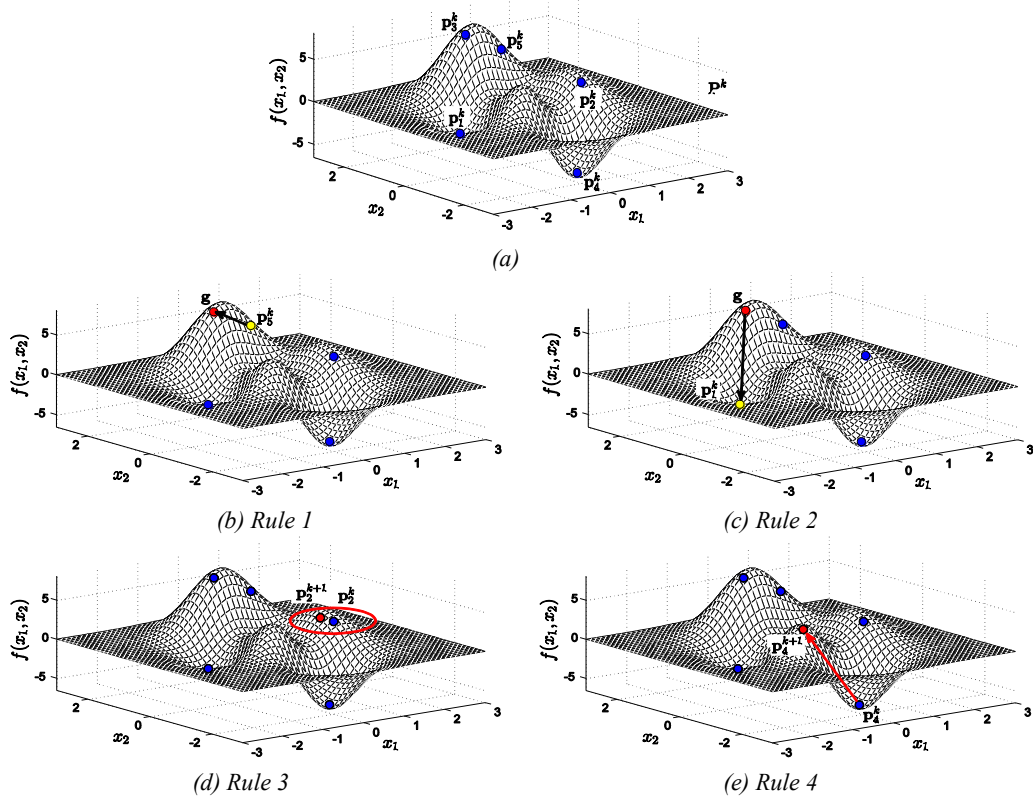


Fig. 3.3. Visual example that expresses the conditions under which action rules are executed. (a) Current configuration of the candidate solution population \mathbf{P}^k , (b) rule 1, (c) rule 2, (d) rule 3 and (e) rule 4.

3.3.1.3. Implementation of the TS fuzzy system

In this section, we will discuss the implementation of the expert knowledge concerning the optimization process in a TS fuzzy system.

I) Membership functions and antecedents

In the rules, two different linguistic variables are considered, distance from de candidate solution \mathbf{p}_i^k to the best solution \mathbf{g} ($D(\mathbf{p}_i^k, \mathbf{g})$) and the fitness value of the candidate solution ($f(\mathbf{p}_i^k)$). Therefore, $D(\mathbf{p}_i^k, \mathbf{g})$ is characterized by two membership functions: short and large (see 3.3.1.1). On the other hand, $f(\mathbf{p}_i^k)$ is modeled by the membership functions good and bad. Fig. 3.4 shows the fuzzy membership functions for both linguistic variables. The distance $D(\mathbf{p}_i^k, \mathbf{g})$ is defined as the Euclidian distance $\|\mathbf{g} - \mathbf{p}_i^k\|$. Therefore, as it is exhibited in Fig. 3.4(a), two complementary membership functions define the relative distance $D(\mathbf{p}_i^k, \mathbf{g})$: short (**S**) and large (**L**). Their support values are 0 and d_{\max} , where d_{\max} represents the maximum possible distance delimited by the search space \mathbf{X} which is defined as follows:

$$d_{\max} = \sqrt{\sum_{s=1}^d (u_s - l_s)^2}, \quad (3.6)$$

where d represents the number of dimensions in the search space \mathbf{X} . In the case of $f(\mathbf{p}_i^k)$, two different membership functions define its relative value: bad (**B**) and good (**G**). Their support values are f_{\min} and f_{\max} . These values represent the minimum and maximum fitness values seen so-far. Therefore, they can be defined as following:

$$f_{\min} = \min_{\substack{i \in \{1, 2, \dots, N\} \\ k \in \{1, 2, \dots, gen\}}} (f(\mathbf{p}_i^k)) \text{ and } f_{\max} = \max_{\substack{i \in \{1, 2, \dots, N\} \\ k \in \{1, 2, \dots, gen\}}} (f(\mathbf{p}_i^k)) \quad (3.7)$$

From Eq. (3.7), it is evident that $f_{\max} = f(\mathbf{g})$. If a new minimum or maximum value of $f(\mathbf{p}_i^k)$ is detected during the evolution process, it replaces the past values of f_{\min} or f_{\max} . Fig. 3.4(b) shows the membership functions that describe $f(\mathbf{p}_i^k)$.

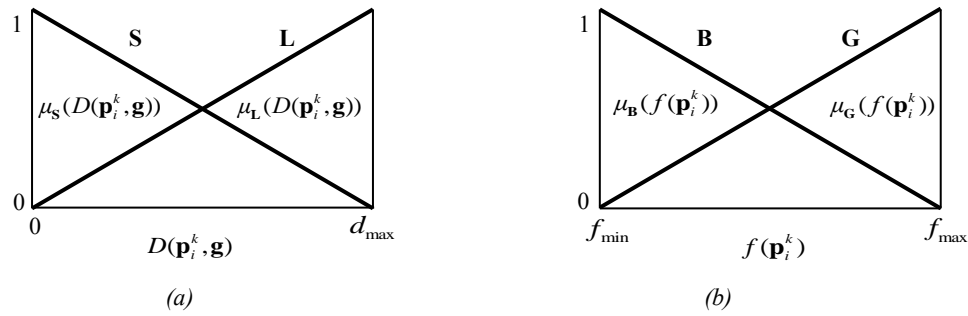


Fig. 3.4. Membership functions for (a) distance $D(\mathbf{p}_i^k, \mathbf{g})$ and (b) for $f(\mathbf{p}_i^k)$.

Considering the membership functions defined in Fig. 3.4, the degree of fulfilment of the antecedent $\beta_w(\mathbf{x})$ for each rule ($w \in [1, 2, 3, 4]$) is defined in Table 3.1.

Rule	Degree of fulfilment $\beta_w(\mathbf{x})$
1	$\beta_1(\mathbf{p}_i^k) = \min(\mu_S(D(\mathbf{p}_i^k, \mathbf{g})), \mu_G(f(\mathbf{p}_i^k)))$
2	$\beta_2(\mathbf{p}_i^k) = \min(\mu_S(D(\mathbf{p}_i^k, \mathbf{g})), \mu_B(f(\mathbf{p}_i^k)))$
3	$\beta_3(\mathbf{p}_i^k) = \min(\mu_L(D(\mathbf{p}_i^k, \mathbf{g})), \mu_G(f(\mathbf{p}_i^k)))$
4	$\beta_4(\mathbf{p}_i^k) = \min(\mu_L(D(\mathbf{p}_i^k, \mathbf{g})), \mu_B(f(\mathbf{p}_i^k)))$

Table 3.1. Degree of fulfilment of the antecedent $\beta_w(\mathbf{x})$ for each rule ($w \in [1, 2, 3, 4]$).

II) Actions or consequents

Actions or Consequents are functions which can be modeled by any function as long as it can appropriately describe the desired behavior of the system within the fuzzy region specified by the antecedent of a rule i ($i \in [1, 2, 3, 4]$). The consequents of the four rules are modeled by using the following behaviors.

Rule 1. Attraction

$$At(\mathbf{p}_i^k) = \left| f_{\max} - f(\mathbf{p}_i^k) \right| \cdot (\mathbf{g} - \mathbf{p}_i^k) \cdot \alpha_1, \quad (3.8)$$

where α_1 represents a tuning factor. Under this rule, the function $At(\mathbf{p}_i^k)$ produces a change of position in the direction of the attraction vector $(\mathbf{g} - \mathbf{p}_i^k)$. The magnitude depends on the difference of the fitness values between \mathbf{g} and \mathbf{p}_i^k .

Rule 2. Repulsion

$$Rep(\mathbf{p}_i^k) = |f_{\max} - f(\mathbf{p}_i^k)| \cdot (\mathbf{g} + \mathbf{p}_i^k) \cdot \alpha_2, \quad (3.9)$$

where α_2 represents a tuning factor.

Rule 3. Refining or perturbation.

$$Ref(\mathbf{p}_i^k) = |f_{\max} - f(\mathbf{p}_i^k)| \cdot \mathbf{v} \cdot \gamma, \quad (3.10)$$

where $\mathbf{v} = \{v_1, v_2, \dots, v_d\}$ is a random vector where each component represents a random number between -1 and 1 whereas γ represents a tuning factor. In this rule, the function $Ref(\mathbf{p}_i^k)$ generates a random position within the limits specified by $\pm |f_{\max} - f(\mathbf{p}_i^k)|$.

Rule 4. Random substitution.

$$Ran(\mathbf{p}_i^k) = \mathbf{r}, \quad (3.11)$$

where $\mathbf{r} = \{r_1, r_2, \dots, r_d\}$ is a random vector where each component r_u represents a random number between the lower (l_u) and upper (u_u) limits of the search space \mathbf{X} .

III) Inference of the TS model.

The global change of position $\Delta \mathbf{p}_i^k$ of the TS fuzzy system is composed as the concatenation of the local behaviors produced by the four rules, and can be seen as the weighted mean of the consequents:

$$\Delta \mathbf{p}_i^k = \frac{At(\mathbf{p}_i^k) \cdot \beta_1(\mathbf{p}_i^k) + Rep(\mathbf{p}_i^k) \cdot \beta_2(\mathbf{p}_i^k) + Ref(\mathbf{p}_i^k) \cdot \beta_3(\mathbf{p}_i^k) + Ran(\mathbf{p}_i^k) \cdot \beta_4(\mathbf{p}_i^k)}{\beta_1(\mathbf{p}_i^k) + \beta_2(\mathbf{p}_i^k) + \beta_3(\mathbf{p}_i^k) + \beta_4(\mathbf{p}_i^k)}, \quad (3.12)$$

Once $\Delta \mathbf{p}_i^k$ has been calculated, the new position \mathbf{p}_i^{k+1} is calculated as follows:

$$\mathbf{p}_i^{k+1} = \mathbf{p}_i^k + \Delta \mathbf{p}_i^k, \quad (3.13)$$

3.3.2. Computational procedure

The proposed algorithm is implemented as an iterative process in which several operations are executed. Such operations can be summarized in the form of pseudo-code in Algorithm 3.1. The proposed method uses as input information the number of candidate solutions (N), the maximum

number of generations (*Maxgen*), and the tuning parameters α_1 , α_2 , γ . Similar to other metaheuristic algorithms, in the first step (line 2), the algorithm initiates producing the set of N candidate solutions with values that are uniformly distributed between the pre-specified lower and upper limits. These candidate solutions represent the first population \mathbf{P}^0 . After initialization, the best element \mathbf{g} in terms of its fitness value is selected (line 3). Then, for each particle \mathbf{p}_i^k its distance to the best value \mathbf{g} is calculated (line 6). With $D(\mathbf{p}_i^k, \mathbf{g})$ and $f(\mathbf{p}_i^k)$, the search optimization strategy implemented in the fuzzy system is applied (lines 7-9). Under such circumstances, the antecedents (line 7) and consequents (line 8) are computed while the final displacement $\Delta \mathbf{p}_i^k$ is obtained as a result of the operation performed by the TS model (line 9). Afterwards, the new position \mathbf{p}_i^{k+1} is updated (line 10). Once the new population \mathbf{P}^{k+1} is obtained as a result of the iterative operation of lines 6-10, the best value \mathbf{g} is updated (line 12). This cycle is repeated until the maximum number of the iterations *Maxgen* has been reached.

Algorithm 3.1. Pseudo-code for the proposed Fuzzy method	
1. Input: $N, Maxgen, \alpha_1, \alpha_2, \gamma, k=0$.	
2. $\mathbf{P}^k \leftarrow \text{Initialize}(N)$;	
3. $\mathbf{g} \leftarrow \text{SelectBestParticle}(\mathbf{P}^k)$;	
4. while $k \leq Maxgen$ do	
5. for ($i=1; i > N; i++$)	
6. $D(\mathbf{p}_i^k, \mathbf{g}) \leftarrow \text{CalculateTheDistancetoTheBest}(\mathbf{p}_i^k, \mathbf{g})$;	
7. $[\beta_1, \beta_2, \beta_3, \beta_4] \leftarrow \text{EvaluateAntecedents}(D(\mathbf{p}_i^k, \mathbf{g}), f(\mathbf{p}_i^k))$;	Fuzzy System
8. $[At, Rep, Ref, Ran] \leftarrow \text{EvaluateConsequents}(\mathbf{p}_i^k, \mathbf{g}, f(\mathbf{p}_i^k))$;	
9. $\Delta \mathbf{p}_i^k \leftarrow \text{InferenceTS}(\beta_1, \beta_2, \beta_3, \beta_4, At, Rep, Ref, Ran)$;	
10. $\mathbf{p}_i^{k+1} \leftarrow \mathbf{p}_i^k + \Delta \mathbf{p}_i^k$	
11. end for	
12. $\mathbf{g} \leftarrow \text{SelectBestParticle}(\mathbf{P}^{k+1})$;	
13. $k \leftarrow k+1$	
14. end while	
15. Output: \mathbf{g}	

Algorithm 3.1. Summarized operations of the proposed Fuzzy method.

3.4. Discussion about the proposed methodology

In this section, several important characteristics of the proposed algorithm are discussed. First, in sub-section 3.4.1, interesting operations of the optimization process are analyzed. Next, in sub-section 3.4.2 the modelling properties of the proposed approach are highlighted.

3.4.1. Optimization algorithm

A metaheuristic algorithm is conceived as a high-level problem-independent methodology that consists of a set of guidelines and operations to develop an optimization strategy. In the proposed

methodology, a Fuzzy system is generated based on expert observations about the optimization process. The final Fuzzy system then performs various Fuzzy Logic operations to produce a new candidate solution \mathbf{p}_i^{k+1} from the current solution \mathbf{p}_i^k . During this process, the following operations are involved:

1. Determination of the degree of membership between the input data ($D(\mathbf{p}_i^k, \mathbf{g}), f(\mathbf{p}_i^k)$) and the defined Fuzzy sets ("short & large" or "good & bad").
2. Calculation of the degree of relevance for each rule based on the degree of fulfilment $\beta_w(\mathbf{x})$ for each rule ($w \in [1, 2, 3, 4]$) in the antecedent part of the rule.
3. Evaluation of the consequent of each rule: *At, Rep, Ref, Ran*.
4. Derivation of the new candidate solution \mathbf{p}_i^{k+1} based on the weighted mean of the consequent functions, according to the TS model.

Under such circumstances, the generated Fuzzy system is applied over all candidate solutions from \mathbf{P}^k in order to produce the new population \mathbf{P}^{k+1} . This procedure is iteratively executed until a termination criteria has been reached.

3.4.2. Modeling characteristics

Metaheuristic algorithms are widely employed for solving complex optimization problems. Such algorithms have been developed by a combination of deterministic models and randomness, mimicking the behavior of biological or social systems. Most of the metaheuristic methods divide the individual behavior into several processes which show no coupling among them (Nanda & Panda, 2014b; Sørensen, 2015).

In the presented methodology, the produced Fuzzy system models a complex optimization strategy. This modeling is accomplished by a number of Fuzzy IF-THEN rules, each of which describes the local behavior of the model. In particular, the rules express the conditions under which new positions are explored. In order to calculate a new candidate solution \mathbf{p}_i^{k+1} , the consequent actions of all rules are aggregated. In this way, all the actions are presented in the computation of a certain solution \mathbf{p}_i^{k+1} , but with different influence levels. By coupling local behaviors, fuzzy systems are able to reproduce complex global behaviors. An interesting example of such modeling characteristics is rule 1 and rule 2. If these rules are individually analyzed, the attraction and repulsion movements conducted by the functions are completely deterministic. However, when all rules are considered, rule 3 and rule 4 add randomness to the final position of \mathbf{p}_i^{k+1} .

3.5. Experimental study

An illustrative set of 19 functions has been used to examine the performance of our approach. These test functions represent the base functions from the latest competition on single objective optimization problems at CEC2015 (Liang, et al., 2014). Tables 3.17, 3.18 and 3.19 in Section 3.7 show the benchmark functions employed in our experiments. These functions are ordered into three different classes: Unimodal (Table 3.17), Multimodal (Table 3.18) and Hybrid (Table 3.19) test functions. In the tables, n represents the dimension in which the function is operated, $f(\mathbf{x}^*)$ characterizes the optimal value of the function in the position \mathbf{x}^* and S is the defined search space.

The main objective of this section is to present the performance of the proposed algorithm on numeric optimization problems. Moreover, the results of our method are compared with some popular optimization algorithms by using the complete set of benchmark functions. The results of the proposed algorithm are verified by a statistical analysis of the experimental data.

The experimental results are divided into two sub-sections. In the first section, the performance of the proposed algorithm is evaluated with regard to its tuning parameters. In the second section, the overall performance of the presented method is compared to six popular optimization algorithms based on random principles.

3.5.1. Performance evaluation with regard to its own tuning parameters

The three parameters of the rules α_1 , α_2 and γ affect the expected performance of the proposed Fuzzy optimization algorithm. In this sub-section we analyze the behavior of the proposed algorithm considering the different settings of these parameters. All experiments have been executed on a Pentium dual-core computer with 2.53-GHz and 4-GB RAM under MATLAB 8.3. For the sake of simplicity, only the functions from f_1 to f_{14} (unimodal and multimodal) have been considered in the tuning process. In the simulations, all the functions operate with a dimension $n=30$. As an initial condition, the parameters α_1 , α_2 and γ are set to their default values $\alpha_1 = 1.4$, $\alpha_2 = 0.05$ and $\gamma = 0.005$. Then, in our analysis, the three parameters are evaluated one at a time, while the other two parameter remain fixed to their default values. To minimize the stochastic effect of the algorithm, each benchmark function is executed independently a total of 10 times. As a termination criterion, the maximum number of iterations (*Maxgen*) is set to 1000. In all simulations, the population size N is fixed to 50 individuals.

In the first stage, the behavior of the proposed algorithm is analyzed considering different values for α_1 . In the analysis, the values of α_1 vary from 0.6 to 1.6 whereas the values of α_2 and γ remain fixed at 0.05 and 0.005, respectively. In the simulation, the proposed method is executed independently 30 times for each value of α_1 on each benchmark function. The results are registered in Table 3.2. These values represent the average best fitness values (\bar{f}) and the standard deviations (σ_f) obtained in terms of a certain parameter combination of α_1 , α_2 and γ . From Table 3.2, we can conclude that the proposed Fuzzy algorithm with $\alpha_1 = 1.4$ maintains the best performance on functions $f_1 - f_9$, and f_{11} . Under this configuration, the algorithm obtains the best results in 9 out of 14 functions. On the other hand, when the parameter α_1 is set to any other value, the performance of the algorithm is inconsistent, producing generally bad results.

In the second stage, the performance of the proposed algorithm is evaluated considering different values for α_2 . In the experiment, the values of α_2 are varied from 0.01 to 0.1 whereas the values of α_1 and γ remain fixed at 1.4 and 0.005, respectively. The statistical results obtained by the Fuzzy algorithm using different values of α_2 are presented in Table 3.3. From Table 3.3, it is clear that our Fuzzy optimization algorithm with $\alpha_2 = 0.05$ outperforms the other parameter configurations. Under this configuration, the algorithm obtains the best results in 8 of the 14 functions. However, if another parameter set is used, it results in a bad performance.

Finally, in the third stage, the performance of the proposed algorithm is evaluated considering different values for γ . In the simulation, the values of γ are varied from 0.001 to 0.01 whereas the values of α_1 and α_2 remain fixed at 1.4 and 0.05, respectively. Table 3.4 summarizes the results of this experiment. From the information provided by Table 3.4, it can be seen that the proposed

fuzzy algorithm with $\gamma = 0.005$ obtains the best performance on functions $f_1, f_2, f_3, f_4, f_6, f_7, f_{10}, f_{12}$ and f_{13} . However, when the parameter γ takes any other value, the performance of the algorithm is inconsistent. Under this configuration, the algorithm presents the best possible performance, since it obtains the best indexes in 10 out of 14 functions.

α_1		0.6	0.7	0.8	0.9	1	1.2	1.3	1.4	1.5	1.6
f_1	\bar{f}	6.95E-55	7.74E-89	3.97E-167	1.01E-39	1.02E-193	0.00E+00	4.26E-29	3.08E-281	1.15E-28	6.85E-28
	σ_f	3.67E-54	4.24E-88	0.00E+00	5.54E-39	0.00E+00	0.00E+00	2.19E-28	0.00E+00	3.41E-28	1.08E-27
f_2	\bar{f}	6.10E-23	1.14E-53	1.12E-124	2.49E-139	1.08E-158	7.16E-22	1.31E-78	2.66E-207	2.03E-15	7.52E+00
	σ_f	3.34E-22	6.07E-53	4.81E-124	1.36E-138	0.00E+00	3.89E-21	7.18E-78	0.00E+00	4.31E-15	2.35E+01
f_3	\bar{f}	4.69E-10	1.80E-17	2.22E-22	3.23E-22	2.00E-27	4.96E-24	4.11E-27	1.00E-27	2.68E-18	1.93E-11
	σ_f	1.62E-09	6.81E-17	8.87E-22	1.64E-21	4.73E-27	2.66E-23	9.21E-27	1.50E-27	1.12E-17	5.52E-11
f_4	\bar{f}	1.55E-23	1.48E-30	1.51E-130	4.64E-180	2.84E-112	6.13E-19	2.00E-183	3.85E-220	9.09E-16	5.16E-15
	σ_f	7.09E-23	8.12E-30	8.25E-130	0.00E+00	1.56E-111	3.31E-18	0.00E+00	0.00E+00	2.76E-15	6.91E-15
f_5	\bar{f}	2.85E+01	2.85E+01	2.85E+01	2.55E+01	3.85E+01	1.75E+01	2.65E+01	3.04E-03	2.85E+01	2.99E+01
	σ_f	4.38E-02	3.86E-02	3.87E-02	4.37E-02	3.04E-02	3.72E-02	4.50E-02	3.02E-02	4.58E-02	4.72E-02
f_6	\bar{f}	2.15E-02	1.05E-02	1.19E-02	1.57E-02	1.59E-02	1.67E-02	1.69E-02	7.94E-03	1.98E-02	1.92E-02
	σ_f	1.90E-02	3.86E-03	1.13E-02	1.56E-02	1.49E-02	1.37E-02	1.49E-02	1.89E-03	1.80E-02	9.92E-03
f_7	\bar{f}	8.98E-03	3.22E-03	2.22E-03	1.88E-03	1.75E-03	2.07E-03	1.79E-03	1.36E-03	1.59E-03	1.64E-03
	σ_f	1.27E-02	2.65E-03	2.17E-03	1.48E-03	1.62E-03	2.26E-03	1.92E-03	1.10E-03	1.45E-03	1.50E-03
f_8	\bar{f}	-4.95E+03	-6.12E+03	-5.01E+04	-5.13E+04	-4.96E+03	-3.02E+03	-5.14E+04	-5.58E+04	-4.94E+03	-5.21E+03
	σ_f	4.36E+02	5.09E+02	4.15E+02	5.55E+02	5.08E+02	4.78E+02	4.24E+02	4.10E+02	4.85E+02	4.60E+02
f_9	\bar{f}	6.20E+01	2.91E+01	1.70E+01	1.57E+01	5.94E+00	5.21E+00	5.86E+00	4.76E-01	9.94E+00	2.02E+01
	σ_f	6.08E+01	5.32E+01	4.17E+01	4.27E+01	3.17E+01	2.77E+01	2.90E+01	2.38E+00	3.77E+01	5.22E+01
f_{10}	\bar{f}	8.70E-15	7.16E-15	7.99E-15	9.18E-15	9.41E-15	1.04E-14	1.19E-14	8.47E-15	1.07E-14	1.38E-14
	σ_f	5.39E-15	3.92E-15	3.61E-15	2.53E-15	2.57E-15	5.22E-15	6.00E-15	3.82E-15	6.64E-15	7.83E-15
f_{11}	\bar{f}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	3.46E-05
	σ_f	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.32E-04
f_{12}	\bar{f}	8.76E-02	8.32E-02	8.42E-02	7.82E-02	7.70E-02	8.73E-02	9.45E-02	9.59E-01	3.84E+00	6.49E+02
	σ_f	2.74E-02	3.35E-02	4.73E-02	2.30E-02	1.80E-02	2.78E-02	2.15E-02	3.26E+00	8.79E+00	2.65E+03
f_{13}	\bar{f}	2.88E-01	3.30E-01	3.69E-01	3.77E-01	3.99E-01	3.72E-01	4.27E-01	3.91E-01	2.44E+01	1.83E+06
	σ_f	1.42E-01	1.03E-01	1.63E-01	1.57E-01	2.20E-01	1.16E-01	3.78E-01	1.76E-01	9.67E+01	1.00E+07
f_{14}	\bar{f}	-8.43E+02	-8.33E+02	-8.31E+02	-8.29E+02	-8.43E+02	-8.97E+02	-8.98E+02	-8.90E+02	-8.86E+02	-8.84E+02
	σ_f	1.14E+01	9.37E+00	1.06E+01	1.12E+01	1.68E+01	2.54E+01	2.19E+01	1.80E+01	2.13E+01	2.48E+01

Table 3.2. Experimental results obtained by the proposed algorithm using different values of α_1 .

In general, the experimental results shown in Tables 3.2, 3.3 and 3.4 suggest that a proper combination of the parameter values can improve the performance of the proposed method and the quality of solutions. In this experiment we can conclude that the best parameter set is composed by the following values: $\alpha_1 = 1.4$, $\alpha_2 = 0.05$ and $\gamma = 0.005$.

Once the parameters α_1 , α_2 and γ have been experimentally set, it is possible to analyze their influence in the optimization process. In the search strategy, integrated in the Fuzzy system,

α_1 modifies the attraction that a promising individual experiments with regard to the best current element in the population. This action aims to improve the solution quality of the individual, considering that the unexplored region between the promising solution and the best element could contain a better solution. On the other hand, α_2 adjusts the repulsion to which a low quality individual is undergone. This operation intends to enhance the quality of the bad candidate solution through a movement in opposite direction of the best current element. This repulsion is considered, since there is evidence that the unexplored section between the low quality solution and the best current element does not enclose promising solutions. Finally, γ defines the neighborhood around a promising solution, from which a local search operation is conducted. The objective of this process is to refine the quality of each solution that initially maintains an acceptable fitness value.

α_2		0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.1
f_1	\bar{f}	1.01E-39	3.25E-28	3.39E-28	2.35E-28	5.18E-49	1.48E-28	1.36E-28	2.61E-28	2.15E-28	2.29E-28
	σ_f	5.54E-39	9.09E-28	1.02E-27	7.44E-28	2.34E-48	5.19E-28	4.95E-28	8.03E-28	6.36E-28	6.94E-28
f_2	\bar{f}	6.25E-16	3.61E-16	1.67E-22	2.66E-20	7.16E-22	1.15E-19	7.85E-16	3.29E-17	4.82E-16	6.57E-30
	σ_f	2.85E-15	1.97E-15	9.11E-22	1.45E-19	3.89E-21	5.41E-19	2.97E-15	1.53E-16	2.47E-15	3.59E-29
f_3	\bar{f}	4.96E-24	9.65E-27	9.29E-26	2.22E-25	1.97E-27	2.52E-23	2.81E-21	4.94E-22	7.99E-23	6.69E-21
	σ_f	2.66E-23	3.49E-26	4.33E-25	1.12E-24	2.02E-27	1.37E-22	1.53E-20	2.64E-21	2.25E-22	2.13E-20
f_4	\bar{f}	1.96E-15	1.20E-15	3.44E-17	5.99E-18	3.08E-29	1.92E-20	4.91E-26	6.13E-19	3.98E-16	1.42E-28
	σ_f	5.12E-15	4.45E-15	1.33E-16	3.28E-17	1.69E-28	9.01E-20	2.69E-25	3.31E-18	2.18E-15	7.76E-28
f_5	\bar{f}	3.85E+01	3.85E+01	2.55E+01	1.55E+01	1.99E-04	1.85E-01	4.85E-02	1.23E+01	1.35E+01	2.85E+01
	σ_f	4.45E-02	4.42E-02	4.38E-02	4.52E-02	3.74E-02	5.02E-02	3.45E-02	4.12E-02	5.02E-02	4.04E-02
f_6	\bar{f}	1.55E-02	1.47E-02	2.11E-02	2.15E-02	1.07E-03	1.73E-02	1.94E-02	1.78E-02	2.35E-01	2.04E-02
	σ_f	9.87E-03	5.13E-03	2.10E-02	4.72E-03	1.67E-02	6.85E-03	1.90E-02	7.59E-03	1.17E+00	2.21E-02
f_7	\bar{f}	1.03E-03	1.56E-03	1.09E-03	1.42E-03	1.36E-03	2.17E-03	1.88E-03	2.12E-03	2.53E-03	2.55E-03
	σ_f	8.62E-04	1.60E-03	7.41E-04	1.07E-03	1.10E-03	1.22E-03	1.56E-03	2.59E-03	3.04E-03	2.10E-03
f_8	\bar{f}	-3.10E+03	-5.24E+03	-2.17E+03	-5.00E+03	-5.13E+03	-5.01E+03	-6.29E+03	-5.11E+03	-5.24E+03	-5.18E+03
	σ_f	5.08E+02	4.49E+02	4.65E+02	3.83E+02	4.77E+02	3.68E+02	5.35E+02	4.36E+02	5.03E+02	4.41E+02
f_9	\bar{f}	8.98E+00	3.41E-02	5.91E+00	9.16E-02	4.76E-01	8.28E+00	7.75E-02	4.07E+00	1.64E+01	5.83E+00
	σ_f	3.42E+01	1.87E-01	3.14E+01	2.81E-01	2.38E+00	3.15E+01	2.61E-01	2.18E+01	5.02E+01	3.13E+01
f_{10}	\bar{f}	1.12E-14	1.01E-14	1.10E-14	1.17E-14	8.47E-15	9.30E-15	9.89E-15	1.21E-14	1.21E-14	1.20E-14
	σ_f	4.88E-15	3.82E-15	4.86E-15	7.26E-15	3.44E-15	4.71E-15	3.82E-15	6.19E-15	7.11E-15	6.03E-15
f_{11}	\bar{f}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	σ_f	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{12}	\bar{f}	1.03E+00	7.77E-01	9.55E-02	1.93E+00	9.59E-01	4.14E-01	1.17E+00	1.26E+00	1.60E+00	6.41E+00
	σ_f	3.80E+00	2.63E+00	3.50E-02	4.30E+00	3.26E+00	1.75E+00	3.56E+00	4.41E+00	4.70E+00	2.28E+01
f_{13}	\bar{f}	3.54E-01	4.08E-01	1.69E+00	2.38E+00	3.51E-01	4.20E-01	1.17E+00	1.12E+00	8.94E-01	1.34E+00
	σ_f	1.91E-01	2.18E-01	5.25E+00	7.65E+00	1.76E-01	1.69E-01	4.03E+00	3.38E+00	2.53E+00	4.99E+00
f_{14}	\bar{f}	-8.85E+02	-8.84E+02	-8.91E+02	-8.88E+02	-8.90E+02	-8.87E+02	-8.82E+02	-8.86E+02	-8.94E+02	-8.82E+02
	σ_f	2.41E+01	1.81E+01	2.25E+01	2.39E+01	1.80E+01	1.47E+01	2.37E+01	1.63E+01	2.07E+01	1.54E+01

Table 3.3. Experimental results obtained by the proposed algorithm using different values of α_2

γ		0.001	0.002	0.003	0.004	0.005	0.006	0.007	0.008	0.009	0.01
f_1	\bar{f}	3.73E-28	5.52E-29	4.79E-29	1.14E-28	1.01E-39	2.04E-28	1.92E-28	1.20E-28	8.28E-29	1.27E-28

	σ_f	8.01E-28	2.74E-28	2.23E-28	5.10E-28	5.54E-39	7.66E-28	7.31E-28	4.91E-28	4.46E-28	6.93E-28
f_2	\bar{f}	1.78E-16	5.62E-16	6.03E-16	9.19E-17	5.26E-35	4.20E-16	2.44E-22	7.16E-22	3.97E-17	2.41E-18
	σ_f	8.72E-16	2.14E-15	2.30E-15	4.21E-16	2.88E-34	1.63E-15	1.34E-21	3.89E-21	2.18E-16	1.30E-17
f_3	\bar{f}	1.19E-23	5.91E-25	1.31E-23	1.74E-24	2.35E-25	4.96E-24	6.38E-22	1.91E-23	3.41E-21	7.75E-13
	σ_f	6.49E-23	2.61E-24	6.52E-23	5.07E-24	8.29E-25	2.66E-23	2.64E-21	1.04E-22	1.85E-20	4.24E-12
f_4	\bar{f}	3.34E-26	9.36E-16	5.20E-16	5.23E-16	6.13E-19	7.80E-18	5.46E-16	6.14E-16	7.75E-21	4.90E-16
	σ_f	1.52E-25	3.79E-15	2.85E-15	2.86E-15	3.31E-18	4.27E-17	2.99E-15	2.37E-15	4.24E-20	2.56E-15
f_5	\bar{f}	2.75E+01	2.85E+01	2.97E+01	3.85E-04	1.45E-01	3.55E-01	8.35E-01	1.23E+00	2.78E+01	2.85E+01
	σ_f	4.67E-02	4.29E-02	4.47E-02	2.85E-02	4.38E-02	4.52E-02	4.31E-02	3.96E-02	4.18E-02	4.49E-02
f_6	\bar{f}	1.89E-02	1.93E-02	1.60E-02	1.85E-02	1.54E-02	1.57E-02	2.17E-02	2.06E-02	2.15E-02	1.92E-02
	σ_f	1.27E-02	1.50E-02	7.99E-03	1.14E-02	5.55E-03	8.61E-03	1.45E-02	1.91E-02	1.90E-02	1.34E-02
f_7	\bar{f}	1.98E-03	1.76E-03	1.49E-03	1.58E-03	1.32E-03	1.71E-03	1.61E-03	1.95E-03	2.30E-03	1.36E-03
	σ_f	2.02E-03	1.62E-03	2.01E-03	1.56E-03	1.03E-03	1.38E-03	1.92E-03	2.03E-03	2.79E-03	1.10E-03
f_8	\bar{f}	-5.11E+02	-5.05E+03	-5.27E+04	-5.19E+03	-5.13E+04	-4.98E+03	-5.05E+03	-4.12E+02	-5.11E+02	-4.98E+03
	σ_f	5.20E+02	4.42E+02	5.69E+02	4.20E+02	4.77E+02	4.66E+02	4.54E+02	5.31E+02	3.24E+02	5.47E+02
f_9	\bar{f}	1.14E-14	6.94E+00	4.72E+00	1.07E+01	4.76E-01	6.13E+00	8.69E+00	2.16E+01	7.27E-02	1.41E+01
	σ_f	2.75E-14	2.55E+01	2.56E+01	4.05E+01	2.38E+00	3.18E+01	3.15E+01	5.64E+01	2.81E-01	4.31E+01
f_{10}	\bar{f}	1.23E-14	8.70E-15	9.06E-15	1.13E-14	1.04E-14	1.05E-14	9.06E-15	1.26E-14	8.47E-15	8.82E-15
	σ_f	6.29E-15	3.29E-15	2.97E-15	6.79E-15	2.97E-15	6.06E-15	3.82E-15	6.47E-15	5.55E-15	3.58E-15
f_{11}	\bar{f}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.78E-05	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	σ_f	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.85E-04	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{12}	\bar{f}	2.11E+00	9.59E-01	9.95E-01	7.95E-01	9.38E-02	8.96E-01	5.60E-01	1.08E+00	1.32E-01	6.74E-01
	σ_f	7.38E+00	3.26E+00	3.37E+00	3.83E+00	1.99E-02	3.17E+00	2.48E+00	3.73E+00	1.76E-01	3.12E+00
f_{13}	\bar{f}	3.98E-01	4.28E-01	4.12E-01	3.90E-01	3.85E-01	1.06E+00	3.91E-01	4.08E-01	1.14E+00	3.88E-01
	σ_f	2.68E-01	2.24E-01	3.49E-01	1.90E-01	1.51E-01	3.82E+00	1.76E-01	2.24E-01	4.12E+00	2.08E-01
f_{14}	\bar{f}	-8.86E+02	-8.82E+02	-8.91E+02	-8.91E+02	-8.90E+02	-8.93E+02	-8.93E+02	-8.97E+02	-8.89E+02	-8.89E+02
	σ_f	2.29E+01	2.03E+01	2.22E+01	2.03E+01	1.80E+01	2.72E+01	1.91E+01	2.24E+01	1.97E+01	1.98E+01

Table 3.4. Experimental results obtained by the proposed algorithm using different values of γ .

Considering their magnitude, the values of $\alpha_1 = 1.4$, $\alpha_2 = 0.05$ and $\gamma = 0.005$ indicate that the attraction procedure is the most important operation in the optimization strategy. This fact confirms that the attraction process represents the most prolific operation in the Fuzzy strategy, since it searches new solutions in the direction where high fitness values are expected. According to its importance, the repulsion operation holds the second position. Repulsion produces significant small modifications of candidate solutions in comparison to the attraction process. This result indicates that the repulsion process involves an exploration with a higher uncertainty compared with the attraction movement. This uncertainty is a consequence of the lack of knowledge, if the opposite movement may reach a position with a better fitness value. The only available evidence is that in direction of the attraction movement, it is not possible to find promising solutions. Finally, the small value of γ induces a minor vibration for each acceptable candidate solution, in order to refine its quality in terms of fitness value.

3.5.2. Comparison with other optimization approaches

In this subsection, the proposed method is evaluated in comparison with other popular optimization algorithms based on evolutionary principles. In the experiments, we have applied the Fuzzy optimization algorithm to the 19 functions from section 3.7, and the results are compared to those produced by the Harmony Search (HS) method (Geem et al., 2001), the Bat (BAT) algorithm (Yang, 2010), the Differential Evolution (DE) (Storn & Price, 1995), the Particle Swarm Optimization (PSO) method (Kennedy & Eberhart, 1995b), the Artificial Bee Colony (ABC) algorithm (Dervis Karaboga, 2005) and the Co-variance Matrix Adaptation Evolution Strategies (CMA-ES) (Hansen, et al., 1995). These are considered the most popular metaheuristic algorithms currently in use (Boussaïd, et al., 2013). In the experiments, the population size N has been configured to 50 individuals. The operation of the benchmark functions is conducted in 50 and 100 dimensions. In order to eliminate the random effect, each function is tested for 30 independent runs. In the comparison, a fixed number FN of function evaluations has been considered as a stop criterion. Therefore, each execution of a test function consists of $FN=10^4 \cdot n$ function evaluations (where n represents the number of dimensions). This stop criterion has been decided to keep compatibility with similar works published in the literature (Han, et al., 2014; Li, et al., 2016; Meng & Pan, 2016; Yu & Li, 2015).

For the comparison, all methods have been configured with the parameters, which according to their reported references reach their best performance. Such configurations are described as follows:

1. **HS** (Geem et al., 2001): $HCMR=0.7$ and $PARate=0.3$.
2. **BAT** (Yang, 2010): Loudness ($A=2$), Pulse Rate ($r=0.9$), Frequency minimum ($Q_{min} = 0$) and Frequency maximum ($Q_{max} = 1$).
3. **DE** (Storn & Price, 1995): $CR=0.5$ and $F=0.2$.
4. **PSO** (Kennedy & Eberhart, 1995b): $c_1 = 2$ and $c_2 = 2$; the weight factor decreases linearly from 0.9 to 0.2.
5. **ABC** (Dervis Karaboga, 2005): $limit=50$.
6. **CMA-ES** (Hansen et al., 1995): The source code has been obtained from the original author ("Laboratoire de Recherche en Informatique," 2017). In the experiments, some minor changes have been applied to adapt CMA-ES to our test functions, but the main body is unaltered.
7. **FUZZY**: $\alpha_1 = 1.4$, $\alpha_2 = 0.05$ and $\gamma = 0.005$.

Several tests have been conducted for comparing the performance of the proposed Fuzzy algorithm. The experiments have been divided in Unimodal functions (Table 3.17), Multimodal functions (Table 3.18) and Hybrid functions (Table 3.19).

3.5.2.1. Unimodal test functions

In this test, the performance of the Fuzzy algorithm is compared with HS, BAT, DE, PSO, CMA-ES and ABC, considering functions with only one optimum. Such functions are represented by functions f_1 to f_7 in Table 3.17. In the test, all functions have been operated in 50 dimensions ($n=50$). The experimental results obtained from 30 independent executions are presented in Table 3.5. They report the averaged best fitness values (\bar{f}) and the standard deviations (σ_f) obtained in the runs. We have also included the best (f_{Best}) and the worst (f_{Worst}) fitness values obtained during the total number of executions. The best entries in Table 3.5 are highlighted in boldface.

From Table 3.5, according to the averaged best fitness value (\bar{f}) index, we can conclude that the proposed method performs better than the other algorithms in functions f_1 , f_3 , f_4 and f_7 . In the case of functions f_2 , f_5 and f_6 , the CMA-ES algorithm obtains the best results.

Unimodal functions of Table 3.17 with $n=50$.								
		HS	BAT	DE	PSO	CMA-ES	ABC	FUZZY
f_1	\bar{f}	87035.2235	121388.0212	61.1848761	4.39E+03	1.34E-11	3.09E-06	2.30E-29
	σ_f	5262.26532	6933.129294	163.555175	1261.19173	5.1938E-12	3.4433E-06	1.17237E-28
	f_{Best}	76937.413	108807.878	0.03702664	1.65E+03	5.69E-12	2.47E-07	5.17E-114
	f_{Worst}	95804.9747	138224.1125	878.436103	7.37E+03	2.55E-11	1.72E-05	6.42E-28
f_2	\bar{f}	1.3739E+14	4.31636E+17	0.04057031	4.54E+01	9.92E-06	1.39E-03	4.15E-04
	σ_f	3.188E+14	1.53734E+18	0.09738928	16.386199	2.5473E-06	0.00071159	0.00227186
	f_{Best}	1.0389E+10	1633259021	4.03E-12	2.61E+01	5.87E-06	5.62E-04	7.20E-59
	f_{Worst}	1.64E+15	7.60E+18	0.45348954	9.75E+01	1.44E-05	2.98E-03	0.01244379
f_3	\bar{f}	130472.801	297342.4211	55982.8182	1.57E+04	2.89E-03	4.14E+04	1.93E-05
	σ_f	11639.2864	99049.83213	9234.85975	9734.92204	0.00164804	4785.18216	4.2843E-05
	f_{Best}	104514.012	164628.01	36105.5799	4.23E+03	9.88E-04	2.85E+04	1.66E-10
	f_{Worst}	147659.604	563910.1737	70938.4205	4.96E+04	8.88E-03	4.84E+04	0.00018991
f_4	\bar{f}	80.1841708	90.17564768	25.8134455	2.32E+01	3.96E-04	7.35E+01	3.37E-16
	σ_f	2.55950002	1.862675447	6.30765469	3.51409694	8.2083E-05	3.60905231	1.8484E-15
	f_{Best}	73.2799506	86.11297617	15.7894785	1.73E+01	2.57E-04	6.55E+01	7.52E-70
	f_{Worst}	83.8375161	92.78058061	38.8210447	3.06E+01	5.65E-04	7.90E+01	1.01E-14
f_5	\bar{f}	1024.70257	276.2438329	52.5359064	6.04E+02	3.51E-05	4.53E+01	4.85E-04
	σ_f	100.932656	45.12095642	7.69858817	198.334321	0.49723274	1.13628434	0.0389642
	f_{Best}	783.653134	211.6001157	47.1421071	289.29993	1.21E-09	42.1783081	3.30E-09
	f_{Worst}	1211.08532	399.1608511	75.1362468	1126.38574	3.0654249	47.7422282	4.6323356
f_6	\bar{f}	88027.4244	119670.6412	43.5155273	4.51E+03	1.42E-11	4.15E-06	2.18E-07
	σ_f	5783.21576	6818.723503	80.4217558	2036.72193	5.5321E-12	8.5588E-06	0.84607249
	f_{Best}	77394.5062	105958.6224	0.01832758	1705.47866	5.88E-12	6.00E-07	1.18513127
	f_{Worst}	97765.4819	130549.7364	306.098587	13230.6439	2.85E-11	4.79E-05	5.18913374
f_7	\bar{f}	197.476174	116.8196698	0.08164158	4.43E+01	2.82E-02	6.86E-01	3.43E-04
	σ_f	28.808573	16.46542385	0.12240289	17.8200508	0.00499868	0.14547266	0.00447976
	f_{Best}	116.483527	87.64501186	0.01387586	15.7697307	0.0201694	0.41798576	0.00018152
	f_{Worst}	263.233333	156.0245904	0.65353574	85.526355	0.03888318	0.8957427	0.02057915

Table 3.5. Minimization results of unimodal functions of Table 3.17 with $n=50$.

By contrast, the rest of the algorithms presents different levels of accuracy, with ABC being the most consistent. These results indicate that the proposed approach provides better performance than HS, BAT, DE, PSO and ABC for all functions except for the CMA-ES which delivers similar results to those produced by the proposed approach. By analyzing the standard deviation (σ_f) in-

dex in Table 3.5, it becomes clear that the metaheuristic method which presents the best results also normally obtains the smallest deviations.

Wilcoxon test for unimodal functions of Table 3.17 with $n=50$.						
FUZZY vs	HS	BAT	DE	PSO	CMA-ES	ABC
f_1	5.0176E-07▲	9.4988E-08▲	7.3648E-07▲	7.8952E-05▲	7.234E-03▲	5.1480E-04▲
f_2	4.0553E-07▲	2.4620E-08▲	2.0793E-03▲	2.0182E-05▲	0.0937▶	3.0415E-03▲
f_3	2.0189E-08▲	3.7451E-08▲	1.0492E-07▲	4.1590E-05▲	0.0829▶	2.7612E-06▲
f_4	3.5470E-07▲	2.1490E-08▲	3.4081E-06▲	2.0121E-06▲	8.143E-03▲	4.1680E-07▲
f_5	1.0795E-08▲	4.0479E-09▲	2.0354E-07▲	8.1350E-09▲	0.1264▶	1.2541E-07▲
f_6	6.1769E-07▲	6.5480E-08▲	4.5972E-06▲	2.1594E-07▲	0.0741▶	2.1548E-03▲
f_7	4.3617E-07▲	1.9235E-08▲	2.8070E-04▲	5.4890E-06▲	0.1031▶	1.0430E-03▲
▲	7	7	7	7	2	7
▼	0	0	0	0	0	0
▶	0	0	0	0	5	0

Table 3.6. p -values produced by Wilcoxon test comparing FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO, FUZZY vs. CMA-ES and FUZZY vs. ABC over the “averaged best fitness values” from Table 3.5.

To statistically analyze the results of Table 3.5, a non-parametric test known as the Wilcoxon analysis (García, et al., 2009; Wilcoxon, 1945) has been conducted. It allows us to evaluate the differences between two related methods. The test is performed for the 5% (0.05) significance level over the “averaged best fitness values (\bar{f})” data. Table 3.6 reports the p -values generated by Wilcoxon analysis for the pair-wise comparison of the algorithms. For the analysis, five groups are produced: FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO, FUZZY vs. CMA-ES and FUZZY vs. ABC. In the Wilcoxon analysis, it is considered a null hypothesis that there is no notable difference between the two methods. On the other hand, it is admitted as an alternative hypothesis that there is an important difference between the two approaches. In order to facilitate the analysis of Table 3.6, the symbols ▲, ▼, and ▶ are adopted. ▲ indicates that the proposed method performs significantly better than the tested algorithm on the specified function. ▼ symbolizes that the proposed algorithm performs worse than the tested algorithm, and ▶ means that the Wilcoxon rank sum test cannot distinguish between the simulation results of the Fuzzy optimizer and the tested algorithm. The number of cases that fall in these situations are shown at the bottom of the table.

After an analysis of Table 3.6, it is evident that all p -values in the FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO and FUZZY vs. ABC columns are less than 0.05 (5% significance level) which is a strong evidence against the null hypothesis and indicates that the proposed method performs better (▲) than the HS, BAT, DE, PSO and ABC algorithms. This data is statistically significant and shows that it has not occurred by coincidence (i.e. due to the normal noise contained in the process).

In the case of the comparison between FUZZY and CMA-ES, the FUZZY method maintains a better (▲) performance in functions f_1 and f_4 . In functions f_2 , f_3 , f_5 , f_6 and f_7 the CMA-ES presents a similar performance to the FUZZY method. This fact can be seen from the column FUZZY vs. CMA-ES, where the p -values of functions f_2 , f_3 , f_5 , f_6 and f_7 are higher than 0.05 (▶). These results reveal that there is no statistical difference in terms of precision between FUZZY and CMA-ES, when they are applied to the aforementioned functions. In general, the re-

sults of the Wilcoxon analysis demonstrate that the presented FUZZY algorithm performs better than most of the other methods.

Unimodal functions of Table 3.17 with $n=100$.								
		HS	BAT	DE	PSO	CMA-ES	ABC	FUZZY
f_1	\bar{f}	2.19E+05	2.63E+05	3.89E+02	1.43E+04	1.32E-05	2.45E-02	1.89E-16
	σ_f	10311.7753	14201.563	323.607739	2920.78144	3.2095E-06	0.02575058	1.0358E-15
	f_{Best}	1.74E+05	2.30E+05	1.26E+01	9.64E+03	8.00E-06	4.46E-03	6.43E-68
	f_{Worst}	2.30E+05	2.89E+05	1.38E+03	2.09E+04	2.04E-05	1.26E-01	5.67E-15
f_2	\bar{f}	7.24E+37	1.31E+45	5.73E-01	1.51E+02	1.26E-02	9.28E-02	1.89E-08
	σ_f	2.587E+38	6.9056E+45	0.57775559	41.1235147	0.00287131	0.02545392	1.03445E-07
	f_{Best}	5.45E+31	3.36E+34	2.94E-02	9.05E+01	8.91E-03	5.40E-02	4.84E-46
	f_{Worst}	1.35E+39	3.79E+46	2.56E+00	2.52E+02	2.38E-02	0.18353771	5.67E-07
f_3	\bar{f}	4.98E+05	1.15E+06	2.84E+05	7.90E+04	8.76E-04	1.84E+05	2.07E-08
	σ_f	58467.2769	312595.437	27132.9955	34174.7164	0.7743521	20108.5821	0.54899784
	f_{Best}	3.37E+05	4.69E+05	2.31E+05	3.71E+04	8.76E-04	1.27E+05	3.94E-09
	f_{Worst}	616974.994	1942095.52	356515.579	160139.954	145362.58	219982.397	2.25159901
f_4	\bar{f}	9.01E+01	9.46E+01	4.05E+01	2.81E+01	2.16E-06	9.04E+01	2.63E-15
	σ_f	1.11508888	0.85959226	6.84190892	3.2581793	0.03982112	1.8297347	6.0638E-15
	f_{Best}	8.69E+01	9.29E+01	2.70E+01	2.15E+01	1.39E-08	8.37E+01	1.24E-67
	f_{Worst}	9.16E+01	9.62E+01	5.55E+01	3.40E+01	2.91E-01	9.30E+01	2.02E-14
f_5	\bar{f}	2.70E+03	1.15E+03	1.29E+02	3.95E+03	9.09E-04	1.04E+02	9.86E-04
	σ_f	540.831375	88.3793364	18.7799545	641.937017	1.45575683	6.27511831	0.14584341
	f_{Best}	0	980.500576	101.873611	2571.71519	1.32E-05	96.9508931	2.43E-06
	f_{Worst}	3247.18778	1308.54456	167.973469	5316.76433	94.8276069	121.37168	99.203712
f_6	\bar{f}	2.21E+05	2.64E+05	3.70E+02	1.45E+04	1.51E-05	1.92E-02	2.02E-05
	σ_f	9381.48118	18216.585	280.553973	2798.89068	2.2704E-06	0.01816388	2.87413087
	f_{Best}	198863.662	226296.005	16.8023385	9243.44125	1.10E-05	0.0025079	0.15E-05
	f_{Worst}	235307.769	288557.483	1278.66865	20954.2719	1.94E-05	0.09120152	23.7436855
f_7	\bar{f}	1.28E+03	4.67E+02	7.35E-01	4.85E+02	7.03E-02	2.20E+00	4.51E-03
	σ_f	100.811769	54.7947564	0.58830651	125.201537	0.01043989	0.36193003	0.00535871
	f_{Best}	975.480173	351.38694	0.08400938	233.702775	0.04689927	1.43278953	3.59E-05
	f_{Worst}	1424.35137	609.120842	2.38315757	806.350617	0.08989015	2.98777544	0.0213576

Table 3.7. Minimization results of unimodal functions of Table 3.17 with $n=100$.

In addition to the experiments in 50 dimensions, it was also conducted a set of simulations on 100 dimensions to test the scalability of the presented Fuzzy method. In the analysis, it was also employed all the compared algorithms in this test. The simulation results are presented in Tables 3.7 and 3.8, which report the data produced during the 30 executions and the Wilcoxon analysis, respectively. According to the averaged best fitness value (\bar{f}) index from Table 3.7, the proposed method performs better than the other algorithms in functions f_1 , f_2 , f_3 , f_4 and f_7 . In the

case of functions f_5 and f_6 , the CMA-ES algorithm obtains the best results. On the other hand, the rest of the algorithms present different levels of accuracy.

After analyzing Table 3.7, it is clear that the proposed fuzzy method presents slightly better results than CMA-ES in 100 dimensions. From Table 3.8, it is evident that all p -values in the FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO and FUZZY vs. ABC columns are less than 0.05, which indicates that the proposed method performs better than the HS, BAT, DE, PSO and ABC algorithms. In the case of FUZZY vs. CMA-ES, the FUZZY method maintains a better performance in functions f_1 , f_2 and f_4 . In functions f_3 , f_5 , f_6 and f_7 the CMA-ES presents a similar performance to the FUZZY method. This experiment shows that the more dimensions there are, the worse the performance of the CMA-ES is.

Wilcoxon test for unimodal functions of Table 3.17 with $n=100$.						
FUZZY vs	HS	BAT	DE	PSO	CMA-ES	ABC
f_1	3.0150E-08 ▲	8.0794E-08 ▲	6.4901E-06 ▲	2.0146E-07 ▲	7.9460E-04 ▲	4.0168E-05 ▲
f_2	5.3480E-11 ▲	1.1302E-11 ▲	7.4985E-05 ▲	4.7001E-06 ▲	2.4920E-04 ▲	8.2940E-04 ▲
f_3	6.0145E-07 ▲	7.0651E-08 ▲	4.6782E-07 ▲	8.4670E-06 ▲	0.0743 ►	2.3014E-07 ▲
f_4	1.4920E-07 ▲	3.7912E-07 ▲	2.0142E-06 ▲	1.4972E-06 ▲	7.4682E-04 ▲	2.1966E-07 ▲
f_5	8.7942E-06 ▲	5.4972E-06 ▲	9.4662E-05 ▲	2.1580E-07 ▲	0.1851 ►	4.7223E-05 ▲
f_6	2.7301E-07 ▲	4.7920E-07 ▲	8.0493E-05 ▲	5.7942E-06 ▲	0.2451 ►	1.4901E-04 ▲
f_7	1.0458E-07 ▲	5.4201E-06 ▲	2.0051E-04 ▲	7.6190E-06 ▲	0.0851 ►	4.610E-05 ▲
▲	7	7	7	7	3	7
▼	0	0	0	0	0	0
►	0	0	0	0	4	0

Table 3.8. p -values produced by Wilcoxon test comparing FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO, FUZZY vs. CMA-ES and FUZZY vs. ABC over the “averaged best fitness values” from Table 3.7.

3.5.2.2. Multimodal test functions

Contrary to unimodal functions, multimodal functions include many local optima. For this cause, they are, in general, more complicated to optimize. In this test the performance of our algorithm is compared with HS, BAT, DE, PSO, CMA-ES and ABC regarding multimodal functions. Multimodal functions are represented by functions from f_8 to f_{14} in Table 3.18, where the number of local minima increases exponentially as the dimension of the function increases. Under such conditions, the experiment reflects the ability of each algorithm to find the global optimum in the presence of numerous local optima. In the simulations, the functions are operated in 50 dimensions ($n=50$). The results, averaged over 30 executions, are reported in Table 3.9 in terms of the best fitness values (\bar{f}) and the standard deviations (σ_f). The best results are highlighted in boldface. Likewise, p -values of the Wilcoxon test of 30 independent repetitions are exhibited in Table 3.10. In the case of f_8 , f_{10} , f_{11} and f_{14} , the proposed fuzzy method presents a better performance than HS, BAT, DE, PSO, CMA-ES and ABC. For functions f_{12} and f_{13} , the fuzzy approach exhibits a worse performance compared to CMA-ES. Additionally, in the case of function f_9 the proposed method and ABC maintain the best performance compared to HS, BAT, DE, PSO and CMA-ES. The rest of the algorithms present different levels of accuracy, with ABC being the most consistent. In particular, this test yields a large difference in performance, which is directly related to a better trade-off between exploration and exploitation produced by the formulated rules of the proposed Fuzzy method.

Multimodal functions of Table 3.18 with $n=50$.								
		HS	BAT	DE	PSO	CMA-ES	ABC	FUZZY
f_8	\bar{f}	-415.83905	-270.254967	-0232.0393	-1.00E+04	-6.22E+03	-1.94E+04	-2.69E+05
	σ_f	318.326084	474.2974644	799.670519	1139.72504	577.603827	328.074723	338131.298
	f_{Best}	-183.25306	-253.007751	-0830.6009	-12207.27	-910.14987	-0460.0202	-602802.18
	f_{Worst}	-937.01673	-560.016959	-6849.3789	-417.20923	-277.36121	-8598.5718	-1860.8124
f_9	\bar{f}	637.314967	370.181231	94.9321639	2.78E+02	9.12E-03	6.43E-06	1.03E-06
	σ_f	25.9077403	31.0789956	23.8913991	38.1965572	8.314267	2.34825349	1.6781696
	f_{Best}	581.055495	321.398632	49.5787092	2.09E+02	6.91E-04	1.99899276	0
	f_{Worst}	681.505155	450.919651	143.664199	375.979507	342.621828	11.2277505	310.43912
f_{10}	\bar{f}	20.2950743	19.2995398	1.04072229	1.21E+01	8.74E-07	1.69E-02	1.40E-14
	σ_f	0.09866263	0.11146929	0.69779278	1.03376556	1.4921E-07	0.01136192	4.489E-15
	f_{Best}	19.9003135	18.9741996	0.0040944	9.35E+00	5.62E-07	2.92E-03	7.99E-15
	f_{Worst}	20.472367	19.576839	2.78934934	1.38E+01	1.18E-06	5.30E-02	2.22E-14
f_{11}	\bar{f}	786.564993	1072.40695	0.98725915	4.05E+01	9.87E-10	6.23E-03	0.00E+00
	σ_f	49.0195978	70.0220465	0.62998733	12.8397453	4.8278E-10	0.01154936	0
	f_{Best}	658.158623	926.062051	0.00107768	14.1594978	2.92E-10	0.01503879	0
	f_{Worst}	860.983823	1186.93137	2.4153938	64.7187195	2.32E-09	0.053391	0
f_{12}	\bar{f}	557399404	1029876322	1309.87126	4.08E+01	2.58E-12	3.67E-07	1.91E-08
	σ_f	68320767.6	150067294	4319.40539	27.0146375	1.0706E-12	4.1807E-07	0.63138873
	f_{Best}	444164964	763229039	0.20366113	16.607083	9.05E-13	2.14E-08	1.95E-08
	f_{Worst}	700961313	1277767934	17508.9826	136.891908	5.63E-12	1.84E-06	19.8206283
f_{13}	\bar{f}	1163989772	1982187734	29551.4297	1.41E+05	5.03E-11	5.98E-06	9.06E-09
	σ_f	123421334	291495991	137415.981	186740.994	2.7928E-11	7.5042E-06	0.0439066
	f_{Best}	898858903	1250159582	4.37613356	1594.63864	1.37E-11	5.06E-07	6.91E-10
	f_{Worst}	1453640537	2558128052	754699.3	735426.524	1.38E-10	3.48E-05	60.2604938
f_{14}	\bar{f}	-58.679663	-1066.80779	-937.10075	-1.40E+03	-1.84E+03	-1.32E+03	-1.96E+03
	σ_f	40.8885038	61.5793102	13.199329	77.8992722	41.6063335	29.4904745	0.06633944
	f_{Best}	-060.29598	-1213.46466	-958.29881	-527.69665	-915.89813	-395.05118	-958.30818
	f_{Worst}	-93.619964	-988.87492	-915.09727	-275.90292	-732.12078	-1262.8218	-958.04305

Table 3.9. Minimization results of multimodal functions of Table 3.18 with $n=50$.

The results of the Wilcoxon analysis, presented in Table 3.10, statistically demonstrate that the proposed algorithm performs better than HS, DE, BAT, DE and PSO in all test functions (f_8 - f_{14}). In the case of the comparison between FUZZY and CMA-ES, the FUZZY method maintains a better (\blacktriangle) performance in functions f_8 , f_9 , f_{10} , f_{11} and f_{14} .

On the other hand, in functions f_{12} and f_{13} the FUZZY method presents worse results (\blacktriangledown) than the CMA-ES algorithm. However, according to Table 3.10, the proposed FUZZY approach obtains a better performance than ABC in all cases except for function f_9 , where there is no difference in results between the two.

Wilcoxon test for Multimodal functions of Table 3.18 with $n=50$.						
FUZZY vs	HS	BAT	DE	PSO	CMA-ES	ABC
f_8	7.1350E-05 ▲	5.4032E-05 ▲	3.4760E-05 ▲	1.1452E-05 ▲	4.7201E-06 ▲	2.4993E-05 ▲
f_9	4.1640E-05 ▲	2.4886E-05 ▲	6.1472E-04 ▲	2.1235E-05 ▲	4.2910E-04 ▲	0.0783 ►
f_{10}	3.1425E-05 ▲	3.0183E-05 ▲	7.4920E-04 ▲	1.4982E-05 ▲	3.1157E-04 ▲	9.4872E-04 ▲
f_{11}	5.4971E-08 ▲	9.3345E-09 ▲	7.1350E-04 ▲	5.3791E-06 ▲	8.4973E-03 ▲	6.1540E-03 ▲
f_{12}	6.4821E-11 ▲	8.4038E-11 ▲	4.6840E-08 ▲	5.2920E-06 ▲	7.2365E-04 ▼	4.0312E-03 ▲
f_{13}	7.9824E-11 ▲	9.7930E-11 ▲	4.1622E-10 ▲	7.4682E-11 ▲	9.4003E-04 ▼	4.5513E-05 ▲
f_{14}	7.1352E-06 ▲	4.5821E-07 ▲	5.7920E-04 ▲	8.1641E-05 ▲	9.6401E-04 ▲	5.6820E-05 ▲
▲	7	7	7	7	5	6
▼	0	0	0	0	2	0
►	0	0	0	0	0	1

Table 3.10. p -values produced by Wilcoxon test comparing FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO, FUZZY vs. CMA-ES and FUZZY vs. ABC over the “averaged best fitness values” values from Table 3.9.

Multimodal functions of Table 3.18 with $n=100$.								
		HS	BAT	DE	PSO	CMA-ES	ABC	FUZZY
f_8	\bar{f}	-7.87E+03	-4.47E+03	-2.34E+04	-1.48E+04	-8.66E+03	-3.51E+04	-1.62E+05
	σ_f	584.340059	799.31231	3914.77561	1729.15916	831.547363	554.101741	47514.2941
	f_{Best}	-141.81789	-497.01004	-32939.624	-9362.7097	-0177.5382	-6655.8271	-72938.668
	f_{Worst}	-669.02042	-858.38997	-8358.9463	-2408.7377	-375.40434	-3939.9476	-87985.343
f_9	\bar{f}	1.44E+03	9.12E+02	3.98E+02	7.49E+02	2.37E+02	6.49E+01	4.00E-05
	σ_f	41.464352	60.4785656	67.0536747	67.5932069	209.403173	7.90879809	0.00015207
	f_{Best}	1293.19918	772.978721	205.209989	635.167862	74.8466353	49.9922666	0
	f_{Worst}	1503.0147	1033.80416	522.504376	865.893944	806.576683	79.9136576	0.0005994
f_{10}	\bar{f}	2.06E+01	1.98E+01	2.42E+00	1.33E+01	6.14E-04	3.01E+00	3.96E-12
	σ_f	0.06052521	0.08670819	0.82882063	0.87767923	9.6205E-05	0.31154878	2.1504E-11
	f_{Best}	2.05E+01	1.96E+01	1.12E+00	1.16E+01	4.14E-04	2.26E+00	1.51E-14
	f_{Worst}	2.08E+01	2.00E+01	4.63E+00	1.51E+01	8.49E-04	3.59E+00	1.18E-10
f_{11}	\bar{f}	1.96E+03	2.38E+03	4.46E+00	1.18E+02	1.20E-03	1.61E-01	0.00E+00
	σ_f	81.7177655	134.986806	2.67096611	22.637086	0.00028091	0.14553457	0
	f_{Best}	1773.68789	2035.67623	1.02086153	86.1907763	0.00066065	0.01309143	0
	f_{Worst}	2083.84569	2557.11279	12.932624	170.763675	2.25E-03	0.64617359	0
f_{12}	\bar{f}	1.88E+09	2.55E+09	4.85E+04	2.06E+04	2.20E-06	5.96E-03	2.67E+02
	σ_f	110307213	242921094	140231.648	50859.5682	7.1926E-07	0.01533206	1423.14186
	f_{Best}	1.64E+09	1.87E+09	1.86E+00	2.93E+01	1.27E-06	2.91E-05	4.21E-01
	f_{Worst}	2090764763	2966731722	769582.376	251450.989	4.07E-06	0.06229125	7801.38816
f_{13}	\bar{f}	3.54E+09	4.83E+09	6.63E+05	1.77E+06	2.20E+01	4.74E-03	4.71E-05
	σ_f	255789666	477261470	1076152.4	1373210.26	33.1622741	0.00582626	1.2473E-05
	f_{Best}	2936347707	3998731504	3147.16418	417561.443	9.98142144	0.00075547	2.65E-05
	f_{Worst}	3928279153	5901118241	4651437.83	7741055.37	176.314279	0.02480097	7.86E-05
f_{14}	\bar{f}	-1.57E+03	-1.82E+03	-3.83E+03	-2.46E+03	-3.57E+03	-3.78E+03	-2.30E+03

σ_f	79.1693639	76.7852589	36.6930452	97.0301039	59.5947211	22.8558428	65.2812493
f_{Best}	-722.36681	-945.12523	-900.49741	-683.50044	-676.29234	-823.95029	-455.01558
f_{Worst}	-432.17812	-695.67279	-740.63055	-280.67655	-393.55796	-3724.2658	-156.74064

Table 3.11. Minimization results of multimodal functions from Table 3.18 with $n=100$.

In addition to the 50-dimension benchmark function tests, we also performed a series of simulations with 100 dimensions by using the same set of functions in Table 3.18. The results are presented in Tables 3.11 and 3.12, which report the data produced during the 30 executions and the Wilcoxon analysis, respectively.

In Table 3.11, it can be seen that the proposed method performs better than HS, BAT, DE, PSO, CMA-ES and ABC for functions f_8 , f_9 , f_{10} , f_{11} and f_{13} . On the other hand, the CMA-ES maintains better results than HS, BAT, DE, PSO, ABC and the fuzzy optimizer for function f_{12} . Likewise, the DE method obtains better indexes than the other algorithms for function f_{14} .

From the Wilcoxon analysis shown in Table 3.12, the results indicate that the proposed method performs better than the HS, BAT, DE, PSO and ABC algorithms. In the case of FUZZY vs. CMA-ES, the FUZZY method maintains a better performance in all test functions except in problem f_{12} , where the CMA-ES produces better results than the proposed FUZZY method. This experiment also shows that the more dimensions there are, the worse the performance of the CMA-ES is.

Wilcoxon test for Multimodal functions of Table 3.18 with $n=100$.						
FUZZY vs	HS	BAT	DE	PSO	CMA-ES	ABC
f_8	8.1340E-05 ▲	6.4720E-05 ▲	4.6920E-05 ▲	3.1664E-05 ▲	7.1163E-05 ▲	3.7920E-05 ▲
f_9	1.3642E-07 ▲	9.4982E-06 ▲	7.6012E-06 ▲	8.6620E-06 ▲	5.4901E-06 ▲	3.1362E-06 ▲
f_{10}	6.9482E-07 ▲	6.3482E-07 ▲	3.5698E-06 ▲	6.1345E-06 ▲	3.1692E-04 ▲	3.9302E-06 ▲
f_{11}	4.9842E-07 ▲	8.1647E-07 ▲	7.1352E-04 ▲	5.3120E-06 ▲	2.0162E-03 ▲	9.4867E-03 ▲
f_{12}	4.2682E-08 ▲	7.6801E-08 ▲	8.4672E-07 ▲	7.4682E-07 ▲	3.0521E-06 ▼	1.6428E-05 ▲
f_{13}	4.5926E-09 ▲	6.4720E-09 ▲	6.1680E-07 ▲	7.4682E-08 ▲	9.1722E-06 ▲	7.4682E-04 ▲
f_{14}	8.1550E-05 ▲	8.9647E-05 ▲	6.4923E-04 ▼	9.4212E-03 ▲	5.4682E-04 ▲	6.0125E-04 ▲
▲	7	7	6	7	6	7
▼	0	0	1	0	1	0
▶	0	0	0	0	0	0

Table 3.12. p -values produced by Wilcoxon test comparing FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO, FUZZY vs. CMA-ES and FUZZY vs. ABC over the “averaged best fitness values” from Table 3.11.

3.5.2.3. Hybrid test functions

In this test, hybrid functions are employed to test the optimization performance of the proposed approach. Hybrid functions, shown in Table 3.19, are multimodal functions with complex behaviors, since they are built from different multimodal single functions. A detailed implementation of the hybrid functions can be found in (Wong et al., 2015). In the experiments, the performance of

our proposed Fuzzy algorithm is compared with HS, BAT, DE, PSO, CMA-ES and ABC, considering functions f_{15} to f_{19} .

In the first test, all functions have been operated in 50 dimensions ($n=50$). The experimental results obtained from 30 independent executions are presented in Tables 3.13 and 3.14. In Table 3.13, the indexes \bar{f} , σ_f , f_{Best} and f_{Worst} , obtained during the total number of executions, are reported. Furthermore, Table 3.14 presents the statistical Wilcoxon analysis of the averaged best fitness values \bar{f} from Table 3.13.

Hybrid functions of Table 3.19 with $n=50$.								
		HS	BAT	DE	PSO	CMA-ES	ABC	FUZZY
f_{15}	\bar{f}	7.9969E+13	5.1022E+21	12.3509776	9.64E+03	6.36E-06	5.23E-04	2.49E-15
	σ_f	1.3868E+14	2.7945E+22	16.9157032	5195.1729	2.0915E-06	0.00018266	6.6512E-15
	f_{Best}	1.5309E+10	8.1918E+12	0.01189652	4.19E+03	4.16E-06	2.68E-04	3.17E-58
	f_{Worst}	4.7627E+14	1.53E+23	65.8109321	2.50E+04	1.48E-05	1.02E-03	2.53E-14
f_{16}	\bar{f}	2706.73644	3508.41961	73.699317	5.99E+02	5.88E+02	4.91E+01	4.90E+01
	σ_f	103.253645	252.48393	13.915609	114.611979	9.03874746	0.19971751	0.00025806
	f_{Best}	2491.6759	2883.46006	49.0005972	393.824001	48.9964485	48.998348	48.9973691
	f_{Worst}	2876.20757	3869.21201	103.690098	878.998232	81.0717712	49.6854294	48.9981462
f_{17}	\bar{f}	1151151029	2105822689	67405.5759	3.13E+05	5.40E+01	8.96E+02	5.40E+01
	σ_f	113601255	190215425	193321.347	421404.743	0.00020018	132.212771	9.8857E-05
	f_{Best}	909668213	1637035871	413.088353	15529.3525	53.9999308	546.710586	53.9998073
	f_{Worst}	1428940501	2452560936	936409.163	1947952.34	54.0007602	1155.9351	54.000177
f_{18}	\bar{f}	2.0155E+14	3.3427E+19	54.2557544	9.06E+02	5.99E+01	4.90E+01	4.90E+01
	σ_f	3.9073E+14	1.83E+20	10.4970565	291.082635	12.4089453	0.01211373	0
	f_{Best}	911061731	3.2735E+13	49.0002421	530.607446	49.0000116	49.0021079	49
	f_{Worst}	1.85E+15	1.00E+21	97.123625	1597.0748	86.0099556	49.0607764	49
f_{19}	\bar{f}	7.7416E+14	7.7174E+18	-9.5833354	1.17E+06	-1.44E+02	-1.43E+02	2.18E+01
	σ_f	1.6757E+15	4.2201E+19	117.854019	5835860.88	0.39733093	0.29998167	472.608012
	f_{Best}	1343488881	1.939E+10	-43.748394	4511.91824	-44.056723	-43.608756	-3.2609165
	f_{Worst}	7.53E+15	2.31E+20	334.753741	32053489.3	-42.208256	-143.0037	2523.59236

Table 3.13. Minimization results of hybrid functions from Table 3.19 with $n=50$.

According to Table 3.13, the proposed approach maintains a superior performance in comparison to most of the other methods. In the case of f_{15} , f_{16} and f_{18} , the proposed fuzzy method performs better than HS, BAT, DE, PSO, CMA-ES and ABC. For function f_{19} , the fuzzy approach presents a worst performance than CMA-ES or ABC. However, in functions f_{16} and f_{18} , the proposed method and ABC maintain a better performance than HS, BAT, DE, PSO and CMA-ES. For function f_{17} the proposed FUZZY method and CMA-ES perform better than other methods. Therefore, the proposed FUZZY algorithm reaches better \bar{f} values in 4 from 5 different functions. This fact confirms that the FUZZY method is able to produce more accurate solutions than its competitors. From the analysis of σ_f in Table 3.13, it is clear that our FUZZY method obtain a better

consistency than the other algorithms, since its produced solutions present a small dispersion. As it can be expected, the only exception is function f_{19} , where the FUZZY algorithm does not achieve the best performance. Additional to such results, Table 3.13 shows that the proposed FUZZY method attains the best produced solution f_{Best} during the 30 independent executions than the other algorithms, except for f_{19} function. Besides, the worst fitness values f_{Worst} generated by the Fuzzy technique maintain a better solution quality than the other methods excluding function f_{19} . The case of obtaining the best f_{Best} and f_{Worst} indexes reflexes the remarkable capacity of the proposed FUZZY method to produce better solutions through use an efficient search strategy.

Table 3.14 shows the results of the Wilcoxon analysis over the averaged best fitness values \bar{f} from Table 3.13. They indicate that the proposed method performs better than the HS, BAT, DE and PSO algorithms. In the case of FUZZY vs. CMA-ES, the FUZZY method maintains a better performance in all test functions except in problem f_{19} , where the CMA-ES produces better results than the proposed FUZZY method. However, in the comparison between the FUZZY algorithm and ABC, FUZZY obtains the best results in all test functions except in problems f_{18} and f_{16} , where there is no statistical difference between the two methods.

Wilcoxon test for Hybrid functions of Table 3.19 with $n=50$.						
FUZZY vs	HS	BAT	DE	PSO	CMA-ES	ABC
f_{15}	4.6102E-10▲	7.6801E-11▲	8.1253E-07▲	3.1678E-09▲	1.3542E-04▲	5.6932E-05▲
f_{16}	5.6922E-08▲	6.4982E-09▲	6.3142E-04▲	8.4320E-05▲	6.4931E-05▲	0.1560▶
f_{17}	8.6523E-11▲	9.4685E-11▲	6.3352E-09▲	7.3477E-10▲	0.0956▶	4.6501-04▲
f_{18}	3.4962E-11▲	7.6851E-12▲	4.6820E-04▲	5.3102E-06▲	4.8235E-04▲	0.1986▶
f_{19}	7.6301E-10▲	9.3114E-11▲	4.3301E-07▲	6.0021E-09▲	6.3315E-07▼	5.8937E-07▼
▲	5	5	5	5	4	3
▼	0	0	0	0	1	1
▶	0	0	0	0	0	1

Table 3.14. p -values produced by Wilcoxon test comparing FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO, FUZZY vs. CMA-ES and FUZZY vs. ABC over the “averaged best fitness values” from Table 3.13.

In addition to the test in 50 dimensions, a second set of experiments have also conducted in 100 dimensions considering the same set of hybrid functions. Tables 3.15 and 3.16 present the results of the analysis in 100 dimensions. In Table 3.15, the indexes \bar{f} , σ_f , f_{Best} and f_{Worst} , obtained during the total number of executions, are reported. On the other hand, Table 14 presents the statistical Wilcoxon analysis of the averaged best fitness values \bar{f} from Table 3.15.

Table 3.15 confirms the advantage of the proposed method over HS, BAT, DE, PSO, CMA-ES and ABC. After analyzing the results, it is clear that the proposed FUZZY method produces better results than HS, BAT, DE, PSO, CMA-ES and ABC in functions f_{15} - f_{18} . However, it can be seen that the proposed method performs worse than CMA-ES and ABC in function f_{19} . Similar to the case of 50 dimensions, in the experiments of 100 dimensions, the proposed FUZZY algorithm obtains solutions with the smallest level of dispersion (σ_f). This consistency is valid for all functions, except for problem f_{19} , where the CMA-ES obtain the best σ_f value. Considering the f_{Best} and f_{Worst} indexes, similar conclusion can be established that in the case of 50 dimensions.

In 100 dimension, it is also observed that the proposed FUZZY technique surpass all algorithms in the production of high quality solutions.

Hybrid functions of Table 3.19 with $n=100$.								
		HS	BAT	DE	PSO	CMA-ES	ABC	FUZZY
f_{15}	\bar{f}	1.07E+38	1.45E+46	1.67E+02	3.58E+04	8.48E-03	8.13E-02	1.03E-05
	σ_f	2.6648E+38	7.9319E+46	212.77092	19777.8798	0.00248591	0.04144673	5.64261E-05
	f_{Best}	1.88E+28	1.31E+37	5.17E+00	1.58E+04	5.99E-03	4.35E-02	4.50E-44
	f_{Worst}	1.21E+39	4.34E+47	1.15E+03	9.15E+04	1.75E-02	2.66E-01	3.09E-04
f_{16}	\bar{f}	6.46E+03	8.00E+03	1.90E+02	1.36E+03	1.55E+02	1.81E+02	9.90E+01
	σ_f	316.896114	426.761823	28.0273818	129.05773	18.161094	18.5185506	0.00187142
	f_{Best}	5753.69747	7125.84623	148.521771	1116.8234	122.010095	129.807203	98.9958572
	f_{Worst}	6997.63377	8965.08163	260.542179	1681.54333	187.628218	206.634292	99.0057217
f_{17}	\bar{f}	3.57E+09	5.03E+09	6.85E+05	2.10E+06	5.95E+02	3.72E+03	1.09E+02
	σ_f	251070990	401513619	1158365.05	1426951.79	84.33472	459.761456	0.0026802
	f_{Best}	2908492728	4028081811	4767.47553	428429.323	428.761773	2973.30401	108.99997
	f_{Worst}	3953742523	5605713616	4789383.94	6274361.22	784.972289	4756.54052	109.01469
f_{18}	\bar{f}	3.30E+38	5.29E+43	1.33E+02	2.09E+03	1.49E+02	3.34E+02	1.08E+02
	σ_f	1.2038E+39	2.669E+44	20.7615151	481.839456	23.6851947	908.915277	8.76059629
	f_{Best}	5.02E+29	4.38E+34	1.01E+02	1.24E+03	1.12E+02	9.90E+01	99.6507148
	f_{Worst}	5.86E+39	1.47E+45	1.92E+02	3.51E+03	2.10E+02	4.24E+03	134.545048
f_{19}	\bar{f}	1.01E+38	1.49E+44	9.38E+03	6.45E+07	-2.94E+02	-2.01E+02	4.29E+07
	σ_f	3.9907E+38	6.2388E+44	40545.7303	217095412	0.55047399	1.27097202	200132558
	f_{Best}	4.34E+29	1.43E+36	-1.10E+02	4.71E+04	-2.95E+02	-295.68221	-9.45E+01
	f_{Worst}	2.18E+39	3.36E+45	2.23E+05	1.17E+09	-2.92E+02	-88.145624	1.08E+09

Table 3.15. Minimization results of hybrid functions from Table 3.19 with $n=100$.

On the other hand, the data obtained from the Wilcoxon analysis (Table 3.16) demonstrates that the proposed FUZZY method performs better than the other metaheuristic algorithms in all test functions, except in problem f_{18} , where the CMA-ES and ABC produce the best results.

In the Table 3.16, it is also summarized the results of the analysis through the symbols \blacktriangle , \blacktriangledown , and \blacktriangleright . The conclusions of the Wilcoxon test statistically validate the results of Table 3.15. They indicate that the superior performance of the FUZZY method is as a consequence of a better search strategy and not for random effects.

Wilcoxon test for Hybrid functions of Table 3.19 with $n=100$.						
FUZZY vs	HS	BAT	DE	PSO	CMA-ES	ABC
f_{15}	8.4682E-12 \blacktriangle	9.7624E-12 \blacktriangle	6.4950E-07 \blacktriangle	7.0012E-08 \blacktriangle	3.1261E-04 \blacktriangle	7.6823E-04 \blacktriangle
f_{16}	7.6332E-05 \blacktriangle	8.4220E-05 \blacktriangle	6.5010E-04 \blacktriangle	2.0035E-04 \blacktriangle	3.9630E-04 \blacktriangle	6.0012E-11 \blacktriangle

f_{17}	5.3422E-08 ▲	6.8892E-08 ▲	3.3019E-07 ▲	9.0394E-07 ▲	4.2961E-04 ▲	8.6301E-05 ▲
f_{18}	4.9302E-12 ▲	8.3670E-12 ▲	5.6312E-04 ▲	3.4621E-05 ▲	6.0341E-04 ▲	1.3025E-05 ▲
f_{19}	6.9210E-11 ▲	2.4950E-12 ▲	6.3301E-06 ▲	2.0182E-04 ▲	6.3019E-07 ▼	4.1305E-07 ▼
▲	5	5	5	5	4	4
▼	0	0	0	0	1	1
▶	0	0	0	0	0	0

Table 3.16. p -values produced by Wilcoxon test comparing FUZZY vs. HS, FUZZY vs. BAT, FUZZY vs. DE, FUZZY vs. PSO, FUZZY vs. CMA-ES and FUZZY vs. ABC over the “averaged best fitness values” from Table 3.15.

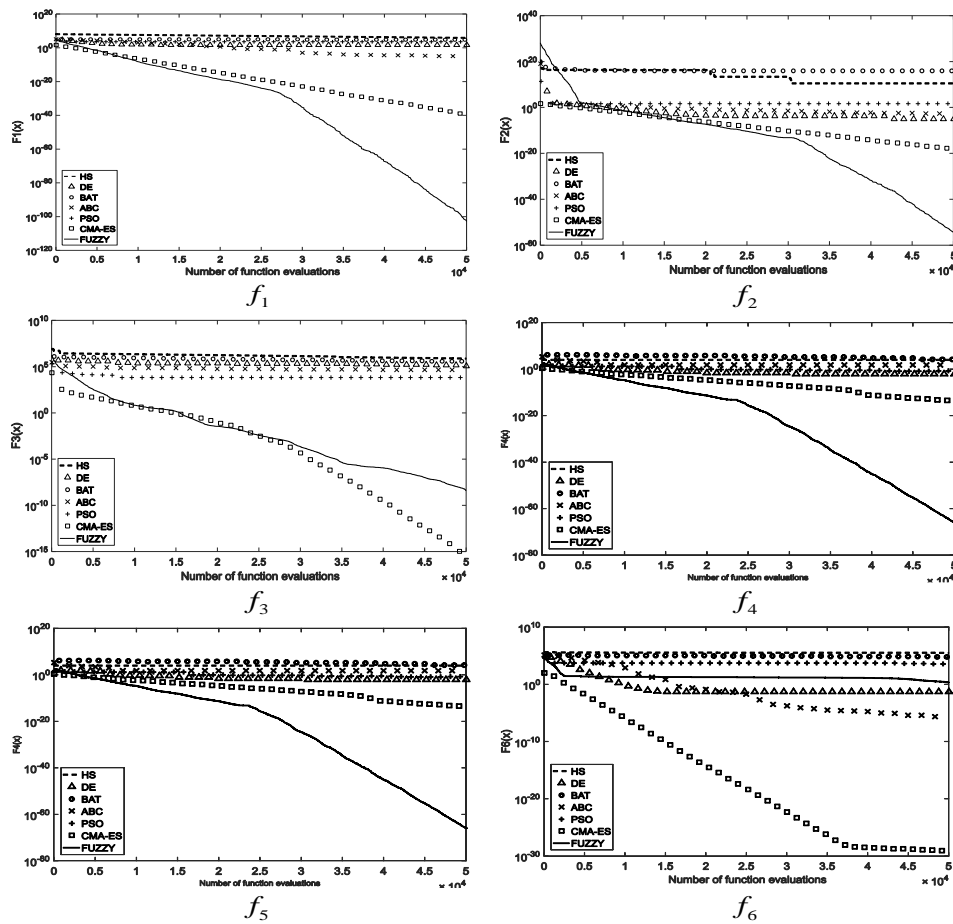


Fig. 3.5. Convergence test results for functions $f_1 - f_6$.

3.5.2.4. Convergence experiments

The comparison of the final fitness value cannot completely describe the searching performance of an optimization algorithm. Therefore, in this section, a convergence test on the seven compared algorithms has been conducted. The purpose of this experiment is to evaluate the velocity with which a compared method reaches the optimum. In the experiment, the performance of each algo-

rithm is considered over all functions ($f_1 - f_{19}$) from Section 3.7, operated in 50 dimensions. In order to build the convergence graphs, we employ the raw simulation data generated in Sections 3.5.2.1, 3.5.2.2 and 3.5.2.3. As each function is executed 30 times for each algorithm, we select the convergence data of the run which represents the median final result. Figures 3.5, 3.6 and 3.7 show the convergence data of the seven compared algorithms. Fig. 3.5 presents the convergence results for functions $f_1 - f_6$, Fig. 3.6 for functions $f_7 - f_{12}$ and Fig. 3.7 for functions $f_{13} - f_{19}$. In the Figures, the x -axis is the elapsed function evaluations, and the y -axis represents the best fitness values found.

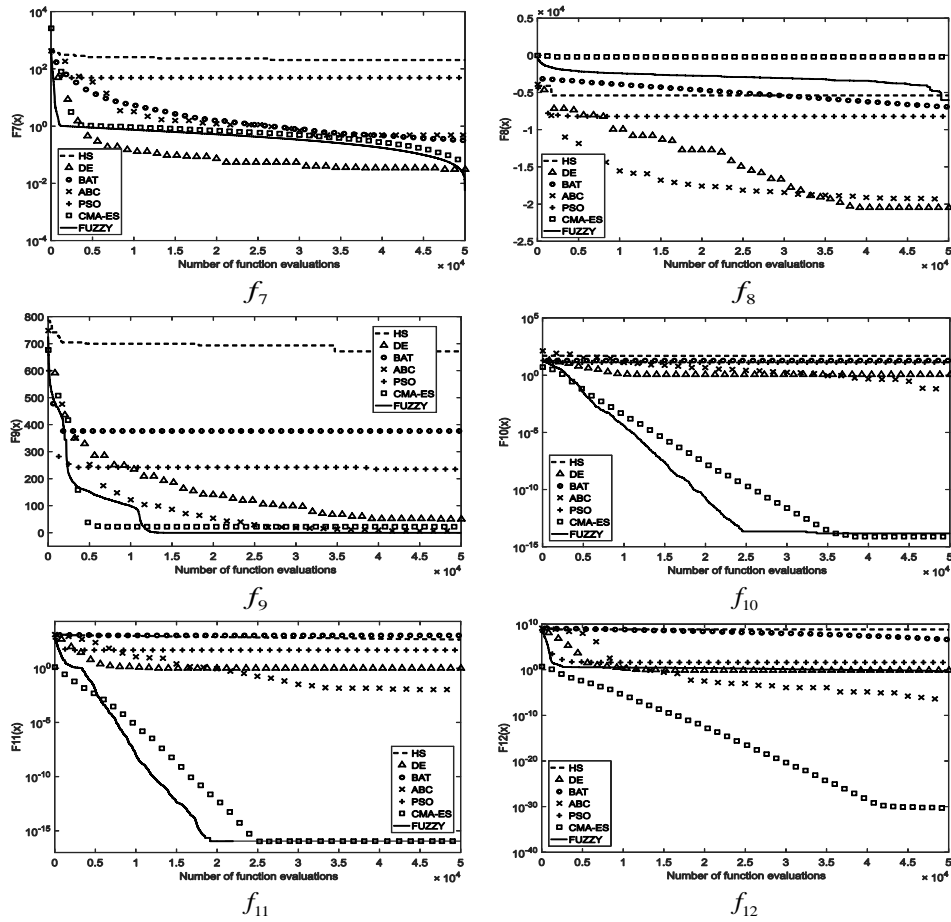


Fig. 3.6. Convergence test results for functions $f_7 - f_{12}$.

From Figure 3.5, it is clear that the proposed FUZZY method presents a better convergence than the other algorithms for functions f_1 , f_2 , f_4 and f_5 . However, for function f_3 and f_6 the CMA-ES reaches faster an optimal value. After an analysis of Figure 3.5, we can say that the proposed Fuzzy method and the CMA-ES algorithm attain the best convergence responses whereas the other techniques maintain slower responses. In Figure 3.6, the convergence graphs show that the proposed fuzzy method obtains the best responses for functions f_9 , f_{10} and f_{11} . In function f_7 , even though the Fuzzy technique finds in a fat way optimal solutions, the DE algorithm presents the best convergence result.

An interesting case is function f_9 , where several optimization methods such as FUZZY, CMA-ES, ABC and DE obtain an acceptable convergence response. In case of function f_8 , the DE and

ABC methods own the best convergence properties. Finally, in function f_{12} , the CMA-ES attains the fastest reaction.

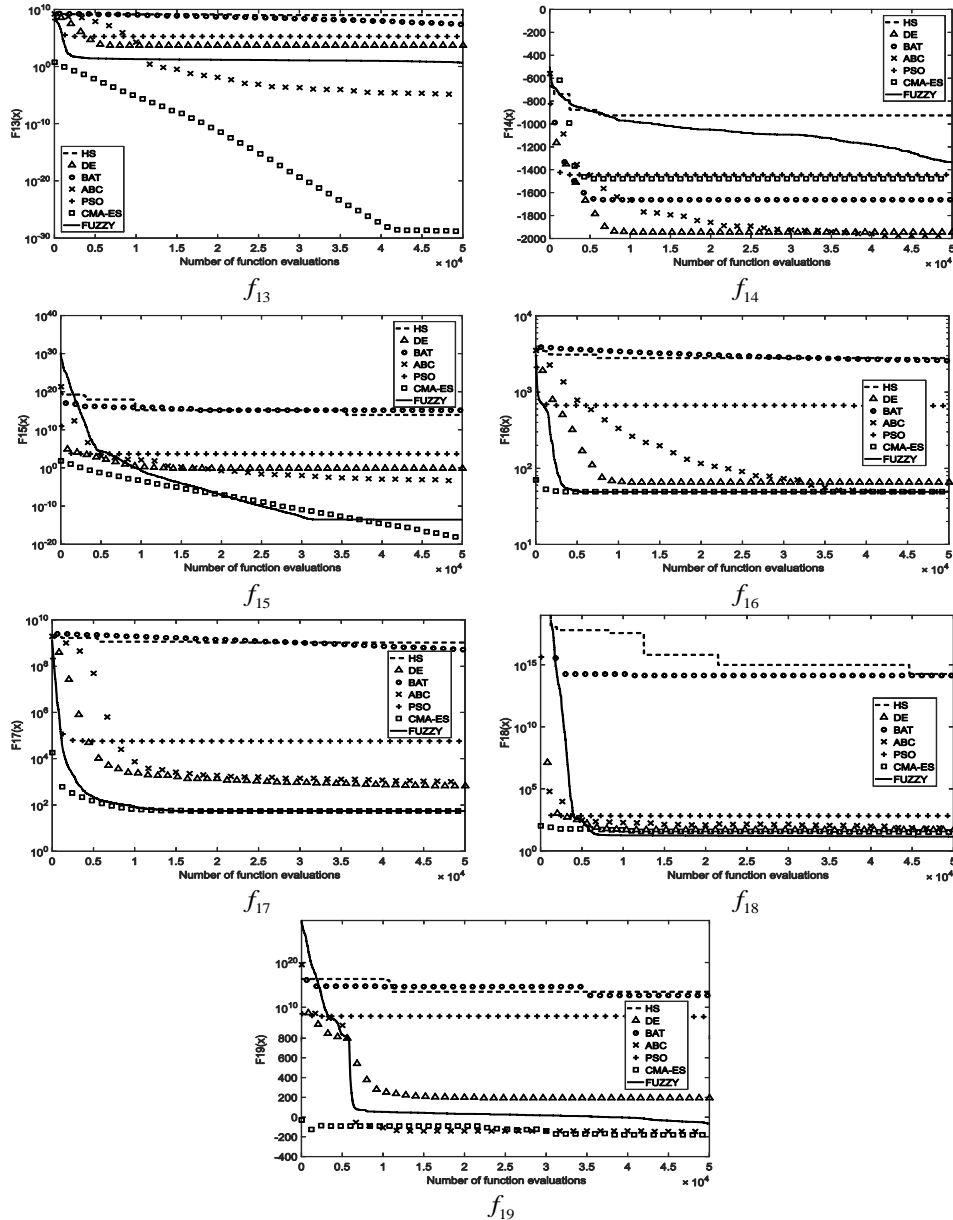


Fig. 3.7. Convergence test results for functions $f_{13} - f_{19}$.

Finally, in Figure 3.7, the convergence responses for functions $f_{13} - f_{19}$ are presented. In function f_{13} of Figure 3.7, the algorithms CMA-ES and ABC obtain the best responses. In case of function f_{14} , DE and ABC find an optimal solution in a prompt way than the other optimization techniques. Although for functions $f_{15} - f_{18}$ the proposed FUZZY algorithm reaches the fastest convergence reaction, the CMA-ES method maintains a similar response. For function f_{19} , the CMA-ES and ABC own the best convergence properties.

Therefore, the convergence speed of the proposed FUZZY method in solving unimodal optimization ($f_1 - f_7$) problems is faster than HS, BAT, DE, PSO, CMA-ES and ABC, except in f_7 , where the CMA-ES reaches the best response. On the other hand, when solving multimodal optimization problems ($f_8 - f_{14}$), the fuzzy algorithm generally converges as fast as or even faster than the compared algorithms. This phenomenon can be clearly observed in Figures 3.6 and 3.7, where the proposed method generates a similar convergence curve to the others, even in the worst case scenario. Finally, after analyzing the performance of all algorithms on hybrid functions ($f_{15} - f_{19}$), it is clear that the convergence response of the proposed approach is not as fast as CMA-ES. In fact, the proposed FUZZY and the CMA-ES algorithms present the best convergence properties when they face the optimization of hybrid functions.

3.5.2.5. Computational complexity

In this section, the computational complexity of all methods is evaluated. Metaheuristic methods are, in general, complex processes with several random operations and stochastic sub-routines. Therefore, it is impractical to conduct a complexity analysis from a deterministic point of view. For that reason, the computational complexity (C) is used in order to evaluate the computational effort of each algorithm. C exhibits the averaged CPU time invested by an algorithm with regard to a common time reference, when it is under operation. In order to assess the computational complexity, the procedure presented in (J. J. Liang et al., 2014) has been conducted. Under this process, C is obtained through the subsequent method:

1. The time reference T_0 is computed. T_0 represents the computing time consumed by the execution of the following standard code:

```

for j=1:1000000
v=0.55+j
v=v+v; v=v/2; v=v*v; v=sqrt(v); v=exp(v); v=v/(v+2);
end
```

2. Evaluate the computing time T_1 for function operation. T_1 Expresses the time in which 200000 runs of function f_9 (multimodal) are executed (only the function without optimization method). In the test, the function f_9 is operated with $n=100$.
3. Calculate the execution time T_2 for the optimization algorithm. T_2 exhibits the elapsed time in which 200000 function evaluations of f_9 are executed (here, optimization method and function are combined).
4. The average time \bar{T}_2 is computed. First, execute the Step 3 five times. Then, extract their average value $\bar{T}_2 = (T_2^1 + T_2^2 + T_2^3 + T_2^4 + T_2^5) / 5$.
5. The computational complexity C is obtained as follows: $C = (\bar{T}_2 - T_1) / T_0$.

Under this process, the computational complexity (C) values of HS, BAT, DE, PSO, CMA-ES, ABC, and FUZZY are obtained. Their values correspond to 77.23, 81.51, 51.20, 36.87, 40.77, 70.17 and 40.91, respectively. A smaller C value indicates that the method is less complex, which allows a faster execution speed under the same evaluation conditions. An analysis of the experiment results shows that although the proposed FUZZY algorithm is slightly more complex than PSO and CMA-ES, their computational complexity (C) values are comparable. Additionally, the three algorithms are significantly less computationally complex than HS, BAT, DE and ABC.

3.6. Conclusions

Recently, several new metaheuristic algorithms have been proposed with interesting results. Most of them use operators based on metaphors of natural or social elements to evolve candidate solutions. Although humans have demonstrated their potential to solve real-life complex optimization problems, the use of human knowledge to build optimization algorithms has been less popular than the natural or social metaphors. In this chapter, a methodology to implement human-knowledge-based optimization strategies has been presented. Under the approach, a conducted search strategy is modeled in the rule base of a Takagi-Sugeno Fuzzy inference system, so that the implemented Fuzzy rules express the conditions under which candidate solutions are evolved during the optimization process.

All the approaches reported in the literature that integrate Fuzzy logic and metaheuristic techniques consider the optimization capabilities of the metaheuristic algorithms for improving the performance of Fuzzy systems. In the presented method, the approach is completely different. Under this new schema, the Fuzzy system directly conducts the search strategy during the optimization process. In this chapter our intent is to propose a methodology for emulating human search strategies in an algorithmic structure. To the best of our knowledge, this is the first time that a Fuzzy system is used as a metaheuristic algorithm.

The presented methodology presents three important characteristics: (1) *Generation*. Under the proposed methodology, Fuzzy Logic provides a simple and well-known method for constructing a search strategy via the use of human knowledge. (2) *Transparency*. It generates fully interpretable models whose content expresses the search strategy as humans can conduct it. (3) *Improvement*. As human experts interact with an optimization process, they obtain a better understanding of successful search strategies capable of finding optimal solutions. As a result, new rules are added so that their inclusion in the existing rule base improves the quality of the original search strategy.

Under the proposed methodology, new rules can be easily incorporated to an already existent system. The addition of such rules allows the capacities of the original system to be extended.

To demonstrate the ability and robustness of our approach, the presented FUZZY algorithm has been experimentally evaluated with a test suite of 19 benchmark functions. To assess the performance of the FUZZY algorithm, it has been compared to other popular optimization approaches based on evolutionary principles currently in use. The results, statistically validated, have confirmed that the presented algorithm outperforms its competitors for most of the test functions in terms of its solution quality and convergence.

3.7. List of benchmark functions

Function	S	Dim	Minimum
$f_1(\mathbf{x}) = \sum_{i=1}^n x_i^2$	$[-100,100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_2(\mathbf{x}) = \sum_{i=1}^n x_i + \prod_{i=1}^n x_i $	$[-10,10]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_3(\mathbf{x}) = \sum_{i=1}^n \left(\sum_{j=1}^i x_j \right)^2$	$[-100,100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_4(\mathbf{x}) = \max_i \{ x_i , 1 \leq i \leq n\}$	$[-100,100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_5(\mathbf{x}) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30,30]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (1, \dots, 1);$ $f(\mathbf{x}^*) = 0$
$f_6(\mathbf{x}) = \sum_{i=1}^n (\lfloor x_i + 0.5 \rfloor)^2$	$[-100,100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_7(\mathbf{x}) = \sum_{i=1}^n i x_i^4 + \text{random}(0,1)$	$[-1.28,1.28]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$

Table 3.17. Unimodal test functions used in the experimental study.

Function	S	Dim	Minimum
$f_8(\mathbf{x}) = \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$	$[-500,500]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (420, \dots, 420);$ $f(\mathbf{x}^*) = -418.9829 \times n$
$f_9(\mathbf{x}) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_{10}(\mathbf{x}) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}\right) - \exp\left(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i)\right) + 20 + \exp$	$[-32,32]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_{11}(\mathbf{x}) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600,600]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_{12}(\mathbf{x}) = \frac{\pi}{n} \left\{ 10 \sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\} + \sum_{i=1}^n u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{(x_i + 1)}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \leq x_i \leq a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50,50]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_{13}(\mathbf{x}) = 0.1 \left\{ \begin{aligned} & \sin^2(3\pi x_1) \\ & + \sum_{i=1}^n (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] \\ & + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \end{aligned} \right\} + \sum_{i=1}^n u(x_i, 5, 100, 4);$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-10,10]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (1, \dots, 1);$ $f(\mathbf{x}^*) = 0$

$f_{14}(x) = \frac{1}{2} \sum_{i=1}^n (x_i^4 - 16x_i^2 + 5x_i)$	$[-5, 5]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (-2.90, \dots, -2.90);$ $f(\mathbf{x}^*) = -39.16599 \times n$
---	-------------	-----------------------	---

Table 3.18. Multimodal test functions used in the experimental study.

Function	S	Dim	Minimum
$f_{15}(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x}) + f_9(\mathbf{x})$	$[-100, 100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = 0$
$f_{16}(\mathbf{x}) = f_9(\mathbf{x}) + f_5(\mathbf{x}) + f_{11}(\mathbf{x})$	$[-100, 100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = n - 1$
$f_{17}(\mathbf{x}) = f_3(\mathbf{x}) + f_5(\mathbf{x}) + f_{10}(\mathbf{x}) + f_{13}(\mathbf{x})$	$[-100, 100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = (1.1 \times n) - 1$
$f_{18}(\mathbf{x}) = f_2(\mathbf{x}) + f_5(\mathbf{x}) + f_9(\mathbf{x}) + f_{10}(\mathbf{x}) + f_{11}(\mathbf{x})$	$[-100, 100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (0, \dots, 0);$ $f(\mathbf{x}^*) = n - 1$
$f_{19}(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x}) + f_8(\mathbf{x}) + f_{10}(\mathbf{x}) + f_{12}(\mathbf{x})$	$[-100, 100]^n$	$n = 50$ $n = 100$	$\mathbf{x}^* = (1, \dots, 1);$ $f(\mathbf{x}^*) = 0$

Table 3.19. Hybrid test functions used in the experimental study.

Chapter 4

Color Segmentation using LVQ Neural Networks

Color segmentation in digital images is a challenging task due to image capture conditions. Typical segmentation algorithms present several difficulties in this process because they do not tolerate variations in color hue corresponding to the same object. In this chapter is presented an approach for color segmentation based on Learning Vector Quantization (LVQ) networks, which conducts the segmentation process by means a color-based pixel classification. In the LVQ networks, neighboring neurons have the capability to learn how to recognize close sections of the input space. The presented segmentation approach classifies the pixels directly by means of the LVQ network. The experimental results over a set of images show the efficiency of the LVQ-based method to satisfactorily segment color despite remarkable illumination problems.

4.1. Introduction

The color discrimination plays an important role in humans for individual object identification. Humans usually do not search in a bookcase for a previously known book solely by its title. We try to remember the color on the cover (e.g., blue) and then search among all the books with a blue cover for the one with the correct title. The same applies to recognizing an automobile in a parking site. In general, humans do not search for model A of company B, but rather we look for a red car. It is only when a red vehicle is spotted, when it is decided according to its geometry, whether that vehicle is the one of the required kinds.

Image segmentation is the first step in image analysis and pattern recognition. It is a critical and essential component but also it is one of the most difficult tasks in image processing. The actual operation of the algorithm determines the quality of the overall image analysis.

Color image segmentation is a process of extracting from the image domain one or more connected regions satisfying the uniformity (homogeneity) criterion (Egmont-Petersen, et al., 2002) which is derived from spectral components (Cheng, et al., 2001; Gonzalez & Woods, 2008). These components are defined within a given color space model such as the RGB model -the most common model, which considers that a color point is defined by the color component levels of the corresponding pixel, i.e. red (R), green (G), and blue (B). Other color spaces can also be employed considering that the performance of an image segmentation procedure is known to depend on the choice of the color space. Many authors have sought to determine the best color space for their specific color image segmentation problems. Unfortunately, there is not an ideal color space to provide satisfying results for the segmentation of all kinds of images.

Image segmentation has been the subject of considerable research activity over the last two decades. Many algorithms have been elaborated for gray scale images. However, the problem of segmentation for color images that implies a lot of information about objects in scenes has received much less attention of the scientific community. Although color information allows a more complete representation of images and more reliable segmentations, processing color images requires computational times considerably larger than those needed for gray-level images as it is very sensitive to illumination changes.

This chapter considers the color image segmentation as a pixel classification problem. By means of the LVQ neural networks and their classification schemes, classes of pixels are detected by analyzing the similarities between the colors of the pixels.

In particular, color image segmentation techniques described in the literature can be categorized into four main approaches: Histogram thresholding and color space clustering; region-based approaches, edge detection, probabilistic methods and soft-computing techniques. The following section discusses on each technique, summarizing their main features.

4.1.1. Histogram thresholding and color space clustering

Histogram thresholding is one of the widely used techniques for monochrome image segmentation. It assumes that images are composed of regions with different gray levels. The histogram of an image can be separated into a number of peaks (modes), each corresponding to one region, and there exists a threshold value corresponding to valley between the two adjacent peaks. As for color images, the situation is different from monochrome images because of multi-features. Multiple histogram-based thresholding divides the color space by thresholding each component histogram.

The classes for color segmentation are built by means of a cluster identification scheme which is performed either by an analysis of the color histogram (Park, et al., 1998) or by a cluster analysis procedure (T. Q. Chen & Lu, 2002). When the classes are constructed, the pixels are assigned to one of them by means of a decision rule and then mapped back to the original image plane to produce the segmentation. The regions of the segmented image are composed of connected pixels which are assigned to the same classes. When the distribution of color points is analyzed in the color space, the procedures generally lead to a noisy segmentation with small regions scattered through the image. Usually, a spatial-based post-processing is performed to reconstruct the actual regions in the image (Nikolaev & Nikolayev, 2004).

4.1.2. Edge detection

Region based approaches, including region growing, region splitting (Ohlander, et al., 1978), region merging (Cheng, et al., 2002) and their combination (Tremeau & Borel, 1997), attempt to group pixels into homogeneous regions. In the region growing approach, a seed region is first selected. Thus, it is expanded to include all homogeneous neighbors, repeating the process until all pixels in the image are classified. One problem with region growing is its inherent dependence on the selection of seed region and the order in which pixels and regions are examined. In the region splitting approach, the initial seed region is simply the whole image. If the seed region is not homogeneous, it is usually divided into four squared sub-regions, which become new seed regions. This process is repeated until all sub-regions are homogeneous. The major drawback of region splitting is that the resulting image tends to mimic the data structure used to represent the image and comes out too square. The region merging approach is often combined with region growing or region splitting to merge similar regions for making a homogeneous section as large as possible.

4.1.3. Probabilistic methods

Probabilistic color segmentation estimates the probability $P_i(x, y) \in [0, 1]$ for a given pixel $I(x, y)$ of belonging to a region i in the image I . Although the probability density $P_i(x, y)$ is usually determined, its parameters are often unknown. In (Jepson, et al., 2003; McKenna, et al., 1999; Raja, et al., 1998) have already discussed color segmentation when the joint distribution of color is modeled by a mixture of Gaussians within a 3-dimensional space. Since no spatial coordinates are incorporated, once the model has been inferred, it needs a spatial grouping step which applies a maximum-vote filter and uses the connected component algorithm.

Isard & MacCormick,(2001) have employed color information to implement particle filtering. Lately, Pérez, et al., (2002) introduced an approach that also uses color histograms and particle filtering for multiple object tracking. Both methods differ in the initialization procedure for the tracker, the model updating, the region shape and the observation of the tracking performance. Bradski (1998) modified the mean-shift algorithm (Camshift) which operates on probability distributions to track colored objects in video frame sequences.

4.1.4. *Soft-Computing techniques*

A trendy issue is the use of soft-computing approaches for image processing systems. Artificial neural network models have been proposed to segment images directly from pixel similarity or discontinuity. More than 200 neural networks used in image processing are presented by Egmont-Petersen et al. (2002) by means of an 2D taxonomy. In (Cheng et al., 2001) it is also discussed on many color image Segmentation techniques, including the histogram Thresholding, characteristic feature Clustering, Edge Detection, Region-based methods, Fuzzy methods, and Neural Networks.

Color segmentation is successfully computed by Self-organizing Maps (SOMs) and competitive networks in (Dong & Xie, 2005; Ong, et al., 2002; Yeo, et al., 2005). In (Ong et al., 2002) a two-stage strategy includes a fixed-size two dimensional feature map (SOM) to capture the dominant colors of an image by unsupervised training. In a second stage, the algorithm combines a variable-sized one-dimensional feature map and color merging to control the number of color clusters that are used for segmentation. The model in (G. Dong & Xie, 2005) is based on a two-step neural network. In the first step, a SOM performs color reduction and then a simulated annealing step searches for the optimal clusters from SOM prototypes. The task involves a procedure of hierarchical prototype learning (HPL) to generate different sizes of color prototypes from the sampled object colors.

4.1.5. *Scheme*

Learning Vector Quantization (LVQ) networks learn to recognize groups of similar input vectors in such a way neurons that locate nearby to others in the neuron layer respond to similar input vectors. The learning is supervised and the inputs vectors into target classes are chosen by the user.

The LVQ algorithm presented in this chapter works only with image pixels, with no dynamic model or probability distribution, which in turn, improves the processing speed and facilitates the implementation process.

The approach naturally avoids the complex structures commonly resulting from other neural methods such as those in (Dong & Xie, 2005; Ong, et al., 2002; Yeo, et al., 2005). It incorporates a decision function which eases the segmentation of the objective color. The method has been applied on several color segmentation problems (face localization and color tracking), showing enough capacity to comprehensively segment color even under illumination differences.

This chapter is organized as follows: Section 4.2 revisits some background concepts while Section 4.3 presents an introductory study of competitive neural networks and their main features. Section 4.4 explains relevant details of LVQ networks and Section 4.5 shows the architecture and characteristics of the presented color-segmentation system, including some practical discussions. Section 4.6 offers a simple explanation on the algorithm's implementation. Section 4.7 reports the results and some discussions, finally in Section 4.8 are presented some conclusions.

4.2. Background Issues

4.2.1. RGB Space Color

Color is perceived by humans as a combination of triple stimuli R (red), G (green), and B (blue) which are usually named as primary colors. From R, G, B representation, it is possible to derive other kinds of color representations (spaces) by using either linear or nonlinear transformations. The RGB color space can be geometrically represented within a 3-dimensional cube as shown in Figure 4.1. The coordinates of each point inside the cube represent the values of red, green and blue components, respectively.

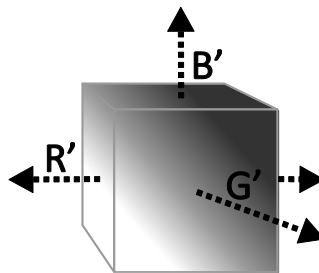


Fig 4.1. RGB Space Color

The laws of color theory are: (1) any color can be created by these three colors and the combination of the three colors is unique; (2) if two colors are equivalent, they will be again equivalent after multiplying or dividing the three components by the same number; (3) the luminance of a mixture of colors is equal to the sum of the luminance of each color. The triple stimuli values that served as the color basis are: 425.8 nm for blue, 546.1 nm for green and 700.0 nm for red. Any color can be expressed by these three color bases.

RGB is the most widely accepted model for television systems and pictures acquired by digital cameras. Video monitors commonly display color images by modulating the intensity of the three primary colors (red, green, and blue) at each pixel of the image (Comaniciu & Meer, 1997). RGB is suitable for color display as it is complicated for color segmentation's purposes, considering the high correlation among the R , G , and B components (Pietikainen, et al., 1996). High correlation refers to the intensity changes which assume that all the three components will change accordingly. The measurement of a color in RGB space does not represent color differences in a uniform scale and hence it is impossible to evaluate the similarity of two colors from their distance in RGB space.

4.2.2. Artificial Neural Networks

Artificial Neural Networks are composed from simple elements that commonly mimic biological systems following parallel arrangements. By nature, a network function is determined by the connections between such neural elements. It is possible to train a neural network to "learn" a given function by adjusting the values of the connections (\mathbf{W} weights) between elements.

A common training algorithm seeks to match a given neural input to a specific target output as shown in Figure 4.2. The network is adjusted by comparing the network's output and the target value, until the network output matches, as close as possible, the target. Typically, a great number of input/target pairs are used following the *supervised learning* scheme to train the network.

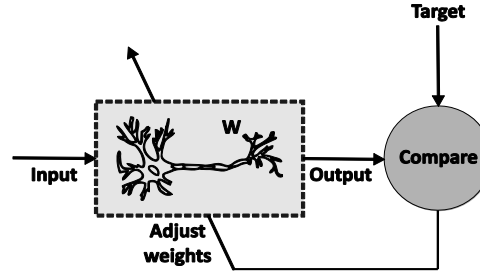


Fig. 4.2. Supervised learning in Neural Networks

Batch training of a network proceeds by making weight and bias changes based on an entire set (batch) of input vectors. Incremental training changes are applied to the weights and biases of a network after the presentation of each individual input vector. Incremental training is sometimes referred as on-line or adaptive training.

Neural networks may be employed to solve several sorts of problems, ranging from pattern recognition, identification, classification, speech, control systems and computational vision. The supervised training methods are widely known in the school. Other kind of networks can be obtained from *unsupervised training* techniques or from direct design methods. Unsupervised networks can be applied, for instance, to identify groups of data.

4.3. Competitive Networks

Competitive Networks (Kohonen, 1989) learn to classify input vectors according to how they are grouped in the input space. They differ from other networks in that neighboring neurons learn to recognize neighboring sections of the input space. Thus, competitive layers learn both the distributions and topology of the input vectors in which they are trained on. The neurons in the layer of a competitive network are arranged originally in physical positions according to a topology pattern such as grid, hexagonal, or random topology.

The architecture of a competitive network is shown in Figure 4.3. The $|Ndist|$ box in the figure receive the input vector \mathbf{p} and the input weight matrix \mathbf{IW} and produces a vector or a matrix \mathbf{S} according to the topological configuration. The elements are the negative distances between the input vector \mathbf{p} and the matrix \mathbf{IW} . The net value v of the competitive layer is computed by finding the negative distance between input vector \mathbf{p} and the weight matrix \mathbf{IW} and then adding the biases \mathbf{b} . If, all biases are zero, the maximum net input that a neuron can have is 0. This occurs when the input vector \mathbf{p} equals the neuron's weight vector contained in the matrix \mathbf{IW} .

The competitive transfer function C receive a net value v and returns outputs of 0 for all neurons except for the *winner*, the neuron associated with the most positive element of input v . Thus, the winner's output is 1. The weights of the winning neuron are adjusted by the Kohonen learning rule. Supposing that the i^{th} neuron wins, the elements of the i^{th} row of the input weight matrix and all neurons within a certain neighborhood radius $Ni(d)$ of the winning neuron are adjusted as shown in Eq. (4.1). In other words, $Ni(d)$ is the neighbor's number around of the winner neuron to be affected.

$${}_i\mathbf{IW}^{l+1}(q) = {}_i\mathbf{IW}^{l+1}(q-1) + \alpha(\mathbf{p}(q) - {}_i\mathbf{IW}^{l+1}(q-1)) \quad (4.1)$$

Here α is the learning rate and $Ni(d)$ contains the index for all of the neurons that lie within a radius d of the i^{th} winning neuron. Thus, when a vector \mathbf{p} is presented, the weights of the winning neuron and its closest neighbors move toward \mathbf{p} . Consequently, after many presentations, neighboring

neurons will have learned vectors similar to each others. The winning neuron's weights are altered accordingly by the learning rate. The weights of neurons in its neighborhood are altered proportional to half of the learning rate. In this work, the learning rate and the neighborhood distance (used to determine which neurons are in the winning neuron's neighborhood) are not altered during training.

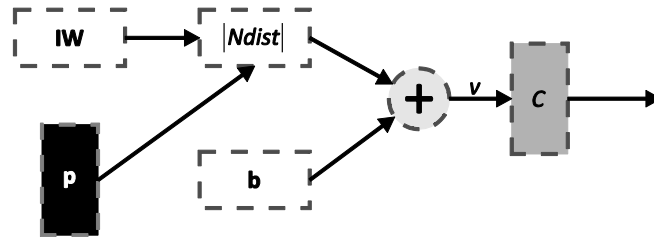


Fig. 4.3. Architecture of a competitive network

To illustrate the concept of neighborhoods, consider the Figure 4.4. Left, it is shown a two-dimensional neighborhood of radius $d = 1$ around neuron 13. Aside it is shown a neighborhood of radius $d = 2$.

These neighborhoods could be written as:

$$N_{13}(1) = (8, 12, 13, 14, 18), N_{13}(2) = (3, 7, 8, 9, 11, 12, 13, 14, 15, 17, 18, 19, 23)$$

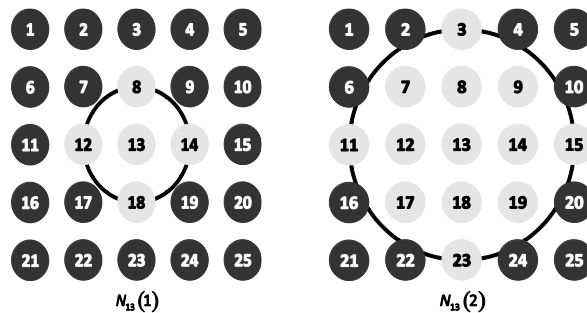


Fig. 4.4. Left, two-dimensional neighborhood with radius $d = 1$. Right, neighborhood with radius $d = 2$.

4.4. Learning Vectors Quantization Vectors

An LVQ Network (Kohonen, 1989) has first, a competitive layer and second, a linear layer. The competitive layer learns to classify input vectors like the networks of the last section. The linear layer transforms the competitive layer's classes into target classifications defined by the user. We refer to the classes learned by the competitive layer as *subclasses* and the classes of the linear layer as *target classes*. Both the competitive and linear layers have one neuron per class. However, the neurons in the competitive layer can be arranged according to a topology pattern.

Thus, the competitive layer can learn **S1** classes, according to how they are grouped in the topological space. These, in turn, are combined by the linear layer to form **S2** target classes. This process can be considered as a lineal transformation carried out on the learned classes **S1** (in unsupervised manner) by the competitive layer to a mapping on **S2** defined by **LW**. This

transformation allows distributing similar patterns around the target neuron in the linear layer. The LVQ Network architecture is shown in Fig. 4.5.

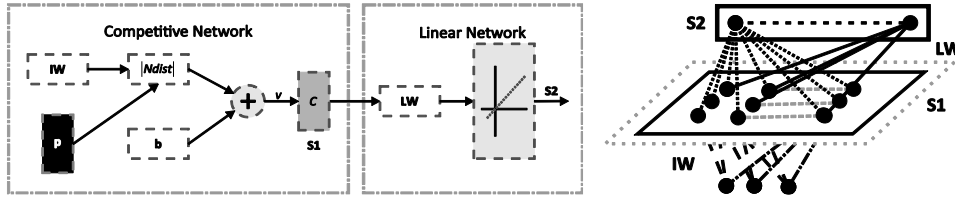


Figure 4.5. Schematic representations of the LVQ net.

4.5. Architecture of the color segmentation system

The core of the proposed algorithm is an LVQ network whose inputs are connected directly to each RGB pixel component of the image \mathbf{I} . The output of the LVQ network is a vector $\mathbf{S2}$ connected to the decision function \mathbf{f}_d . If the RGB components of the original pixel represents the color to be segmented, then the \mathbf{f}_d function output is 1, if not is 0. The result is a new image \mathbf{I}' . The segmentator takes advantage of the LVQ property to learn to recognize neighboring sections of the input space. Figure 4.6 shows the segmentator's architecture.

Considering that the LVQ net is configured with a grid of 6×5 neurons in the competitive layer and 30 one-dimensional output neurons (linear layer), then would be possible to train the competitive network to learn the color-pixel space and its topology (described as the vector \mathbf{P} with elements \mathbf{p}_R , \mathbf{p}_G and \mathbf{p}_B coming from the image). The 6×5 grid in the competitive layer was chosen after considering a tradeoff between the neuron distribution and the computational cost [16]. The size of the linear layer (30) is considered only as being coherent to the neurons contained on the grid (6×5).

The net training is achieved in two phases: Ordering phase and tuning phase. In the ordering phase the neurons of competitive layer learn to recognize groups of similar color vectors in an *unsupervised* manner. Using *supervised* learning, they learn the tuning phase for the linear layer. It was for supported, we suppose that the image \mathbf{I} contains an object \mathbf{O} with the color to be segmented, being \mathbf{p}_o a RGB pixel corresponding to the object, we train the linear network in such a way, that the class of this pixel is assigned in the middle of the linear layer (15). Using the neuron 15th as objective helps to have symmetry in the class distribution. This defines a pattern similarity, depending on the presented neighborhood with regard to this class.

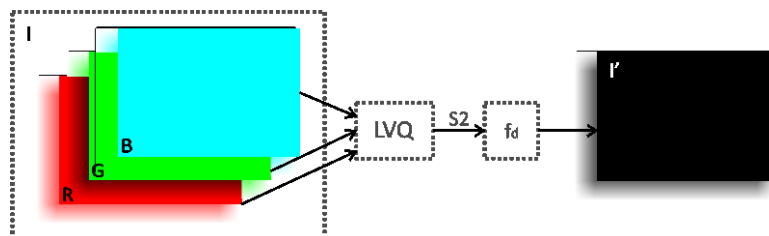


Figure 4.6. Architecture of the color segmentation system

The idea is that the winning neuron activated in the competitive layer (as consequence of have been excited by the RGB color combination to be segmented) will be located halfway of the linear

layer as consequence of the **LW** mapping. This intrinsic LVQ property allows to locate similar colors in the neighboring neurons. Thus, if exists a color vector \mathbf{p}_1 that correspond, in fact, to the same object, however due to the illumination conditions and noise, it is a little different to the target pattern, this color could not be classified by the neuron 15 but for some close to it.

The classification performed by the LVQ network finds a vector of elements categorized by \mathbf{S}^2 of 30 elements corresponding to 30 classes. Each element of \mathbf{S}^2 vector could have two possible values, 1 or 0 and only an element from each vector could be 1, while the other elements will be 0. Then for the color to be segmented, the activation of the neurons is concentrated in the middle of the vector, Thus, neurons nearest to the 15 will have a bigger possibility to be activated, for similar color patterns.

Considering the problematic above described, is necessary to describe a function \mathbf{f}_d who defines the neuron's density which will be taken to consider if a pixel corresponds or not to the color to be segmented, this function will be called in this work "decision function". Is possible to formulate many functions which could solve the decision problem satisfactorily. In this work the Gaussian function has been chosen to resolve the decision problem, although it is possible to use other, including non-symmetrical distributions functions. Figure 4.7 shows graphically the Gaussian function and its relationship with the output layer. Eq. (4.2) shows mathematically this function where \mathbf{g} is the index (1,...,30) of the activated neuron, N is the neuron number of the linear layer (for this paper, $N = 30$) and σ is the standard deviation. Therefore, \mathbf{f}_d has only a calibration parameter represented by σ which determines the generalization capacity of the complete system. Thus, for example, if the value of σ is chosen small enough, the segmentation capacity will be more selective than in the case of a bigger σ .

$$\mathbf{f}_d(\mathbf{g}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mathbf{g} - (N/2))^2}{2\sigma^2}\right) \quad (4.2)$$

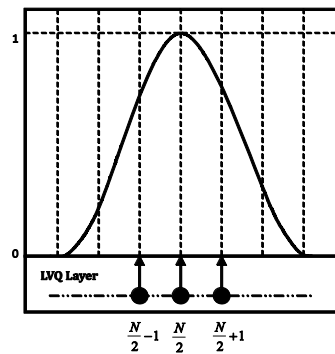


Figure 4.7. Decision function model

4.6. Implementation

The implementation is divided in two parts, the net training and the segmentation application. First, the training process requires an image frame containing the object whose color will be segmented. Then, a pixel's block is selected to train the LVQ net according to the color to be segmented but specifying that this pixel must be located at the 15th neuron that means, at the middle of the 30 neurons array. Using this selection, the patterns are similarly distributed around the 15th neuron.

The weight matrix \mathbf{IW} and \mathbf{LW} are randomly initialized for the training. During the unsupervised training the learning rate $\alpha=0.1$ was used as well as a distance radius $Ni(d)=3$. This conditions assure that winner neurons weights will be affected in a reason of 0.1. While its three neighboring neurons weights (in both ways) be affected in a reason of 0.05. During the training of the lineal layer the values of \mathbf{LW} are calculated. These values transform the classes lineally allowing distributing them around the neuron 15. Initially due to the aleatory configuration, the learned classes by the competitive layer cannot be correctly mapped to the output vector, however after some iteration; these values are classified correctly as consequence of the adaptation of \mathbf{LW} . The Fig. 4.8 shows the classification results of the neurons in the competitive layer arranged in a 5 x 2 grid on an image obtained of Webcam.

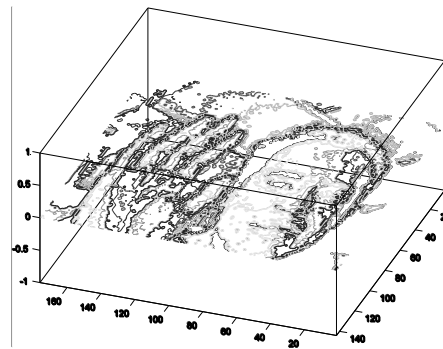


Figure 4.8. Classification performed by a LVQ network with 10 neurons in the competitive layer.

For the segmentator use, a decision function \mathbf{f}_d with parameters $\mu=15$ and $\sigma=3$ was integrated to the previously trained network. The complete system was programmed using Visual C ++ and tested on a PCx86 at 900 MHz with 128 MBytes RAM.

4.7. Results and discussion

In order to test the robustness of the segmentation algorithm, the approach is applied to video-streamed images. The patch of color to be segmented may exhibit illumination changes within the video sequence; however, the algorithm's operation prevails despite such changes.

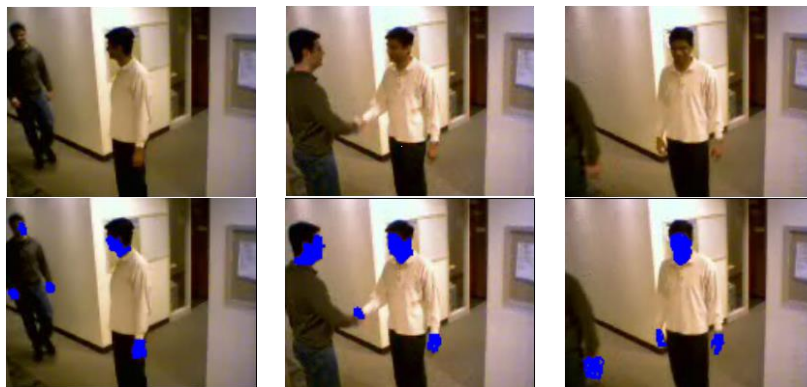


Figure 4.9. Sequence performed by the LVQ segmentator in indoor environment.

The overall performance is tested considering two different experiments. First, the system performance is tested over an indoor footage as shown by Figure 4.9. In order to train the LVQ Network, one video-frame is considered to choose some pixels that belong to the skin of a couple of human individuals who are participating in a handshaking gesture. Once the system is fully trained, a full-long video sequence is tested. It is important to consider all changes in the skin color that are related to variable illumination. The effect is generated as a result of changes in the relative position and orientation of the object with respect to the light source. Despite such changes and the closeness of other similar colors in the scene, the algorithms have been able to acceptably segment the skin color as shown by Figure 4.9.

The second experiment tests the algorithm's sensitivity in response to abrupt changes on illumination. An external setting is used to generate the video sequence which naturally includes changes in sun lighting and therefore shading too. Figure 4.10 shows the resulting sequence once the algorithm is segmenting a red spot representing one backpack. Although the shape of the object of interest exhibits several irregular holes generated from lighting variations, the algorithm is still capable to segment the patch yet if it is located under the shade or passing by a transition zone or under full illumination from the sun.

The overall performance of the algorithm can be successfully compared to other algorithms such as those in (Jang & Kweon, 2001; Nummiaro, et al., 2002). However, the LVQ algorithm works directly on the image pixels with no dynamical model or probability distribution, improving the execution time and simplifying the implementation. However, contrary to other approaches (Salmond, 1990), the average execution time required by the algorithm to classify one pixel is always constant depending solely on the color complexity.

It is remarkable the influence of the σ parameter over the decision function itself. Best results were achieved with σ falling between 3 and 5. It is easy to envision that improving the robustness of the system may evolve considering an adaptable σ parameter. As an example, Figures 4.11(a)-(b) show several cases when using different values for σ .



Figure 4.10. Image sequence as processed by the LVQ segmentator while segmenting for the backpack within an outdoor environment.



Figure 4.11. Results obtained using (a) $\sigma = 4$ and (b) $\sigma = 1$.

The proposed algorithm also exhibits better generalization properties compared to other classical lookup table algorithms. Particularly, if it considers images with changing illumination.

The robustness of the LVQ algorithm facing variable lighting can be compared to the popular Cam-Shift algorithm. Aiming for a fair comparison, three indexes are considered just as it is proposed in (Salmond, 1990). The first performance index is related to “tracking failure”. Both algorithms run until a failure in the tracking process is registered. According to this measurement, a track is assumed to be lost either when the center of the segmented object is sufficiently far from the true center, i.e. when the object position determined by the algorithm does not match to the real position or when the computed center falls outside the image location in five consecutive steps. Finally, the “track-lifetime” is defined according to the number of time steps until the tracking was lost. This index is commonly averaged across all trials. Fig. 4.12 shows the results from these experiments. Both algorithms were tested considering 100 lux as an initial point –a standard office’s illumination. At this point, the averaged track-lifetime is infinite for both algorithms. Measurements are taken in both directions, showing that the LVQ algorithm has a higher robustness, particularly in case of low intensity lighting.

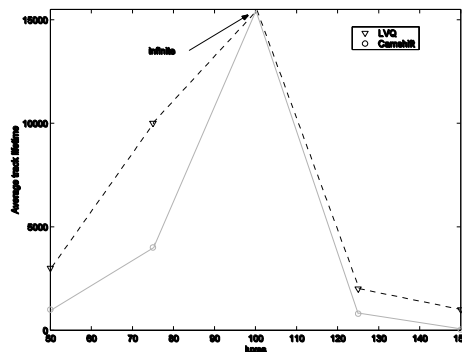


Figure 4.12. Comparison of LVQ and Cam-Shift algorithms.

4.8. Conclusions

This chapter presents an image segmentator approach based on LVQ networks which considers the segmentation process as a pixel classification which is fully based on color. The segmentator operates directly upon the image pixels using the classification properties of the LVQ networks. The algorithm is effectively applied to the segmentation of sampled images showing its capacity to satisfactorily segment color despite remarkable illumination differences, considering indoor and outdoor scenes. The results demonstrate the operation of the LVQ algorithm which in turn is capa-

ble of organizing topologically the input space, accomplishing the segmentation process, despite a small number of neurons.

The presented system has two important features. First, since the LVQ algorithm works directly on the image pixels with no dynamic model or probability distribution, the execution time is faster than other approaches. Second, the algorithm exhibits interesting generalization properties, in particular considering images with changing illumination.

Further increases in the segmentator performance might be reached, if the parameter σ is also adapted using some kind of optimization technique.

Chapter 5

Motion estimation algorithm using Block-matching and Harmony Search Optimization

Motion estimation is one of the major problems in developing video coding applications. On the other hand, Block-matching (BM) algorithms are the most popular methods due to their effectiveness and simplicity for both software and hardware implementations. A BM approach assumes that the movement of pixels within a defined region of the current frame can be modeled as a translation of pixels contained in the previous frame. During this procedure is obtained a motion vector by minimizing a certain matching metric that is produced between the current frame and the previous frame. However, the evaluation of such matching measurement is computationally expensive and represents the most consuming operation in the BM process. Therefore, BM motion estimation can be viewed as an optimization problem whose goal is to find the best-matching block within a search space. Harmony Search (HS) algorithm is a metaheuristic optimization method inspired by the music improvisation process, in which a musician polishes the pitches to obtain a better state of harmony. In this chapter, a BM algorithm that combines HS with a fitness approximation model is presented. The approach uses motion vectors belonging to the search window as potential solutions. A fitness function evaluates the matching quality of each motion vector candidate. In order to minimize computational time, the approach incorporates a fitness calculation strategy to decide which motion vectors can be only estimated or actually evaluated. Guided by the values of such a fitness calculation strategy, the set of motion vectors is evolved through HS operators until the best possible motion vector is identified. The presented method has been compared to other BM algorithms in terms of velocity and coding quality and its experimental results demonstrate that the algorithm exhibits the best balance between coding efficiency and computational complexity.

5.1. Introduction

Motion estimation plays important roles in a number of applications such as automobile navigation, video coding, surveillance cameras and so forth. The measurement of the motion vector is a fundamental problem in image processing and computer vision, which has been faced using several approaches (Bohlooli & Jamshidi, 2012; Cirrincione & Cirrincione, 2003; Kang, et al., 2012; Risinger & Kaikhah, 2008). The goal is to compute an approximation to the 2-D motion field – a projection of the 3-D velocities of surface points onto the imaging surface.

Video coding is currently utilized in a vast amount of applications ranging from fixed and mobile telephony, real-time video conferencing, DVD and high-definition digital television. Motion Estimation (ME) is an important part of any video coding system, since it can achieve significant compression by exploiting the temporal redundancy existing in a video sequence. Several ME methods have been studied aiming for a complexity reduction at video coding, such as block matching (BM) algorithms, parametric-based models (Tzovaras, et al., 1999), optical flow (Barron, et al., 1994) and recursive techniques (Skowronski, 1999). Among such methods, BM seems to be the most popular technique due to its effectiveness and simplicity for both software and hardware implementations (Huang, et al., 2006). Furthermore, in order to reduce the computational complexity in ME, many BM algorithms have been proposed and used in implementations

of various video compression standards such as MPEG-4 (*VISUAL ISO/IEC 14496-2 Committee Draft*, 1998) and H.264 (Joint Video Team (JVT), 2002).

In BM algorithms, the video frames are partitioned in non-overlapping blocks of pixels. Each block is predicted from a block of equal size in the previous frame. Specifically, for each block in the current frame, we search for a best matching block within a searching window in the previous frame that minimizes a certain matching metric. The most used matching measure is the Sum of Absolute Differences (SAD) which is computationally expensive and represents the most consuming operation in the BM process. The best matching block found represents the predicted block, whose displacement from the previous block is represented by a transitional motion vector (MV). Therefore, BM is essentially an optimization problem, with the goal of finding the best matching block within a search space.

The full search algorithm (FSA) (Jain & Jain, 1981) is the simplest block-matching algorithm that can deliver the optimal estimation solution regarding a minimal matching error as it checks all candidates one at a time. However, such exhaustive search and full-matching error calculation at each checking point yields an extremely computational expensive BM method that seriously constraints real-time video applications.

In order to decrease the computational complexity of the BM process, several BM algorithms have been proposed considering the following three techniques:

(1) using a fixed pattern: which means that the search operation is conducted over a fixed subset of the total search window. The Three Step Search (TSS) (Jong, et al., 1994), the New Three Step Search (NTSS) (Li, et al., 1994), the Simple and Efficient TSS (SES) (Lu & Liou, 1997), the Four Step Search (4SS) (Po & Ma, 1996) and the Diamond Search (DS) (Zhu & Ma, 2000) are some of its well-known examples. Although such approaches have been algorithmically considered as the fastest, they are not able eventually to match the dynamic motion-content, delivering false motion vectors (image distortions).

(2) Reducing the search points: in this method, the algorithm chooses as search points exclusively those locations which iteratively minimize the error-function (SAD values). This category includes: the Adaptive Rood Pattern Search (ARPS) (Nie & Ma, 2002), the Fast Block Matching Using Prediction (FBMAUPR) (Liaw, et al., 2009), the Block-based Gradient Descent Search (BBGD) (Liu & Feig, 1996) and the Neighbourhood Elimination algorithm (NE) (Saha, et al., 2011). Such approaches assume that the error-function behaves monotonically, holding well for slow-moving sequences; however, such properties do not hold true for other kind of movements in video sequences (Chow & Liou, 1993), which risks on algorithms getting trapped into local minima.

(3) Decreasing the computational overhead for every search point, which means the matching cost (SAD operation) is replaced by a partial or a simplify version that features less complexity. The New pixel-Decimation (ND) (Saha, et al., 2008), the Efficient Block Matching Using Multilevel Intra and Inter-Sub-blocks (Li et al., 1994) and the Successive Elimination Algorithm (Chen, et al., 2002), all assume that all pixels within each block move by the same amount and a good estimate of the motion could be obtained through only a fraction of the pixel pool. However, since only a fraction of pixels enters into the matching computation, the use of these regular sub-sampling techniques can seriously affect the accuracy of the detection of motion vectors due to noise or illumination changes.

Another popular group of BM algorithms employ spatio-temporal correlation, using the neighboring blocks in spatial and temporal domain. The main advantage of these algorithms is that they alleviate the local minimum problem to some extent. Since the new initial or predicted search center

is usually closer to the global minimum, the chance of getting trapped in a local minimum decrease. This idea has been incorporated by many fast block motion estimation algorithms such as the enhanced predictive zonal search (EPZS) (Tourapis, 2002) and the UMHexagonS (Chen et al., 2002). However, the information delivered by the neighboring blocks occasionally conduces to false initial search points, producing distorted motion vectors. Such problem is typically caused when very small objects moves during the image sequence (Nisar, et al., 2012).

Alternatively, evolutionary approaches such as genetic algorithms (GA) (Holland, 1992) and particle swarm optimization (PSO) (Kennedy & Eberhart, 1995b) are well known for locating the global optimum in complex optimization problems. Despite of such fact, only few evolutionary approaches have specifically addressed the problem of BM, such as the light-weight genetic block matching (LWG) (Lin & Wu, 1998), the genetic four-step search (GFSS) (Wu & So, 2003) and the PSO-BM (Yuan & Shen, 2008). Although these methods support an accurate identification of the motion vector, their spending times are very long in comparison to other BM techniques.

On the other hand, the Harmony Search (HS) algorithm introduced by Geem et al., (2001) is one of the population-based evolutionary heuristics algorithms which are based on the metaphor of the improvisation process that occurs when a musician searches for a better state of harmony. The HS generates a new candidate solution from all existing solutions. In HS, the solution vector is analogous to the harmony in music, and the local and global search schemes are analogous to musician's improvisations. In comparison to other metaheuristics in the literature, HS imposes fewer mathematical requirements as it can be easily adapted for solving several sorts of Engineering Optimization challenges (Mahdavi, et al., 2007; Omran & Mahdavi, 2008).

Furthermore, numerical comparisons have demonstrated that the evolution for the HS is faster than GA (Lee & Geem, 2005; Lee, et al., 2005; Mahdavi et al., 2007), attracting ever more attention. It has been successfully applied to solve a wide range of practical optimization problems such as structural optimization, parameter estimation of the nonlinear Muskingum model, design optimization of water distribution networks, vehicle routing, combined heat and power economic dispatch, design of steel frames, bandwidth-delay-constrained least-cost multicast routing, computer vision, among others (Ayvaz, 2007; Cuevas, et al., 2012; Geem, 2005, 2006, 2008; Kim, et al., 2001; Lee & Geem, 2004, 2005; Lee et al., 2005).

A main difficulty applying HS to solve real-world problems is that it usually needs a large number of fitness evaluations before an acceptable result can be obtained. In practice, however, fitness evaluations are not always straightforward because either an explicit fitness function does not exist (an experiment is needed instead) or the evaluation of the fitness function is computationally demanding.

Moreover, since random numbers are involved in the calculation of new individuals, they may encounter the same positions (repetition) that other individuals have visited in previous iterations, especially when the individuals are confined to a small area.

The problem of considering expensive fitness evaluations has already been faced in the field of evolutionary algorithms (EA) and is better known as fitness approximation (Jin, 2005). In such approach, the idea is to estimate the fitness value of so many individuals as it is possible instead of evaluating the complete set. Such estimations are based on an approximate model of the fitness landscape. Thus, the individuals to be evaluated and those to be estimated are determined following some fixed criteria which depend on the specific properties of the approximate model (Jin, 2011). The models involved at the estimation can be built during the actual EA run, since EA repeatedly sample the search space at different points (Branke & Schmidt, 2005).

There are many possible approximation models, and several have already been used in combination with EA (e.g. polynomials (Zhou, et al., 2005), the kriging model (Ratle, 2001), the feed-

forward neural networks that includes multi-layer Perceptrons (Lim, et al., 2010) and radial basis-function networks (Ong, et al., 2008)). These models can be either global, which make use of all available data or local which make use of only a small set of data around the point where the function is to be approximated. Local models, however, have a number of advantages (Branke & Schmidt, 2005): they are well-known and suitably established techniques with relatively fast speeds. Moreover, they consider the intuitively most important information: the closest neighbors.

In this chapter, is presented a BM algorithm that combines HS with a fitness approximation model. Since the presented method approaches the BM process as an optimization problem, its overall operation can be formulated as follows: First, a population of individuals is initialized where each individual represents a motion vector candidate (a search location). Then, the set of HS operators is applied at each iteration in order to generate a new population. The procedure is repeated until convergence is reached whereas the best solution is expected to represent the most accurate motion vector. In the optimization process, the quality of each individual is evaluated through a fitness function which represents the SAD value corresponding to each motion vector. In order to save computational time, the approach incorporates a fitness estimation strategy to decide which search locations can be only estimated or actually evaluated. The method has been compared to other BM algorithms in terms of velocity and coding quality. Experimental results show that the HS-BM algorithm exhibits the best trade-off between coding efficiency and computational complexity.

The overall chapter is organized as follows: Section 5.2 holds a brief description about the HS method In Section 5.3, the fitness calculation strategy for solving the expensive optimization problem is presented. Section 5.4 provides background about the BM motion estimation issue while Section 5.5 exposes the final BM algorithm as a combination of HS and the fitness calculation strategy. Section 5.6 demonstrates experimental results for the presented approach over standard test sequences and some conclusions are drawn in Section 5.7.

5.2. Harmony Search Algorithm

5.2.1. The Harmony Search Algorithm

In the basic HS, each solution is called a “harmony” and is represented by an n -dimension real vector. An initial population of harmony vectors are randomly generated and stored within a Harmony Memory (HM). A new candidate harmony is thus generated from the elements in the HM by using a memory consideration operation either by a random re-initialization or a pitch adjustment operation.

Finally, the HM is updated by comparing the new candidate harmony and the worst harmony vector in the HM. The worst harmony vector is replaced by the new candidate vector in case it is better than the worst harmony vector in the HM. The above process is repeated until a certain termination criterion is met. The basic HS algorithm consists of three basic phases: HM initialization, improvisation of new harmony vectors and updating of the HM. The following discussion addresses details about each stage.

5.2.1.1. Initializing the problem and algorithm parameters

In general, the global optimization problem can be summarized as follows: $\min f(\mathbf{x}) : x(j) \in [l(j), u(j)]$, $j = 1, 2, \dots, n$, where $f(\mathbf{x})$ is the objective function, $\mathbf{x} = (x(1), x(2), \dots, x(n))$ is the set of design variables, n is the number of design variables, and $l(j)$ and $u(j)$ are the lower and upper bounds for the design variable $x(j)$, respectively. The parameters for HS are the harmony memory size, i.e., the number of solution vectors lying on the

harmony memory (HM), the harmony-memory consideration rate (*HMCR*), the pitch adjusting rate (*PAR*), the distance bandwidth (*BW*) and the number of improvisations (*NI*) which represents the total number of iterations. It is obvious that the performance of HS is strongly influenced by parameter values which determine its behavior.

5.2.1.2. Harmony memory initialization

In this stage, initial vector components at HM, i.e., *HMS* vectors, are configured. Let $\mathbf{x}_i = \{x_i(1), x_i(2), \dots, x_i(n)\}$ represent the i -th randomly-generated harmony vector: $x_i(j) = l(j) + (u(j) - l(j)) \cdot \text{rand}(0,1)$ for $j = 1, 2, \dots, n$ and $i = 1, 2, \dots, HMS$, where $\text{rand}(0,1)$ is a uniform random number between 0 and 1.

Then, the HM matrix is filled with the *HMS* harmony vectors as follows:

$$\text{HM} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{HMS} \end{bmatrix} \quad (5.1)$$

5.2.1.3. Improvisation of new harmony vectors

In this phase, a new harmony vector \mathbf{x}_{new} is built by applying the following three operators: memory consideration, random re-initialization and pitch adjustment. Generating a new harmony is known as ‘improvisation’. In the memory consideration step, the value of the first decision variable $x_{new}(1)$ for the new vector is chosen randomly from any of the values already existing in the current HM i.e., from the set $\{x_1(1), x_2(1), \dots, x_{HMS}(1)\}$. For this operation, a uniform random number r_1 is generated within the range $[0, 1]$. If r_1 is less than *HMCR*, the decision variable $x_{new}(1)$ is generated through memory considerations; otherwise, $x_{new}(1)$ is obtained from a random re-initialization between the search bounds $[l(1), u(1)]$. Values of the other decision variables $x_{new}(2), x_{new}(3), \dots, x_{new}(n)$ are also chosen accordingly. Therefore, both operations, memory consideration and random re-initialization, can be modelled as follows:

$$x_{new}(j) = \begin{cases} x_i(j) \in \{x_1(j), x_2(j), \dots, x_{HMS}(j)\} & \text{with probability } HMCR \\ l(j) + (u(j) - l(j)) \cdot \text{rand}(0,1) & \text{with probability } 1-HMCR \end{cases} \quad (5.2)$$

Every component obtained by memory consideration is further examined to determine whether it should be pitch-adjusted. For this operation, the Pitch-Adjusting Rate (*PAR*) is defined as to assign the frequency of the adjustment and the Bandwidth factor (*BW*) to control the local search around the selected elements of the HM. Hence, the pitch adjusting decision is calculated as follows:

$$x_{new}(j) = \begin{cases} x_{new}(j) = x_{new}(j) \pm \text{rand}(0,1) \cdot BW & \text{with probability } PAR \\ x_{new}(j) & \text{with probability } (1-PAR) \end{cases} \quad (5.3)$$

Pitch adjusting is responsible for generating new potential harmonies by slightly modifying original variable positions. Such operation can be considered similar to the mutation process in evolutionary algorithms. Therefore, the decision variable is either perturbed by a random number be-

tween 0 and BW or left unaltered. In order to protect the pitch adjusting operation, it is important to assure that points lying outside the feasible range $[l, u]$ must be re-assigned i.e., truncated to the maximum or minimum value of the interval.

5.2.1.4. Updating the harmony memory

After a new harmony vector x_{new} is generated, the harmony memory is updated by the survival of the fit competition between x_{new} and the worst harmony vector x_w in the HM. Therefore x_{new} will replace x_w and become a new member of the HM in case the fitness value of x_{new} is better than the fitness value of x_w .

5.2.2. Computational procedure

The computational procedure of the basic HS can be summarized as follows (Liaw et al., 2009):

Step 1	Set the parameters HMS , $HMCR$, PAR , BW and NI .
Step 2	Initialize the HM and calculate the objective function value of each harmony vector.
Step 3	Improve a new harmony \mathbf{x}_{new} as follows: for ($j = 1$ to n) do if ($r_1 < HMCR$) then $x_{new}(j) = x_a(j)$ where a is element of $(1, 2, \dots, HMS)$ randomly selected if ($r_2 < PAR$) then $x_{new}(j) = x_{new}(j) \pm r_3 \cdot BW$ where $r_1, r_2, r_3 \in \text{rand}(0,1)$ end if if $x_{new}(j) < l(j)$ $x_{new}(j) = l(j)$ end if if $x_{new}(j) > u(j)$ $x_{new}(j) = u(j)$ end if else $x_{new}(j) = l(j) + r \cdot (u(j) - l(j))$, where $r \in \text{rand}(0,1)$ end if end for
Step 4	Update the HM as $\mathbf{x}_w = \mathbf{x}_{new}$ if $f(\mathbf{x}_{new}) < f(\mathbf{x}_w)$
Step 5	If NI is completed, the best harmony vector \mathbf{x}_b in the HM is returned; otherwise go back to step 3.

5.3. Fitness approximation method

Evolutionary algorithms that use fitness approximation aim to find the global minimum of a given function considering only a very few numbers of function evaluations and a large number of estimations, based on an approximate model of the function landscape. In order to apply such approach, it is necessary that the objective function implicates a very expensive evaluation and consists of few dimensions (up to five) (Luo, et al., 2011). Recently, several fitness estimators have been reported in the literature (Lim et al., 2010; Ong et al., 2008; Ratle, 2001; Zhou et al., 2005) in

which the number of function evaluations is considerably reduced to hundreds, dozens, or even less. However, most of these methods produce complex algorithms whose performance is conditioned to the quality of the training phase and the learning algorithm in the construction of the approximation model.

In this chapter, we explore the use of a local approximation scheme based on the nearest-neighbor-interpolation (NNI) for reducing the function evaluation number. The model estimates the fitness values based on previously evaluated neighboring individuals which have been stored during the evolution process. At each generation, some individuals of the population are evaluated through the accurate (real) fitness function while the other remaining individuals are only estimated. The positions to be accurately evaluated are determined based on their proximity to the best individual or regarding their uncertain fitness value.

5.3.1 Updating the individual database

In a fitness approximation method, every evaluation of an individual produces one data point (individual position and fitness value) that is potentially taken into account for building the approximation model during the evolution process. Therefore, in our approach, we keep all seen-so-far evaluated individuals and their respective fitness values within a history array \mathbf{T} which is employed to select the closest neighbor and to estimate the fitness value of a new individual. Thus, each element of \mathbf{T} consists of two parts: the individual position and its respective fitness value. The array \mathbf{T} begins with null elements in the first iteration. Then, as the optimization process evolves, new elements are added. Since the goal of a fitness approximation approach is to evaluate the least possible number of individuals, only few elements are contained in \mathbf{T} .

5.3.2 Fitness calculation strategy

This section explains the strategy to decide which individuals are to be evaluated or estimated. The presented fitness calculation scheme estimates most of fitness values to reduce the computational overhead at each generation. In the model, those individuals positioned nearby the individual with the best fitness value at the array \mathbf{T} (Rule 1) are evaluated by using the actual fitness function. Such individuals are important as they possess a stronger influence over the evolution process than the others. Moreover, it also evaluates those individuals placed in regions of the search space with no previous evaluations (Rule 2). Fitness values for these individuals are uncertain since there is no close reference (close points contained in \mathbf{T}) to calculate their estimates.

The remaining individuals, for which there exist a close point that is previously evaluated and its fitness value is not the best contained in the array \mathbf{T} , are estimated using the NNI criterion (Rule 3). Thus, the fitness value of an individual is approximated by assigning the same fitness value that the nearest individual stored in \mathbf{T} .

Therefore, the fitness computation model follows three important rules to evaluate or estimate fitness values:

1. **Exploitation rule (evaluation).** If a new individual (search position) P is located closer than a distance d with respect to the nearest individual L_q contained in \mathbf{T} ($q = 1, 2, 3, \dots, m$; where m is the number of elements contained in \mathbf{T}), whose fitness value F_{L_q} corresponds to the best fitness value, then the fitness value of P is evaluated by using the actual fitness function. Figure 5.1(a) draws the rule procedure.

2. **Exploration rule (evaluation).** If a new individual P is located further away than a distance d with respect to the nearest individual L_q contained in \mathbf{T} , then its fitness value is evaluated by using the actual fitness function. Figure 5.1(b) outlines the rule procedure.
3. **NNI rule (estimation).** If a new individual P is located closer than a distance d with respect to the nearest individual L_q contained in \mathbf{T} , whose fitness value F_{L_q} does not correspond to the best fitness value, then its fitness value is estimated assigning it the same fitness that L_q ($F_P = F_{L_q}$). Figure 1c sketches the rule procedure.

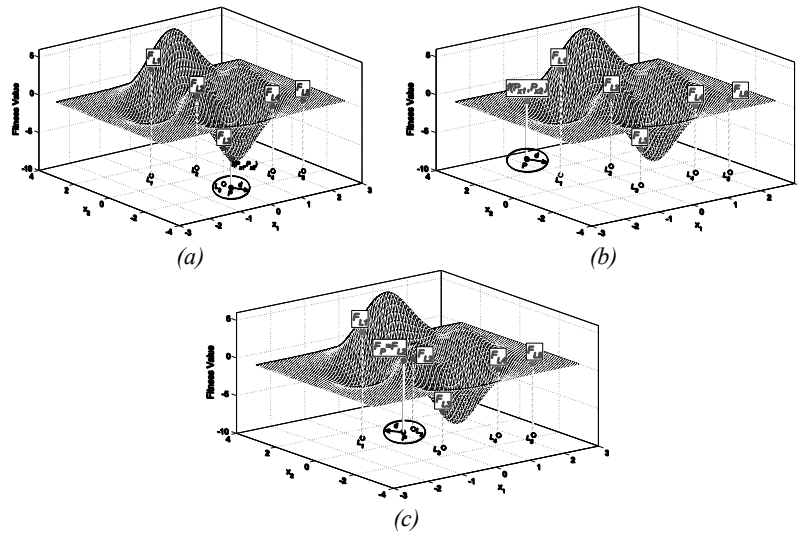


Fig. 5.1. The fitness calculation strategy. (a) According to the rule 1, the individual (search position) P is evaluated since it is located closer than a distance d with respect to the nearest individual location L_3 . Therefore, the fitness value F_{L_3} corresponds to the best fitness value (minimum). (b) According to the rule 2, the search point P is evaluated and there is no close reference within its neighborhood. (c) According to rule 3, the fitness value of P is estimated by means of the NNI-estimator, assigning $F_P = F_{L_2}$.

The d value controls the trade-off between the evaluation and the estimation of search locations. Typical values of d range from 1 to 4. Values close to 1 improve the precision at the expense of a higher number of fitness evaluations (the number of evaluated individuals is more than the number of estimated). On the other hand, values close to 4 decrease the computational complexity at the price of poor accuracy (decreasing the number of evaluation and increasing the number of estimations). After exhaustive experimentation, it has been determined that a value of $d=3$ represents the best trade-off between computational overhead and accuracy, so it is used throughout the study.

The presented method, from an optimization perspective, favors the exploitation and exploration in the search process. For the exploration, the method evaluates the fitness function of new search locations which have been located far from previously calculated positions. Additionally, it also estimates those that are closer.

For the exploitation, the presented method evaluates the fitness function of those new searching locations which are placed nearby the position that holds the minimum fitness value seen so far. Such fact is considered as a strong evidence that the new location could improve the “best value” (the minimum) already found.

With the purpose of knowing which rule must be applied by the fitness approximation strategy, considering a new search position P , it is only necessary to identify the closest individual L_q which is contained in \mathbf{T} . Then, it is inquired if the positional relationship between them and the fitness value F_{L_q} of L_q fulfill the properties imposed by each rule (distance, if F_{L_q} is the best fitness values contained in \mathbf{T} , etc). As the number of elements of the array \mathbf{T} is very limited, the computational complexity resulting from such operations is negligible.

Fig. 5.1 illustrates the procedure of fitness computation for a new solution (point P). In the problem, the objective function f is minimized with respect to two parameters (x_1, x_2) . In all figures (Figs. 5.1(a), (b) and (c)), the individual database array \mathbf{T} contains five different elements $(L_1, L_2, L_3, L_4, L_5)$ with their corresponding fitness values $(F_{L_1}, F_{L_2}, F_{L_3}, F_{L_4}, F_{L_5})$. Figures 5.1(a) and (b) show the fitness evaluation ($f(x_1, x_2)$) of the new solution P , following the rule 1 and 2 respectively, whereas Fig. 5.1(c) present the fitness estimation of P using the NNI approach which is laid by rule 3.

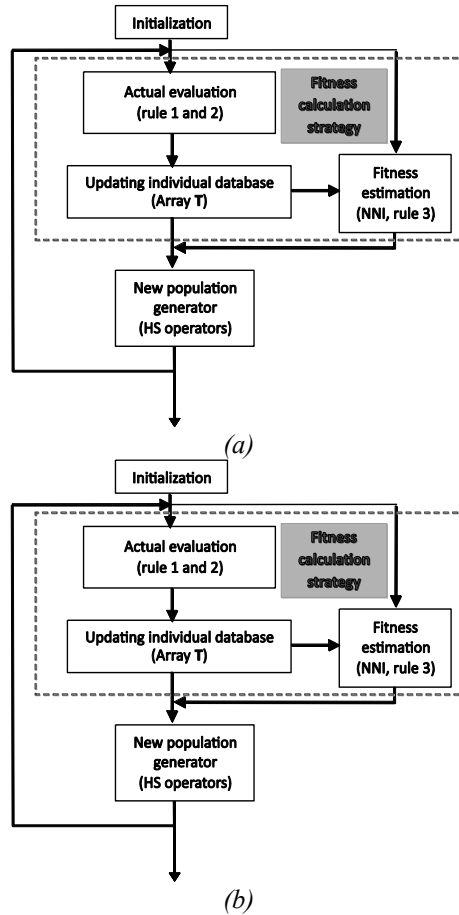


Fig. 5.2. Differences between the conventional HS and the HS optimization method presented in this chapter. (a) Conventional HS and (b) the HS algorithm including the fitness calculation strategy

5.3.3. HS optimization method

The coupling of HS and the fitness approximation strategy is presented in this chapter as an optimization approach. The only difference between the conventional HS and the enhanced HS method is the fitness calculation scheme.

In the presented algorithm, only some individuals are actually evaluated (Rules 1 and 2) at each generation. All other fitness values for the rest are estimated using the NNI-approach (Rule 3). The estimation is executed by using the individuals previously calculated which are contained in the array \mathbf{T} .

Fig.5.2 shows the difference between the conventional HS and the presented version. It is clear that two new blocks have been added, the fitness estimation and the updating individual database. Both elements and the actual evolution block, represent the fitness calculation strategy just as it has been explained at Section 5.3.2. As a result, the HS approach can substantially reduce the number of function evaluations preserving the good search capabilities of HS.

5.4. Motion estimation and Block-Matching

For motion estimation, in a BM algorithm, the current frame of an image sequence I_t is divided into non-overlapping blocks of $N \times N$ pixels. For each template block in the current frame, the best matched block within a search window (S) of size $(2W + 1) \times (2W + 1)$ in the previous frame I_{t-1} is determined, where W is the maximum allowed displacement. The position difference between a template block in the current frame and the best matched block in the previous frame is called the Motion Vector (MV) (see Fig. 5.3). Therefore, BM can be viewed as an optimization problem, with the goal of finding the best MV within a search space.

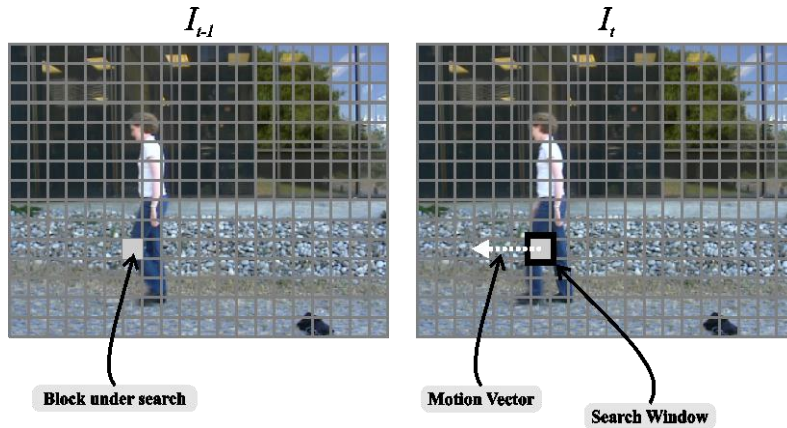


Fig. 5.3. Block Matching procedure.

The most well-known matching criterion for BM algorithms is the sum of absolute difference (SAD). It is defined in Eq. (5.5) considering a template block at position (x, y) in the current frame and the candidate block at position $(x + \hat{u}, y + \hat{v})$ in the previous frame I_{t-1} .

$$\text{SAD}(\hat{u}, \hat{v}) = \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} |g_t(x+i, y+j) - g_{t-1}(x+\hat{u}+i, y+\hat{v}+j)| \quad (5.4)$$

where $g_t(\cdot)$ is the gray value of a pixel in the current frame I_t and $g_{t-1}(\cdot)$ is the gray level of a pixel in the previous frame I_{t-1} . Therefore, the MV in (u, v) is defined as follows:

$$(u, v) = \arg \min_{(u, v) \in S} \text{SAD}(\hat{u}, \hat{v}) \quad (5.5)$$

where $S = \{(\hat{u}, \hat{v}) \mid -W \leq \hat{u}, \hat{v} \leq W \text{ and } (x + \hat{u}, y + \hat{v}) \text{ is a valid pixel position in } I_{t-1}\}$. As it can be seen, the computing of such matching criterion is a consuming time operation which represents the bottle-neck in the BM process.

In the context of BM algorithms, the FSA is the most robust and accurate method to find the MV. It tests all possible candidate blocks from I_{t-1} within the search area to find the block with minimum SAD. For the maximum displacement of W , the FSA requires $(2W + 1)^2$ search points. For instance, if the maximum displacement W is ± 7 , the total search points are 225. Each SAD calculation requires $2N^2$ additions and the total number of additions for the FSA to match a 16×16 block is 130,560. Such computational requirement makes the application of FSA difficult for real time applications.

5.5. Block-Matching algorithm based on Harmony Search with the estimation strategy

FSA finds the global minimum (the accurate MV), considering all locations within the search space S . Nevertheless, the approach has a high computational cost for practical use. In order to overcome such a problem, many fast algorithms have been developed yielding only a poorer precision than the FSA. A better BM algorithm should spend less computational time on searching and obtaining accurate motion vectors (MVs).

The BM algorithm presented at this chapter is comparable to the fastest algorithms and delivers a similar precision to the FSA approach. Since most of fast algorithms use a regular search pattern or assume a characteristic error function (uni-modal) for searching the motion vector, they may get trapped into local minima considering that for many cases (i.e., complex motion sequences) an uni-modal error is no longer valid.

Fig. 5.4 shows a typical error surface (SAD values) which has been computed around the search window for a fast-moving sequence. On the other hand, the presented BM algorithm uses a non-uniform search pattern for locating global minimum distortion.

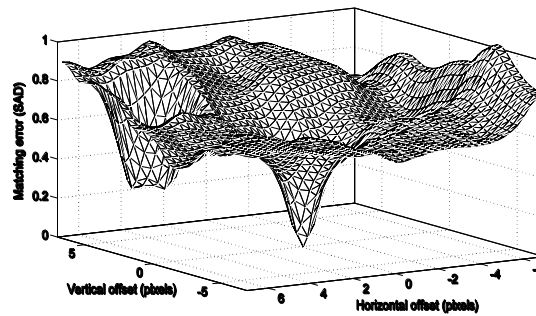


Fig. 5.4. Common non-unimodal error surface with multiple local minimum error points

Under the effect of the HS operators, the search locations vary from generation to generation, avoiding to get trapped into a local minimum. Besides, since the presented algorithm uses a fitness calculation strategy for reducing the evaluation of the SAD values, it requires fewer search positions.

In the algorithm, the search space S consists of a set of 2-D motion vectors \hat{u} and \hat{v} representing the x and y components of the motion vector, respectively. The particle is defined as:

$$P_i = \{\hat{u}_i, \hat{v}_i \mid -W \leq \hat{u}_i, \hat{v}_i \leq W\} \quad (5.6)$$

where each particle i represents a possible motion vector. In this chapter, the search windows, considered in the simulations, are set to ± 8 and ± 16 pixels. Both configurations are selected in order to compare the results with other approaches presented in the literature.

5.5.1. Initial population

The first step in HS optimization is to generate an initial group of individuals. The standard literature of evolutionary algorithms generally suggests the use of random solutions as the initial population, considering the absence of knowledge about the problem (Goldberg, 1989).

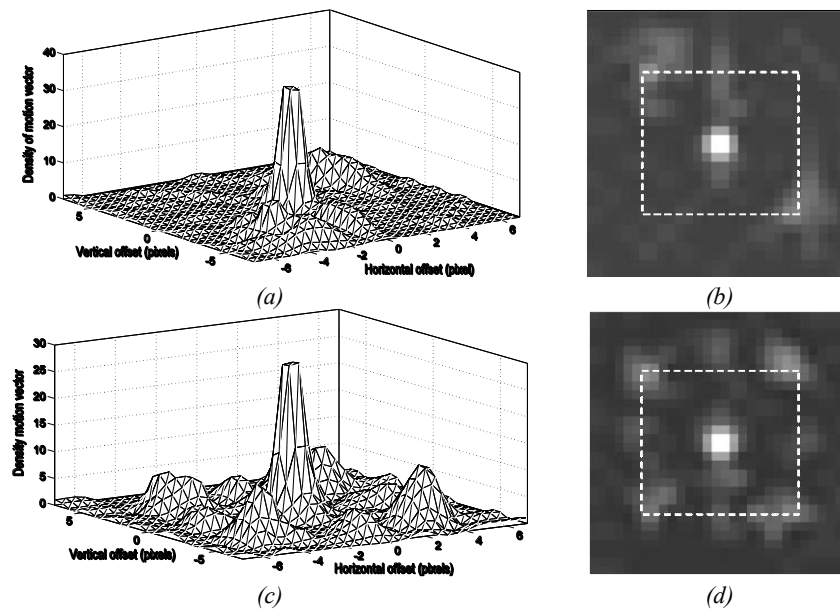


Fig. 5.5. Motion vector distribution for *Foreman* and *Stefan* sequences. (a)-(b) MV distribution for the *Foreman* sequence. (c)-(d) MV distribution for the *Stefan* sequence.

However, several studies (Xiao, et al., 2011; Luque, et al., 2011; Soak & Lee, 2012; Xiao, 2008) have demonstrated that the use of solutions generated through some domain knowledge to set the initial population (i.e., non-random solutions) can significantly improve its performance. In order to obtain appropriate initial solutions (based on knowledge), an analysis over the motion vector distribution was conducted. After considering several sequences (see Table 5.1 and Fig. 5.9), it can be seen that 98% of the MVs are found to lie at the origin of the search window for a slow-moving sequence such as the one at *Container*, whereas complex motion sequences, such as the *Carphone*

and the *Foreman* examples, have only 53.5% and 46.7% of their MVs in the central search region. The *Stefan* sequence, showing the most complex motion content, has only 36.9%. Figure 5.5 shows the surface of the MV distribution for the *Foreman* and the *Stefan*.

On the other hand, although it is less evident, the MV distribution of several sequences shows small peaks at some locations lying away from the center as they are contained inside a rectangle that is shown in Fig. 5.5(b) and 5.5(d) by a white overlay. Real-world moving sequences concentrate most of the MVs under a limit due to the motion continuity principle (Zhu & Ma, 2000).

Therefore, in this chapter, initial solutions are selected from five fixed locations which represent points showing the higher concentration in the MV distribution, just as it is shown by Figure 5.6.

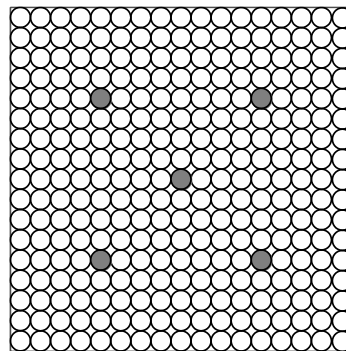


Fig. 5.6. Fixed pattern of five elements in the search window of ± 8 , used as initial solutions.

Since most movements suggest displacements near to the center of the search window (Jong et al., 1994; Li et al., 1994), the initial solutions shown by Fig. 5.6 are used as initial position for the HS algorithm. This consideration is taken regardless of the search window employed (± 8 or ± 16).

5.5.2. Tuning of the HS algorithm

The performance of HS is strongly influenced by parameter values which determine its behavior. HS incorporates several parameters such as the population size, the operator's probabilities (as *HMCR* and *PAR*) or the total number of iterations (*NI*).

Determining the most appropriate parameter values for a determined problem is a complex issue, since such parameters interact to each other in a highly nonlinear manner and there are not mathematical models of such interaction. Throughout the years, two main types of methods have been proposed for setting up parameter values of an evolutionary algorithm: off-line and on-line strategies (Montero & Riff, 2011).

An off-line method (called tuning) searches for the best set of parameter values through experimentation. Once defined, these values remain fixed. Such methodology is appropriate when the optimization problem maintains the same properties (dimensionality, multimodality, unconstrained, etc) each time that the EA is applied.

On the other hand, on-line methods focus on changing parameter values during the search process of the algorithm. Thus, the strategy must decide when to change parameter values and determine new values. Therefore, these methods are indicated when EA faces optimizations problems with dimensional variations or restriction changes, etc.

Considering that the optimization problem outlined by the BM process maintains the same properties (same dimensions and similar error landscapes), the off-line method has been used for tuning the HS algorithm. Therefore, after exhaustive experimentation, the following parameters have been found as the best parameter set, $HMCR=0.7$, $PAR=0.3$.

Considering that the presented approach is tested by using two different search windows (± 8 and ± 16), the values of BW and NI have different configurations depending on the selected search window. Therefore, it is employed $BW=8$ and $NI=25$ in the case of a window search of ± 8 whereas the case of ± 16 , it uses $BW=16$ and $NI=45$. Once such configurations are defined, the parameter set is kept for all test sequences through all experiments.

5.5.3. The HS-BM algorithm

The goal of our BM-approach is to reduce the number of evaluations of the SAD values (real fitness function) avoiding any performance loss and achieving an acceptable solution. The HS-BM method is listed below:

Step 1	Set the HS parameters. $HMCR=0.7$, $PAR=0.3$, $BW=8$ in case of a search window of ± 8 and 16 in case of ± 16 .
Step 2	Initialize the harmony memory with five individuals ($HMS=5$), where each decision variable u and v of the candidate motion vector MV_a is set according to the fixed pattern shown in Fig. 5.6. Considering $a \in (1, 2, \dots, HMS)$. Define also the individual database array \mathbf{T} , as an empty array.
Step 3	Compute the fitness values for each individual according to the fitness calculation strategy presented in Section 5.3. Since all individuals of the initial population fulfil rule 2 conditions, they are evaluated through a real fitness function (calculating the real SAD values).
Step 4	Update the new evaluations in the individual database array \mathbf{T} .
Step 5	Determine the candidate solution MV_w of HMS holding the worst fitness value.
Step 6	<pre> Improvise a new harmony MV_{new} such that: for ($j = 1$ to 2) do if ($r_1 < HMCR$) then $MV_{new}(j) = MV_a(j)$ where a is element of $(1, 2, \dots, HMS)$ randomly selected else if ($r_2 < PAR$) then $MV_{new}(j) = MV_{new}(j) \pm r_3 \cdot BW$ where $r_1, r_2, r_3 \in (0, 1)$ if $MV_{new}(j) < l(j)$ $MV_{new}(j) = l(j)$ end if if $MV_{new}(j) > u(j)$ $MV_{new}(j) = u(j)$ end if end if else $MV_{new}(j) = 1 + \text{round}(r \cdot E_p)$, where $r \in (-1, 1)$, $E_p = 8$ or 16. end if end for $E_p = 8$, in case of a search window of ± 8 and 16 in case of ± 16. </pre>

Step 7	Compute the fitness value of MV_{new} by using the fitness calculation strategy presented in Section 5.3.
Step 8	Update the new evaluation in the individual database array \mathbf{T} .
Step 9	Update HM . In case that the fitness value (evaluated or approximated) of the new solution MV_{new} , is better than the solution MV_w , such position is selected as an element of HM , otherwise the solution MV_w remains.
Step 10	Determine the best individual of the current new population. If the new fitness (SAD) value is better than the old best fitness value, then update \hat{u}_{best} , \hat{v}_{best} .
Step 11	If the number of iterations (NI) has been reached (25 in the case of a search window of ± 8 and 45 for ± 16), then the MV is \hat{u}_{best} , \hat{v}_{best} ; otherwise go back to Step 5.

Thus, the presented HS-BM algorithm considers different search locations, 30 in the case of a search window of ± 8 and 50 for ± 16 , during the complete optimization process (which consists of 25 and 45 different iterations depending on the search window, plus the five initial positions). However, only a few search locations are evaluated using the actual fitness function (5-14, in the case of a search window of ± 8 and 7-22, for ± 16) while the remaining positions are just estimated. Therefore, as the evaluated individuals and their respective fitness values are exclusively stored in the array \mathbf{T} , the resources used for the management of such data are negligible.

Figure 5.7 shows two search-patterns examples that have been generated by the HS-BM approach. Such patterns exhibit the evaluated search-locations (rule 1 and 2) in white-cells, whereas the minimum location is marked in black. Grey-cells represent cells that have been estimated (rule 3) or not visited during the optimization process.

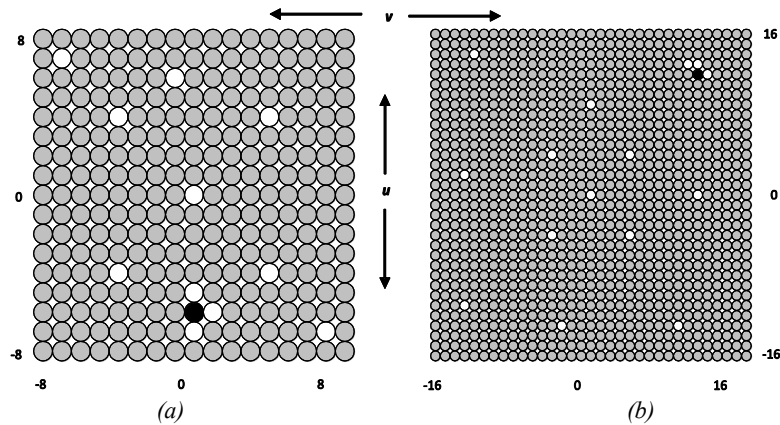


Fig. 5.7. Search-patterns generated by the HS-BM algorithm. (a) Search window pattern ± 8 with solution $\hat{u}_{best}^1 = 0$ and $\hat{v}_{best}^1 = -6$. (b) Search window pattern ± 16 with solution

$$\hat{u}_{best}^2 = 11 \text{ and } \hat{v}_{best}^2 = 12 .$$

5.5.4. Discussion on the accuracy of the fitness approximation strategy

HS has been found to be capable of solving several practical optimization problems. A distinguishing feature of HS is about its operation with only a population of individuals. It uses multiple candidate solutions at each step. This requires the computation of the fitness function for each candidate at every iteration. The ability to locate the global optimum depends on sufficient exploration

of the search space which requires the use of enough individuals. Under such circumstances, this work intends to couple the HS method with a fitness approximation model in order to replace (when it is feasible) the use of an expensive fitness function to compute the quality of several individuals.

Similar to other EA approaches, HS maintains two different phases on its operation: exploitation and exploration (Tan, et al., 2009). Exploitation (local search) refers to the action of refining the best solutions found so far whereas exploration (global search) represents the process of generating semi-random individuals in order to capture information of unexplored areas. In spite of this, the optimization process is guided by the best individuals seen-so-far (Wang, et al., 2011). They are selected more frequently and thereby modified or combined by the evolutionary operators in order to generate new promising individuals.

Therefore, the main concern in using fitness approximation models, is to accurately calculate the quality of those individuals which either hold great possibilities of grasping an excellent fitness value (individuals that are too close to one of the best individuals seen-so-far), or do not have reference about their possible fitness values (individuals located in unexplored areas) (Buche, et al., 2005; Tenne, 2012).

Most of the fitness approximation methods proposed in the literature (Lim et al., 2010; Ong et al., 2008; Ratle, 2001) use interpolation models in order to compute the fitness value of new individuals. Since the estimated fitness value is approximated considering other individuals which might be located far away from the position to be calculated, it introduces big errors that harshly affects the optimization procedure (Jin, 2011). Different to such methods, in our approach, the fitness values are calculated using three different rules which promote the evaluation of individuals that require particular accuracy (Rule 1 and Rule 2).

On the other hand, the strategy estimates those individuals which according to the evidence known so-far (elements contained in the array \mathbf{T}) represent unacceptable solutions (bad fitness values). Such individuals do not play an important role in the optimization process, therefore their accuracy is not considered critical (Branke & Schmidt, 2005; Giannakoglou, et al., 2006).

It is important to emphasize that the presented fitness approximation strategy has been designed considering some of the BM process particularities. Error landscapes in BM, due to the continuity principle (Saha et al., 2011; Tai, et al., 2007; Nie & Ma, 2002) of video sequences, present the following particularity: the closer neighbors to one global/local minimum (a motion vector with a low SAD value) decrement their SAD value as they approach to it. Such behavior is valid even in the most complex movement types. Under such circumstances, when it is necessary to calculate the fitness value of a search position which is close to one of the search positions previously visited (according to array \mathbf{T}) and whose fitness value was unacceptable, its fitness value is estimated according to Rule 3. This decision is taken considering that there is a strong evidence to consider such position as a bad individual from which it is not necessary to get a good accuracy level.

Fig. 5.8 presents the optimization procedure achieved by the combination between HS and the fitness approximation strategy presented in this chapter over a complex movement case. The example illustrates the fitness strategy operation for a complex movement considering a search window of ± 8 .

Fig. 5.8(a) shows the error landscape (SAD values) in a 3-D view, whereas Fig. 5.8(b) depicts the search positions calculated by the fitness approximation strategy over the SAD values that are computed for all elements of the search window as reference (for the sake of representation, both Figures are normalized from 0 to 1). Yellow squares indicate evaluated search positions whereas blue squares represent the estimated ones. Since random numbers are involved by HS in the gener-

ation of new individuals, they may encounter same positions (repetition) that other individuals have visited in previous iterations. Circles represent search positions that have been selected several times during the optimization procedure.

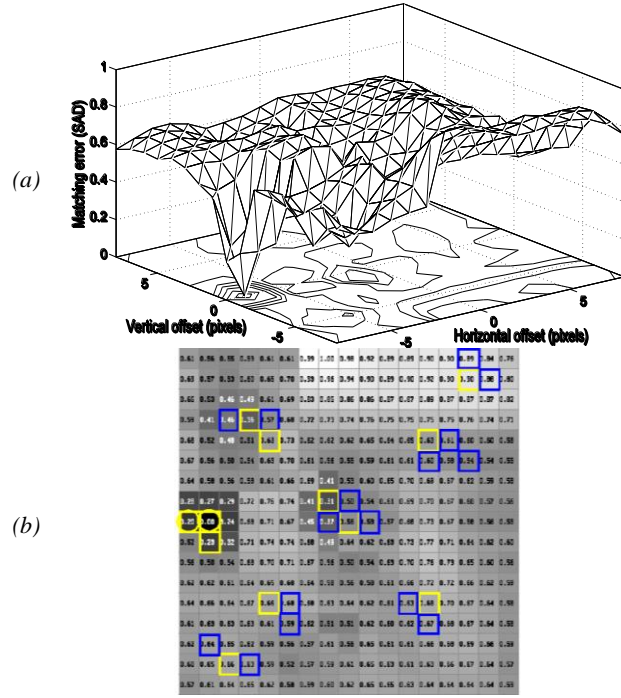


Fig. 5.8. Example of the optimization procedure: (a) Error landscape (SAD values) in a 3-D view. (b) Search positions calculated by the fitness approximation strategy over the SAD values which are computed for all elements of the search window of size ± 8 .

The problem of accuracy, in the estimation process, can also be appreciated through a close analysis from the red dashed square of Fig. 5.8(b). As the blue squares represent the estimated search positions according to the Rule 3, their fitness values are both assigned to 0.68 substituting their actual value of 0.63 and 0.67 respectively. Thus, considering that such individuals present an unacceptable solution (according to the elements stored in the array \mathbf{T}), the differences in the fitness value are negligible for the optimization process.

From Fig. 5.8(b), it can be seen that although the fitness function considers 30 individuals only 12 are actually evaluated by the fitness function (note that circle positions represent multiple evaluations).

5.6. Experimental results

5.6.1. HS-BM results

This section presents the results of comparing the HS-BM algorithm with other existing fast BMAs. The simulations have been performed over the luminance component of popular video sequences that are listed in Table 5.1. Such sequences consist of different degrees and types of motion including QCIF (176x144), CIF (352x288) and SIF (352x240) respectively. The first four sequences are *Container*, *Carphone*, *Foreman* and *Akiyo* in QCIF format. The next two sequences are *Stefan* in CIF format and *Football* in SIF format. Among such sequences, *Container* has gen-

tle, smooth and low motion changes and consists mainly of stationary and quasi-stationary blocks. *Carphone*, *Foreman* and *Akiyo* have moderately complex motion getting a “medium” category regarding its motion content. Rigorous motion which is based on camera panning with translation and complex motion content can be found in the sequences of *Stefan* and *Football*. Figure 5.9 shows a sample frame from each sequence.

Each picture frame is partitioned into macro-blocks with the sizes of 16×16 ($N=16$) pixels for motion estimation, where the maximum displacement within the search range W is of ± 8 pixels in both horizontal and vertical directions for the sequences *Container*, *Carphone*, *Foreman*, *Akiyo* and *Stefan*. The sequence *Football* has been simulated with a window size W of ± 16 , which requires a greater number of iterations (8 iterations) by the HS-BM method.



Fig. 5.9. Test video sequences.

In the comparison, two relevant performance indexes have been considered: the distortion performance and the search efficiency.

Sequence	Format	Total frames	Motion type
<i>Container</i>	QCIF(176x144)	299	Low
<i>Carphone</i>	QCIF(176x144)	381	Medium
<i>Foreman</i>	QCIF(352x288)	398	Medium
<i>Akiyo</i>	QCIF(352x288)	211	Medium
<i>Stefan</i>	CIF(352x288)	89	High
<i>Football</i>	SIF(352x240)	300	High

Table 5.1. Test sequences used in the comparison test.

In order to compare the performance of the HS-BM approach, different search algorithms such as FSA, TSS (Jong et al., 1994), 4SS (Po & Ma, 1996), NTSS (Li et al., 1994), BBGD (Liu & Feig, 1996), DS (Zhu & Ma, 2000), NE (Saha et al., 2011), ND (Saha et al., 2008), LWG (Lin & Wu, 1998), GFSS (Wu & So, 2003) and PSO-BM (Yuan & Shen, 2008) have been all implemented in

our simulations. For comparison purposes, all six video sequences in Fig. 5.8 have been all used. All simulations are performed on a Pentium IV 3.2 GHz PC with 1GB of memory.

Distortion performance

First, all algorithms are compared in terms of their distortion performance which is characterized by the Peak-Signal-to-Noise-Ratio (PSNR) value. Such value indicates the reconstruction quality when motion vectors, which are computed through a BM approach, are used. In PSNR, the signal is the original data frames whereas the noise is the error introduced by the calculated motion vectors. The PSNR is thus defined as:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{255^2}{MSE} \right) \quad (5.7)$$

where MSE is the mean square between the original frames and those compensated by the motion vectors. Additionally, as an alternative performance index, it is used in the comparison the PSNR degradation ratio (D_{PSNR}). This ratio expresses in percentage (%) the level of mismatch between the PSNR of a BM approach and the PSNR of the FSA which is considered as reference. Thus the D_{PSNR} is defined as

$$D_{\text{PSNR}} = - \left(\frac{\text{PSNR}_{\text{FSA}} - \text{PSNR}_{\text{BM}}}{\text{PSNR}_{\text{FSA}}} \right) \cdot 100\% \quad (5.8)$$

Table 5.2 shows the comparison of the PSNR values and the PSNR degradation ratios (D_{PSNR}) among the BM algorithms. The experiment considers the six image sequences presented in Fig. 5.8. As it can be seen, in the case of the slow-moving sequence *Container*, the PSNR values (the D_{PSNR} ratios) of all BM algorithms are similar. For the medium motion content sequences such as *Carphone*, *Foreman* and *Akiyo*, the approaches consistent of fixed patterns (TSS, 4SS and NTSS) exhibit the worst PSNR value (high D_{PSNR} ratio) except for the DS algorithm. On the other hand, BM methods that use evolutionary algorithms (LWG, GFSS, PSO-BM and HS-BM) present the lowest D_{PSNR} ratio, only one step under the FSA approach which is considered as reference. Finally, approaches based on the error-function minimization (BBGD and NE) and pixel-decimation (ND), show an acceptable performance. For the high motion sequence of *Stefan*, since the motion content of these sequences is complex producing error surfaces with more than one minimum, the performance, in general, becomes worst for most of the algorithms especially for those based on fixed patterns. In the sequence *Football*, which has been simulated with a window size of ± 16 , the algorithms based on the evolutionary algorithms present the best PSNR values. Such performance is because evolutionary methods adapt better to complex optimization problems where the search area and the number of local minima increase. As a summary of the distortion performance, the last column of Table 5.2 presents the average PSNR degradation ratio (D_{PSNR}) obtained for all sequences. According to such values, the HS-BM method is superior to any other approach. Due to the computation complexity, the FSA is considered just as a reference. The best entries are bold-cased in Table 5.2.

Search efficiency

The search efficiency is used in this section as a measurement of computational complexity. The search efficiency is calculated by counting the average number of search points (or the average number of SAD computations) for the MV estimation.

In Table 5.3, the search efficiency is compared, where the best entries are bold-cased. Just above FSA, some evolutionary algorithms such as LWG, GFSS and PSO-BM hold the highest number of search points per block. On the contrary, the HS-BM algorithm can be considered as a fast approach as it maintains a similar performance to DS.

From data shown in Table 5.3, the average number of search locations, corresponding to the HS-BM method, represents the number of SAD evaluations (the number of SAD estimations are not considered whatsoever). Additionally, the last two columns of Table 5.3 present the number of search locations that have been averaged (over the six sequences) and their performance rank. According to these values, the HS-BM method is ranked in the first place.

Algorithm	Container $W=\pm 8$		Carphone $W=\pm 8$		Foreman $W=\pm 8$		Akiyo $W=\pm 8$		Stefan $W=\pm 8$		Football $W=\pm 16$		Total Average (D_{PSNR})
	PSNR	D_{PSNR}	PSNR	D_{PSNR}	PSNR	D_{PSNR}	PSNR	D_{PSNR}	PSNR	D_{PSNR}	PSNR	D_{PSNR}	
FSA	43.18	0	31.51	0	31.69	0	29.07	0	25.95	0	23.07	0	0
TSS	43.10	-0.20	30.27	-3.92	29.37	-7.32	26.21	-9.84	21.14	-18.52	20.03	-13.17	-8.82
4SS	43.12	-0.15	30.24	-4.01	29.34	-7.44	26.21	-9.84	21.41	-17.48	20.10	-12.87	-8.63
NTSS	43.12	-0.15	30.35	-3.67	30.56	-3.57	27.12	-6.71	22.52	-13.20	20.21	-12.39	-6.61
BBGD	43.14	-0.11	31.30	-0.67	31.00	-2.19	28.10	-3.33	25.17	-3.01	22.03	-4.33	-2.27
DS	43.13	-0.13	31.26	-0.79	31.19	-1.59	28.00	-3.70	24.98	-3.73	22.35	-3.12	-2.17
NE	43.15	-0.08	31.36	-0.47	31.23	-1.47	28.53	-2.69	25.22	-2.81	22.66	-1.77	-1.54
ND	43.15	-0.08	31.35	-0.50	31.20	-1.54	28.32	-2.56	25.21	-2.86	22.60	-2.03	-1.59
LWG	43.16	-0.06	31.40	-0.36	31.31	-1.21	28.71	-1.22	25.41	-2.09	22.90	-0.73	-0.95
GFSS	43.15	-0.06	31.38	-0.40	31.29	-1.26	28.69	-1.28	25.34	-2.36	22.92	-0.65	-1.01
PSO-BM	43.15	-0.07	31.39	-0.38	31.27	-1.34	28.65	1.42	25.39	-2.15	22.88	-0.82	-1.03
HS-BM	43.16	-0.03	31.49	-0.03	31.63	-0.21	29.01	-0.18	25.89	-0.20	23.01	-0.20	-0.18

Table 5.2. PSNR values and D_{PSNR} comparison of the BM methods

Algorithm	Container $W=\pm 8$	Carphone $W=\pm 8$	Foreman $W=\pm 8$	Akiyo $W=\pm 8$	Stefan $W=\pm 8$	Football $W=\pm 16$	Total Average	Rank
FSA	289	289	289	289	289	1089	422.3	12
TSS	25	25	25	25	25	25	25	8
4SS	19	25.5	24.8	27.3	25.3	25.6	24.58	7
NTSS	17.2	21.8	22.1	23.5	25.4	26.5	22.75	6
BBGD	9.1	14.5	14.5	13.2	17.2	22.3	15.13	3
DS	7.5	12.5	13.4	11.8	15.2	17.8	13.15	2
NE	11.7	13.8	14.2	14.5	19.2	24.2	16.36	5
ND	10.8	13.4	13.8	14.1	18.4	25.1	16.01	4
LWG	75	75	75	75	75	75	75	11
GFSS	60	60	60	60	60	60	60	10
PSO-BM	32.5	48.5	48.1	48.5	52.2	52.2	47	9
HS-BM	8.0	12.2	11.2	11.5	17.1	15.2	12.50	1

Table 5.3. Averaged number of visited search points per block for all ten BM methods.

The average number of search points visited by the HS-BM algorithm ranges from 9.2 to 17.3, representing the 4% and the 7.4% respectively in comparison to the FSA method. Such results demonstrate that our approach can significantly reduce the number of search points. Hence, the HS-BM algorithm presented in this chapter is comparable to the fastest algorithms and delivers a similar precision to the FSA approach.

5.6.2. Results on H.264

Other set of experiments have been performed in JM-12.2 [64] of H.264/AVC reference software. In the simulations, we compare FS, DS (Zhu & Ma, 2000), EPZS (Tourapis, 2002), TSS (Jong et al., 1994), 4SS (Po & Ma, 1996), NTSS (Li et al., 1994), BBGD (Liu & Feig, 1996) and the HS-BM algorithm in terms of coding efficiency and computational complexity.

For encoding purposes JM-12.2 Main Encoder Profile has been used. For each test sequence only the first frame has been coded as “I frame” and the remaining frames are coded as P frames. Only one reference frame has been used.

Each pixel in the image sequences is uniformly quantized to 8 bits. Sum of absolute difference (SAD) distortion function is used as the block distortion measure (BDM). Image formats used are QCIF, CIF and SIF meanwhile sequences are tested at 30 fps (frames per second). The simulation platform in our experiments is a PC with Intel Pentium IV 2.66 GHz CPU.

The test sequences used for our experiments are *Container*, *Akiyo* and *Football*. These sequences exhibit a variety of motion that is generally encountered in real video. For the sequences *Container* and *Akiyo* a search window of ± 8 is selected meanwhile for the football sequence a search window of ± 16 is considered. The group of experiments has been performed over such sequences at four different quantization parameters (QP=28,32,36,40) in order to test the algorithms at different transmission conditions.

a) Coding efficiency

In the first experiment, the performance of the HS-BM algorithm is compared to other BM algorithms regarding the coding efficiency. Two different performance indexes are used for evaluating the coding quality: the PSNR Gain and the increasing of the Bit Rate.

In order to comparatively assess the results, two additional indexes, called PSNR loss and Bit Rate Incr., relate the performance of each method with the FSA performance as a reference. Such indexes are calculated as follows:

$$\text{PSNR loss} = \text{PSNR FSA} - \text{PSNR algorithm} \quad (5.9)$$

$$\text{Bit Rate Incr.} = \left(\frac{\text{Bit Rate algorithm} - \text{Bit Rate FSA}}{\text{Bit Rate FSA}} \right) \cdot 100 \quad (5.10)$$

Tables 5.4 – 5.6 show a coding efficiency comparison among BM algorithms. It is observed, from experimental results, that the HS-BM algorithm holds an effective coding quality because the loss in terms of PSNR and the increase of the Bit rate are low with an average of 1.6dB and -0.04%, respectively. Such coding performance is similar to the one produced by the EPZS method whereas it is much better than the obtained by other BM algorithms which possess the worst coding quality.

b) Computational complexity

In the second experiment, we have compared the performance of the HS-BM algorithm to other BM algorithms in terms of computational overhead. As the JM-12.2 platform allows to simulate BM algorithms in real time conditions, we have used such results in order to evaluate their performances.

Three different performance indexes are used for evaluating the computational complexity; they are the Averaged Coding Time (ACT), Instruction Number (IN) and Consumed Memory (CM). The ACT is the averaged time employed to codify a complete frame (the averaged time consumed after finding all the corresponding motion vectors for a frame). IN represents the number of instructions used to implement each algorithm in the JM-12.2 profile. CM considers the memory size used by the JM-12.2 platform in order to manage the data that are employed by each BM algorithm.

BM	Coding efficiency				Computational complexity		
	PSNR	Bit-rate (Kbits/s)	PNSR loss (dB)	Bit-rate increase (%)	ACT (ms)	IN	CM (Bytes)
FSA	36.06	41.4	-	-	133.2	122	3072
DS	36.04	43.4	0.02	4.83	6.33	138	420
EPZS	36.04	41.3	0.02	-0.20	19.5	621	8972
TSS	34.01	45.2	2.05	9.17	2.1	100	180
4SS	35.22	44.7	0.84	7.97	2.8	100	204
NTSS	35.76	44.3	0.30	7.00	3.7	110	256
BBGD	35.98	42.1	0.08	1.70	9.1	256	1024
HS-BM	36.04	41.5	0.02	0.20	3.8	189	784

Table 5.4. Coding efficiency results for the *container* sequence, considering a window size W of ± 8 .

BM	Coding efficiency				Computational complexity		
	PSNR	Bit-rate (Kbits/s)	PNSR loss (dB)	Bit-rate increase (%)	ACT (ms)	IN	CM (Bytes)
FSA	38.19	25.6	-	-	133.2	122	3072
DS	38.11	25.9	0.08	1.20	7.45	138	420
EPZS	38.19	25.3	-	-1.20	22.1	621	8972
TSS	30.32	29.3	7.87	14.45	2.1	100	180
4SS	32.42	28.4	5.77	10.93	2.8	100	204
NTSS	33.57	27.2	4.62	6.25	3.7	110	256
BBGD	35.21	26.8	2.98	4.68	10.1	256	1024
HS-BM	38.17	25.5	0.02	-0.40	3.9	189	784

Table 5.5. Simulation results for the *Akiyo* sequence, considering a window size W of ± 8 .

Tables 5.4 – 5.6 show the computational complexity comparison among the BM algorithms. It is observed from the experimental results that the HS-BM algorithm possesses a competitive ACT value (from 3.8 to 4.1 milliseconds) in comparison to other BM algorithms. This fact reflexes that although the cost of applying the fitness approximation strategy represents an overhead that is not required in most fast BM methods, such overhead is negligible in comparison to the cost of the number of fitness evaluations which have been saved. The ACT values, presented by the HS-BM, are lightly superior to those produced by the fast BM methods (TSS, 4SS and NTSS) whereas it is much better than those generated by the EPZS algorithm which possesses the worst computational performance. On the other hand, the resources (in terms of number of instructions IN and required

memory CM) needed by the HS-BM approach are considered as standard in software and hardware architectures.

BM	Coding efficiency				Computational complexity		
	PSNR	Bit-rate (Kbits/s)	PNSR loss (dB)	Bit-rate increase (%)	ACT (ms)	IN	CM (Bytes)
FSA	34.74	98.85	-	-	245.7	122	12288
DS	32.22	99.89	2.52	1.02	10.36	144	600
EPZS	34.72	98.81	0.02	-0.04	26.8	678	20256
TSS	27.12	106.42	7.62	7.65	2.9	113	180
4SS	27.91	105.29	6.83	6.51	3.1	113	204
NTSS	29.11	104.87	5.63	6.09	4.2	113	256
BBGD	29.76	103.96	4.98	5.19	16.41	268	2048
HS-BM	34.73	98.91	0.01	0.06	4.1	201	1024

Table 5.6. Simulation results for the *Football* sequence, considering a window size W of ± 16 .

5.7. Conclusions

In this chapter, a Block-Matching algorithm that combines Harmony Search with a fitness approximation model is presented. The approach uses as potential solutions the motion vectors belonging to the search window. A fitness function evaluates the matching quality of each motion vector candidate. To save computational time, the approach incorporates a fitness calculation strategy to decide which motion vectors can be estimated or actually evaluated. Guided by the values given by such fitness calculation strategy, the set of motion vectors are evolved using the HS operators so the best possible motion vector can be identified.

Since the presented algorithm does not consider any fixed search pattern during the BM process or any other movement assumption, a high probability for finding the true minimum (accurate motion vector) is expected regardless of the movement complexity contained in the sequence. Therefore, the chance of being trapped into a local minimum is reduced in comparison to other BM algorithms.

The performance of HS-BM has been compared to other existing BM algorithms by considering different sequences which present a great variety of formats and movement types. Experimental results demonstrate that the presented algorithm maintains the best balance between coding efficiency and computational complexity.

Although the experimental results indicate that the HS-BM method can yield better results on complicated sequences, it should be noticed that the aim of this chapter is to show that the fitness approximation can effectively serve as an attractive alternative to evolutionary algorithms for solving complex optimization problems, yet demanding fewer function evaluations.

Chapter 6

Multi-Threshold Segmentation using Learning Automata

Multi-Threshold selection for image segmentation is considered as a critical pre-processing step for image analysis, pattern recognition and computer vision. This chapter explores the use of the Learning Automata (LA) algorithm to compute the thresholding points for segmentation purposes. LA is a heuristic method which is able to solve complex optimization problems with interesting results in parameter estimation. Different to other optimization approaches, LA explores in the probability space providing appropriate convergence properties and robustness. In this chapter the segmentation task is considered as an optimization problem and the LA is used to generate the image multi-threshold points. In this approach, one 1-D histogram of a given image is approximated through a Gaussian mixture model whose parameters are calculated using the LA algorithm. Each Gaussian function approximating the histogram represents a pixel class and therefore a thresholding point. Experimental results show fast convergence of the method, avoiding the typical sensitivity to initial conditions.

6.1. Introduction

Several image processing applications aim to detect and mark relevant features which may be later analyzed to perform several high-level tasks. In particular, image segmentation seeks to group pixels within meaningful regions. Commonly, gray levels belonging to the object, are substantially different from those featuring the background. Thresholding is thus a simple but effective tool to isolate objects of interest; its applications include several classics such as document image analysis, whose goal is to extract printed characters (Abak, et al., 1997; Kamel & Zhao, 1993), logos, graphical content, or musical scores; also it is used for map processing which aims to locate lines, legends, and characters (Trier & Jain, 1995). Moreover, it is employed for scene processing, seeking for object detection, marking (Bhanu, 1986a) and for quality inspection of materials (Sezgin & Sankur, 2001; Sezgin & Taşaltın, 2000).

Thresholding selection techniques can be classified into two categories: bi-level and multi-level. In the former, one limit value is chosen to segment an image into two classes: one representing the object and the other one segmenting the background. When distinct objects are depicted within a given scene, multiple threshold values have to be selected for proper segmentation, which is commonly called multilevel thresholding.

A variety of thresholding approaches have been proposed for image segmentation, including conventional methods (Guo & Pandit, 1998; Pal & Pal, 1993; Sahoo, et al., 1988; Snyder, et al., 1990) and intelligent techniques (see for instance (Chen & Wang, 2005; Lai, 2006)). Extending the segmentation algorithms to a multilevel approach may cause some inconveniences: (i) they may have no systematic or analytic solution when the number of classes to be detected increases and (ii) they may also show a slow convergence and/or high computational cost (Chen & Wang, 2005).

In this chapter, the segmentation algorithm is based on a parametric model holding a probability density function of gray levels which groups a mixture of several Gaussian density functions (Gaussian mixture). Mixtures represent a flexible method of statistical modelling as they are em-

ployed in a wide variety of contexts (Böhning & Seidel, 2003). Gaussian mixture has received considerable attention in the development of segmentation algorithms despite its performance is influenced by the shape of the image histogram and the accuracy of the estimated model parameters (Gupta & Sortrakul, 1998). The associated parameters can be calculated considering an approximated maximum a posterior estimation (MAP) or the maximum likelihood (ML) estimation, considering the Expectation Maximization (EM) algorithm (Dempster, et al., 1977; Zhang, et al., 2003) or Gradient-based methods (Park, et al., 2000).

The EM algorithm provides a simple alternative procedure for computing posterior density or likelihood functions. However, its slow convergence speed has been pointed out as the most serious practical problem (Redner & Walker, 1984). When the EM is used aside Gaussian mixtures, the convergence speed depends on the separation of component populations within such mixture. Therefore, the EM algorithms are very sensitive to the choice of the initial values (Park & Ozeki, 2009). Moreover, the EM algorithm also tends to converge to local minima (Ma, et al., 2000; Xu & Jordan, 1996). A feasible way for solving this problem is to choose several sets of initial values, applying the EM algorithm and finally choosing the best outcome-set as the best estimation. By doing so, it can alleviate the influence of the initial values on the algorithm but increasing the computational cost. Additionally, the EM algorithm fails to converge if one or some variances of the Gaussian mixture approach to zero as it has been demonstrated when big objects with uniform intensities have undergone segmentation (Gupta & Sortrakul, 1998).

On the other hand, Gradient-based methods are also computationally expensive and may easily get stuck within local minima (Gupta & Sortrakul, 1998). Redner and Walkerin argued in (Redner & Walker, 1984) -a widely-cited article, that Newtonian methods (such as Levenberg-Marquardt, LM) should generally be preferred over EM particularly for unconstrained optimization problems. However, they must be modified in order to be used within Gaussian mixtures, where there are probabilistic constraints on the parameters (Xu & Jordan, 1996) that may result in singularities.

In the parameter space of mixture models, the singularities occur when two or more components are exactly overlapped, and they can be dismissed as a single component. Recent works (Olsson, et al., 2007; Park & Ozeki, 2009) have shown that singularities cause slow convergence in Newtonian and quasi-Newtonian methods while they are applied to determine parameters of Gaussian mixtures.

Despite gradient-based methods and the EM algorithm seem to have different mechanisms for parameter updating. Xu & Jordan (1996) have established a relationship between the gradient of the log likelihood and the updating step within the parameter space while using the EM algorithm. They found that the EM algorithm can be viewed as a variable metric gradient algorithm with a projection matrix changing at each step and behaving just like a function of the current parameter value. In the EM algorithm, the new value of the parameter $k+1$ is thus chosen close to the previous value k mimicking gradients methods. Therefore, the updating rule may get the EM algorithm easily stuck within local minima (Olsson et al., 2007).

In this chapter, an alternative approach using an optimization algorithm based on learning automata for determining the parameters of a Gaussian mixture is presented. The Learning Automata (LA) (Najim & Poznyak, 1994; Narendra & Thathachar, 1989) is an adaptive decision making method that operate in unknown random environments while progressively improve their performance via a learning process. Since LA theorists study the optimization under uncertainty, it is very useful for optimization of multi-modal functions when the function is unknown and only noise-corrupted evaluations are available. In such algorithms, a probability density function, which is defined over the parameter space, is used to select the next point. The reinforcement signal (objective function) and the learning algorithm are used by the learning automata (LA) to update the probability density function at each stage. LA has been successfully applied to solve different sorts

of engineering problems such as pattern recognition (Zahiri, 2008), adaptive control (Zeng, et al., 2000) signal processing (Howell & Gordon, 2001) and power systems (Wu, 1995).

One main advantage of the LA method is that it does not need knowledge of the environment or any other analytical reference to the function to be optimized. Additionally, one interesting advantage of LA lies on the fact that it offers fast convergence mainly when it is considered for the estimation of many parameters (Thathachar & Sastry, 2002). Other methods such as Gradient and the EM which make use of iterative updating procedures within the parameter space, may exhibit slow convergence or local minima trapping. However, LA is focused on the probability space (Zeng & Liu, 2005) leading to global optimization (Beigy & Meybodi, 2006) by allowing any element of the action set (or parameter) to be chosen. Such fact makes LA insensitive to initial values.

Recently, more effective LA-based algorithms have been proposed for multimodal complex function optimization (Beigy & Meybodi, 2006; Howell, et al., 1997; Thathachar & Sastry, 2002; Zeng & Liu, 2005). It has also been experimentally shown that the performance of such optimization algorithms is comparable to or better than the genetic algorithm (GA) in (Zeng & Liu, 2005). On the other hand, the algorithm known as continuous action reinforcement learning automata (CARLA) (Frost, 1998), has been used for parameter identification of particularly complex systems, showing the effectiveness of the approach with interesting results on adaptive control (Frost, 1998; Howell & Best, 2000; Howell et al., 1997; Kashki, et al., 2008) and digital filter design (Howell & Gordon, 2001).

In this chapter, the segmentation process is considered as an optimization problem approximating the 1-D histogram of a given image by means of a Gaussian mixture model. The operation parameters are calculated through the CARLA algorithm. Each Gaussian contained within the histogram represents a pixel class and therefore belongs to the thresholding points. The experimental results, presented in this work, demonstrate that LA exhibits fast convergence, relatively low computational cost and no sensitivity to initial conditions by keeping an acceptable segmentation of the image, i.e. a better mixture approximation in comparison to the EM or gradient based algorithms.

The chapter is organized as follows. Section 6.2 presents the Gaussian approximation to the histogram while Section 6.3 introduces the LA algorithm. Section 6.4 shows the most important implementation issues. Experimental results for the proposed approach are presented in Section 6.5 and some relevant conclusions are discussed in Section 6.6.

6.2. Gaussian approximation

Let consider an image holding L gray levels $[0, \dots, L-1]$ whose distribution is displayed within a histogram $h(g)$. In order to simplify the description, the histogram is normalized just as a probability distribution function, yielding:

$$h(g) = \frac{n_g}{N}, \quad h(g) > 0, \quad (6.1)$$

$$N = \sum_{g=0}^{L-1} n_g, \quad \text{and} \quad \sum_{g=0}^{L-1} h(g) = 1,$$

where n_g denotes the number of pixels with gray level g and N being the total number of pixels in the image.

The histogram function can thus be contained into a mix of Gaussian probability functions of the form:

$$p(x) = \sum_{i=1}^K P_i \cdot p_i(x) = \sum_{i=1}^K \frac{P_i}{\sqrt{2\pi\sigma_i}} \exp\left[\frac{-(x-\mu_i)^2}{2\sigma_i^2}\right] \quad (6.2)$$

with P_i being the probability of class i , $p_i(x)$ being the probability distribution function of gray-level random variable x in class i , with μ_i and σ_i being the mean and standard deviation of the i -th probability distribution function and K being the number of classes within the image. In addition, the constraint $\sum_{i=1}^K P_i = 1$ must be satisfied.

The mean square error is used to estimate the $3K$ parameters P_i , μ_i and σ_i , $i = 1, \dots, K$. For instance, the mean square error between the Gaussian mixture $p(x_i)$ and the experimental histogram function $h(x_i)$ is now defined as follows:

$$J = \frac{1}{n} \sum_{j=1}^n [p(x_j) - h(x_j)]^2 + \omega \cdot \left| \left(\sum_{i=1}^K P_i \right) - 1 \right| \quad (6.3)$$

Assuming an n -point histogram and ω being the penalty associated with the constrain $\sum_{i=1}^K P_i = 1$.

In general, the estimation of the parameters that minimize the square error produced by the Gaussian mixture is not a simple problem. A straightforward method is to consider the partial derivatives of the error function to zero, obtaining a set of simultaneous transcendental equations (Gonzalez & Woods, 2008). However, an analytical solution is not available considering the non-linear nature of the equations. The algorithms therefore make use of an iterative approach which is based on the gradient information or maximum likelihood estimation, just like the EM algorithm. Unfortunately, such methods may also get easily stuck within local minima.

For the EM algorithm and the gradient-based methods, the new parameter point lies within a neighbourhood distance of the previous parameter point. However, this is not the case for the LA's adaptation algorithm which is based on stochastic principles.

The new operating point is thus determined by a parameter probability function and therefore it can be far from the previous operating point. This gives the algorithm a higher ability to locate and pursue a global minimum.

It has been shown by many works in the literature that intelligent approaches may actually provide a satisfactory performance for image processing problems (Baştürk & Günay, 2009; Chen & Wang, 2005; Lai, 2006; Lai & Tseng, 2001; Tseng & Lai, 1999). The LA approach was chosen aiming into find appropriate parameters and their corresponding threshold values yet relying on the LA convergence characteristics and its immunity to initial values.

6.3. Learning Automata (LA)

LA operates by selecting actions via a stochastic process. Such actions operate within an environment while being assessed according to a measure of the system performance. Figure 6.1(a)

shows the typical learning system architecture. The automaton selects an action (\mathbf{X}) probabilistically. Such actions are applied to the environment and the performance evaluation function provides a reinforcement signal β . This is used to update the automaton's internal probability distribution whereby actions that achieve desirable performance are reinforced via an increased probability.

Likewise, those underperforming actions are penalised or left unchanged depending on the particular learning rule which has been employed. Over time, the average performance of the system will improve until a given limit is reached. In terms of optimization problems, the action with the highest probability would correspond to the global minimum as demonstrated by rigorous proofs of convergence available in (Najim & Poznyak, 1994; Narendra & Thathachar, 1989).

A wide variety of learning rules have been reported in the literature. One of the most widely used algorithms is the linear reward/inaction (L_{RI}) scheme, which has been shown to guaranteed convergence properties (see (Najim & Poznyak, 1994; Narendra & Thathachar, 1989)). In response to action \mathbf{x}_i , which is selected at time step k , the probabilities are updated as follows:

$$\begin{aligned} p_i(n+1) &= p_i(n) + \theta \cdot \beta(n) \cdot (1 - p_i(n)) \\ p_j(n+1) &= p_j(n) - \theta \cdot \beta(n) \cdot p_j(n), \text{ if } i \neq j \end{aligned} \quad (6.4)$$

being θ a learning rate parameter and $0 < \theta < 1$ and $\beta \in [0, 1]$ the reinforcement signal; $\beta = 1$ indicates the maximum reward and $\beta = 0$ is a null reward. Eventually, the probability of successful actions will increase to become close to unity. In case that a single and foremost successful action prevails, the automaton is deemed to have converged.

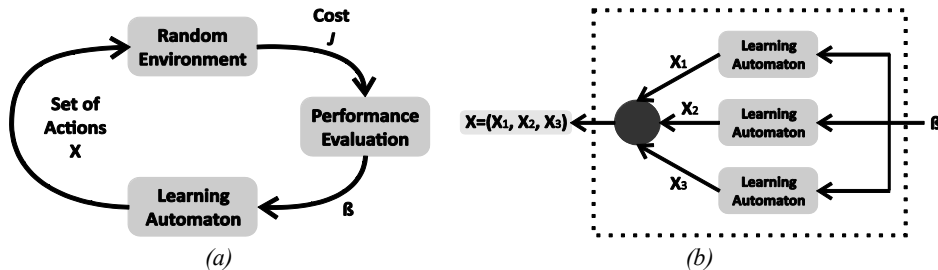


Fig. 6.1. (a) Reinforcement learning system and (b) Interconnected automata.

With a large number of discrete actions, the probability of selecting any particular action becomes low and the convergence time can become excessive. To avoid this, LA can be connected in a parallel setup as the one shown in Figure 6.1(b). Each automaton operates a smaller number of actions and the 'team' works together in a co-operative manner. This scheme can also be used where multiple actions are required.

Discrete stochastic LA can be used to determine global optimal parameters for optimization applications within multi-modal mean-square error surfaces. However, the discrete nature of the automata requires the discretization of a continuous parameter space while the quantization level tends to reduce the convergence rate. Therefore, a sequential approach is adopted for the CARLA implementation (Frost, 1998), overcoming the problem by means of an initial coarse quantization. The method may be refined again by using a re-quantization around the most successful action later on.

6.3.1. CARLA Algorithm

The continuous action reinforcement learning automata (CARLA) is developed as an extension of the discrete stochastic LA for applications involving searching of continuous action space in a random environment (Howell & Gordon, 2001). Several CARLA can be connected in parallel similarly to discrete automata (Figure 6.1(b)), in order to search multidimensional action spaces. Although the interconnection of the automata is through the environment, no direct inter-automata communication exists.

The automaton's discrete probability distribution is replaced by a continuous probability density function which is used as the basis for action selection. It operates a reward/inaction learning rule similar to the discrete LA shown in Equation (6.4). Successful actions receive an increase on the probability of being selected in the future via a Gaussian neighborhood function which augments the probability density in the vicinity of such successful action.

Table 6.1 shows the generic pseudo-code for the CARLA algorithm. The initial probability distribution can be chosen as being uniform over a desired range. After a considerable number of iterations, it converges to a probability distribution with a global maximum around the best action value (Beigy & Meybodi, 2006).

CARLA Algorithm
Initialize the probability density function to a uniform distribution
Repeat
Select an action using its probability density function
Execute action on the environment
Receive cost/reward for previous action
Update performance evaluation function β
Update probability density function
Until stopping condition is reached.

Table 6.1. Generic pseudo-code for the CARLA algorithm.

If action x (parameter) is defined over the range (x_{\min}, x_{\max}) , the probability density function $f(x, n)$ at iteration n is updated according to the following rule:

$$f(x, n+1) = \begin{cases} \alpha \cdot [f(x, n) + \beta(n) \cdot H(x, r)] & \text{if } x \in (x_{\min}, x_{\max}) \\ 0 & \text{otherwise} \end{cases} \quad (6.5)$$

with α being chosen to re-normalize the distribution according to the following condition

$$\int_{x_{\min}}^{x_{\max}} f(x, n+1) dx = 1 \quad (6.6)$$

with $\beta(n)$ being again the reinforcement signal from the performance evaluation and $H(x, r)$ being a symmetric Gaussian neighborhood function centered on $r = x(n)$. It yields

$$H(x, r) = \lambda \cdot \exp\left(-\frac{(x-r)^2}{2\sigma^2}\right) \quad (6.7)$$

with λ and α being parameters that determine the height and width of the neighborhood function. They are defined in terms of the range of actions as follows:

$$\sigma = g_w \cdot (x_{\max} - x_{\min}) \quad (6.8)$$

$$\lambda = \frac{g_h}{(x_{\max} - x_{\min})} \quad (6.9)$$

The speed and resolution of learning are thus controlled by free parameters g_w and g_h . Let action $x(n)$ be applied to the environment at iteration n , returning a cost or performance index $J(n)$.

Current and previous costs are stored as a reference set $R(n)$. The median and minimum values J_{med} and J_{min} may thus be calculated by means of $\beta(n)$, which is defined as follows:

$$\beta(n) = \max \left\{ 0, \frac{J_{\text{med}} - J(n)}{J_{\text{med}} - J_{\text{min}}} \right\} \quad (6.10)$$

To avoid problems with infinite storage requirements and to allow the system to adapt to changing environments, only the last m values of the cost functions are stored in $R(n)$. Equation (6.10) limits $\beta(n)$ to values between 0 and 1 and only returns nonzero values for those costs that are below the median value. It is easy to understand how $\beta(n)$ affects the learning process as follows: during the learning, the performance and the number of selecting actions can be wildly variable, generating extremely high computing costs. However, $\beta(n)$ is insensitive to such extremes and to high values of $J(n)$ resulting from a poor choice of actions. As the learning continues, the automaton converges towards more worthy regions of the parameter space as such actions are chosen to be evaluated more often. When more of such responses are being received, J_{med} gets reduced. Decreasing J_{med} in $\beta(n)$ effectively enables the automaton to refine its reference around better responses (previously received), and hence resulting in a better discrimination between selected actions.

In order to define an action value $x(n)$ which has been associated to a given probability density function, an uniformly distributed pseudo-random number $z(n)$ is generated within the range of $[0,1]$. Simple interpolation is thus employed to equate this value to the cumulative distribution function:

$$\int_{x_{\min}}^{x(n)} f(x, n) dx = z(n) \quad (6.11)$$

For implementation purposes, the distribution is stored at discrete points with an equal inter-sample probability. Linear interpolation is used to determine values at intermediate positions (see full details in (Howell & Gordon, 2001)).

6.4. Implementation

Four different pixel classes are used to segment the images. The idea is to show the effectiveness of the algorithm and its performance against other algorithms solving the same task. The implementation can easily be transferred to cases with a greater number of pixel classes.

To approach the histogram of an image by 4 Gaussian functions (one for each pixel class), it is necessary to calculate the optimum values of the 3 parameters (P_i , μ_i and σ_i) for each Gaussian

function (in this case, 12 values according to Equation (6.2)). This problem can be solved by optimizing equation 3, considering that function $p(x)$ gathers 4 Gaussian functions.

The parameters to be optimized are summarized in Table 6.2, with k_p^i being the parameter representing the a priori probability (P), k_σ^i holding the variance (σ) and k_μ^i representing the expected value (μ) of the Gaussian function i .

In the LA optimization, each parameter is considered like an Automaton which is able to choose actions. Such actions correspond to values assigned to the parameters by a probability distribution within the interval. All intervals considered in this work are defined as $k_p^i \in [0,0.5]$, $k_\sigma^i \in [0,128]$, and $k_\mu^i \in [0,255]$.

For this 12-dimensional problem, 12 different automata will be created to represent parametric approach of the corresponding histogram. One of the main advantages of the LA algorithm regarding multi-dimensional problems is that the automata are coupled only through the environment, thus each automaton operates independently during optimization.

Thus, at each instant n , each automaton chooses an action according to their probability distribution which can be represented in a vector $A(n)=\{k_p^1, k_\sigma^1, k_\mu^1, \dots, k_p^4, k_\sigma^4, k_\mu^4\}$. This vector represents a certain approach to the histogram. Then, the quality of the approach is evaluated (according to Equation (6.3)) and converted into a reinforcement signal $\beta(n)$ (through Equation 6.10). After the reinforcement value $\beta(n)$ is defined as a product of the elected approach $A(n)$, the distribution of probability is updated for $n+1$ of each automaton (according to the Equation (6.5)). To simplify parameters in Equation (6.8) and (6.9), they will take the same value for the 12 automata, such that $g_w = 0.02$ and $g_h = 0.3$. In this work, the optimization process considers a limit up to 2000 iterations.

The optimization algorithm can thus be described as follows:

i	Set iteration number $n=0$.
ii	Define the action set $A(n)=\{k_p^1, k_\sigma^1, k_\mu^1, \dots, k_p^4, k_\sigma^4, k_\mu^4\}$ such that $k_p^i \in [0,0.5]$, $k_\sigma^i \in [0,128]$ and $k_\mu^i \in [0,255]$.
iii	Define probability density functions at iteration n : $f(k_p^i, n)$, $f(k_\sigma^i, n)$ and $f(k_\mu^i, n)$
iv	Initialize $f(k_p^i, n)$, $f(k_\sigma^i, n)$ and $f(k_\mu^i, n)$ as a uniform distribution between the defined limits.
v	Repeat while $n \leq 2000$
(a)	Using a pseudo-random number generator for each automaton, select $z_p^i(n)$, $z_\sigma^i(n)$ and $z_\mu^i(n)$ uniformly between 0 and 1.
(b)	Select $k_p^i \in [0,0.5]$, $k_\sigma^i \in [0,128]$ and $k_\mu^i \in [0,255]$ where the area under the probability density function is $\int_0^{k_p^i(n)} f(k_p^i, n) = z_p^i(n)$, $\int_0^{k_\sigma^i(n)} f(k_\sigma^i, n) = z_\sigma^i(n)$ and $\int_0^{k_\mu^i(n)} f(k_\mu^i, n) = z_\mu^i(n)$.
(c)	Evaluate the performance using Eq. (6.3).
(d)	Obtain the minimum, J_{\min} , and median, J_{med} of $J(n)$.
(e)	Evaluate $\beta(n)$ via Eq. (6.10).

(f)	Update the probability density functions $f(k_p^i, n)$, $f(k_\sigma^i, n)$ and $f(k_\mu^i, n)$ using Eq. (6.5).
(g)	Increment iteration number n .

The learning system searches within the 12-dimensional parameter space aiming for reducing the values of J in Equation (6.3).

The final step is to determine the optimal threshold values T_i . In this case, the pixel classification corresponds to the maximum likelihood (ML) estimator. The classes can be determined by simple thresholding following standard methods, just as it is illustrated in the Figure 6.2.

Parameters			Gaussian
k_p^1	k_σ^1	k_μ^1	1
k_p^2	k_σ^2	k_μ^2	2
k_p^3	k_σ^3	k_μ^3	3
k_p^4	k_σ^4	k_μ^4	4

Table 6.2. Parameters to be optimized by the LA algorithm.

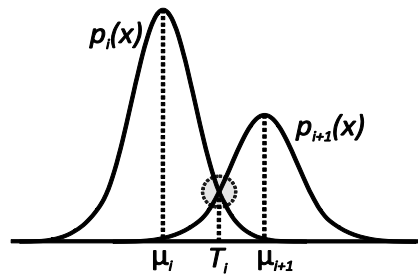


Figure 6.2. Thresholding points determination.

6.5. Experimental Results

This section presents the experimental work with the LA algorithm. The discussion is divided into two parts: the first one shows the performance of the proposed LA algorithm while the second discusses on a comparison between the LA segmentator, the EM algorithm and the Levenberg-Marquardt method.

6.5.1. LA algorithm performance in image segmentation

This section presents two experiments to analyze the LA's performance considering a segmentation mixture of four classes while the original histogram of the image is approached by the LA method. In order to test consistency, 10 independent repetitions are made for each experiment.

The first test considers the histogram shown by Figure 6.3(b) while Figure 6.3(a) presents the original image. After applying the LA algorithm (as it is explained in the previous section), a minimum is obtained (Equation (6.3)), as the point is defined by $k_p^1=0.094$, $k_\sigma^1=6$, $k_\mu^1=15$, $k_p^2=0.1816$, $k_\sigma^2=29$, $k_\mu^2=63$, $k_p^3=0.2733$, $k_\sigma^3=10$, $k_\mu^3=93$, $k_p^4=0.4503$, $k_\sigma^4=30$, and $k_\mu^4=163$. The values of such parameters define four different Gaussian functions which are clearly visible in Figure 6.4. The original histogram and its approximation by the Gaussian mixture are visually compared in Figure 6.5.

The evolution of the probability density parameters which in turn represent the expected values $f(k_\mu^1, n)$, $f(k_\mu^2, n)$, $f(k_\mu^3, n)$ and $f(k_\mu^4, n)$ of the Gaussian functions are shown in Figure 6.6. It can be seen that most of the convergence is achieved at the first 1050 iterations, as subsequent steps yield a bit of sharpening in the distribution's shape. The final highest probability value obtained from the distribution ($n=2000$) corresponds to the final parameter value.

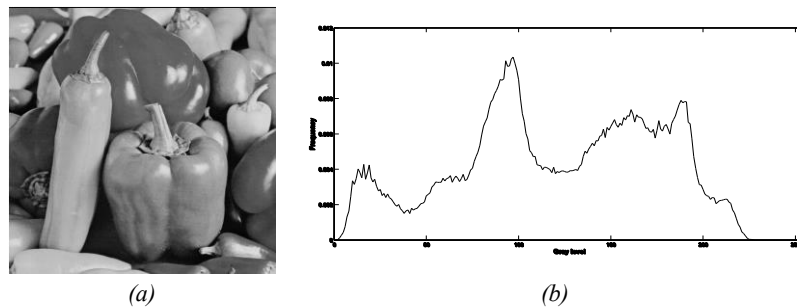


Figure 6.3. (a) Original image used on the first experiment, (b) and its histogram.

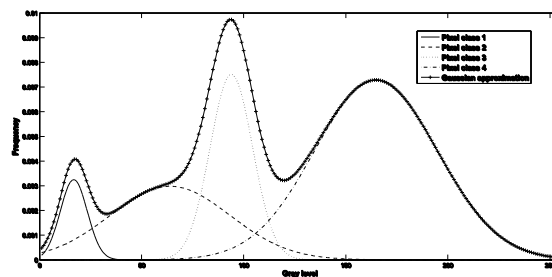


Figure 6.4. Gaussian functions obtained by LA.

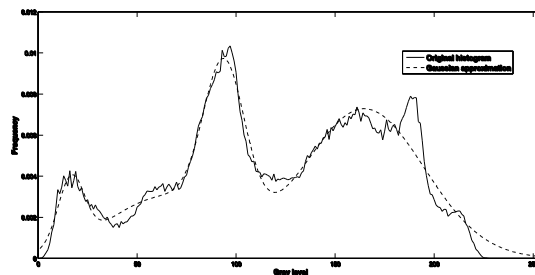


Figure 6.5. Comparison between the original histogram and its approach.

From the Gaussian functions obtained by LA in Figure 6.4, the threshold values T_i are calculated using well-known methods. Considering such values, the segmented image is shown in Figure 6.7.

For the second experiment the image shown in Figure 6.8 is tested. The method aims to segment the image into four different classes using the LA approach. After executing the algorithm according to the parameters defined in Section 6.4, the resulting Gaussian functions approximating the histogram are shown in Figure 6.9(a).

The comparison between the original image and its Gaussian approximation is shown in Figure 6.9(b). It is clear that the algorithm approaches each of all the pixel concentrations distributed within the histogram but the first one, which is presented approximately around the intensity value seven. This effect shows that the algorithm discards the smallest pixel accumulation as it prefers to cover classes that contribute to generate smaller errors during optimization of the Equation (6.3). Such results can improve if five-pixel classes were used instead.

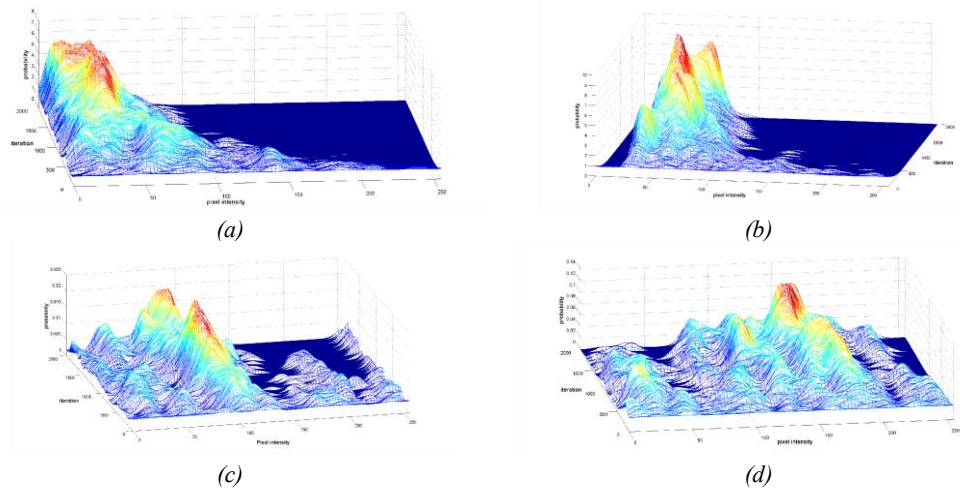


Figure 6.6. Evolution of the probability densities parameters and their expected values of the Gaussian functions (a) $f(k_{\mu}^1, n)$, (b) $f(k_{\mu}^2, n)$, (c) $f(k_{\mu}^3, n)$ and (d) $f(k_{\mu}^4, n)$.



Figure 6.7. Image segmented in four classes by the LA method.

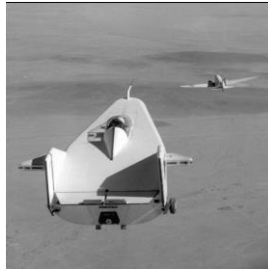


Figure 6.8. Original image used in the second experiment.

From the Gaussian mixture obtained by the LA method (Figure 6.9(a)), the threshold values T_i are calculated again using well-known methods. Figure 6.10 shows the segmented image after the detection task. Figure 6.11 shows the separation of each class after applying the LA algorithm.

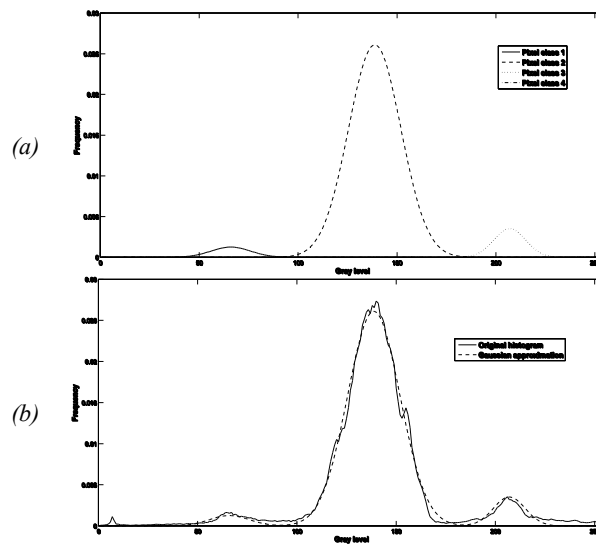


Figure 6.9. (a) Gaussian functions obtained by the LA algorithm and (b) its comparison to the original histogram.



Figure 6.10. Segmentation obtained by LA.

6.5.2. Comparing the LA algorithm vs. the EM and LM methods

This section discusses on the comparison between LA and other algorithms such as the EM algorithm and one Levenberg-Marquardt (LM) method. The discussion is focused on the following issues: first, sensitivity to the initial conditions; second, singularities and third, convergence and computational costs.

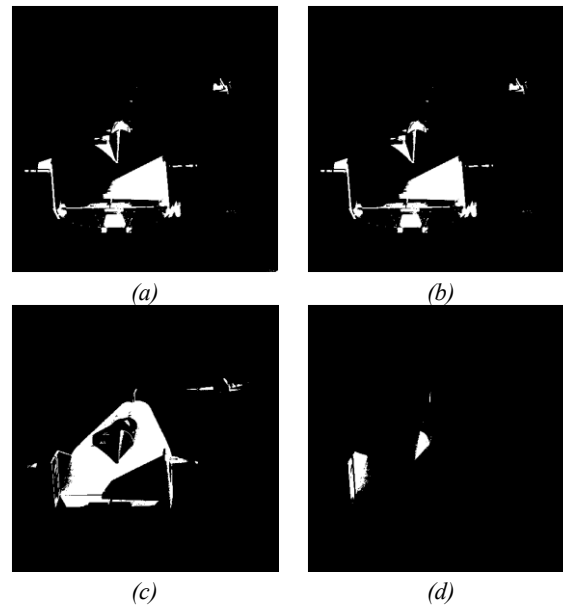


Figure 6.11. Class separation as it is produced by the LA algorithm. (a) Pixel class 1, (b) pixel class 2, (c) Pixel class 3, and (d) Pixel class 4.

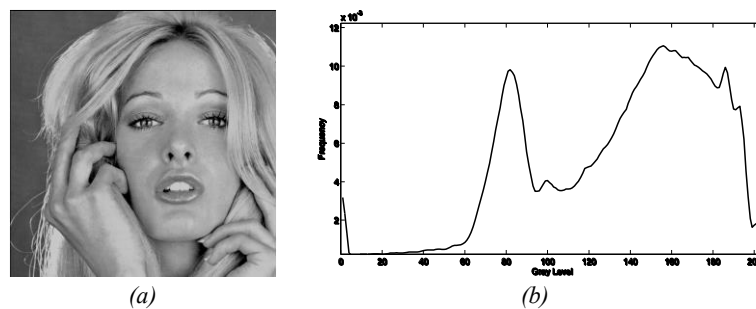


Figure 6.12. (a) Original image used for the comparison on initial conditions and (b) its corresponding histogram.

a) Sensitivity to the initial conditions. In this experiment, initial values for all methods are initialized in different values while the same histogram is considered for the approximation task. The final parameters representing the Gaussian mixture after convergence are reported. Figure 6.12(a) shows the image used in this comparison while Figure 6.12(b) pictures the histogram. All experiments are conducted several times in order to assure consistency. Only two different initial states with the highest variation are reported in Table 6.3. Likewise, Figure 6.13 shows the obtained segmented images considering the two initial conditions reported by Table 6.3. In the LA case, the algorithm does not require initialization as it works with random initial values; however, in order

to assure a valid comparison, the same initial values are considered for the EM, the LM and the LA method.

By analyzing the information in Table 6.3, the sensitivity of the EM algorithm to initial conditions becomes evident. Figure 6.13 shows a clear pixel misclassification in some sections of the image as a consequence of such sensitivity.

Parameters	Initial condition 1	EM	LM	LA	Initial condition 2	EM	LM	LA
k_{μ}^1	40.6	33.13	32.12	32.10	10	20.90	31.80	32.92
k_{μ}^2	81.2	81.02	82.05	82.01	100	82.78	80.85	82.12
k_{μ}^3	121.8	127.52	127	126.95	138	146.67	128	127.01
k_{μ}^4	162.4	167.58	166.80	166.72	200	180.72	165.90	166.62
k_{σ}^1	15	25.90	25.50	25.51	10	18.52	20.10	25.11
k_{σ}^2	15	9.78	9.70	9.66	5	12.52	9.81	9.68
k_{σ}^3	15	17.72	17.05	17.10	8	20.5	15.15	17.12
k_{σ}^4	15	17.03	17.52	17.55	22	10.09	18.00	17.15
k_p^1	0.25	0.0313	0.0310	0.312	0.20	0.0225	0.0312	0.312
k_p^2	0.25	0.2078	0.2081	0.2078	0.30	0.2446	0.2079	0.2088
k_p^3	0.25	0.2508	0.2500	0.2510	0.20	0.5232	0.2502	0.2500
k_p^4	0.25	0.5102	0.5110	0.5103	0.30	0.2098	0.5108	0.5103

Table 6.3. Comparison between the EM, the LM and the LA algorithm, considering two different initial conditions.

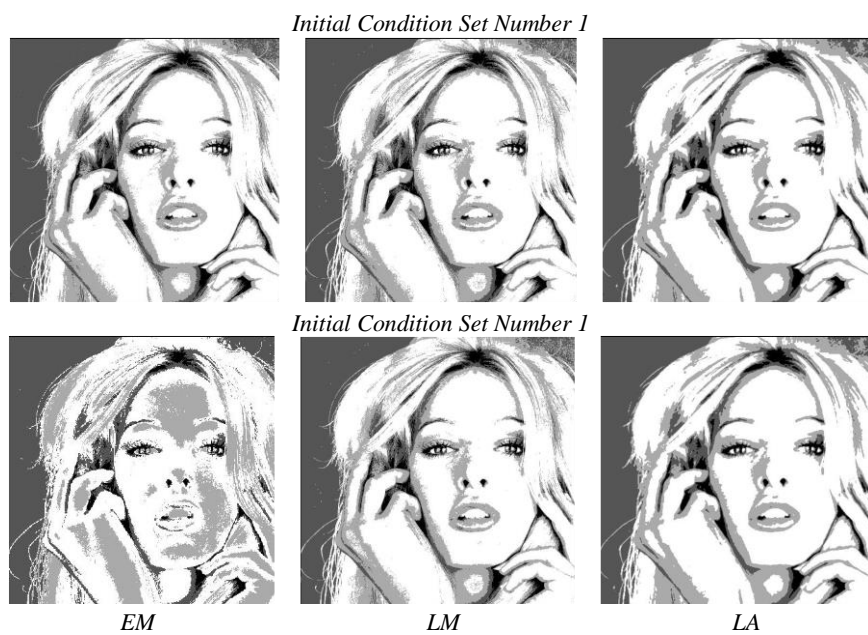


Figure 6.13. Segmented images after applying the EM, the LM and the LA algorithm with different initial conditions.

b) Singularities. The experiment aims to test the LA performance under certain circumstances on which it is well-reported in the literature (Gupta & Sortrakul, 1998; Park & Ozeki, 2009) that the EM and the LM have underperformed. Two cases are relevant to such purpose. First, the Gaussian variance is small or near to zero, i.e. big objects are present in the image with a homogeneous intensity value (Gupta & Sortrakul, 1998). Second, the LM algorithm exhibits a slow convergence when the Gaussians are overlapped (Olsson et al., 2007; Park & Ozeki, 2009). For both cases, the EM method never reaches convergence. The benchmark image and its histogram are shown in Figure 6.14.

Case 1. The experiment shows the lack of convergence of the EM algorithm when a small or near to zero Gaussian variance is considered. The test consists on using all the algorithms to obtain the Gaussian mixture parameters that approximate the histogram shown in the Figure 6.14(b). It is evident that only 4 classes are relevant. In order to assure consistency, the experiment is repeated over 100 times with different initial conditions. The results show that the EM method never converge to an acceptable value whatsoever. Table 6.4 shows the results for the LM and the LA algorithm as they are averaged over 100 experiments.

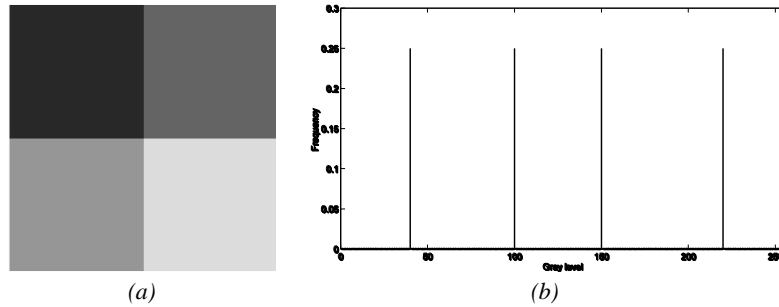


Figure 6.14. (a) Original image used by the singularity experiment, and (b) its histogram.

Parameters	LM	LA
\bar{k}_μ^1	42.6	40.1
\bar{k}_μ^2	98.3	99.89
\bar{k}_μ^3	153.7	150.05
\bar{k}_μ^4	220.1	220.01
\bar{k}_σ^1	7	0.05
\bar{k}_σ^2	12	0.07
\bar{k}_σ^3	5	0.10
\bar{k}_σ^4	0.3	0.03
\bar{k}_p^1	0.20	0.0313
\bar{k}_p^2	0.3	0.2078
\bar{k}_p^3	0.25	0.2508
\bar{k}_p^4	0.25	0.5102
iterations	997	1050

Table 6.4. Comparison between the LM and the LA algorithms using variances values close to zero.

By analyzing data in Table 6.4, it is clear that the LM and the LA algorithms are able to successfully segment the image shown in Figure 6.14(a). The LM method converges a little faster than the LA algorithm. However, it shows a sub-optimal approximation to a local minimum (see Figure 6.15(a)).

Case 2. This case analyzes the slow convergence of the LM method when the parameters of the Gaussian mixture are overlapped. For the experiment, the Gaussian's overlapping is caused by considering initial values falling on the same position. Although the results fully match with those in Case 1 (see Table 6.4), the differences on the required iterations are evident. For instance, the LA method requires nearly 1000 iterations while the LM method as much as 2300 iterations - averaging 100 experiments for both cases. The convergence speed in the LA method is clearly not affected by such singularity.

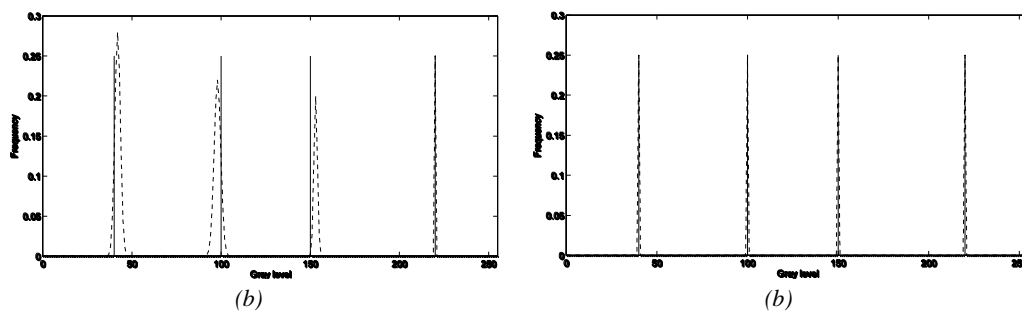


Figure 6.15. Graphical view of approximations using near zero variances with: a) the LM algorithm and b) the LA method.

Iterations	(a)	(b)	(c)	(d)
Time elapsed				
EM	1855 2.72s	1833 2.70s	1861 2.73s	1870 2.73s
LM	985 4.03s	988 4.04s	945 4.98s	958 4.98s
LA	970 1.51s	991 1.53s	951 1.48s	951 1.48s

Table 6.5. Iterations and time requirements of the EM, the LM and the LA algorithm as they are applied to segment benchmark images (see Figure 6.16).

c) Convergence and computational cost. The experiment aims to measure the number of required steps and the computing time spent by the EM, the LM and the LA algorithm required to calculate the parameters of the Gaussian mixture in benchmark images (see Figure 6.16(a-c)). All experiments consider four classes. Table 6.5 shows the averaged measurements as they are obtained from 20 experiments. It is evident that the EM is the slowest to converge (iterations) and the LM shows the highest computational cost (time elapsed) because it requires complex Hessian approximations. On the other hand, the LA shows an acceptable compromise between its convergence time and its computational cost. Finally, Figure 6.16 below shows the segmented images as they are generated by each algorithm.

6.6. Conclusions

In this chapter, an automatic image multi-threshold approach based on Learning Automata (LA) is proposed. The segmentation process is considered to be similar to an optimization problem. The algorithm approximates the 1-D histogram of a given image using a Gaussian mixture model whose parameters are calculated through the LA algorithm CARLA. Each Gaussian function approximating the histogram represents a pixel class and therefore one threshold point.

Experimental evidence shows that LA algorithm has an acceptable compromise between its convergence time and its computational cost when it is compared to the Expectation-Maximization (EM) method and the Levenberg-Marquardt (LM) algorithm. Additionally, the LA algorithm also exhibits a better performance under certain circumstances (singularities) on which it is well-reported in the literature (Gupta & Sortrakul, 1998; Park & Ozeki, 2009) that the EM and the LM have underperformed. Two cases are reported: First, when Gaussian variance is small or near to zero (i.e. big objects are presented on the image with a homogeneous intensity value). Second, it is when the parameters of the Gaussian mixture are overlapped. Finally, the results have shown that the stochastic search accomplished by the LA method shows a consistent performance with no regard of the initial value and still showing a greater chance to reach the global minimum.



Figure 6.16. Original benchmark images a)-c), and segmented images obtained by the EM, the LM and the LA algorithms.

Chapter 7

Fuzzy-based System for Corner Detection

Corner detection is an important task in computer vision problems due to the complexity of determining the shape of different regions within an image. Real-life image data are always inexact due to inherent uncertainties that may arise from the imaging capture process such as defocusing, illumination changes, noise, etc. Therefore, the localization and detection of corners has become a difficult task under research, in order to accomplish the detection process under such imperfect situations. On the other hand, Fuzzy systems are well known for their efficient handling capacities when they face uncertainty and incompleteness. Fuzzy systems use modelling concepts in the same way as a human do. Under these circumstances, corners could be modelled by means of linguistic rules. This chapter presents a corner detection algorithm which employs Fuzzy reasoning. The robustness of the presented algorithm is compared to well-known conventional corner detectors and its performance is also tested over a number of benchmark images to illustrate the efficiency of the algorithm under uncertainty.

7.1. Introduction

The human visual system has a highly developed capability for detecting many classes of patterns including visually significant arrangements of image elements. From the psychovisual aspect, points representing high curvature are one of the dominant classes of patterns that play an important role in almost all real-life image analysis applications (Fischler & Wolf, 1994; Loupias & Sebe, 2000; Lowe, 1985). These points encode a significant amount of shape information. Corners are generally formed at the junction of different edge segments which may be the meeting (or crossing) of two edges.

Cornerness of an edge segment depends solely on the curvature formed at the meeting point of two-line segments. Corner detection is one of the fundamental tasks in computer vision and it can be regarded as a special type of feature segmentation. Extracted corners can be used for measurement and/or recognition purposes.

A large number of algorithms already exist in the literature. In particular, corner detection on gray level images can be classified into two main approaches. In the first approach, the gray level image is first converted into its binary version for extraction of boundaries using some thresholding technique. After a successful extraction of boundaries, the corners or the high curvature points are detected using directional codes or other polygonal approximation techniques (Freeman & Davis, 1977).

In the second approach, the gray level image is taken directly as an input for corner detection. In this paper, the discussion is restricted to the second approach only. Most of the general-purpose detectors based on gray level, use either a topology-based or an auto-correlation-based approach. Topology based corner detectors, mainly use gradients and surface curvature to define the measure of cornerness. Points are marked as corners, if the value of cornerness exceeds some predefined threshold condition. Alternatively, a measure of curvature can be obtained using auto-

correlation (Kitchen & Rosenfeld, 1982; Rattarangsi & Chin, 1992; Rosenfeld & Johnston, 1973; Teh & Chin, 1989; Zheng, et al., 1999).

There exist several classical corner detection algorithms for estimating corner points. Such detectors are based on a local structure matrix which consists on the first partial derivatives of the intensity function. A clear example is the Harris feature point detector (Harris & Stephens, 1988), which is based on a comparison: the measure of the corner strength - which is defined by the method and is based on a local structure matrix - is compared to an appropriately chosen concrete threshold. Another well-known corner detector is the SUSAN (Smallest Univalued Segment Assimilating Nucleus) detector which is based on brightness comparison (Smith & Brady, 1997). It does not depend on image derivatives. The SUSAN area will reach a minimum while the nucleus lies on a corner point. The effectiveness of the above-mentioned algorithms is acceptable. Recent studies such as (Zou, et al., 2008) demonstrate that the Harris corner detector performs better for several circumstances in comparison to the SUSAN algorithm.

Data from natural images are always imprecise and noisy due to inherent uncertainties that may arise from the imaging process (such as defocusing, wide variations of illuminations, etc.). Thus, precise localization and detection of corners become difficult under such imperfect situations.

On the other hand, Fuzzy systems are well known for efficiently handling of impreciseness and incompleteness (Pal, et al., 2000; Yu, et al., 2007; Zadeh, 1965) due to imperfection of data. Therefore, it may result reasonable to model corner properties using a fuzzy rule-based system as they have been successfully applied to image processing in a wide variety of applications (Basak & Pal, 2005; Jacquey, et al., 2008; Karmakar & Dooley, 2002).

This chapter seeks to contribute to enhance the application of Fuzzy Logic to image processing, just as it has been proposed in (Russo, 1999). The method adopts a template-based rule-driven procedure and has been specifically developed to deal with topics related to image processing purposes. The presented method is able to address many different processing tasks (Kim, Lee, & Kweon, 2004; Liang & Looney, 2003; Tizhoosh, 2003) and to produce better results than classical methods when applied to some critical issues such as noise (Russo, 2004; Tizhoosh, 2003; Yüksel, 2007).

Only few Fuzzy approaches have specifically addressed the problem of corner detection for general purposes. Banerjee & Kundu (2008) have proposed an algorithm to extract significant gray level corner points. The measure of cornerness in each point is computed by means of the fuzzy edge strength and the gradient direction. Different corner Fuzzy-sets are obtained by considering different threshold values from the fuzzy edge map. However, the algorithm's main drawback is that it uses several feature detectors which operate at different stages, yielding a high computational load.

On the other hand, Várkonyi-Kóczy (1993) have proposed, a Fuzzy Corner Detector that employs a local structure matrix. It builds a continuous transient between the localized and not localized corner points. The algorithm uses a fuzzy pre-filter that improve the quality of the image under process. Despite both Fuzzy approaches show a good performance, they demand an expensive computing load in comparison to other classical algorithms such as the Harris method or SUSAN.

This chapter presents a new robust algorithm to extract significant gray level corner points. The method is derived from a Fuzzy-rule approach which aims to detect corners even under complex conditions. In the presented approach, the measure of "cornerness" for each pixel in the image is computed by Fuzzy rules (represented as templates) which are applied to a set of pixels belonging to a rectangular window. As the algorithm scans each pixel of the image at a time, a new pixel of

the resulting image is generated by Fuzzy reasoning. Hence, the possible uncertainty contained in the window-neighborhood is handled by using an appropriate rule base (template set).

Experimental evidence shows the effectiveness of such method for detecting corners under several conditions. A comparison between one state-of-the-art Fuzzy-based method (Banerjee & Kundu, 2008) and the Harris algorithm (Harris & Stephens, 1988) demonstrates the performance of the Fuzzy-based method.

This chapter is organized as follows: Section 7.2 briefly describes the mathematical approach and the fuzzy model used in this work. Section 7.3 describes the experimental results while Section 7.4 describes the performance comparison regarding to other methods. On the other hand, Section 7.5 offers some conclusions about the development and performance of this technique.

7.2. Fuzzy rule-based System

7.2.1. Fuzzy System

Most of the approaches for corner detection are easy to implement and demand a low computational load. However, their effective operation largely relies on the fact that noisiness must be limited. In this section, a more robust technique is proposed. The new procedure is set to deliver a better performance for noisy environments. The Fuzzy system is simple to implement and still fast in computation if it is compared to some existing Fuzzy methods (Banerjee & Kundu, 2008; Várkonyi-Kóczy, 1993). Also, it can be easily extended to detect other features. In the proposed approach, the fuzzy rules are applied to a set of pixels belonging to a rectangular $N \times N$ window (usually 3×3 pixels), where the gray-level differences between the center pixel $p_{m,n}$ and its surrounding pixels are computed and stored within matrix E as follows:

$$E = \begin{bmatrix} p_{m,n} - p_{m-1,n-1} & p_{m,n} - p_{m-1,n} & p_{m,n} - p_{m-1,n+1} \\ p_{m,n} - p_{m,n-1} & 0 & p_{m,n} - p_{m,n+1} \\ p_{m,n} - p_{m+1,n-1} & p_{m,n} - p_{m+1,n} & p_{m,n} - p_{m+1,n+1} \end{bmatrix} \quad (7.1)$$

where m and n represent the coordinates of the central pixel. If the neighborhood is a homogenous region, then E contains values near zero.

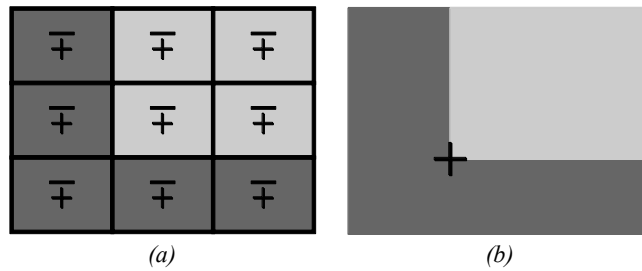


Fig. 7.1. Region shaping with respect to gray level differences: (a) the resulting template and (b) the real corner that originates the template.

In the case of corners, the matrix E possesses a specific configuration depending on the corner type. These divide E in two connected regions, one with positive (pixel type **A**) and another with negative (pixel type **B**) difference values (see Figure 7.1). The reasoning structure uses two different types of rules: the **THEN-rules** and the **ELSE-rules** (don't care conditions) respectively. Each THEN-rule includes a determined pixel configuration as antecedent and only one pixel as conse-

quent. Antecedents are related to a corner existence test and the consequent to its presence or absence. The rule-base gathers many Fuzzy rules (THEN-rules) and only one ELSE-rule (i.e. do-not-care rule). Therefore, only relevant rules (i.e. configurations) are formulated as THEN-rules while other not important configurations may be handled as a group of ELSE-rules.

The set of THEN-rules lies on the very core of the algorithm. The rules must deliver successful structure detection, i.e. corners in this case, while still cancelling other inconsistencies such as noise. Such tradeoff may be solved by using a reduced set of rules (configurations) which in turn represent the minimum number in order to coherently detect the structure as it is required by a given application. Such procedure allows dealing with noisy pixels or imprecision.

The proposed corner detector considers twelve THEN-rules that represent the same number of possible corner configurations and only one ELSE-rule as it is graphically explained by Fig. 7.2. It may be also possible to consider some other corner configurations. However, it may reduce the algorithm's ability to deal with noise or uncertainty (Russo, 1999; Tizhoosh, 2003; Yüksel, 2007). Despite using a reduced rule base, the performance in the detection process can be considered acceptable when it is compared to other algorithms solving the same task. The rule base (THEN-rules and ELSE-rule) supporting the detector algorithm is shown in Table 7.1.

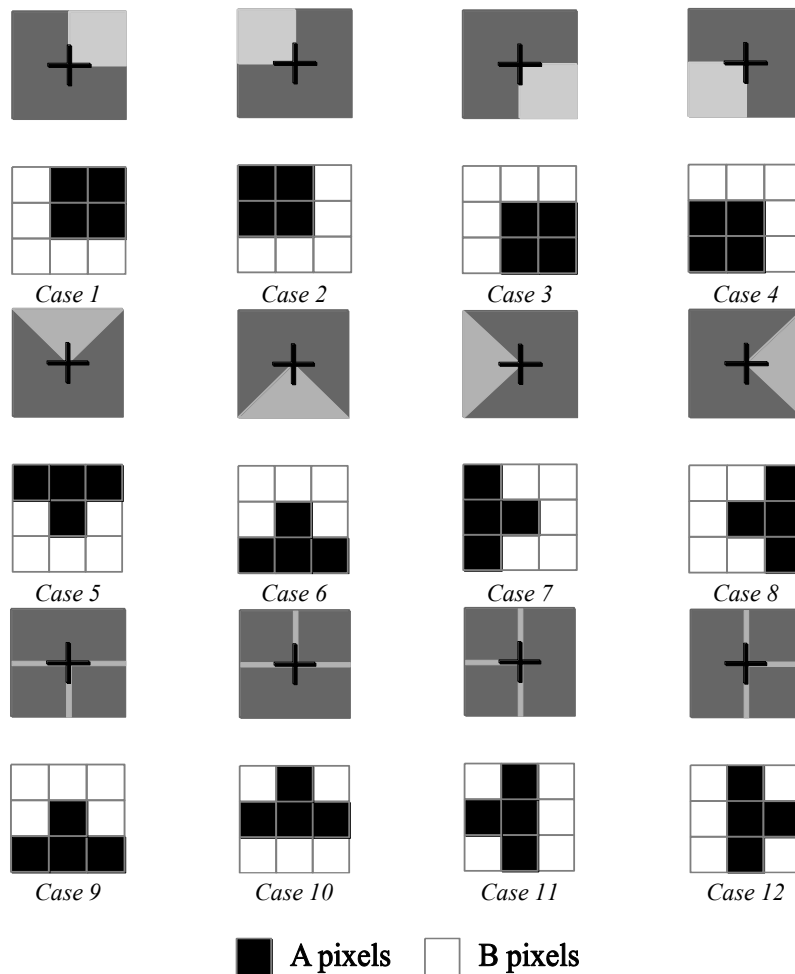






Fig. 7.2. Different corner cases to be considered for building the fuzzy rules. The image region containing the corner is shown in the upper section while the resulting 3x3 template is shown below each case. Each rule has the following form:

<p>If the corner structure in E possesses positive elements</p> <p>and the opposite region possesses negative elements,</p> <p>then the pixel represent a corner,</p> <p>else the pixel does not represent a corner</p>	   	(7.2)
--	--	-------

The principle can be explained as follows: If one region of the neighborhood, according to any of the twelve cases, contains positive/negative differences with respect to the center pixel, and if any other region contains the opposite (negative/positive) differences with respect to the center pixel, then the center pixel is a corner (see Fig. 7.2). The procedure can be considered as the evaluation of each one of the 12 different THEN-rules (configurations), yielding two auxiliary matrices E^p and E^n as follows:

$$E^p(i, j) = \begin{cases} 1 & \text{if } E(i, j) \geq 0 \\ 0 & \text{else} \end{cases} \tag{7.3}$$

$$E^n(i, j) = \begin{cases} 1 & \text{if } E(i, j) < 0 \\ 0 & \text{else} \end{cases} \tag{7.4}$$

where i, j represents the row and column of the matrix E ($i, j \in (1, 2, 3)$), Eq. (7.1).































<i>Rule 1</i>		<i>Rule 2</i>		<i>Rule 3</i>		<i>Rule 4</i>	
IF	THEN	IF	THEN	IF	THEN	IF	THEN
							
<i>Rule 5</i>		<i>Rule 6</i>		<i>Rule 7</i>		<i>Rule 8</i>	
IF	THEN	IF	THEN	IF	THEN	IF	THEN
							
<i>Rule 9</i>		<i>Rule 10</i>		<i>Rule 11</i>		<i>Rule 12</i>	
IF	THEN	IF	THEN	IF	THEN	IF	THEN
							
<i>Rule 13</i>							
IF				THEN			
						
Otherwise							
Corner				No Corner			
							

Table 7.1. The rule base (THEN-rules and ELSE-rule) supporting the corner detector algorithm.

For the case that all elements of E^p / E^n are ones (meaning all elements of $E(i, j)$ are positives or negatives), it is possible to construct regions A and B within the window-neighborhood according to the existing relative differences. Thus, the values of E^p and E^n can be recalculated as follows:

$$\begin{aligned} E^p(i, j) &= \begin{cases} 1 & \text{if } E(i, j) \leq t_h \\ 0 & \text{else} \end{cases} \\ E^n(i, j) &= \begin{cases} 1 & \text{if } E(i, j) > t_h \\ 0 & \text{else} \end{cases} \end{aligned} \quad (7.5)$$

For all the elements of E^p being ones, and

$$\begin{aligned} E^p(i, j) &= \begin{cases} 1 & \text{if } E(i, j) \geq -t_h \\ 0 & \text{else} \end{cases} \\ E^n(i, j) &= \begin{cases} 1 & \text{if } E(i, j) < -t_h \\ 0 & \text{else} \end{cases} \end{aligned} \quad (7.6)$$

For all the elements of E^n being ones, t_h is a threshold that controls the sensitivity of the considered differences. Typical values for t_h normally fall into the interval [5,35]. The lowest value of 5 would yield a higher detector's sensitivity which may detect a great number of corners corresponding to noisy intensity changes which are commonly found in images.

On the other hand, a maximum value of 35 would detect corners matching to a significant difference between several objects in the structure, i.e. object whose pixels may be considered as being connected.

Although the selection of the best value for t_h clearly depends on the particular application, a good compromise can be obtained by taking a value on approximately half the overall interval, i.e. $t_h = 20$.

The membership values $\mu_c(m, n)$ (where $c=1,2,\dots,12$) are computed depending on the corner types (see Fig. 7.2). Such values represent the antecedents of each employed THEN-rule. They can be calculated as follows:

$$\begin{aligned} \mu_c(m, n) &= \frac{1}{20} \max \left[\left(\sum_{ij \in A} E^p(i, j) \right) \cdot \left(\sum_{ij \in B} E^n(i, j) \right), \right. \\ &\quad \left. \left(\sum_{ij \in B} E^p(i, j) \right) \cdot \left(\sum_{ij \in A} E^n(i, j) \right) \right] \end{aligned} \quad (7.7)$$

Expression (7.7) considers a normalization factor equal to 20 which represents the maximum possible value, i.e. the highest product of the multiplication among the pixels between E^p and E^n . Hence, the membership value $\mu_c(m, n)$ falls between 0 and 1.

Equation (7.7) can be considered as the numerical implementation of the generic rule previously defined by Eq. (7.2).

If Rule 1 (case 1) is considered as an example, the expressions corresponding to expression (7.7) would thus be:

$$\begin{aligned}
 \sum_{ij \in A} E^p(i, j) &= E^p(1, 2) + E^p(1, 3) + E^p(2, 2) + E^p(2, 3) \\
 \sum_{ij \in B} E^n(i, j) &= E^n(1, 1) + E^n(2, 1) + E^n(3, 1) + E^n(3, 2) + E^n(3, 3) \\
 \sum_{ij \in B} E^p(i, j) &= E^p(1, 1) + E^p(2, 1) + E^p(3, 1) + E^p(3, 2) + E^p(3, 3) \\
 \sum_{ij \in A} E^n(i, j) &= E^n(1, 2) + E^n(1, 3) + E^n(2, 2) + E^n(2, 3)
 \end{aligned} \tag{7.8}$$

Analogously to (7.8), membership values $\mu_2(i, j), \dots, \mu_{12}(i, j)$ for other rules (cases) can be calculated. Finally, the 12 fuzzy rules can be added into a single fuzzy value using the **max** (maximum) operator. The final fuzzy value represents the linguistic meaning of cornerness yielding:

$$\mu_{\text{cornerness}}(m, n) = \max(\mu_1(m, n), \mu_2(m, n), \dots, \mu_{12}(m, n)) \tag{7.9}$$

The pixels whose value $\mu_{\text{cornerness}}(m, n)$ are near to one, belong to a feature similar to a corner, while values near to zero would represent any other feature.

7.2.2. Robustness

This kind of corner detection clearly differs from other classical procedures in several ways. Conventional corner detectors look usually for the explicit corner location by means of detecting the zero-crossing of derivatives in different directions. On the contrary the presented approach detects the entire area where the corner could lie.

In particular, gradient-based methods are normally highly sensitive to the noise in real images and being mainly affected by the impulsive noise. Also, most of the corner detection algorithms incorporate several pre-filters (Harris & Stephens, 1988; Lowe, 1985; Moravec, 1997; Smith & Brady, 1997), which allow attenuation but do not eliminate impulsive noise.

On the other hand, Fuzzy detectors allow corner marking despite noisy environments either by implementing Fuzzy pre-filtering that eliminates uncertainty on the image or by incorporating Fuzzy sets for modeling imprecision (Banerjee & Kundu, 2008).

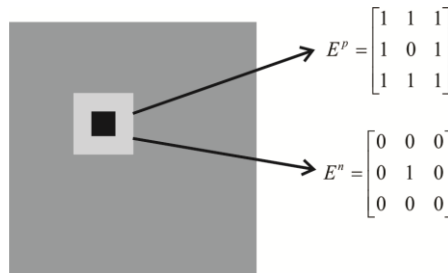


Fig. 7.3. The effect of the impulsive noise in matrices E^+ and E^- . Matrices E^+ and E^- would contain only ones or zeros depending on the gray-level difference.

The method presented in this chapter considers vagueness due to noise and grayness ambiguity to be handled by the Fuzzy rules introduced in expression (7.2). If the image of Figure 7.3 is consid-

ered, with a pixel holding a gray value different from its neighbors is located within a homogeneous region. This situation can be considered as impulsive noise.

Under these circumstances, matrices E^p and E^p would contain only ones or zeros depending on the gray-level difference. Therefore, the values used to calculate the membership functions in expression (7.7), for any of the twelve cases, would yield

$$\left(\sum_{ij \in R_p} E^p(i, j) \right) \cdot \left(\sum_{ij \in R_n} E^p(i, j) \right) \approx 0 \quad (7.10)$$

$$\left(\sum_{ij \in R_n} E^p(i, j) \right) \cdot \left(\sum_{ij \in R_p} E^n(i, j) \right) \approx 0 \quad (7.11)$$

Now, considering the values from expressions (7.10) and (7.11) and a noisy pixel, the resulting value of its cornerness can be calculated by expression (7.9) as $\mu_{cornerness}(i, j) \approx 0$.

The impulsive noise is thus classified by the Fuzzy system as a homogeneous region. In the same way, the central pixel would not be marked as corner for cases not considered in Table 7.2 which normally represent noisy configurations. It is mainly because the inference system works with ELSE-rules.

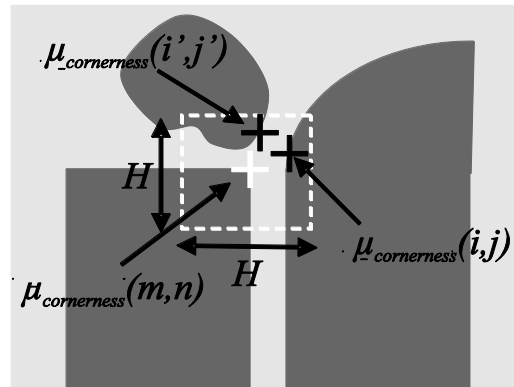


Fig. 7.4. Neighborhood method for corner selection. Example, where $\mu_{cornerness}(m,n)$ represents the cornerness of the pixel currently under evaluation, by assuming $\mu_{cornerness}(m,n) \geq t_c$. Inside the window $H \times H$ that has been established around it, there exist other two pixels $p_{i,j}$ and $p_{i',j'}$, whose values $\mu_{cornerness}(i,j)$ and $\mu_{cornerness}(i',j')$ are lower than $\mu_{cornerness}(m,n)$. Therefore, a point $p_{m,n}$ can thus be considered as a corner within the image.

7.2.3. Corner Selection

In order to detect corners, it would be enough to choose an appropriate threshold t_c . If $\mu_{cornerness}(m,n) \geq t_c$, then the pixel $p_{m,n}$ can be assumed as such. Under these assumptions, the value t_c must be selected as close to 1 as it is likely to assure that pixel $p_{m,n}$ may be a corner. However, a more convenient approach is to choose a small threshold value t_c whose value allows detecting a wider number of corners despite a higher uncertainty.

The corner selection process can therefore be explained as follows:

For each pixel, if $\mu_{cornerness}(m,n) \geq t_c$, a neighborhood of $H \times H$ dimension is established around it (commonly $H > N$). The pixel $p_{m,n}$ is thus selected as a corner if its value $\mu_{cornerness}(m,n)$ is maximum within the neighborhood $H \times H$, otherwise it does not represent a corner point.

7.3. Experimental Results

Different sorts of images have been tested in order to analyze the performance of the method for corner detection. Such benchmark set includes image alterations such as blurring, illumination change, impulsive noise etc.

Table 7.2 presents the parameters of the presented algorithm used in this chapter. Once they have been determined experimentally, they are kept for all the test images through all experiments.

t_h	t_c	H
20	0.7	10

Table 7.2. Parameter setup for the proposed corner detector

First, Figure 7.5(b) shows the value of $\mu_{cornerness}(m,n)$, as it is computed by the fuzzy system according to Eq. (7.9) to detect corners in a real image. In Figure 7.5(a), the blue crosses represent the corners obtained using the corner selection procedure explained in sub-section 7.2.3.

Figure 7.6 shows the algorithm's performance on different image conditions such as the case with variable illumination and blurring. Figures 7.6(a)-(b) present the performance of the fuzzy corner detector as it is applied to over-exposed and over-illuminated images. The effect of high illumination on the images was made by applying a linear transformation of the form $I(i,j) + 80$.

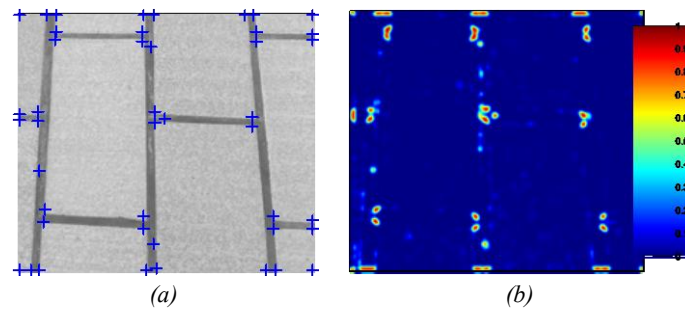


Fig. 7.5. (a) Detected corners using the proposed approach, and (b) values of $\mu_{cornerness}(m,n)$

On the other hand, Figures 7.6(c)-(d) show the effectiveness of the presented detector using low-illuminated or sub-exposed images. Such effect was made by another linear transformation: $I(i,j) - 40$. The images in Figures 7.6(e)-(f) illustrate the sensitivity of the fuzzy detector to blurring.

Such steamed up effect was made by applying a low-pass filter to the original images, with a 5x5 kernel as follows:

$$h(i, j) = \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad (7.12)$$

From results shown in Figure 7.6, it can be observed as the fuzzy detector exhibits immunity to changes in illumination, see for instance Figures 7.6(a)-(7b) and 7.6(c)-(d). However, it also shows sensitivity to blurring in Figures 7.6(e)-(f). For the case of blurring images, the detector is able to find all the corners over the simulated image in 7.6(e).

The latter figure exhibits low distortion in the homogeneous gray levels within the image as a consequence of the filter operation. On the other hand, some sensitivity may be lost while applying the detector to the real image shown in Figure 7.6(f). Moreover, after applying distortion to the image, several points that do not belong to a corner as such have been wrongly marked as corners.

Despite all previous comments, the Fuzzy detector was able to detect in Figure 7.6(f) the corners which delimit the object's shape. This is not a common feature of other corner detectors (Harris & Stephens, 1988; Azriel Rosenfeld & Johnston, 1973; Smith & Brady, 1997; Zou et al., 2008).

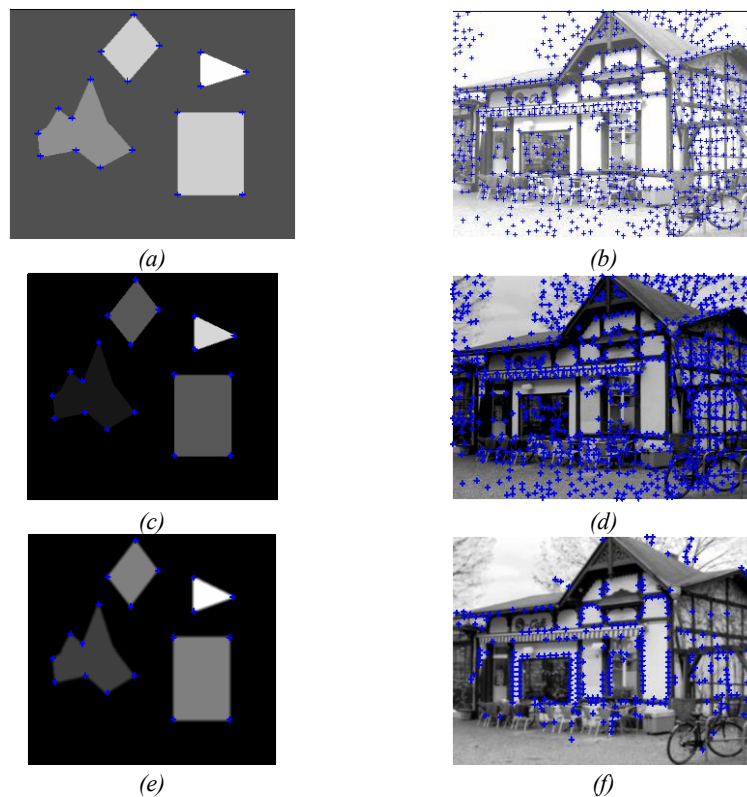


Fig. 7.6. Performance of the fuzzy corner detector over different conditions on the image: (a)-(b) over-exposition or high illumination, (c)-(d) sub-exposition or low illumination and (e)-(f) blurring.

7.4. Performance Comparison

A variety of quantitative evaluation methods for corner detection algorithms have been proposed in the literature (Mokhtarian & Mohanna, 2006; Schmid, Mohr, & Bauckhage, 2000; Zou et al., 2008). Following the criteria in (Mokhtarian & Mohanna, 2006), the performance analysis considers the Harris algorithm (Harris & Stephens, 1988), the fuzzy method presented by Banerjee & Kundu (2008) and the approach presented in this chapter. A quantitative comparison over three criteria is presented: stability, noise immunity and computational effort. The study aims analyze the performance objectively.

The parameters for each detector algorithm are set as follows:

For the Harris algorithm, the gradient operators $[-2 \ -1 \ 0 \ 1 \ 2]$ and $[-2 \ -1 \ 0 \ 1 \ 2]^T$ are set in directions u and v separately. The Gaussian smoothing filter employs a Gaussian window function of size 7×7 and a standard deviation of 2 with $k=0.06$. The parametric setup appears as the best set following data in (Zou et al., 2008) and considering lots of hand tuning experiments.

For the fuzzy method proposed by Banerjee & Kundu, the parameter are set following guidelines from (Banerjee & Kundu, 2008), with a Gaussian window function of size 3×3 and a standard deviation of 2, $\mu_d(P) \geq 0.9$ and $T_h = 0.2$. Finally, the parameters of the presented approach are set according to the Table 7.2.

7.4.1. Stability Criterion

Two frames in an image sequence are processed by the algorithm to detect corners. If the corner's positions are unchanged from one frame to the next one, the algorithm can be regarded as stable. However, the gray-level value of each pixel would normally vary in actual images because of several factors affecting the image.

If the algorithm is applied to a given image, then it cannot be assured the number and position of all detected corners would be exactly the same. Therefore, absolute stability is almost non-existent. A factor η to measure the stability of a corner algorithm can be defined as follows:

$$\eta = \frac{|A_1 \cap A_2|}{\min(|A_1|, |A_2|)} \times 100\%, \quad (7.13)$$

where A_1 and A_2 representing the corner sets for the first and the second frame respectively (the intersection operator \cap stands for common corners); $|A_i|$ represents the number of elements in A_i set and the overall numerator holds the number of corresponding corners in two frames. From expression (7.8), it can be concluded that a greater η yields a more stable corner detection algorithm.

Fifty pairs of images holding different contrast and brightness levels are gathered in order to compare the presented fuzzy detector and other classic methods.

Figure 7.7(a) shows the comparison with respect to the stability factor, where the horizontal axis represents the image pair number and the vertical axis represents the value of such stability factor. The average stability factor of Harris detector is 75%, while the Fuzzy method Banerjee & Kundu holds 70% and the proposed Fuzzy detector shows 83%.

7.4.2. Noise Immunity

Noise immunity is measured by factor ρ which it can be defined as follows:

$$\rho = \frac{B_1 \cap B_2}{\max(|B_1|, |B_2|)} \times 100\%, \quad (7.14)$$

where B_1 is the corner set of the original image and B_2 is the corner set of the image with added noise. In this case, the maximum operator seeks to consider that false corners have been added as a result of additive noise. As ρ increases, it can be assumed that the algorithm's ability to avoid noisy corners is stronger.

One experiment is focus on comparing such noise immunity among methods. Fifty images with 10% of added impulsive noise are considered.

Figure 7.7(b) shows the noise immunity factor, with the Harris detector showing 9%, the fuzzy method Banerjee & Kundu holding 65% and the proposed fuzzy detector showing 80%.

7.4.3. Computational Effort

The speed and computational effort of a corner detector algorithm must meet demands for real-time tasks, regarding speed and required processing time. The runtime of an algorithm can be a reference to its overall computational effort. In order to compare the three algorithms, fifty pairs of images are considered in order to register the algorithm's runtime for testing images holding 320×240 pixels.

The average runtime for the Harris method, the fuzzy Banerjee & Kundu algorithm and the Fuzzy-based corner detectors is 1.8686s, 6.2125s and 0.878s respectively, as all are tested under the MatLab© R2008b environment.

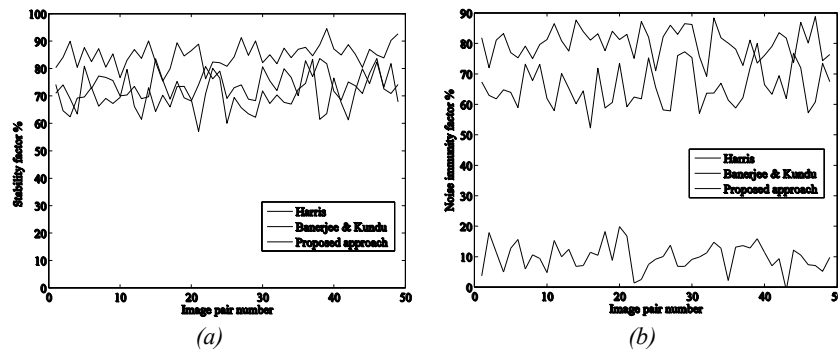


Fig. 7.7. Performance comparison among corner detectors. (a) Stability factor and (b) noisy immunity factor.

7.4.4. Comparison Results

Table 7.3 shows a final comparison between all the methods. The presented fuzzy detector can be considered as equally stable as the Harris method. It also shows stronger noise immunity being slightly superior to the Fuzzy detector proposed by Banerjee & Kundu. The presented corner detector can also be regarded as the algorithm showing the best computational performance.

Corner Detector	Stability \pm Std Dev (%)	Noise \pm Std Dev (%)	Time \pm Std Dev (s)
Harris	75 \pm 5.5	9 \pm 4.4	1.8686 \pm 0.3
Fuzzy Banerjee & Kundu	70 \pm 7.8	65 \pm 7.1	6.2125 \pm 0.21
The presented fuzzy-based detector	83 \pm 4.1	80 \pm 4.6	0.878 \pm 0.11

Table 7.3. Performance comparison among the three corner detectors considered by the study. Figures 7.8, 7.9 and 7.10 shows the performance of the detector algorithms considered in the study while analyzing a number of benchmark images.

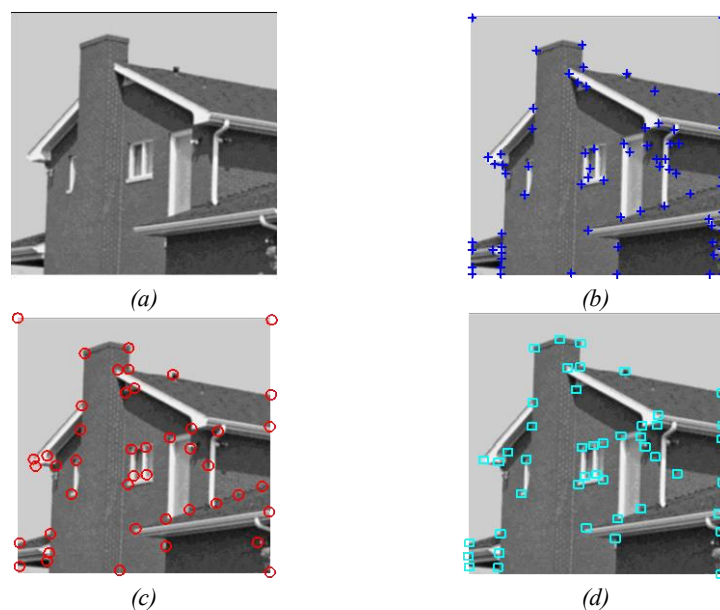


Fig. 7.8. House: (a) Original image, (b) the Harris algorithm, (c) the fuzzy Banerjee & Kundu method and (d) the Fuzzy-based presented detector.

7.5. Conclusions

This chapter has presented a corner detection algorithm which models the structure of a potential corner in images based on a Fuzzy rule set. The method is able to tolerate implicit imprecision and impulsive noise.

Experimental evidence suggests that the fuzzy-based presented algorithm produces better results than other common methods such as the Harris detector and the Fuzzy approach proposed by Banerjee & Kundu (2008).

The presented algorithm is able to successfully identify corners on images holding different uncertainty conditions. However, it is also sensitive to blurring in particular when a steaming up effect is produced by considering neighborhood window wider than the one previously considered for building the Fuzzy model of corners (templates). Such fact shall not be considered as inconvenient because the Fuzzy-based algorithm is still capable of identifying corners over similar blurring levels than those of conventional algorithms.

The presented detector is stable and has shown robustness to impulsive noise which in turn represents its major advantage over the Harris method considering that impulsive noise is commonly found in real-time images.

Although the algorithm exhibits a tolerance to imprecision that matches the performance of the Banerjee & Kunduand, the presented approach requires a lighter computational cost for analyzing benchmark images.

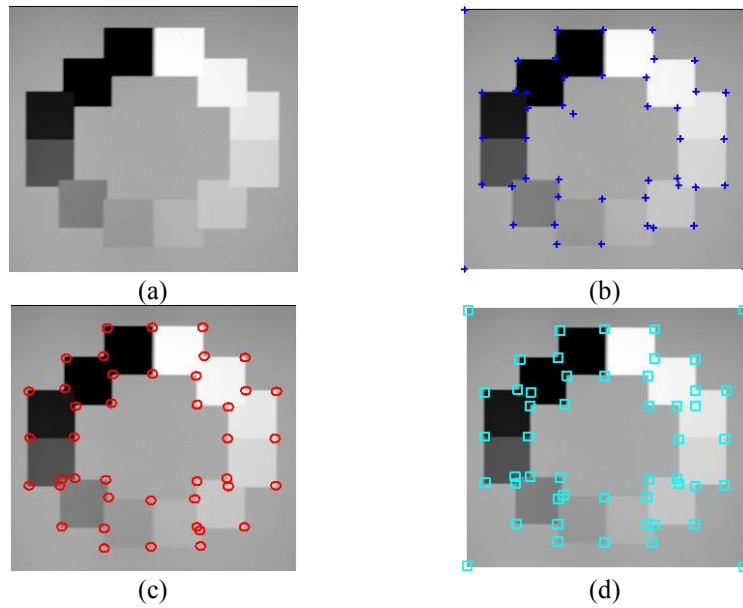


Fig. 7.9. Circle: (a) Original image, (b) Harris algorithm, (c) the fuzzy Banerjee & Kundu method and (d) the presented fuzzy-based detector.

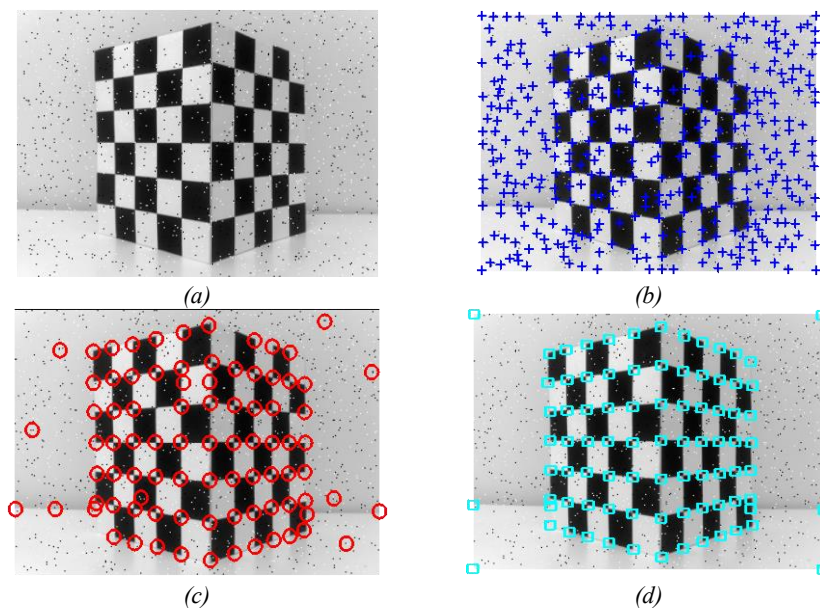


Fig. 7.10. Noisy chessboard image: (a) Original image and the output after applying (b) the Harris algorithm, (c) the Fuzzy Banerjee & Kundu method and (d) the presented Fuzzy-based detector.

Chapter 8

Clonal Selection algorithm applied to Circle detection

Automatic circle detection in digital images is considered an important and complex task for the computer vision community. Consequently, recently, a tremendous amount of research has been devoted to find an optimal circle detector. This chapter presents an algorithm for the automatic detection of circular shapes from complicated and noisy images without making use of the conventional Hough transform principles. The presented algorithm is based on the Artificial Immune Optimization (AIO) technique, known as the Clonal Selection algorithm (CSA). The CSA is an effective method for searching and optimizing following the Clonal Selection Principle (CSP) in the human immune system which generates a response according to the relationship between antigens (Ag), i.e. patterns to be recognized and antibodies (Ab) i.e. possible solutions. The algorithm uses the encoding of three points as candidate circles (x,y,r) over the edge image. An objective function evaluates if such candidate circles (Ab) are actually present in the edge image (Ag). Guided by the values of this objective function, the set of encoded candidate circles are evolved using the CSA so that they can fit to the actual circles on the edge map of the image. Experimental results over several synthetic as well as natural images with varying range of complexity validate the efficiency of the presented technique with regard to accuracy, speed, and robustness.

8.1. Introduction

Bio-inspired computing (Brabazon & O'Neill, 2006) lies within the realm of Natural Computing, a field of research that is concerned with both the use of biology as an inspiration for solving computational problems and the use of the natural world experiences to solve real world problems. The increasing interest in this field lies in the fact that nowadays the world is facing more and more complex, large, distributed and ill-structured systems, while on the other hand, people notice that the apparently simple structures and organizations in nature are capable of dealing with most complex systems and tasks with ease. Bio-inspired computing has proved to be useful in various application areas. Following features from optimization, pattern recognition, shape detection and machine learning, the Bio-inspired algorithms have recently gained considerable research interest from the computer vision community.

Currently, bio-inspired algorithms are widely applied to solve challenging computer vision problems. For instance, Chih-Chih Lai, (2006) have applied the Particle Swarm Optimization (PSO) algorithm for image segmentation. Le Hgarat-Masclé, et al., (2007) proposed a non-stationary Markov model-based image regularization algorithm, which uses another swarm intelligence algorithm known as Ant Colony Optimization (ACO). In (Hammouche, et al., 2008), the authors proposed a multilevel method that allows the determination of the appropriate number of thresholds for image segmentation. Such method combines a Genetic Algorithm (GA) with a wavelet transform. More recently, Baştürk & Günay, (2009) have proposed an image edge detector based on a cellular neural network which is optimized by the Differential Evolution (DE) algorithm.

The problem of detecting circular features holds paramount importance for image analysis, in particular for industrial applications such as automatic inspection of products and components, aided vectorization of drawings, target detection, etc (Costa & Cesar, 2001). Many methods have been

developed to solve the shape-detection problem (Peura, et al., 1997). Solving the object location is normally approached from two viewpoints: deterministic techniques which include the application of Hough transform (Yuen, et al., 1990), geometric hashing, template or model matching techniques (Iivarinen, et al., 1997; Jones, et al., 1990). On the other hand, stochastic techniques include random sample consensus (Fischler & Bolles, 1981), simulated annealing (Bongiovanni, et al., 1995) and genetic algorithms (GA) (Roth & Levine, 1994).

Template and model matching techniques are the first approaches to be successfully applied to shape localization. Shape coding techniques and combination of shape properties are used to represent such objects. The main drawback of these techniques is related to the contour extraction from real images. Additionally, it is difficult for models to deal with pose invariance unless only simple objects are considered.

Commonly, circle detection in digital images is performed by means of Circular Hough Transform (Muammar & Nixon, 1989). A typical Hough-based approach employs an edge detector and uses edge information to infer locations and radius values. Peak detection is then applied by averaging, filtering and histogramming the transform space. However, such approach requires a large storage space -given the 3-D cells needed to store the parameters (x, y, r) , the computational complexity yielding low processing speeds.

The accuracy of the detected circle's parameters is poor, under noisy conditions (Atherton & Kerbyson, 1993). The required processing time for Circular Hough Transform makes it prohibitive to be deployed in real time applications, in particular for digital images with significant width and height and a densely populated area around edge pixels. In order to overcome such a problem, other researchers have proposed new approaches based on the Hough transform (HT) such as the probabilistic HT (Fischler & Bolles, 1981; Shaked, et al., 1996b), the randomized HT (RHT) (Xu et al., 1990) and the Fuzzy Hough transform (FHT) (Han, et al., 1994a). In (Lu & Tan, 2008), the authors proposed a novel approach based on RHT called Iterative Randomized Hough Transformation (IRHT) that achieves better results on complex images and noisy environments. The algorithm iteratively applies the randomized Hough transform (RHT) to a region of interest in the image which is determined from the latest estimation of ellipse/circle parameters.

Shape recognition can also be approached using stochastic search methods such as Genetic Algorithms (GA). In particular, GA has recently been applied to important shape detection tasks e.g. Roth & Levine, (1994) proposed use of GA for primitive extraction of images. Lutton & Martinez, (1994), developed a further improvement of the aforementioned method. Yao, et al., (2004), came up with a multi-population GA method to detect ellipses. In (Lu & Tan, 2008), GA was used for template matching when the pattern has been the subject of an unknown affine transformation.

Ayala-Ramirez et al., (2006), presented a GA-based circle detector that is capable of detecting multiple circles on real images, but it fails frequently when detecting imperfect circles. Recently, Dasgupta, et al., (2010) proposed an automatic circle detector using the bacterial foraging algorithm as optimization method. For the case of ellipsoidal detection Rosin, (1999) proposed an ellipse fitting algorithm that uses five points.

On the other hand, biological inspired methods can successfully be transferred into novel computational paradigms as shown by the successful development of artificial neural networks, evolutionary algorithms, swarming algorithms and so. The Human Immune System (HIS) is a highly evolved, parallel and distributed adaptive system (Goldsby, 2005) that exhibits remarkable abilities that can be imported into important aspects in the field of computation. This emerging field is known as Artificial Immune Systems (AIS) (De Castro & Timmis, 2002) which is a computational system fully inspired by the immunology theory and its functions, including principles and models.

AIS have recently reached considerable research interest from different communities (Dasgupta, 2006), focusing on several aspects of optimization, pattern recognition, abnormality detection, data analysis and machine learning. Artificial Immune Optimization has been successfully applied to tackle numerous challenging optimization problems with remarkable performance in comparison to other classical techniques (Wang, et al., 2004).

Clonal Selection algorithm (CSA) (De Castro & Von Zuben, 2002) is one of the most widely employed AIO approaches. The CSA is a relatively novel evolutionary optimization algorithm which has been built on the basis of the Clonal Selection Principle (CSP) (Ada & Nossal, 1987) of HIS. The CSP explains the immune response when an antigenic pattern is recognized by a given antibody. In the clonal selection algorithm, the antigen (Ag) represents the problem to be optimized and its constraints, while the antibodies (Ab) are the candidate solutions of the problem. The antibody-antigen affinity indicates the matching as well between the solution and the problem. The algorithm performs the selection of antibodies based on affinity either by matching against an antigen pattern or by evaluating the pattern via an objective function. In mathematical grounds, CSA has the ability of getting out of local minima while simultaneously operating over a pool of points within the search space. It does not use the derivatives or any of its related information as it employs probabilistic transition rules instead of deterministic ones. Despite to its simple and straightforward implementation, it has been extensively employed in the literature for solving several kinds of challenging engineering problems (Campelo, et al., 2005; Coello & Cortes, 2005; Dong, et al., 2007).

This chapter presents an algorithm for the automatic detection of circular shapes from complicated and noisy images with no consideration of the conventional Hough transform principles. The presented algorithm is based on a recently developed Artificial Immune Optimization (AIO) technique, known as the Clonal Selection algorithm (CSA). The algorithm uses the encoding of three non-collinear edge points as candidate circles (x,y,r) in the edge image of the scene. An objective function evaluates if such candidate circles (Ab) are actually present in the edge image (Ag). Guided by the values of this objective function, the set of encoded candidate circles are evolved using the CSA so that they can fit into the actual circles within the edge map of the image. The approach generates a sub-pixel circle detector which can effectively identify circles in real images despite circular objects exhibiting a significant occluded portion. Experimental evidence shows the effectiveness of such method for detecting circles under different conditions. Comparison to one state-of-the-art GA-based method (Han et al., 1994a) and a randomized Hough transform approach (IRHT) (Bongiovanni et al., 1995) on multiple images demonstrates a better performance of the presented method.

The chapter is organized as follows: Section 8.2 provides a brief CSA explanation. Section 8.3 formulates the approach and studies the main features of the CSA method as it is used to detect circles in images. Section 8.4 shows the experimental results of applying our method to the recognition of circles in different image conditions. Finally, Section 8.5 discusses several conclusions.

8.2. Clonal Selection Algorithm

In natural immune systems, only the antibodies (Abs) which are able to recognize the intrusive antigens (non-self cells) are to be selected to proliferate by cloning (Goldsby, 2005). Therefore, the fundament of the clonal optimization method is that only capable Abs will proliferate. Particularly, the underlying principles of the CSA are borrowed from the CSP as follows:

- Maintenance of memory cells which are functionally disconnected from repertoire,
- Selection and cloning of most stimulated Abs,

- Suppression of non-stimulated cells,
- Affinity maturation and re-selection of clones showing the highest affinities, and
- Mutation rate proportional to Abs affinities.

From immunology concepts, an antigen is any substance that forces the immune system to produce antibodies against it. Regarding the CSA systems, the antigen concept refers to the pending optimization problem which focuses on circle detection. In CSA, B cells, T cells and antigen-specific lymphocytes are generally called antibodies. An antibody is a representation of a candidate solution for an antigen, e.g. the prototype circle in this work. A selective mechanism guarantees that those antibodies (solutions) that better recognize the antigen and therefore may elicit the response, are to be selected holding long life spans. Therefore such cells are to be named memory cells (**M**).

8.2.1. Definitions

In order to describe the CSA, the notation includes boldfaced capital letters indicating matrices and boldfaced small letters indicating vectors. Some relevant concepts are also revisited below:

- (i) Antigen: the problem to be optimized and its constraints (circle detection).
- (ii) Antibody: the candidate solutions of the problem (circle candidates).
- (iii) Affinity: the objective function measurement for an antibody (circle matching).

The limited-length character string \mathbf{d} is the coding of variable vector \mathbf{x} as $\mathbf{d} = \text{encode}(\mathbf{x})$; and \mathbf{x} is called the decoding of antibody \mathbf{d} following $\mathbf{x} = \text{decode}(\mathbf{d})$.

Set \mathbf{I} is called the antibody space namely $\mathbf{d} \in \mathbf{I}$. The antibody population space is thus defined as:

$$\mathbf{I}^m = \{\mathbf{D} : \mathbf{D} = (\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m), \quad \mathbf{d}_k \in \mathbf{I}, \quad 1 \leq k \leq m\} \quad (8.1)$$

where the positive integer m is the size of antibody population $\mathbf{D} = \{\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m\}$ which is an m -dimensional group of antibody \mathbf{d} , being a spot within the antibody space \mathbf{I} .

8.2.2. CSA Operators

Based on (Gong, et al., 2009), the CSA implements three different operators: the clonal proliferation operator (T_P^C), the affinity maturation operator (T_M^A) and the clonal selection operator (T_S^C). $\mathbf{A}(k)$ is the antibody population at time k that represents the set of antibodies \mathbf{a} , such as $\mathbf{A}(k) = \{\mathbf{a}_1(k), \mathbf{a}_2(k), \dots, \mathbf{a}_n(k)\}$. The evolution process of CSA can be described as follows:

$$\mathbf{A}(k) \xrightarrow{T_P^C} \mathbf{Y}(k) \xrightarrow{T_M^A} \mathbf{Z}(k) \cup \mathbf{A}(k) \xrightarrow{T_S^C} \mathbf{A}(k+1) \quad (8.2)$$

8.2.2.1 Clonal proliferation operator (T_P^C)

Define

$$\mathbf{Y}(k) = T_P^C(\mathbf{A}(k)) = [T_P^C(\mathbf{a}_1(k)), T_P^C(\mathbf{a}_1(k)), \dots, T_P^C(\mathbf{a}_n(k))] \quad (8.3)$$

where $\mathbf{Y}(k) = T_p^c(\mathbf{A}(k)) = \mathbf{e}_i \cdot \mathbf{a}_i(k)$ $i = 1, 2, \dots, n$, and \mathbf{e}_i is a q_i -dimensional identity column vector. Function $\text{round}(x)$, gets x to the least integer bigger than x . There are various methods for calculating q_i . In this work, it is calculated as follows:

$$q_i(k) = \text{round} \left[N_c \cdot \frac{F(\mathbf{a}_i(k))}{\sum_{j=1}^n F(\mathbf{a}_j(k))} \right] \quad i = 1, 2, \dots, n \quad (8.4)$$

where N_c is called the clonal size. The value of $q_i(k)$ is proportional to the value of $F(\mathbf{a}_i(k))$. After clonal proliferation, the population becomes

$$\mathbf{Y}(k) = \{\mathbf{Y}_1(k), \mathbf{Y}_2(k), \dots, \mathbf{Y}_n(k)\} \quad (8.5)$$

where

$$\begin{aligned} \mathbf{Y}_i(k) &= \{\mathbf{y}_{ij}(k)\} = \{\mathbf{y}_{i1}(k), \mathbf{y}_{i2}(k), \dots, \mathbf{y}_{iq_i}(k)\} \text{ and} \\ \mathbf{y}_{ij}(k) &= \mathbf{a}_i(k), \quad j = 1, 2, \dots, q_i. \quad i = 1, 2, \dots, n \end{aligned} \quad (8.6)$$

8.2.2.2. Affinity maturation operator (T_M^A)

The affinity maturation operation is performed by hypermutation. Random changes are introduced into the antibodies just like it happens in the immune system. Such changes may lead to increase the affinity. The hypermutation is performed by the operator T_M^A which is applied to the population $\mathbf{Y}(k)$ as it is obtained by clonal proliferation $\mathbf{Z}(k) = T_M^c(\mathbf{Y}(k))$. The mutation rate is calculated using the following equation (De Castro & Von Zuben, 2002):

$$\alpha = e^{(-\rho \cdot F(ab))} \quad (8.7)$$

being α the mutation rate, F being the objective function value of the antibody (ab) as it is normalized between $[0, 1]$ and ρ being a fixed step. In (Cutello, et al., 2005), it is demonstrated the importance of including the factor ρ into Eq. (8.7) to improve the algorithm performance. The way ρ modifies the shape of the mutation rate is shown by Figure 8.1.

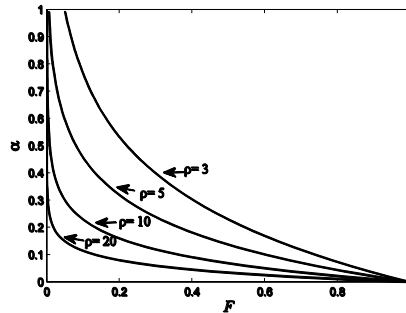


Fig. 8.1. Hypermutation rate versus fitness, considering some size steps

The number of mutations held by a clone with objective function value F , is equal to $L \cdot \alpha$, considering L as the length of the antibody 22 bits are used in this chapter. For the binary encoding, mu-

tation operation can be done as follows: each gene within an antibody may be replaced by its opposite number (i.e. 0-1 or 1-0). Following the affinity maturation operation, the population becomes:

$$\begin{aligned} \mathbf{Z}(k) &= \{\mathbf{Z}_1(k), \mathbf{Z}_2(k), \dots, \mathbf{Z}_n(k)\} \\ \mathbf{Z}_i(k) &= \{\mathbf{z}_{ij}(k)\} = \{\mathbf{z}_{i1}(k), \mathbf{z}_{i2}(k), \dots, \mathbf{z}_{iq_i}(k)\} \text{ and} \\ \mathbf{z}_{ij}(k) &= T_M^A(\mathbf{y}_{ij}(k)), \quad j=1, 2, \dots, q_i \quad i=1, 2, \dots, n \end{aligned} \quad (8.8)$$

where T_M^A is the operator as it is defined by Eq. 8.7 and applied onto the antibody \mathbf{y}_{ij} .

8.2.2.3. Clonal selection operator (T_S^C)

Define $\forall i=1, 2, \dots, n$, $\mathbf{b}_i(k) \in \mathbf{Z}_i(k)$ as the antibody with the highest affinity in $\mathbf{Z}_i(k)$, then $\mathbf{a}_i(k+1) = T_S^C(\mathbf{Z}_i(k) \cup \mathbf{a}_i(k))$, where T_S^C is defined as:

$$T_S^C(\mathbf{Z}_i(k) \cup \mathbf{a}_i(k)) = \begin{cases} \mathbf{b}_i(k) & \text{if } F(\mathbf{a}_i(k)) < F(\mathbf{b}_i(k)) \\ \mathbf{a}_i(k) & \text{if } F(\mathbf{a}_i(k)) \geq F(\mathbf{b}_i(k)) \end{cases} \quad (8.9)$$

where $i=1, 2, \dots, n$.

Each step of the CSA may be defined as follows:

Step 1	Initialize randomly a population (Pinit), a set $h = P_r + n$ of candidate solutions of subsets of memory cells (\mathbf{M}) which is added to the remaining population (\mathbf{P}_r), with the total population being $\mathbf{P}_T = \mathbf{P}_r + \mathbf{M}$, with \mathbf{M} holding n memory cells.
Step 2	Select the n best individuals of the population \mathbf{P}_T to build $\mathbf{A}(k)$, according to the affinity measure (objective function).
Step 3	Reproduce (T_P^C) population $\mathbf{A}(k)$ proportionally to their affinity with the antigen and generate a temporary population of clones $\mathbf{Y}(k)$. The clone number is an increasing function of the affinity with the antigen (Eq. 8.4).
Step 4	Mutate (T_M^A) the population $\mathbf{Y}(k)$ of clones according to the affinity of the antibody to the antigen (Eq. 8.7). A matured antibody population $\mathbf{Z}(k)$ is thus generated.
Step 5	Re-select (T_S^C) the best individuals from $\mathbf{Z}(k)$ and $\mathbf{A}(k)$ to compose a new memory set $\mathbf{M} = \mathbf{A}(k+1)$.
Step 6	Add random P_r novel antibodies (diversity introduction) to the new memory cells \mathbf{M} to build \mathbf{P}_T .
Step 7	Stop if any criteria are reached, otherwise return to Step 2.

Figure 8.2 shows the full draw of the CSA. The clone number in Step 3 is defined according to Eq. (8.4). Although a unique mutation operator is used in Step 5, the mutated values of individuals are inversely proportional to their fitness by means of Eq. (8.7), i.e. the more Ab shows a better fitness, the less it may change.

The similarity property (Gong, et al., 2008) within the Abs can also affect the convergence speed of the CSA. The idea of the antibody addition based on the immune network theory is introduced

for providing diversity to the newly generated Abs in \mathbf{M} , which may be similar to those already in the old memory \mathbf{M} . Holding such a diverse Ab pool, the CSA can avoid being trapped into local minima (Gao, et al., 2009), contrasting to well-known genetic algorithms (GA) which usually tend to bias the whole population of chromosomes towards only the best candidate solution (Langdon & Poli, 2002). Therefore, it can effectively handle challenging multimodal optimization tasks (Tang & Qiu, 2006; Wang, et al., 2005; Xu & Zhang, 2007; Yoo & Hajela, 1999).

The management of population includes a simple and direct searching algorithm for globally optimal multi-modal functions. This is also another clear difference in comparison to other evolutionary algorithms, like GA, because it does not require crossover but only cloning and hypermutation of individuals in order to use affinity as selection mechanism. The CSA is adopted in this work in order to find the circle parameters (x, y, r) that better represent the actual circles in the image.

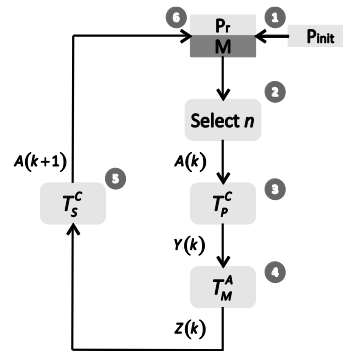


Fig. 8.2. Basic flow diagram of clonal selection algorithm (CSA).

8.3. Circle detection using CSA

Circles are represented in this work by means of parameters of a well-known second degree equation (see Equation 8.10), that passes through three points (Roth & Levine, 1994) in the edge space of the image. Images are preprocessed by an edge detection method which uses a single-pixel contour detector. Such task is accomplished by the classical Canny algorithm which stores locations for each edge point. Therefore, such points are the only potential candidates to define circles by considering triplets. All the edge points in the image are then stored within a vector array $P = \{p_1, p_2, \dots, p_{N_p}\}$ with N_p as the total number of edge pixels contained in the image. The algorithm stores the (x_i, y_i) coordinates for each edge pixel p_i in the edge vector.

In order to construct each of the circle candidates (or antibodies within the AIS-framework), the indexes i_1 , i_2 and i_3 of three non-collinear edge points must be combined, assuming the circle's contour goes through points p_{i_1} ; p_{i_2} ; p_{i_3} . A number of candidate solutions are generated randomly for the initial pool. The solutions will thus evolve through the application of the CSA as the evolution takes place over the pool until a minimum is reached and the best individual is considered as the solution for the circle detection problem.

Applying classic methods based on Hough Transform for circle detection would normally require huge amounts of memory and consume large computation time. In order to reach a sub-pixel resolution –just like the method discussed in this chapter, they also consider three edge points to cast a vote for the corresponding point within the parameter space. Such methods also require an evidence-collecting step which is also implemented by the method in this chapter.

As the overall evolution process evolves, the objective function improves at each generation by discriminating non-plausible circles and locating others by avoiding a visit to other image points.

The following discussion clearly explains the required steps to formulate the circle detection task just as an AIO optimization problem.

8.3.1. Individual representation

Each antibody C of the pool uses three edge points as elements. In this representation, the edge points are stored according to one index that is relative to their position within the edge array P . In turn, the procedure will encode an Ab as the circle that passes through three points p_i, p_j and p_k ($C = \{p_i, p_j, p_k\}$).

Each circle C is represented by three parameters: x_0, y_0 and r , being (x_0, y_0) the (x, y) coordinates of the center of the circle and r its radius. The equation of the circle passing through the three edge points can thus be computed as follows:

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \quad (8.10)$$

considering

$$\mathbf{A} = \begin{bmatrix} x_j^2 + y_j^2 - (x_j^2 + y_j^2) & 2 \cdot (y_j - y_i) \\ x_k^2 + y_k^2 - (x_i^2 + y_i^2) & 2 \cdot (y_k - y_i) \end{bmatrix} \quad (8.11)$$

$$\mathbf{B} = \begin{bmatrix} 2 \cdot (x_j - x_i) & x_j^2 + y_j^2 - (x_i^2 + y_i^2) \\ 2 \cdot (x_k - x_i) & x_k^2 + y_k^2 - (x_i^2 + y_i^2) \end{bmatrix}$$

$$x_0 = \frac{\det(\mathbf{A})}{4((x_j - x_i)(y_k - y_i) - (x_k - x_i)(y_j - y_i))} \quad (8.12)$$

$$y_0 = \frac{\det(\mathbf{B})}{4((x_j - x_i)(y_k - y_i) - (x_k - x_i)(y_j - y_i))}$$

and

$$r = \sqrt{(x_0 - x_d)^2 + (y_0 - y_d)^2} \quad (8.13)$$

being $\det(\cdot)$ the determinant and $d \in \{i, j, k\}$. Figure 8.3 illustrates the parameters defined by Equations 8.10 to 8.13.

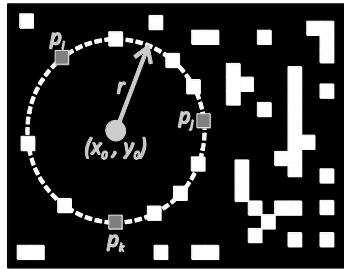


Fig. 8.3. Circle candidate (individual) built from the combination of points p_i, p_j and p_k .

Therefore it is possible to represent the shape parameters (for the circle, $[x_0, y_0, r]$) as a transformation R of the edge vector indexes i, j and k .

$$[x_0, y_0, r] = R(i, j, k) \quad (8.14)$$

with R being the transformation calculated after the previous computations of x_0, y_0 , and r .

By exploring each index as an individual parameter, it is possible to sweep the continuous space looking for the shape parameters using the AIO through the CSA. This approach reduces the search space by eliminating unfeasible solutions.

8.3.2. Objective function or matching function

A circumference may be calculated as a virtual shape in order to measure the matching factor between C and the presented circle in the image (antigenic). It must be also validated, i.e. if it really exists in the edge image. The test for such points is $S = \{s_1, s_2, \dots, s_{N_s}\}$, with N_s representing the number of test points over which the existence of an edge point will be verified.

The test S is generated by the Midpoint Circle Algorithm (MCA) (Bresenham, 1977) which determines the required points for drawing a circle considering the radius r and the center point (x_0, y_0) . The MCA employs the circle equation $x^2 + y^2 = r^2$ with only the first octant. It draws a curve starting at point $(r, 0)$ and proceeds upwards-left by using integer additions and subtractions. See full details in (Van Aken, 1984).

The MCA aims to calculate the points N_s which are required to represent the circle considering coordinates $S = \{s_1, s_2, \dots, s_{N_s}\}$. Although the algorithm is considered the quickest providing a sub-pixel precision, it is important to assure that points lying outside the image plane must not be considered as they must be included in N_s , thus protecting the MCA operation.

The matching function or objective function $J(C)$ represents the matching (or error) resulting from pixels S for the circle candidate and the pixels that actually exist in the edge image, yielding:

$$J(C) = 1 - \frac{\sum_{i=1}^{N_s} E(x_i, y_i)}{N_s} \quad (8.15)$$

with $E(x_i, y_i)$ accumulating the number of expected edge points (the points in S) that are actually present in the edge image. N_s is the number of pixels within the perimeter of the circle that correspond to C , currently under testing.

Therefore, the algorithm aims to minimize $J(C)$, given that a smaller value implies a better response (matching) of the ‘‘circularity’’ operator. The optimization process can thus be stopped after the maximum number of epochs is reached, and the individuals are clearly defined satisfying the threshold. The stopping criterion depends on the a priori knowledge about the application context.

8.3.3. Implementation of CSA

In this work, an antibody will be represented (in binary form) by a bit chain of the form:

$$c = \langle c_1, c_2, \dots, c_L \rangle \quad (8.16)$$

where c representing a point in an L -dimensional (with L bits) space,

$$c \in \mathcal{S}^L \quad (8.17)$$

The CSA implementation can be stated as follows:

1. An original pool of N antibodies is generated, considering the size of 22 bits.
2. The n best Ab's are selected based on the matching function. They will represent the memory set.
3. Best Ab's are cloned.
4. Perform hyper-mutation of the cloned Ab's following the affinity between antibodies and antigens while generating one improved antibody pool.
5. From the hyper-mutated pool, the Ab's with the highest affinity are to be re-selected.
6. As for the original pool, the Ab's with the lowest affinity are replaced improving the overall cells set.

Once the above steps are completed, the process is started again, until one Ab shows the best matching i.e. finding the minimum value of $J(C)$. In this work, the algorithm considers three index points embedded into a single Ab to represent one circle. Each single index has the variable P_i (with $i=1, 2, 3$) representing the Hamming shape-space by means of a 22-bits word over the following range:

$$P_i : [1, N_p] \quad (8.18)$$

considering N_p the total number of edge pixels contained in the image. Hence, the first step is to generate the initial antibody pool by means of:

$$AB = 2 \cdot \text{rand}(N, S_p) - 1; \quad (8.19)$$

where S_p represents the bit size as it is assigned to each of N initial Abs (twenty two for this work). In order to perform the mapping from binary string to base 10, it yields

$$\left((c_L, \dots, c_2, c_1) \right)_2 = \left(\sum_{i=0}^{21} c_i \cdot 2^i \right)_{10} = r' \quad (8.20)$$

Finding the corresponding real value for r :

$$r = r' \cdot \frac{r_{\max}}{2^{22} - 1} \quad (8.21)$$

by using r_{\max} to represent N_p .

8.4. Experimental results

8.4.1. Parametric setup

Table 8.1 presents the parameters of CSA used in this work. Once they have been determined experimentally, they are kept for all the test images through all experiments.

h	n	N_c	ρ	P_r	L	T_c	$ITER$
120	100	80	10	20	22	0.01	400

Table 8.1. Parameter setup for the CSA detector

All the experiments are performed on a Pentium IV 2.5 GHz computer under C language programming. All the images are preprocessed by the standard Canny edge-detector using the image-processing toolbox for MATLAB R2008a.

For comparison purposes, the CSA algorithm is tested against the IRHT and the GA circle detectors to each image individually.

For the GA algorithm described in (Ayala-Ramirez et al., 2006), the population size is 70, the crossover probability is 0.55, the mutation probability is 0.10 and number of elite individuals is 2. The roulette wheel selection and the 1-point crossover are applied. The parameter setup and the fitness function follow the configuration suggested also in (Ayala-Ramirez et al., 2006). The parameter values are defined as suggested in the IRHT algorithm proposed for Lu & Tan, (2008). In IRHT, the most important parameters are grouped into the vector Δ_c which defines the desired set of enlargements of the circle/ellipse parameters to build a new region of interest. In this comparison, Δ_c is considered as $\Delta_c = [0.5 \cdot \sigma_x \quad 0.5 \cdot \sigma_x \quad 0.5 \cdot \sigma_a \quad 0.5 \cdot \sigma_b \quad 0]$. Such configuration is chose according to (Lu & Tan, 2008) as such values make the algorithm insensitive to noise images.

8.4.2. Error score and success rate

Real-life images rarely contain perfectly-shaped circles. Therefore, in order to test the accuracy of the CSA approach, the results are compared to a ground-truth circle (see (Dasgupta et al., 2010)) which is manually detected from the original edge-map. The parameters $(x_{true}, y_{true}, r_{true})$ of the ground-truth circle are computed using the Equations 8.10-8.13, over the three circumference points from the manually detected circle. If the center and the radius of such circle are found by the algorithm, defining (x_D, y_D) and r_D , then the error score defined as follows:

$$Es = \eta \cdot (|x_{true} - x_D| + |y_{true} - y_D|) + \mu \cdot |r_{true} - r_D| \quad (8.22)$$

The first term represents the shift of the center of the detected circle as it is compared to the ground-truth circle. The second term accounts for the difference between their radii. η and μ are two weights associated to each term in expression (8.22). They may be chosen according to the required accuracy as $\eta = 0.05$ and $\mu = 0.1$. This particular choice of parameters ensures that the radii difference is strongly weighted than difference of center positions between the manually detected and the machine-detected circle. It is assumed that if the Es is found to be less than 1, then the algorithm gets a success. Otherwise, it is considered to have failed in detecting the edge-circle. Notice that for $\eta = 0.05$ and $\mu = 0.1$ yields $Es < 1$ which means that the maximum tolerated differ-

ence of radius length is 10 pixels while the maximum mismatch in the location of the center can be up to 20 pixels. From this viewpoint, the success rate (SR) is defined as percentage of reaching success after a certain number of trials.

8.4.3. Presentation of results

Figure 8.4 provides three synthetic images and their counterparts after processing with CSA. Figure 8.5 presents the same experimental results on natural images. In order to test the robustness of the algorithm, salt and pepper noise have been added to the synthetic images before applying the algorithm. Likewise, the natural image shown by Figure 5b is also corrupted with salt and pepper noise. It also illustrates the performance of the algorithm considering noisy and corrupted pixels.

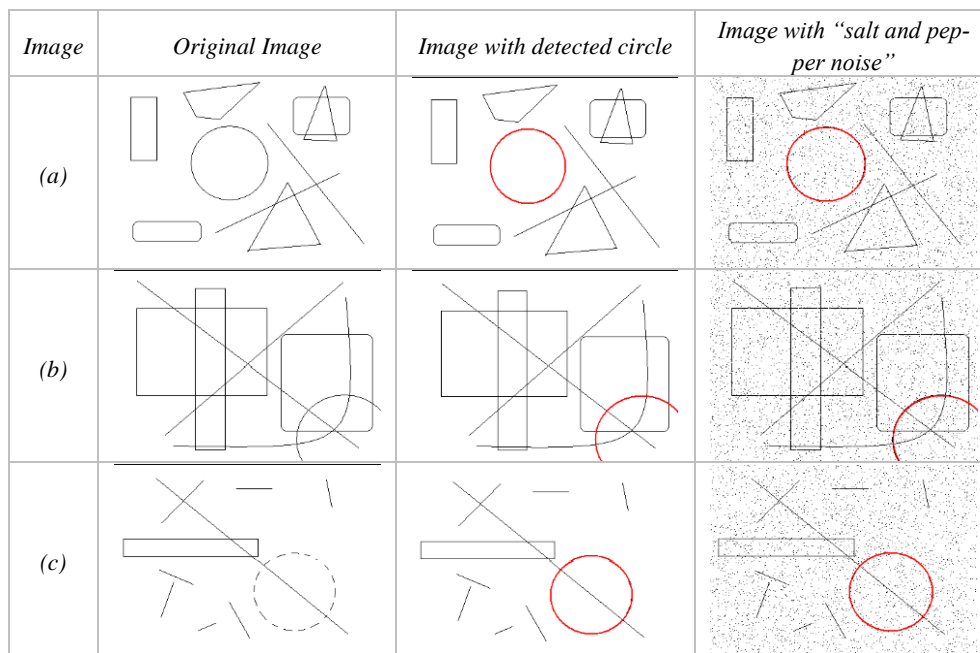


Fig. 8.4. Synthetic images and their detected circles.

As real-life images rarely contain perfectly-shaped circles, the presented algorithm must approximate the circle that better fits into imperfect shapes within a noisy image. Such circle would therefore correspond to the better match in the objective function $J(C)$.

Considering the benchmark images and their corresponding edge maps shown by Figure 8.4, Figure 8.5 provides a visual performance illustration of the GA algorithm (Ayala-Ramirez et al., 2006), the IRHT algorithm (Lu & Tan, 2008) and the presented approach, over three challenging problem instances, i.e. occluded circle, uneven circumference and synthetic noisy image).

The results are averaged over 35 independent runs of each algorithm. It is interesting to observe that the deviation between the detected circle and actual circle is the smallest under the CSA detector. Table 8.2 shows the averaged execution time, the success rate (in %), and averaged error score -following Eq. 8.22, for the three competitor algorithms over six test images shown by Figures 8.4 and 8.5. Table 8.3 contents the results after processing noisy images shown by Figure 8.4. The best results are marked in bold for both Tables.





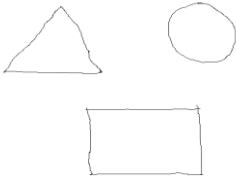
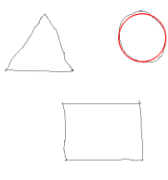
<i>Image</i>	<i>Original Image</i>	<i>Image with detected circle</i>
(a)		
(b)		
(c)		

Fig. 8.5. Natural images and their detected circles.


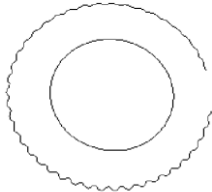


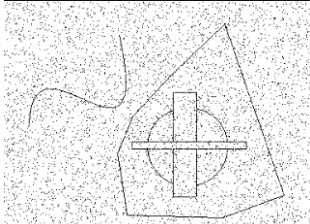
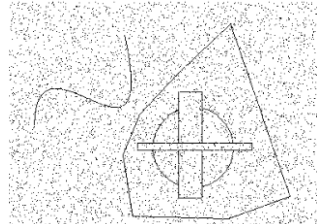
<i>Image</i>	<i>Original Image</i>	<i>Edge-map</i>
(a)		
(b)		
(c)		

Fig. 8.6. Complex benchmark images and their corresponding edge maps.

A close inspection of Tables 8.2 and 8.3 reveals that the CSA method is able to achieve the highest success rate and minimum error tracking least computational time in majority of the cases.

Image	Average time \pm Standard deviation (s)			Success rate (SR) (%)			Es \pm Standard deviation		
	GA	IRHT	CSA	GA	IRHT	CSA	GA	IRHT	CSA
Synthetic images									
(a)	1.18 $\pm(0.20)$	2.10 $\pm(0.80)$	0.52 $\pm(\mathbf{0.10})$	100	92	100	0.77 $\pm(0.081)$	0.62 $\pm(0.070)$	0.40 $\pm(\mathbf{0.051})$
(b)	1.24 $\pm(0.39)$	1.80 $\pm(0.65)$	0.46 $\pm(\mathbf{0.24})$	95	81	98	0.62 $\pm(0.050)$	0.45 $\pm(0.023)$	0.37 $\pm(\mathbf{0.085})$
(c)	2.16 $\pm(0.11)$	3.18 $\pm(0.36)$	0.60 $\pm(\mathbf{0.19})$	91	82	100	0.60 $\pm(0.041)$	0.57 $\pm(0.041)$	0.31 $\pm(\mathbf{0.024})$
Natural Images									
(a)	2.11 $\pm(0.51)$	2.61 $\pm(0.52)$	1.12 $\pm(\mathbf{0.37})$	90	92	100	0.77 $\pm(0.031)$	0.82 $\pm(0.043)$	0.43 $\pm(\mathbf{0.055})$
(b)	2.91 $\pm(0.34)$	3.21 $\pm(0.14)$	1.61 $\pm(\mathbf{0.17})$	92	90	100	0.97 $\pm(0.055)$	1.02 $\pm(0.136)$	0.51 $\pm(\mathbf{0.041})$
(c)	3.82 $\pm(0.97)$	4.36 $\pm(0.17)$	1.95 $\pm(\mathbf{0.41})$	88	81	98	1.21 $\pm(0.102)$	1.42 $\pm(0.155)$	0.59 $\pm(\mathbf{0.073})$

Table 8.2. Averaged execution time and success rate of the GA, the IRHT and the CSA method, over the six test images shown by Figures 8.4 and 8.5.

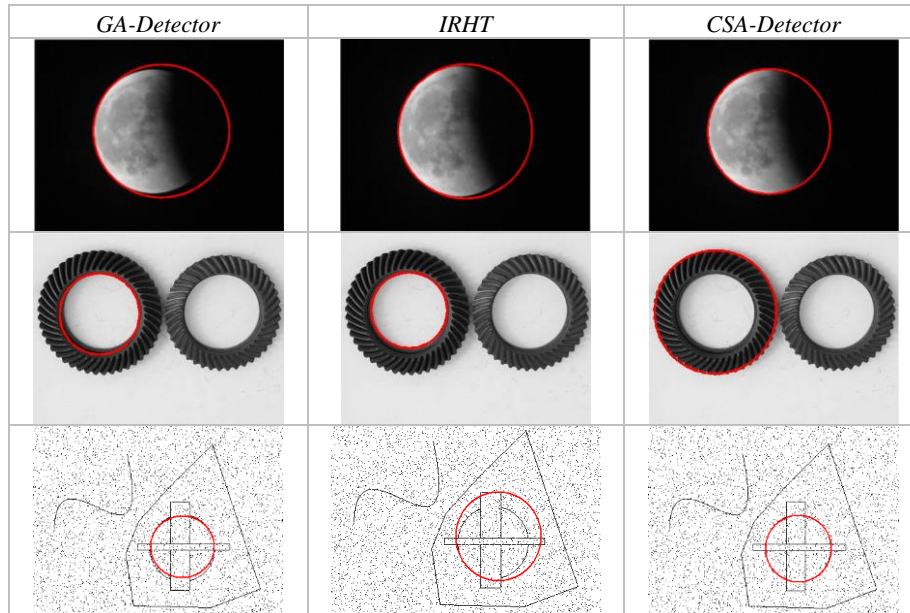


Fig. 8.7. Performance of the CSA method, the GA and the IRHT over complex images.

Image	Average time \pm Standard deviation (s)			Success rate (SR) (%)			Es \pm Standard deviation		
	GA	IRHT	CSA	GA	IRHT	CSA	GA	IRHT	CSA
Synthetic noisy images									
(a)	2.11 $\pm(0.31)$	3.04 $\pm(0.29)$	0.57 $\pm(\mathbf{0.13})$	100	92	100	0.87 $\pm(0.071)$	0.71 $\pm(0.051)$	0.54 $\pm(\mathbf{0.071})$
(b)	2.50 $\pm(0.39)$	2.80 $\pm(0.17)$	0.51 $\pm(\mathbf{0.11})$	91	80	97	0.67 $\pm(0.081)$	0.61 $\pm(0.048)$	0.31 $\pm(\mathbf{0.015})$
(c)	3.02 $\pm(0.63)$	4.11 $\pm(0.71)$	0.64 $\pm(\mathbf{0.33})$	93	78	100	0.71 $\pm(0.036)$	0.77 $\pm(0.044)$	0.42 $\pm(\mathbf{0.011})$

Table 8.3. Averaged execution time and success rate of the GA, the IRHT and the CSA method over three noisy images shown by Figure 8.4.

8.5. Conclusions

This chapter has presented an algorithm for the automatic detection of circular shapes from complicated and noisy images with no consideration of the conventional Hough transform principles. The presented method is based on a newly developed Artificial Immune Optimization (AIO) technique, known as the Clonal Selection algorithm (CSA). To the best of our knowledge, the CSA has not been yet applied to any such circle detection task until date. The algorithm uses the encoding of three non-collinear edge points as circle candidates within the edge image of the scene. An objective function evaluates if a given circle candidate is actually present in the edge image (Ag). Guided by the values of the objective function, the set of encoded candidate circles are evolved using the CSA so that they can fit into the actual circles in the edge map of the image. As it can be

observed from the results shown by Figures 8.4, 8.5 and 8.7, our approach detects the circle in complex images with little visual distortion despite the presence of noisy background pixels.

An important feature is to consider the circle detection problem as an optimization approach. Such view enables the algorithm to detect arcs or occluded circles still matching imperfect circles. The CSA is capable of finding circle parameters according to $J(C)$ instead of making a review of all circle candidates towards detecting occluded or imperfect circles as it is commonly done by other methods.

In order to test the circle detection accuracy, a score function is used (Eq. 8.22) following the work in (Dasgupta et al., 2010). It can objectively evaluate the mismatch between a manually detected circle and a machine-detected shape. We demonstrated that the CSA method outperforms both the GA (as described in (Ayala-Ramirez et al., 2006)) and the IRHT (as described in (Lu & Tan, 2008)) within a statistically significant framework.

Although the Hough Transform methods for circle detection also use three edge points to cast a vote for the potential circular shape in the parameter space, they would require huge amounts of memory and longer computational time to obtain a sub-pixel resolution. In the HT-based methods, the parameter space is quantized and the exact parameters for a circle are often not equal to the quantized parameters, therefore it rarely finds the exact parameters of a circle in the image (Chen & Chung, 2001). However, the presented CSA method does not employ the quantization of the parameter space. In our approach, the detected circles are directly obtained from equations 8.10–8.13, still reaching sub-pixel accuracy.

Although Figure 8.6 indicates that the CSA method can yield better results on complicated and noisy images in comparison to the GA and the IRHT methods, notice that the aim of this chapter is to show that the Artificial Immune Systems can effectively serve as an attractive alternative to evolutionary algorithms which have been employed before to successfully extract circular shapes in images.

Chapter 9

States of Matter Algorithm applied to Pattern Detection

Pattern Detection (PD) plays an important role in several image processing applications such as feature tracking, object recognition, stereo matching and remote sensing. PD involves two critical aspects: similarity measurement and search strategy. The simplest available PD method finds the best possible coincidence between the images through an exhaustive computation of the Normalized cross-correlation (NCC) values (similarity measurement) for all elements of the source image (search strategy). However, the use of such approach is strongly restricted, since the NCC evaluation is a computationally expensive operation. Recently, several PD algorithms, based on evolutionary approaches, have been proposed to reduce the number of NCC operations by calculating only a subset of search locations. In this chapter, is presented an algorithm based on the States of Matter with the purpose of reduce the number of search locations in the PD process. In the presented approach, individuals emulate molecules that experiment state transitions which represent different exploration–exploitation levels. In the algorithm, the computation of search locations is drastically reduced by incorporating a fitness calculation strategy which indicates when it is feasible to calculate or only estimate the NCC value for new search locations. Conducted simulations show that the presented method achieves the best balance over other PD algorithms, in terms of estimation accuracy and computational cost.

9.1. Introduction

Pattern Detection (PD), which measures the degree of similarity between two image sets that are superimposed on one another, is one of the most important and challenging subjects in digital photogrammetry, object recognition, stereo matching, feature tracking, remote sensing, and computer vision (Brunelli, 2009). It relies on calculating at each position of the image under examination a correlation or distortion function that measures the degree of similarity to the template image, and the best matching is obtained when the similarity value is maximized.

Generally, a process like pattern detection, involves two critical aspects: similarity measurement and search strategy (Grailu, et al., 2009). It is used a matching criterion, typically the Normalized cross-correlation (NCC) which is computationally expensive and represents the most consuming operation in the PD process (Krattenthaler, et al., 1994).

The full search algorithm (Rosenfeld & VanderBrug, 1977; Tanimoto, 1981; Uenohara & Kanade, 1997) is the simplest PD algorithm that can deliver the optimal detection with respect to a maximal NCC coefficient as it checks all pixel-candidates one at a time. However, such exhaustive search and the NCC calculation at each checking point, yields an extremely computational expensive PD method that seriously constraints its use for several image processing applications.

Recently, several PD algorithms, based on evolutionary approaches, have been proposed to reduce the number of NCC operations by calculating only a subset of search locations. Such approaches have produced several robust detectors using different optimization methods such as Genetic algorithms (GA) (Dong et al., 2011), Particle Swarm Optimization (PSO) (Liu, et al., 2012; Wu et al., 2009) and Imperialist competitive algorithm (ICA) (Xu, et al., 2010). Although these algorithms

allow reducing the number of search locations, they do not explore the whole region effectively and often suffers premature convergence which conducts to sub-optimal detections. The reason of these problems is the operators used for modifying the particles. In such algorithms, during their evolution, the position of each agent in the next iteration is updated yielding an attraction towards the position of the best particle seen so-far (Adra & Fleming, 2011; Chen, et al., 2009). This behavior produces that the entire population, as the algorithm evolves, concentrates around the best particle, favoring the premature convergence and damaging the particle diversity.

Every evolutionary algorithm (EA) needs to address the issue of exploration-exploitation of the search space. Exploration is the process of visiting entirely new points of a search space whilst exploitation is the process of refining those points within the neighborhood of previously visited locations, in order to improve their solution quality. Pure exploration degrades the precision of the evolutionary process but increases its capacity to find new potential solutions.

On the other hand, pure exploitation allows refining existent solutions but adversely driving the process to local optimal solutions. Therefore, the ability of an EA to find a global optimal solution depends on its capacity to find a good balance between the exploitation of found-so-far elements and the exploration of the search space (Tan et al., 2009). So far, the exploration–exploitation dilemma has been an unsolved issue within the framework of EA.

In this chapter, a novel nature-inspired algorithm, called the States of Matter Search (SMS) is proposed for solving the PD problem. The SMS algorithm is based on the simulation of the states of matter phenomenon. In SMS, individuals emulate molecules which interact to each other by using evolutionary operations based on the physical principles of the thermal-energy motion mechanism. Such operations allow the increase of the population diversity and avoid the concentration of particles within a local minimum.

The presented approach combines the use of the defined operators with a control strategy that modifies the parameter setting of each operation during the evolution process. The algorithm is devised by considering each state of matter at one different exploration–exploitation rate. Thus, the evolutionary process is divided into three stages which emulate the three states of matter: gas, liquid and solid. At each state, molecules (individuals) exhibit different behaviors. Beginning from the gas state (pure exploration), the algorithm modifies the intensities of exploration and exploitation until the solid state (pure exploitation) is reached. As a result, the approach can substantially improve the balance between exploration–exploitation yet preserving the good search capabilities of an evolutionary approach.

However, one particular difficulty in applying any EA to real-world problems is about its demand for a large number of fitness evaluations before delivering a satisfying result. Fitness evaluations are not always straightforward in many applications as either an explicit fitness function does not exist, or the fitness evaluation is computationally expensive. Furthermore, since random numbers are involved in the calculation of new individuals, they may encounter same positions (repetition) that have been visited by other individuals at previous iterations, particularly when individuals are confined to a finite area.

The problem of considering expensive fitness evaluations has already been faced in the field of evolutionary algorithms (EA) and is better known as fitness approximation (Jin, 2005). In such approach, the idea is to estimate the fitness value of so many individuals as it is possible instead of evaluating the complete set. Such estimations are based on an approximate model of the fitness landscape. Thus, the individuals to be evaluated and those to be estimated are determined following some fixed criteria which depend on the specific properties of the approximate model (Jin, 2011b).

The models involved at the estimation can be built during the actual EA run, since EA repeatedly samples the search space at different points (Branke & Schmidt, 2005). There are many possible approximation models which have been used in combination with EA (e.g. polynomials (Zhou et al., 2005), the kriging model (Ratle, 2001), the feed-forward neural networks that includes multi-layer Perceptrons (Lim, et al., 2010b) and radial basis-function networks (Ong, et al., 2008b)).

In this chapter, a new algorithm based on SMS is presented to reduce the number of search locations in the PD process. The algorithm uses a simple fitness calculation approach which is based on the Nearest Neighbor Interpolation (NNI) algorithm in order to estimate the fitness value (NCC operation) for several candidate solutions (search locations). As a result, the approach can not only substantially reduce the number search positions (by using the SMS approach), but also to avoid the NCC evaluation for many of them (by incorporating the NNI strategy). The presented method achieves the best balance over other PD algorithms, in terms of both estimation accuracy and computational cost.

The overall chapter is organized as follows: Section 9.2 holds a description about the SMS algorithm. In Section 9.3, the fitness calculation strategy for solving the expensive optimization problem is presented. Section 9.4 provides backgrounds about the PD process while Section 9.5 exposes the final PD algorithm as a combination of SMS and the fitness calculation strategy. Section 9.6 demonstrates experimental results for the presented approach over standard test images and some conclusions are drawn in Section 9.7.

9.2. SMS-Algorithm

The matter can take different phases which are commonly known as states. Traditionally, three states of matter are known: solid, liquid, and gas. The differences among such states are based on forces which are exerted among particles composing a material (Ceruti & Rubin, 2007).

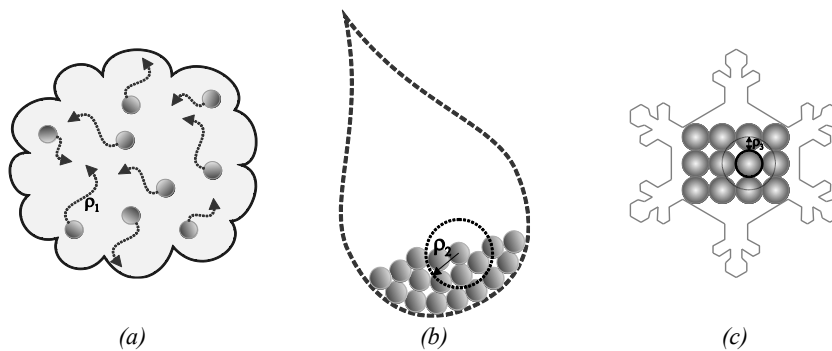


Fig. 9.1. Different states of matter: (a) gas, (b) liquid, and (c) solid.

In the gas phase, molecules present enough kinetic energy so that the effect of intermolecular forces is small (or zero for an ideal gas), while the typical distance between neighboring molecules is greater than the molecular size. A gas has no definite shape or volume but occupies the entire container in which it is confined. Fig. 9.1(a) shows the movements exerted by particles in a gas state. The movement experimented by the molecules represent the maximum permissible displacement ρ_1 among particles (Chowdhury & Stauffer, 2000). In a liquid state, intermolecular forces are more restrictive than those in the gas state. The molecules have enough energy to move relatively to each other still keeping a mobile structure. Therefore, the shape of a liquid is not definite but is determined by its container.

Fig. 9.1(b) presents a particle movement ρ_2 within a liquid state. Such movement is smaller than those considered by the gas state but larger than the solid state. In the solid state, particles (or molecules) are packed together closely with forces among particles being strong enough so that the particles cannot move freely but only vibrate. As a result, a solid has a stable, definite shape and a definite volume. Solids can only change their shape by force, as when they are broken or cut.

Fig. 9.1(c) shows a molecule configuration in a solid state. Under such conditions, particles are able to vibrate (being perturbed) considering a minimal ρ_3 distance (Chowdhury et al., 2000). In this chapter, a nature-inspired algorithm known as the States of Matter Search (SMS) is presented for solving global optimization problems. The SMS algorithm is based on the simulation of the states of matter phenomenon that considers individuals as molecules which interact to each other by using evolutionary operations based on the physical principles of the thermal-energy motion mechanism. The algorithm is devised by considering each state of matter at one different exploration–exploitation ratio. Thus, the evolutionary process is divided into three stages which emulate the three states of matter: gas, liquid and solid. In each state, individuals exhibit different behaviors.

9.3. States of matter search (SMS)

9.3.1. Definition of Operators

In the approach, individuals are considered as molecules whose positions on a multidimensional space are modified as the algorithm evolves. The movement of such molecules is motivated by the analogy to the motion of thermal-energy.

The velocity and direction of each molecule's movement are determined by considering the collision, the attraction forces and the random phenomena experimented by the molecule set (Cengel, 2014). In our approach, such behaviors have been implemented by defining several operators such as the direction vector, the collision and the random positions operators, all of which emulate the behavior of actual physics laws.

The direction vector operator assigns a direction to each molecule in order to lead the particle movement as the evolution process takes place. On the other side, the collision operator mimics those collisions that are experimented by molecules as they interact to each other. A collision is considered when the distance between two molecules is shorter than a determined proximity distance. The collision operator is thus implemented by interchanging directions of the involved molecules.

In order to simulate the random behavior of molecules, the proposed algorithm generates random positions following a probabilistic criterion that considers random locations within a feasible search space.

The next section presents all operators that are used in the algorithm. Although such operators are the same for all the states of matter, they are employed over a different configuration set depending on the particular state under consideration.

9.3.1.1. Direction vector

The direction vector operator mimics the way in which molecules change their positions as the evolution process develops. For each n -dimensional molecule \mathbf{p}_i from the population \mathbf{P} , it is assigned an n -dimensional direction vector \mathbf{d}_i which stores the vector that controls the particle

movement. Initially, all the direction vectors ($\mathbf{D} = \{\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_{N_p}\}$) are randomly chosen within the range of $[-1, 1]$.

As the system evolves, molecules experiment several attraction forces. In order to simulate such forces, the proposed algorithm implements the attraction phenomenon by moving each molecule towards the best so-far particle.

Therefore, the new direction vector for each molecule is iteratively computed considering the following model:

$$\mathbf{d}_i^{k+1} = \mathbf{d}_i^k \cdot \left(1 - \frac{k}{gen}\right) \cdot 0.5 + \mathbf{a}_i, \quad (9.1)$$

where \mathbf{a}_i represents the attraction unitary vector calculated as $\mathbf{a}_i = (\mathbf{p}^{best} - \mathbf{p}_i) / \|\mathbf{p}^{best} - \mathbf{p}_i\|$, being \mathbf{p}^{best} the best individual seen so-far, while \mathbf{p}_i is the molecule i of population \mathbf{P} . k represents the iteration number whereas gen involves the total iteration number that constitutes the complete evolution process.

Under this operation, each particle is moved towards a new direction which combines the past direction, which was initially computed, with the attraction vector over the best individual seen so-far. It is important to point out that the relative importance of the past direction decreases as the evolving process advances. This particular type of interaction avoids the quick concentration of information among particles and encourages each particle to search around a local candidate region in its neighborhood, rather than interacting to a particle lying at distant region of the domain.

The use of this scheme has two advantages: first, it prevents the particles from moving toward the global best position in early stages of algorithm and thus makes the algorithm less susceptible to premature convergence; second, it encourages particles to explore their own neighborhood thoroughly, just before they converge towards a global best position. Therefore, it provides the algorithm with local search ability enhancing the exploitative behavior.

In order to calculate the new molecule position, it is necessary to compute the velocity \mathbf{v}_i of each molecule by using:

$$\mathbf{v}_i = \mathbf{d}_i \cdot v_{init} \quad (9.2)$$

being v_{init} the initial velocity magnitude which is calculated as follows:

$$v_{init} = \frac{\sum_{j=1}^n (b_j^{high} - b_j^{low})}{n} \cdot \beta \quad (9.3)$$

where b_j^{low} and b_j^{high} are the low j parameter bound and the upper j parameter bound respectively, whereas $\beta \in [0, 1]$.

Then, the new position for each molecule is updated by:

$$p_{i,j}^{k+1} = p_{i,j}^k + v_{i,j} \cdot \text{rand}(0,1) \cdot \rho \cdot (b_j^{high} - b_j^{low}) \quad (9.4)$$

where $0.5 \leq \rho \leq 1$.

9.3.1.2. Collision

The collision operator mimics the collisions experimented by molecules while they interact to each other. Collisions are calculated if the distance between two molecules is shorter than a determined proximity value. Therefore, if $\|\mathbf{p}_i - \mathbf{p}_q\| < r$, a collision between molecules i and q is assumed; otherwise, there is no collision, considering $i, q \in \{1, \dots, N_p\}$ such that $i \neq q$. If a collision occurs, the direction vector for each particle is modified by interchanging their respective direction vectors as follows:

$$\mathbf{d}_i = \mathbf{d}_q \text{ and } \mathbf{d}_q = \mathbf{d}_i \quad (9.5)$$

The collision radius is calculated by:

$$r = \frac{\sum_{j=1}^n (b_j^{high} - b_j^{low})}{n} \cdot \alpha \quad (9.6)$$

where $\alpha \in [0, 1]$.

Under this operator, a spatial region enclosed within the radius r is assigned to each particle. In case the particle regions collide to each other, the collision operator acts upon particles by forcing them out of the region. The radius r and the collision operator provide the ability to control diversity throughout the search process. In other words, the rate of increase or decrease of diversity is predetermined for each stage.

Unlike other diversity-guided algorithms, it is not necessary to inject diversity into the population when particles gather around a local optimum because the diversity will be preserved during the overall search process. The collision incorporation therefore enhances the exploratory behavior in the proposed approach.

9.3.1.3. Random positions

In order to simulate the random behavior of molecules, the proposed algorithm generates random positions following a probabilistic criterion within a feasible search space.

For this operation, a uniform random number r_m is generated within the range $[0, 1]$. If r_m is smaller than a threshold H , a random molecule's position is generated; otherwise, the element remains with no change. Therefore, such operation can be modeled as follows:

$$p_{i,j}^{k+1} = \begin{cases} b_j^{low} + \text{rand}(0,1) \cdot (b_j^{high} - b_j^{low}) & \text{with probability } H \\ p_{i,j}^{k+1} & \text{with probability } (1-H) \end{cases} \quad (9.7)$$

where $i \in \{1, \dots, N_p\}$ and $j \in \{1, \dots, n\}$.

9.3.1.4. Best Element Updating

Despite this updating operator does not belong to State of Matter metaphor, it is used to simply store the best so-far solution. In order to update the best molecule \mathbf{p}^{best} seen so-far, the best found

individual from the current k population $\mathbf{p}^{best,k}$ is compared to the best individual $\mathbf{p}^{best,k-1}$ of the last generation. If $\mathbf{p}^{best,k}$ is better than $\mathbf{p}^{best,k-1}$ according to its fitness value, \mathbf{p}^{best} is updated with $\mathbf{p}^{best,k}$, otherwise \mathbf{p}^{best} remains with no change. Therefore, \mathbf{p}^{best} stores the best historical individual found so-far.

9.3.2. SMS algorithm

The overall SMS algorithm is composed of three stages corresponding to the three States of Matter: the gas, the liquid and the solid state. Each stage has its own behavior. In the first stage (gas state), exploration is intensified whereas in the second one (liquid state) a mild transition between exploration and exploitation is executed. Finally, in the third phase (solid state), solutions are refined by emphasizing the exploitation process.

9.3.2.1. General procedure

At each stage, the same operations are implemented. However, depending on which state is referred, they are employed considering a different parameter configuration. The general procedure in each state is shown as pseudo-code in Algorithm 9.1. Such procedure is composed by five steps and maps the current population \mathbf{P}^k to a new population \mathbf{P}^{k+1} . The algorithm receives as input the current population \mathbf{P}^k and the configuration parameters ρ , β , α , and H , whereas it yields the new population \mathbf{P}^{k+1} .

9.3.2.2. The complete algorithm

The complete algorithm is divided into four different parts. The first corresponds to the initialization stage, whereas the last three represent the States of Matter. All the optimization process, which consists of a *gen* number of iterations, is organized into three different asymmetric phases, employing 50% of all iterations for the gas state (exploration), 40% for the liquid state (exploration-exploitation) and 10% for the solid state (exploitation). The overall process is graphically described by Figure 9.2. At each state, the same general procedure (see Algorithm 9.1) is iteratively used considering the particular configuration predefined for each State of Matter. Figure 9.3 shows the data flow for the complete SMS algorithm.

Initialization

The algorithm begins by initializing a set \mathbf{P} of N_p molecules ($\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{N_p}\}$). Each molecule position \mathbf{p}_i is a n -dimensional vector containing the parameter values to be optimized. Such values are randomly and uniformly distributed between the pre-specified lower initial parameter bound b_j^{low} and the upper initial parameter bound b_j^{high} , just as it is described by the following expressions:

$$p_{i,j}^0 = b_j^{low} + rand(0,1) \cdot (b_j^{high} - b_j^{low})$$

$$j = 1, 2, \dots, n; \quad i = 1, 2, \dots, N_p, \quad (9.8)$$

where j and i , are the parameter and molecule index respectively whereas zero indicates the initial population. Hence, p_i^j is the j -th parameter of the i -th molecule.

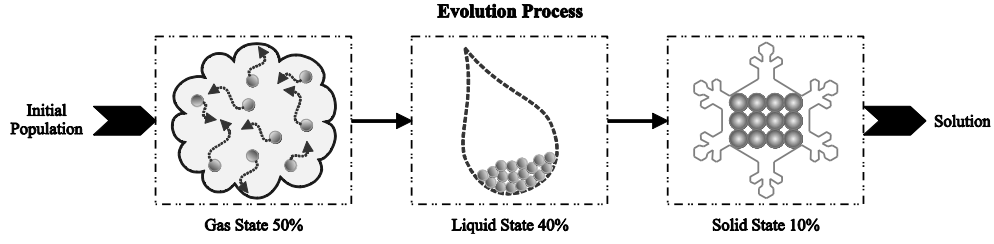


Fig. 9.2. Evolution process in the proposed approach.

Step 1:	Evaluate the fitness value of each particle and find the best element of the population \mathbf{P} $\mathbf{p}^{best} \in \{\mathbf{P}\} J(\mathbf{p}^{best}) = \max \{J(\mathbf{p}_1), J(\mathbf{p}_2), \dots, J(\mathbf{p}_{N_p})\}$
Step 2:	Calculate v_{init} and r $v_{init} = \frac{\sum_{j=1}^n (b_j^{high} - b_j^{low})}{n} \cdot \beta \quad r = \frac{\sum_{j=1}^n (b_j^{high} - b_j^{low})}{n} \cdot \alpha$
Step 3:	Compute the new molecules by using the Direction vector operator 9.3.1.1 <pre> for ($i=1$; $i < N_p + 1$; $i++$) $\mathbf{a}_i = (\mathbf{p}^{best} - \mathbf{p}_i) / \ \mathbf{p}^{best} - \mathbf{p}_i\$ for ($j=1$; $j < n + 1$; $j++$) $d_{i,j}^{k+1} = d_{i,j}^k \cdot \left(1 - \frac{k}{gen}\right) \cdot 0.5 + a_{i,j}$ $v_{i,j} = d_{i,j}^{k+1} \cdot v_{init}$ $p_{i,j}^{k+1} = p_{i,j}^k + v_{i,j} \cdot \text{rand}(0,1) \cdot \rho \cdot (b_j^{high} - b_j^{low})$ end for end for </pre>
Step 4:	Solve collisions by using the Collision operator 9.3.1.2 <pre> for ($i=1$; $i < N_p + 1$; $i++$) for ($j=1$; $j < N_p + 1$; $j++$) if ($(\ \mathbf{p}_i - \mathbf{p}_j\ < r)$ and ($i \neq j$)) $\mathbf{t} = \mathbf{d}_i$ $\mathbf{d}_i = \mathbf{d}_j$ $\mathbf{d}_j = \mathbf{t}$ end if end for end for </pre>
Step 5:	Generate new random positions by using the Random positions operator 9.3.1.3 <pre> for ($i=1$; $i < N_p + 1$; $i++$) if ($r_m < H$) then; where $r_m \in \text{rand}(0,1)$ for ($j=1$; $j < n + 1$; $j++$) $p_{i,j}^{k+1} = b_j^{low} + \text{rand}(0,1) \cdot (b_j^{high} - b_j^{low})$ end for end if end for </pre>

Algorithm 9.1. General procedure executed by all the states of matter.

Gas state

In the gas state, molecules experiment severe displacements and collisions. Such state is characterized by random movements produced by non-modeled molecule phenomena (Cengel, 2014). Therefore, the ρ value from the direction vector operator is set to a value near to one so that the molecules can travel longer distances. Similarly, the H value representing the random positions operator is also configured to a value around one, in order to allow the random generation for other molecule positions. The gas state is the first phase and lasts for the 50% of all iterations which compose the complete optimization process. The computational procedure for the gas state can be summarized as follows:

Step 1	Set the parameters $\rho \in [0.8, 1]$, $\beta = 0.8$, $\alpha = 0.8$ and $H = 0.9$ being consistent with the gas state.
Step 2	Apply the general procedure which is illustrated in Algorithm 9.1.
Step 3	If the 50% of the total iteration number is completed ($1 \leq k \leq 0.5 \cdot gen$), then the process continues to the liquid state procedure; otherwise go back to step 2.

Liquid state

Although molecules currently at the liquid state exhibit restricted motion in comparison to the gas state, they still show a higher flexibility with respect to the solid state. Furthermore, the generation of random positions which are produced by non-modeled molecule phenomena is scarce (Hecht & Bueche, 2011). For this reason, the ρ value from the direction vector operator is bounded to a value between 0.3 and 0.6.

Similarly, the random position operator H is configured to a value near to zero in order to allow the random generation of fewer molecule positions. In the liquid state, collisions are also less common than in gas state, so the collision radius, that is controlled by α , is set to a smaller value in comparison to the gas state.

The liquid state is the second phase and lasts the 40% of all iterations which compose the complete optimization process. The computational procedure for the liquid state can be summarized as follows:

Step 4	Set the parameters $\rho \in [0.3, 0.6]$, $\beta = 0.4$, $\alpha = 0.2$ and $H = 0.2$ being consistent with the liquid state.
Step 5	Apply the general procedure that is defined in Algorithm 9.1.
Step 6	If the 90% (50% from the gas state and 40% from the liquid state) of the total iteration number is completed ($0.5 \cdot gen < k \leq 0.9 \cdot gen$), then the process continues to the solid state procedure; otherwise go back to step 5.

Solid state

In the solid state, forces among particles are stronger so that particles cannot move freely but only vibrate. As a result, effects such as collision and generation of random positions are not considered (Turner & Betts, 1974).

Therefore, the ρ value of the direction vector operator is set to a value near to zero indicating that the molecules can only vibrate around their original positions.

The solid state is the third phase and lasts for the 10% of all iterations which compose the complete optimization process. The computational procedure for the solid state can be summarized as follows:

Step 7	Set the parameters $\rho \in [0, 0.1]$, $\beta = 0.1$, $\alpha = 0$ and $H = 0$ being consistent with the solid state.
Step 8	Apply the general procedure that is defined in Algorithm 9.1.
Step 9	If the 100% of the total iteration number is completed ($0.9 \cdot \text{gen} < k \leq \text{gen}$), the process is finished; otherwise go back to step 8.

It is important to clarify that the use of this particular configuration ($\alpha = 0$ and $H = 0$) disables the collision and generation of random positions operators which have been illustrated in the general procedure.

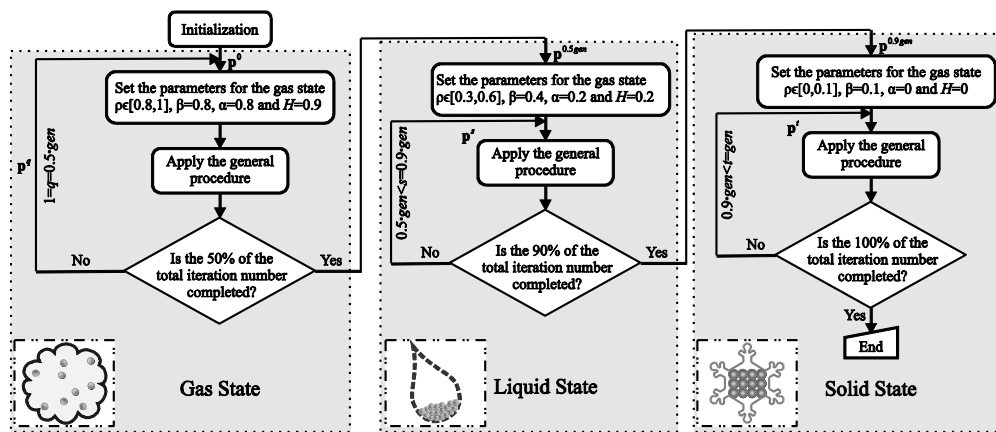


Fig. 9.3. Data flow of the complete SMS algorithm

9.3. Fitness approximation method

Evolutionary methods based on fitness approximation aim to find the global optimum of a given function considering only a very few numbers of function evaluations. In order to apply such approach, it is necessary that the objective function portrait the following conditions: According to (Ratle, 2001) (1) it must be very costly to evaluate and (2) must have few dimensions (up to five). Recently, several fitness estimators have been reported in the literature (Branke & Schmidt, 2005; Jin, 2005; Jin, 2011b; Zhou et al., 2005), where the function evaluation number is considerably reduced (to hundreds, dozens, or even less). However, most of these methods produce complex algorithms whose performance is conditioned to the quality of the training phase and the learning algorithm in the construction of the approximation model.

In this chapter, we explore the use of a local approximation scheme, based on the nearest-neighbor-interpolation (NNI), in order to reduce the function evaluation number. The model estimates the fitness values based on previously evaluated neighboring individuals, stored during the evolution process. At each generation, some individuals of the population are evaluated with the accurate (real) objective function, while the remaining individuals' fitnesses are estimated. The individuals to be evaluated accurately are determined based on their proximity to the best fitness value or uncertainty.

9.3.1. Updating individual database

In our fitness calculation approach, during the evolution process, every evaluation or estimation of an individual produces a data point (individual position and fitness value) that is potentially taken into account for building the approximation model. Therefore, we keep all seen so far evaluations in a history array \mathbf{T} , and then just select the closest neighbor to estimate the fitness value of a new individual. Thus, all data are preserved and potentially available for use, while the construction of the model is still fast since only the most relevant data points are actually used to construct the model.

9.3.2. Fitness calculation strategy

In the presented fitness calculation scheme, most of the fitness values are estimated to reduce the calculation time in each generation. In the model, it is evaluated (using the real fitness function) those individuals that are near the individual with the best fitness value contained in \mathbf{T} (rule 1). Such individuals are important, since they will have a stronger influence on the evolution process than other individuals.

Moreover, it is also evaluated those individuals in regions of the search space with few previous evaluations (rule 2). The fitness values of these individuals are uncertain; since there is no close reference (close points contained in \mathbf{T}) in order to calculate their estimates.

The rest of the individuals are estimated using NNI (rule 3). Thus, the fitness value of an individual is estimated assigning it the same fitness value that the nearest individual stored in \mathbf{T} .

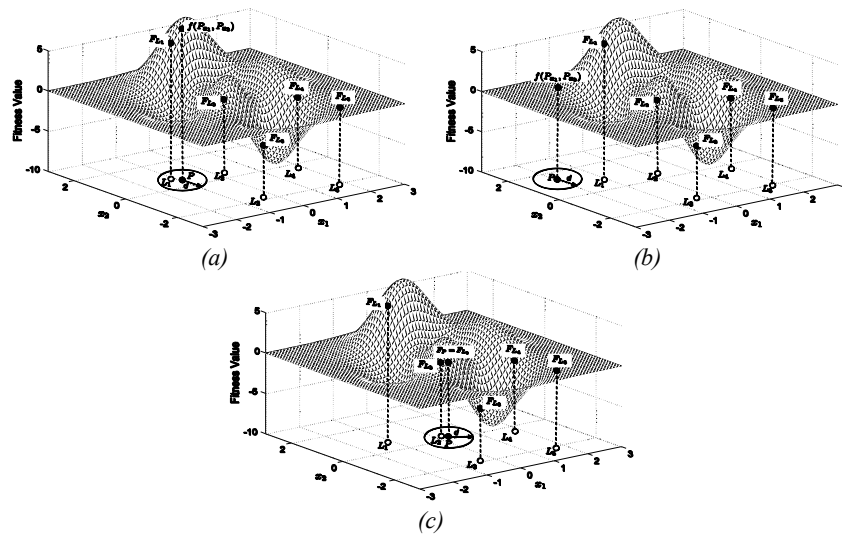


Fig. 9.4. The fitness calculation strategy. (a) According to the rule 1, the individual (search position) P is evaluated ($J(P)$), since it is located closer than a distance d with respect to the nearest individual location L_1 whose fitness value F_{L_1} corresponds to the best fitness value (maximum so-far). (b) According to the rule 2, the search point P is evaluated ($J(P)$), as there is no close reference in its neighborhood. (c) According to the rule 3, the fitness value of P is estimated ($\tilde{J}(P)$) by means of the NNI-estimator, assigning $F_P = F_{L_2}$.

For the sake of clarity, it is considered that the fitness value of i is evaluated by the true fitness function using the representation $J(i)$ whereas $\tilde{J}(i)$ indicates that the fitness value of the individual i has been estimated using an alternative model. Therefore, the estimation model follows 3 different rules in order to evaluate or estimate the fitness values:

1. If the new individual (search position) P is located closer than a distance d with respect to the nearest individual location L_q whose fitness value F_{L_q} corresponds to the best fitness value stored in \mathbf{T} , then the fitness value of P is evaluated using the true fitness function ($J(P)$). Figure 9.4(a) draws the rule procedure.
2. If the new individual P is located longer than a distance d with respect to the nearest individual location L_q whose fitness value F_{L_q} has been already stored in \mathbf{T} , then its fitness value is evaluated using the true fitness function ($J(P)$). Figure 9.4(b) outlines the rule procedure.
3. If the new individual P is located closer than a distance d with respect to the nearest individual location L_q whose fitness value F_{L_q} has been already stored in \mathbf{T} , then its fitness value is estimated ($\tilde{J}(P)$) assigning it the same fitness that L_q ($F_p = F_{L_q}$). Figure 9.4(c) sketches the rule procedure.

The d value controls the trade off between the evaluation and estimation of search locations. Typical values of d range from 5 to 10; in this chapter, the value of 7 has been selected. Thus, the proposed approach favors the exploitation and exploration in the search process. For the exploration, the estimator evaluates the true fitness function of new search locations that have been located far from the positions already calculated. Meanwhile, it also estimates those that are closer. For the exploitation, the proposed method evaluates the effective fitness function of those new searching locations that are placed near to the position with the minimum fitness value seen so far, aiming to improve its minimum.

Step 1:	Evaluate or estimate the fitness value of each particle and find the best element of the population \mathbf{P} for ($i=1; i < N_p + 1; i++$) If (\mathbf{p}_i fulfils rule 1 or rule 2) then $J(\mathbf{p}_i)$ If (\mathbf{p}_i fulfils rule 3) then $\tilde{J}(\mathbf{p}_i)$ update \mathbf{T} end for $\mathbf{p}^{best} \in \{\mathbf{T}\} J(\mathbf{p}^{best}) = \max \{ J(\mathbf{p}_1), \tilde{J}(\mathbf{p}_2), \dots, J(\mathbf{p}_{N_p}) \}$
Step 2:	Calculate v_{init} and r $v_{init} = \frac{\sum_{j=1}^n (b_j^{high} - b_j^{low})}{n} \cdot \beta \quad r = \frac{\sum_{j=1}^n (b_j^{high} - b_j^{low})}{n} \cdot \alpha$
The other steps are similar to those presented in Algorithm 9.1.	

Algorithm 9.2. Enhanced general procedure executed by all the states of matter. The procedure incorporates the fitness calculation strategy in order to reduce the number of function evaluations.

The three rules show that the fitness calculation strategy is simple and straightforward. Fig. 9.4 illustrates the procedure of fitness computation for a new solution (point P) considering the three different rules. In the problem, the objective function J is maximized with respect to two param-

ters (x_1, x_2) . In all figures (Figs. 9.4(a), (b) and (c)) the individual database array \mathbf{T} contains five different elements $(L_1, L_2, L_3, L_4, L_5)$ with their corresponding fitness values $(F_{L_1}, F_{L_2}, F_{L_3}, F_{L_4}, F_{L_5})$.

Figures 9.4(a) and (b) show the fitness evaluation $(J(P_{x_1}, P_{x_2}))$ of the new solution P following the rule 1 and 2 respectively, whereas Fig. 9.4(c) present the fitness estimation of P $(\tilde{J}(P))$ using the NNI approach considered by rule 3.

9.3.3. Proposed optimization SMS method

In this section, it has been proposed a fitness calculation approach in order to accelerate the SMS algorithm. Only the fitness calculation scheme shows difference between the proposed SMS and the enhanced one. In the modified SMS, only some individuals are actually evaluated (rules 1 and 2) in each generation. The fitness values of the rest are estimated using the NNI-approach (rule 3). The estimation is executed using the individual database (array \mathbf{T}).

Fig. 9.5 shows the difference between the original SMS and the modified one. In the Figure, it is clear that two new blocks have been added, the fitness estimation and the updating individual database. Both elements, together with the actual evaluation block, represent the fitness calculation strategy presented in this sub-section. The incorporation of the fitness calculation strategy modifies only the step 1 of the general procedure shown in Algorithm 9.1. Such step is extended by incorporating the decision rules (whether the individual i is $J(i)$ or $\tilde{J}(i)$) and the sub-system that updates the \mathbf{T} array. Algorithm 9.2 illustrates the enhanced procedure. As a result, the SMS approach can substantially reduce the number of function evaluations preserving its good search capabilities.

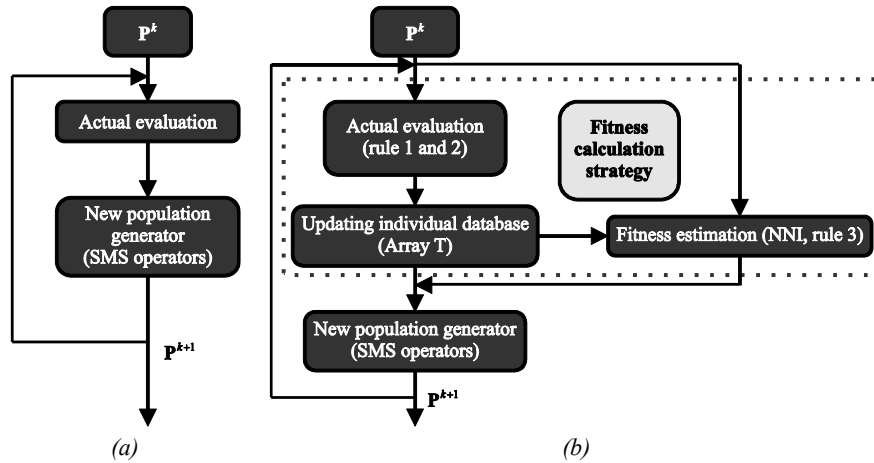


Fig. 9.5. Differences between the original SMS and the modified SMS. (a) Conventional SMS and (b) SMS algorithm included the fitness calculation strategy

9.4. Pattern Detection process

Consider the problem of localizing a given reference image (template) R within a larger intensity image I , which we call the source image. The task is to find those positions where the contents of the reference image R and the corresponding sub-image of I are either the same or most similar. If it is denoted by $R_{u,v}(x, y) = R(x - u, y - v)$, the reference image R shifted by the distance (u, v)

in the horizontal and vertical directions, respectively, then the matching problem (illustrated in Fig. 9.6) can be summarized as:

Given are the source image I and the reference image R .

Find the offset (u,v) inside of the search region S , such that the similarity between the shifted reference image $T_{u,v}$, and

The corresponding sub-image of I is a maximum.

To successfully solve this task, several issues need to be addressed such as determining a minimum similarity value for accepting a match and developing a good search strategy for finding, in a fast way, the optimal displacement.

Several Pattern Detection algorithms (Chen et al., 2009; Dong et al., 2011; Liu et al., 2012; Wu et al., 2009; Xu et al., 2010) have been proposed to reduce the number of search positions, using evolutionary approaches as search strategy. Among the similarity criteria, NCC is the most effective and robust method that allow to measure the resemblance between R and its coincident region within I , at each displacement (u,v) .

The NCC value between a given image I of size $M \times N$, and a template image R of size $m \times n$ at the displacement (u,v) is given by:

$$NCC(u,v) = \frac{\sum_{i=1}^m \sum_{j=1}^n [I(u+i, v+j) - \bar{I}(u,v)] \cdot [R(i,j) - \bar{R}]}{\left[\sum_{i=1}^m \sum_{j=1}^n I(u+i, v+j) - \bar{I}(u,v) \right]^{\frac{1}{2}} \cdot \left[\sum_{i=1}^m \sum_{j=1}^n R(i,j) - \bar{R} \right]^{\frac{1}{2}}} \quad (9.9)$$

where $\bar{I}(u,v)$ is the grey-scale average intensity of the source image in the region coincident with the template image R and \bar{R} is the grey-scale average intensity of the template image. Such values are defined as:

$$\begin{aligned} \bar{I}(u,v) &= \frac{1}{m \cdot n} \sum_{i=1}^m \sum_{j=1}^n I(u+i, v+j) \\ \bar{R} &= \frac{1}{m \cdot n} \sum_{i=1}^m \sum_{j=1}^n R(i,j) \end{aligned} \quad (9.10)$$

Therefore, the point (u,v) which presents the best possible resemblance between R and I is defined as follows:

$$(u,v) = \arg \max_{(\hat{u}, \hat{v}) \in S} NCC(\hat{u}, \hat{v}) \quad (9.11)$$

where $S = \{(\hat{u}, \hat{v}) \mid 1 \leq \hat{u} \leq M - m, 1 \leq \hat{v} \leq N - n\}$.

Fig. 9.7 illustrates the PD process considering Fig. 9.7(a) and 9.7(b) as the source and template image respectively. It is important to point out that the template image (9.7(b)) is similar but not equal to the coincident pattern, contained in the source image (9.7(a)). Fig. 9.7(c) shows the NCC values (color-encoded) calculated in all locations of the search region S . On the other hand, Fig. 9.7(d) presents the NCC surface which exhibits the highly multi-modality nature of the PD problem.

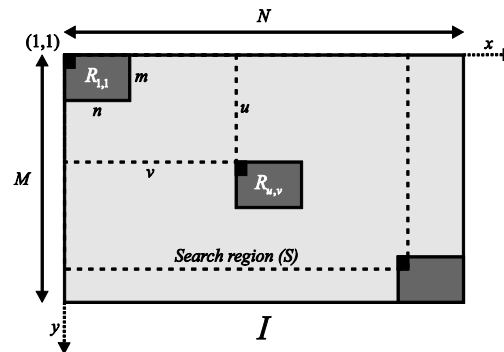


Fig. 9.6. Geometry of pattern detection. The reference image R is shifted across the search image I by an offset (u, v) using the origins of the two images as the reference points. The dimensions of the source image $(M \times N)$ and the reference image $(m \times n)$ determine the maximal search region (S) for this comparison.

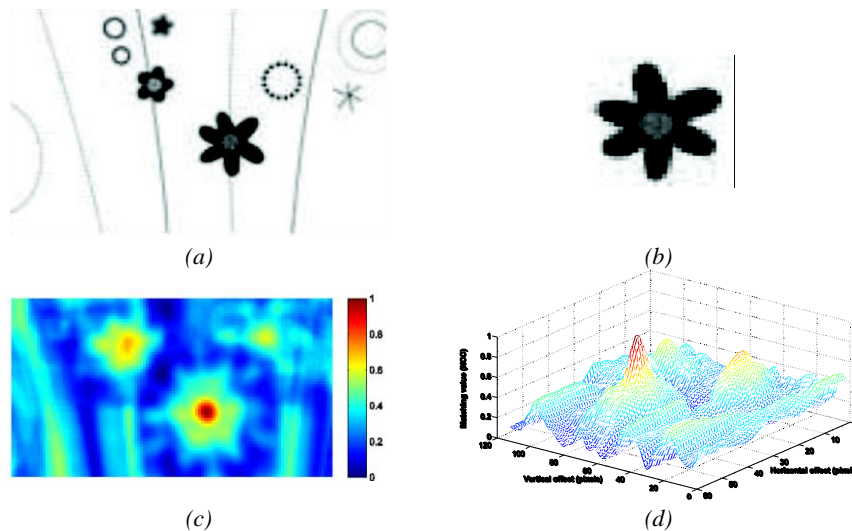


Fig. 9.7. Pattern detection process. (a) Example source image, (b) template image, (c) color-encoded NCC values and (d) NCC multi-modal surface.

9.5. PD algorithm based on SMS with the estimation strategy

The simplest available PD method finds the global maximum (the accurate detection point (u, v)), considering all locations within the search space S . Nevertheless, the approach has a high computational cost for its practical use. Several PD algorithms (Dong et al., 2011; Liu et al., 2012; Wu et al., 2009; Xu et al., 2010) have been proposed to accelerate the search process by calculating only a subset of search locations. Although these algorithms allow reducing the number of search locations, they do not explore the whole region effectively and often suffers premature convergence which conducts to sub-optimal detections. The cause of these problems is the operators used for modifying the particles. In such algorithms, during their evolution, the position of each agent in the next iteration is updated yielding an attraction towards the position of the best particle seen so-far (Lim et al., 2010b; Ong et al., 2008b). This behavior produces that the entire population, as the algorithm evolves, concentrates around the best coincidence seen so-far, favoring the premature convergence in a local minimum of the multi-modal surface. Therefore, a better PD al-

gorithm should spend less computational time on the search strategy and get the optimum match position.

In the SMS-based algorithm, individuals represent search positions (u, v) which move throughout the search space S . The NCC coefficient, used as a fitness value, evaluates the matching quality presented between the template image R and the source image I , for a determined search position (individual). The number of NCC evaluations is drastically reduced by considering a fitness calculation strategy which indicates when it is feasible to calculate or only estimate the NCC values for new search locations. Guided by the fitness values (NCC coefficients), the set of encoded candidate positions are evolved using the SMS operators until the best possible resemblance has been found. In the algorithm, the search space S consists of a set of 2-D search positions \hat{u} and \hat{v} representing the x and y components of the detection locations, respectively. Each particle is thus defined as:

$$P_i = \{(\hat{u}_i, \hat{v}_i) \mid 1 \leq \hat{u}_i \leq M - m, 1 \leq \hat{v}_i \leq N - n\} \quad (9.12)$$

9.5.2. The SMS-PD algorithm

The goal of our PD-approach is to reduce the number of evaluations of the NCC values (actual fitness function) avoiding any performance loss and achieving the optimal solution. The SMS-PD method is listed below:

Step 1	Set the SMS parameters.
Step 2	Initialize the population of 5 random individuals $\mathbf{P} = \{P_1, \dots, P_5\}$ inside of the search region S and the individual database array \mathbf{T} , as an empty array.
Step 3	Compute the fitness values for each individual according to the fitness calculation strategy presented in Section 9.3.
Step 4	Update new evaluations in the individual database array \mathbf{T} .
Step 5	Set the parameters $\rho \in [0.8, 1]$, $\beta = 0.8$, $\alpha = 0.8$ and $H=0.9$ being consistent with the gas state.
Step 6	Apply the enhanced general procedure which is illustrated in Algorithm 9.2.
Step 7	If the 50% of the total iteration number is completed ($1 \leq k \leq 0.5 \cdot \text{gen}$), then the process continues to the liquid state procedure; otherwise go back to step 6.
Step 8	Set the parameters $\rho \in [0.3, 0.6]$, $\beta = 0.4$, $\alpha = 0.2$ and $H=0.2$ being consistent with the liquid state.
Step 9	Apply the enhanced general procedure which is illustrated in Algorithm 9.2.
Step 10	If the 90% (50% from the gas state and 40% from the liquid state) of the total iteration number is completed ($0.5 \cdot \text{gen} < k \leq 0.9 \cdot \text{gen}$), then the process continues to the solid state procedure; otherwise go back to step 9.
Step 11	Set the parameters $\rho \in [0.0, 0.1]$ and $\beta = 0.1$, $\alpha = 0$ and $H=0$ being consistent with the solid state.
Step 12	Apply the enhanced general procedure which is illustrated in Algorithm 9.2.
Step 13	If the 100% of the total iteration number is completed ($0.9 \cdot \text{gen} < k \leq \text{gen}$), the process is finished; otherwise go back to step 11.
Step 14	If the number of target iterations has been reached, then determine the best individual (matching position) of the final population is $\hat{u}_{best}, \hat{v}_{best}$.

The proposed SMS-PD algorithm considers multiple search locations during the complete optimization process. However, only a few of them are evaluated using the true fitness function whereas all other remaining positions are just estimated.

Figure 9.8 shows a search-pattern that has been generated by the SMS-PD approach considering the problem exposed in Fig. 9.7. Such pattern exhibits the evaluated search-locations (rule 1 and 2) in red-cells, whereas the minimum location is marked in green. Blue-cells represent those that have been estimated (rule 3) whereas gray-intensity-cells were not visited at all, during the optimization process.

Since most of fast PD methods employ optimization algorithms that face difficulties with multi-modal surfaces, they may get trapped into local minima and find sub-optimal detections. On the other hand, the proposed approach allows finding out the optimal solution due to a better balance between the exploration and exploitation of the search space.

Under the effect of the SMS operators, the search locations vary from generation to generation, avoiding staying trapped into a local minimum. Besides, since the proposed algorithm uses a fitness calculation strategy for reducing the evaluation of the NCC values, it requires fewer search positions.

As example Fig. 9.8 shows how the SMS-PD algorithm found the optimal detection, evaluating only the 11% of the feasible search locations.

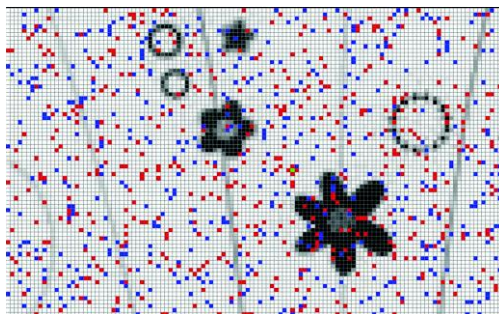


Fig. 9.8. Search-pattern generated by the SMS-PD algorithm. Red points represent the evaluated search positions whereas blue points indicate the estimated locations. The Green point exhibits the optimal match detection.

9.6. Experimental results

In order to verify the feasibility and effectiveness of our proposed algorithm in this work, series of comparative experiments with other PD algorithms are also given. Simulations have been performed over a set of images which are shown in Fig. 9.9. The proposed approach has been applied to the experimental set whose results have been compared to those produced by the ICA-PD method (Xu et al., 2010) and the PSO-PD algorithm (Liu et al., 2012). These are considered state-of-the-art algorithms whose results have been recently published.

The maximum iteration number for the experimental set has been set to 300. Such stop criterion has been selected to maintain compatibility to similar works reported in the literature (Dong et al., 2011; Liu et al., 2012; Wu et al., 2009; Xu et al., 2010).

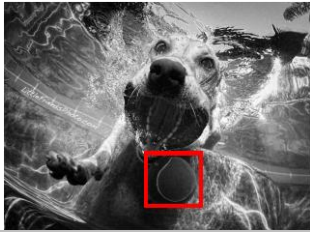
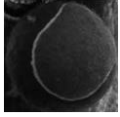
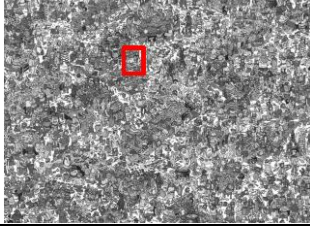





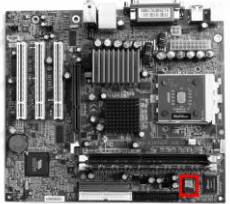

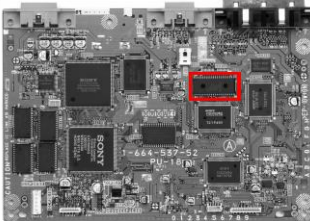

<i>Image</i>		<i>Template</i>	<i>Properties</i>
	(a)		<i>Dog</i> Image size 574x800 Template size 131x141
	(b)		<i>Waldo One</i> Image size 768x1024 Template size 59x36
	(c)		<i>Waldo Two</i> Image size 768x1024 Template size 59x61
	(d)		<i>Map</i> Image size 843x1417 Template size 55x71
	(e)		<i>Board one</i> Image size 2563x2779 Template size 209x176
	(f)		<i>Board Two</i> Image size 2400x3200 Template size 248x455

Fig. 9.9. Experimental set used in the comparisons for the SMS-PD algorithm.

The parameter setting for each algorithm in the comparison is described as follows:

1. ICA-PD: The parameters are set to $NumOfCountries = 100$, $NumOfImper = 10$, $NumOfColony = 90$, $T_{max} = 300$, $\zeta = 0.1$, $\varepsilon_1 = 0.15$ and $\varepsilon_2 = 0.9$. Such values are the best parameter set for this algorithm according to (Xu et al., 2010).

2. PSO-PD: The parameters are set to particle number=100, $c_1 = 1.5$ and $c_2 = 1.5$; besides, the particle velocity is initialized between $[-4, 4]$.
3. SMS-PD: The algorithm was configured by using: Particle number= 100 and $d=3$.

Once all algorithms were configured with such values, they are used without modification during the experiments. The comparisons are analyzed considering three performance indexes: the average elapsed time (At), the success rate (Sr), the Average number of checked locations (AsL) and the average number of function evaluations (AfE). The Average elapsed time (At) indicates the time in seconds employed during the execution of each single experiment. The success rate (Sr) represents the number of executions in percentage in which the algorithms find out successfully the optimal detection point. The Average number of checked locations (AsL) exhibits the number of search locations which has been visited during a single experiment. The average number of function evaluations (AfE) indicates the number of times that the NCC coefficient is computed. In order to assure statistic consistency, all these performance indexes are averaged considering a determined number of executions.

The results after 30 runs are reported in Table 9.1 where the best outcome for each image is bold-faced. According to this table, SMS-PD delivers better results than ICA and PSO for all images. In particular, the test remarks the largest difference in the success rate (Sr) and the average number of checked locations (AsL). Such facts are directly related to a better trade-off between exploration and exploitation, and the incorporation of the fitness calculation strategy, respectively. Fig. 9.10 present the matching evolution curve for each image considering the average best NCC value seen so-far for all the algorithms employed in the comparison.

Image	Algorithm	Average elapsed time (At)	Success rate (Sr)%	Average number of checked locations (AsL)	Average number of function evaluations (AfE)
(a)	ICA-TM	12.345	88.12	32000	32000
	PSO-TM	10.862	80.84	31250	31250
	SMS-TM	2.854	100	30000	5640
(b)	ICA-TM	17.534	70.23	82000	82000
	PSO-TM	16.297	62.45	81784	81784
	SMS-TM	3.643	100	60000	9213
(c)	ICA-TM	24.342	70.21	82000	82000
	PSO-TM	23.174	60.33	81784	81784
	SMS-TM	6.871	99	60000	8807
(d)	ICA-TM	26.249	67.12	220,512	220,512
	PSO-TM	26.381	58.12	210,784	210,784
	SMS-TM	6.937	98	100,512	18,506
(e)	ICA-TM	37.231	90.54	578,400	578,400
	PSO-TM	35.925	85.27	578,400	578,400
	SMS-TM	10.214	100	220,512	54,341
(f)	ICA-TM	40.287	89.78	578,400	578,400
	PSO-TM	38.298	81.47	578,400	578,400
	SMS-TM	10.719	100	220,512	57,981

Table 9.1. Performance comparison of ICA-PD, PSO-PD and the proposed approach for the experimental set shown in Fig. 9.9.

From Table 9.1, it turns out that the average cost of our algorithm is 6.873 s, while the average cost of the ICA-PD and the PSO-PD algorithms are 26.331 and 25.156, respectively. Such fact demonstrates that SMS-PD spends less time on image matching in reference to its counterparts. According to Table 9.4, the SMS-PD presents a better performance than the other two algorithms in terms of effectiveness, since it detects practically in all experiments the optimal detection point. On the other hand, although the three algorithms visit approximately the same number of search location, the proposed algorithm evaluates (NCC evaluation) a minimal number of them. It is important to recall that such evaluation represents the main computational cost associated to the PD process.

Image	SMS-PD vs. ICA-PD	SMS-PD vs. PSO-PD
(a)	1.52E-10	1.78E-10
(b)	3.23E-12	5.47E-12
(c)	1.56E-12	2.67E-12
(d)	3.21E-12	5.87E-12
(e)	4.87E-12	7.58E-12
(f)	2.11E-12	4.49E-12

Table 9.2. p -values produced by Wilcoxon's test comparing SMS-PD vs. ICA-PD and SMS-PD vs. PSO-PD over the average number of function evaluations (AfE) values from Table 9.1.

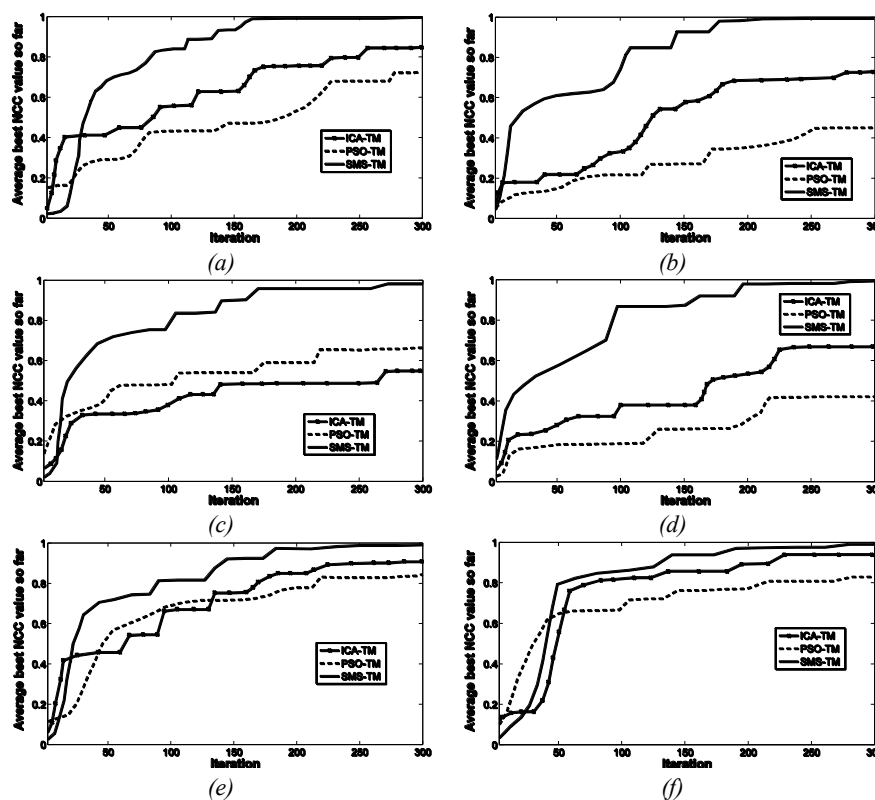


Fig. 9.10. Evolution curves for ICA-PD, PSO-PD and the proposed SMS-PD considering the average best NCC value seen so-far, each curve corresponds to the image of the experimental set.

A non-parametric statistical significance proof known as the Wilcoxon's rank sum test for independent samples (García et al., 2008; Wilcoxon, 1945) has been conducted over the average number of function evaluations (AfE) data of Table 9.1, with an 5% significance level. Table 9.2 reports the p -values produced by Wilcoxon's test for the pair-wise comparison of the average

number of function evaluations (AfE) of four groups. Such groups are formed by SMS-PD vs. ICA-PD and SMS-PD vs. PSO-PD. As a null hypothesis, it is assumed that there is no significant difference between mean values of the two algorithms. The alternative hypothesis considers a significant difference between the AfE values of both approaches.

All p -values reported in Table 9.2 are less than 0.05 (5% significance level) which is a strong evidence against the null hypothesis. Therefore, such evidence indicates that SMS-PD results are statistically significant and that it has not occurred by coincidence (i.e. due to common noise contained in the process).

9.7. Conclusions

In this chapter, a novel nature-inspired algorithm, called the States of Matter Search (SMS) has been proposed for solving the pattern detection (PD). The SMS algorithm is based on the simulation of the states of matter phenomenon. In SMS, individuals emulate molecules which interact to each other by using evolutionary operations based on the physical principles of the thermal-energy motion mechanism. Such operations allow the increase of the population diversity and avoid the concentration of particles within a local minimum. The presented approach combines the use of the defined operators with a control strategy that modifies the parameter setting of each operation during the evolution process. The algorithm is devised by considering each state of matter at one different exploration–exploitation rate. Thus, the evolutionary process is divided into three stages which emulate the three states of matter: gas, liquid and solid. At each state, molecules (individuals) exhibit different behaviors. Beginning from the gas state (pure exploration), the algorithm modifies the intensities of exploration and exploitation until the solid state (pure exploitation) is reached.

The approach also incorporates a simple fitness calculation approach which is based on the Nearest Neighbor Interpolation (NNI) algorithm in order to estimate the fitness value (NCC operation) for several candidate solutions (search locations). The method is able to save computational time by identifying which NCC values can be just estimated or must be calculated instead. As a result, the approach can not only substantially reduce the number search positions (by using the SMS approach), but also to avoid the NCC evaluation for many of them (by incorporating the NNI strategy). The presented method achieves the best balance over other PD algorithms, in terms of both estimation accuracy and computational cost. As a result, the approach can substantially reduce the number of functions evaluations, yet preserving the good search capabilities of SMS.

Since the proposed algorithm is designed to have a better exploration-exploitation balance than other evolutionary algorithms, a high probability for finding the true matching point (accurate detection point) is expected regardless of the high multi-modality nature of the PD process.

The performance of the proposed approach has been compared to other existing PD algorithms by considering different images which present a great variety of formats and complexities. Experimental results demonstrate the high performance of the proposed method in terms of elapsed time and the number of NCC evaluations.

Chapter 10

Artificial Bee Colony algorithm applied to Multi-threshold Segmentation

Image segmentation is a very important task in Computer Vision community, due to its capabilities for further steps that lead to recognizing patterns in digital images. Thus, the process of thresholding selection has become an interesting area, in recent years this procedure has been investigated as an optimization problem. On the other Hand, ABC is a nature inspired algorithm based on the intelligent behaviour of honey-bees which has been successfully used to solve complex real life optimization problems. In this chapter, a multi-thresholding approach, in which an image 1-D histogram is approximated by means of a Gaussian mixture model is presented. In the approach, the parameters are calculated by using the ABC algorithm. Under this method, each Gaussian function represents a pixel class; hence a threshold. The presented ABC approach shows fast convergence and low sensitivity to initial conditions. Experimental results has shown the ABC-method's capability to perform multi-threshold selection and interesting advantages in comparison to other algorithms.

10.1. Introduction

Several image processing applications aim to detect and classify relevant features which may be later analyzed to perform several high-level tasks. In particular, image segmentation seeks to group pixels within meaningful regions. Commonly, gray levels belonging to the object, are substantially different from those featuring the background. Thresholding is thus a simple but effective tool to isolate objects of interest; its applications include several classics such as document image analysis, whose goal is to extract printed characters (Abak, et al., 1997b; Kamel & Zhao, 1993), logos, graphical content, or musical scores; also it is used for map processing which aims to locate lines, legends, and characters (Trier & Jain, 1995). Moreover, it is employed for scene processing, seeking for object detection, marking (Bhanu, 1986b) and for quality inspection of materials (Sezgin & Sankur, 2001; Sezgin & Taşaltın, 2000).

Thresholding selection techniques can be classified into two categories: bi-level and multi-level. In the former, one limit value is chosen to segment an image into two classes: one representing the object and the other one segmenting the background. When distinct objects are depicted within a given scene, multiple threshold values have to be selected for proper segmentation, which is commonly called multilevel thresholding.

A variety of thresholding approaches have been proposed for image segmentation, including conventional methods (Guo & Pandit, 1998; Pal & Pal, 1993; Sahoo et al., 1988; Snyder et al., 1990) and intelligent techniques (Chen & Wang, 2005; Lai, 2006). Extending the segmentation algorithms to a multilevel approach may cause some inconveniences: (i) they may have no systematic or analytic solution when the number of classes to be detected increases and (ii) they may also show a slow convergence and/or high computational cost (Chen & Wang, 2005).

In this work, the segmentation algorithm is based on a parametric model holding a probability density function of gray levels which groups a mixture of several Gaussian density functions (Gaussian mixture). Mixtures represent a flexible method of statistical modelling as they are employed in

a wide variety of contexts (Gonzalez & Woods, 2008). Gaussian mixture has received considerable attention in the development of segmentation algorithms despite its performance is influenced by the shape of the image histogram and the accuracy of the estimated model parameters (Gupta & Sortrakul, 1998). The associated parameters can be calculated considering the Expectation Maximization (EM) algorithm (Dempster et al., 1977; Zhang et al., 2003) or Gradient-based methods such as Levenberg-Marquardt (LM) (Park et al., 2000). However, EM algorithms are very sensitive to the choice of the initial values (Park & Ozeki, 2009), meanwhile Gradient-based methods are computationally expensive and may easily get stuck within local minima (Gupta & Sortrakul, 1998). Therefore, some researchers have attempted to develop methods based on modern global optimization algorithms such as the Learning Automata (LA) (Cuevas, et al., 2011) and Differential Evolution algorithm (DE) (Cuevas, et al., 2010). In this chapter, an alternative approach using an optimization algorithm for determining the parameters of a Gaussian mixture is presented.

On other hand, Karaboga, (2005) has presented a metaheuristic algorithm for solving numerical optimization problems known as the artificial bee colony (ABC) method. Inspired by the intelligent foraging behavior of a honeybee swarm, the ABC algorithm consists of three essential components: food source positions, nectar-amounts and several honey-bee classes. Each food source position represents a feasible solution for the problem under consideration. The nectar-amount for a food source represents the quality of such solution according to its fitness value. Each bee-class symbolizes one particular operation for generating new candidate food source positions (i.e. candidate solutions).

The ABC algorithm starts by producing a randomly distributed initial population (food source locations). After initialization, an objective function evaluates whether such candidates represent an acceptable solution (nectar-amount) or not. Guided by the values of such objective function, candidate solutions are evolved through different ABC operations (honey-bee types). When the fitness function (nectar-amount) cannot be further improved after a maximum number of cycles, its related food source is assumed to be abandoned and replaced by a new randomly chosen food source location.

The performance of ABC algorithm has been compared to other metaheuristic methods such as Genetic Algorithms (GA), Differential Evolution (DE) and Particle Swarm Optimization (PSO) (Karaboga & Basturk, 2008; Karaboga & Akay, 2009). The results have shown that ABC can produce optimal solutions yet more effectively than other methods for several optimization problems. Such characteristics have motivated the use of ABC to solve different sorts of engineering problems within different fields such as signal processing (Karaboga, 2009), flow shop scheduling (Pan, et al., 2011), structural inverse analysis (Kang, et al., 2009), clustering (Karaboga & Ozturk, 2011; Zhang, et al., 2010) and electromagnetism (Ho & Yang, 2009).

This chapter presents the use of the Artificial Bee Colony (ABC) algorithm to compute threshold selection for image segmentation. In this approach, the segmentation process is considered as an optimization problem approximating the 1-D histogram of a given image by means of a Gaussian mixture model. The operation parameters are calculated through the ABC algorithm. Each Gaussian function approximating the histogram represents a pixel class and therefore a threshold point in the segmentation scheme.

The experimental results, presented in this work, demonstrate that ABC exhibits fast convergence, relatively low computational cost and no sensitivity to initial conditions by keeping an acceptable segmentation of the image, i.e. a better mixture approximation in comparison to the EM or gradient based algorithms.

The chapter is organized as follows: Section 10.2 presents the Gaussian approximation of the histogram while Section 10.3 discusses on the ABC algorithm. Section 10.4 formulates the threshold

determination with Section 10.5 presenting all experimental results after the presented approach is implemented. Section 10.6 summarizes a full discussion on the algorithm performance.

10.2. Gaussian approximation

Let consider an image holding L gray levels $[0, \dots, L-1]$ whose distribution is displayed within a histogram $h(g)$. In order to simplify the description, the histogram is normalized just as a probability distribution function, yielding:

$$\begin{aligned} h(g) &= \frac{n_g}{N}, \quad h(g) > 0, \\ N &= \sum_{g=0}^{L-1} n_g, \quad \text{and} \quad \sum_{g=0}^{L-1} h(g) = 1, \end{aligned} \quad (10.1)$$

where n_g denotes the number of pixels with gray level g and N being the total number of pixels in the image. The histogram function can thus be contained into a mix of Gaussian probability functions of the form:

$$p(x) = \sum_{i=1}^K P_i \cdot p_i(x) = \sum_{i=1}^K \frac{P_i}{\sqrt{2\pi}\sigma_i} \exp\left[-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right] \quad (10.2)$$

with P_i being the probability of class i , $p_i(x)$ being the probability distribution function of gray-level random variable x in class i , with μ_i and σ_i being the mean and standard deviation of the i -th probability distribution function and K being the number of classes within the image. In addition, the constraint $\sum_{i=1}^K P_i = 1$ must be satisfied.

The mean square error is used to estimate the $3K$ parameters P_i , μ_i and σ_i , $i = 1, \dots, K$. For instance, the mean square error between the Gaussian mixture $p(x_i)$ and the experimental histogram function $h(x_i)$ is defined as follows:

$$J = \frac{1}{n} \sum_{j=1}^n [p(x_j) - h(x_j)]^2 + \omega \cdot \left| \left(\sum_{i=1}^K P_i \right) - 1 \right| \quad (10.3)$$

Assuming an n -point histogram as in (Gonzalez & Woods, 2008) and ω being the penalty associated with the constrain $\sum_{i=1}^K P_i = 1$.

In general, the parameter estimation that minimizes the square error produced by the Gaussian mixture is not a simple problem. A straightforward method is to consider the partial derivatives of the error function to zero by obtaining a set of simultaneous transcendental equations (Gonzalez & Woods, 2008). However, an analytical solution is not always available considering the non-linear nature of the equation which in turn yields the use of iterative approaches such as gradient-based or maximum likelihood estimation. Unfortunately, such methods may also get easily stuck within local minima or be time expensive.

In the case of other algorithms such as the EM algorithm and the gradient-based methods, the new parameter point lies within a neighbourhood distance of the previous parameter point. However, this is not the case for the ABC adaptation algorithm which is based on stochastic principles. New operating points are thus determined by a parameter probability function that may yield points lying far away from previous operating points, providing the algorithm with a higher ability to locate and pursue a global minimum.

10.3. Artificial Bee Colony (ABC) algorithm

The ABC algorithm assumes the existence of a set of operations that may resemble some features of the honey bee behavior. For instance, each solution within the search space includes a parameter set representing food source locations. The “fitness value” refers to the food source quality that is strongly linked to the food’s location. The process mimics the bee’s search for valuable food sources yielding an analogous process for finding the optimal solution.

10.3.1. Biological bee profile

The minimal model for a honey bee colony consists of three classes: employed bees, onlooker bees and scout bees. The employed bees will be responsible for investigating the food sources and sharing the information with recruit onlooker bees. They, in turn, will make a decision on choosing food sources by considering such information.

The food source having a higher quality will have a larger chance to be selected by onlooker bees than those showing a lower quality. An employed bee, whose food source is rejected as low quality by employed and onlooker bees, will change to a scout bee to randomly search for new food sources. Therefore, the exploitation is driven by employed and onlooker bees while the exploration is maintained by scout bees. The implementation details of such bee-like operations in the ABC algorithm are described in the next sub-section.

10.3.2. Description of the ABC algorithm

Resembling other metaheuristic approaches, the ABC algorithm is an iterative process. It starts with a population of randomly generated solutions or food sources. The following three operations are applied until a termination criterion is met (Karaboga & Akay, 2009):

1. Send the employed bees.
2. Select the food sources by the onlooker bees.
3. Determine the scout bees.

10.3.2.1. Initializing the population

The algorithm begins by initializing N_p food sources. Each food source is a D -dimensional vector containing the parameter values to be optimized, which are randomly and uniformly distributed between the pre-specified lower initial parameter bound x_j^{low} and the upper initial parameter bound x_j^{high} .

$$x_{j,i} = x_j^{low} + \text{rand}(0,1) \cdot (x_j^{high} - x_j^{low});$$

$$j = 1, 2, \dots, D; \quad i = 1, 2, \dots, N_p. \quad (10.4)$$

with j and i being the parameter and individual indexes respectively. Hence, $x_{j,i}$ is the j th parameter of the i th individual.

10.3.2.2. Send employed bees

The number of employed bees is equal to the number of food sources. At this stage, each employed bee generates a new food source in the neighborhood of its present position as follows:

$$\begin{aligned} v_{j,i} &= x_{j,i} + \phi_{j,i}(x_{j,i} - x_{j,k}), \text{ with} \\ k &\in \{1, 2, \dots, N_p\}; j \in \{1, 2, \dots, D\} \end{aligned} \quad (10.5)$$

$x_{j,i}$ is a randomly chosen j parameter of the i th individual and k is one of the N_p food sources, satisfying the condition $i \neq k$. If a given parameter of the candidate solution v_i exceeds its predetermined boundaries, that parameter should be adjusted in order to fit the appropriate range. The scale factor $\phi_{j,i}$ is a random number between $[-1, 1]$. Once a new solution is generated, a fitness value representing the profitability associated with a particular solution is calculated. The fitness value for a minimization problem can be assigned to each solution v_i by the following expression:

$$fit_i = \begin{cases} \frac{1}{1 + J_i} & \text{if } J_i \geq 0 \\ 1 + abs(J_i) & \text{if } J_i < 0 \end{cases} \quad (10.6)$$

where J_i is the objective function to be minimized. A greedy selection process is thus applied between v_i and x_i . If the nectar-amount (fitness) of v_i is better, then the solution x_i is replaced by v_i ; otherwise, x_i remains.

10.3.2.3. Select the food sources by the onlooker bees

Each onlooker bee (the number of onlooker bees corresponds to the food source number) selects one of the proposed food sources, depending on their fitness value, which has been recently defined by the employed bees. The probability that a food source will be selected can be obtained from the following equation:

$$Prob_i = \frac{fit_i}{\sum_{i=1}^{N_p} fit_i} \quad (10.7)$$

where fit_i is the fitness value of the food source i , which is related to the objective function value (J_i) corresponding to the food source i . The probability of a food source being selected by onlooker bees increases with an increase in the fitness value of the food source. After the food source is selected, onlooker bees will go to the selected food source and select a new candidate food source position inside the neighborhood of the selected food source.

The new candidate food source can be expressed and calculated by Eq.(10.5). In case the nectar-amount, i.e., fitness of the new solution, is better than before, such position is held; otherwise, the last solution remains.

10.3.2.4. Determine the scout bees

If a food source i (candidate solution) cannot be further improved through a predetermined trial number known as “limit”, the food source is assumed to be abandoned and the corresponding employed or onlooker bee becomes a scout. A scout bee explores the searching space with no previous information, i.e., the new solution is generated randomly as indicated by Eq.(10.4). In order to verify if a candidate solution has reached the predetermined “limit”, a counter A_i is assigned to each food source i . Such a counter is incremented consequent to a bee-operation failing to improve the food source’s fitness.

10.4. Determination of Thresholding Values

In order to determine optimal threshold values, it is considered that the data classes are organized such that $\mu_1 < \mu_2 < \dots < \mu_K$. Therefore, threshold values are obtained by computing the overall probability error of two adjacent Gaussian functions, yielding:

$$E(T_h) = P_{h+1} \cdot E_1(T_h) + P_h \cdot E_2(T_h), \quad (10.8)$$

$$h = 1, 2, \dots, K-1$$

considering

$$E_1(T_h) = \int_{-\infty}^{T_h} p_{h+1}(x) dx, \quad (10.9)$$

and

$$E_2(T_h) = \int_{T_h}^{\infty} p_h(x) dx, \quad (10.10)$$

$E_1(T_h)$ is the probability of mistakenly classifying the pixels in the $(h + 1)$ -th class belonging to the h -th class, while $E_2(T_h)$ is the probability of erroneously classifying the pixels in the h -th class belonging to the $(h + 1)$ -th class. P_j 's are the a priori probabilities within the combined probability density function, and T_h is the threshold value between the h -th and the $(h + 1)$ -th classes. One T_h value is chosen such as the error $E(T_h)$ is minimized. By differentiating $E(T_h)$ with respect to T_h and equating the result to zero, it is possible to use the following equation to define the optimum threshold value T_h :

$$AT_h^2 + BT_h + C = 0 \quad (10.11)$$

considering

$$A = \sigma_h^2 - \sigma_{h+1}^2$$

$$B = 2 \cdot (\mu_h \sigma_{h+1}^2 - \mu_{h+1} \sigma_h^2)$$

$$C = (\sigma_h \mu_{h+1})^2 - (\sigma_{h+1} \mu_h)^2 + 2 \cdot (\sigma_h \sigma_{h+1})^2 \cdot \ln \left(\frac{\sigma_{h+1} P_h}{\sigma_h P_{h+1}} \right) \quad (10.12)$$

Although the above quadratic equation has two possible solutions, only one of them is feasible, i.e. a positive value falling within the interval.

10.5. Experimental Results

By considering that the mixture parameters are extracted from the fitness function J (Eq.(10.3)) after applying the ABC algorithm, three experiments are set to evaluate the performance of the presented algorithm. The first one considers the well-known image “The Camera-man” which is shown by Figure 10.1a, with its corresponding histogram shown by Figure 10.1b. The goal is to segment the image in three different pixel classes. According to Eq. (10.2), during learning, the ABC algorithm adjusts nine parameters, following the minimization procedure conducted by Eq. (10.3). In this experiment, a population of 20 (N_p) bees is considered, with ten employed and ten onlooker bees. Each candidate holds 9 dimensions, such as:

$$I_N = \{P_1^N, \sigma_1^N, \mu_1^N, P_2^N, \sigma_2^N, \mu_2^N, P_3^N, \sigma_3^N, \mu_3^N\} \quad (10.13)$$

with N representing the individual's number. The parameters (P, σ, μ) are randomly initialized, but assuming some restrictions to each parameter (for example μ must fall between 0 and 255).

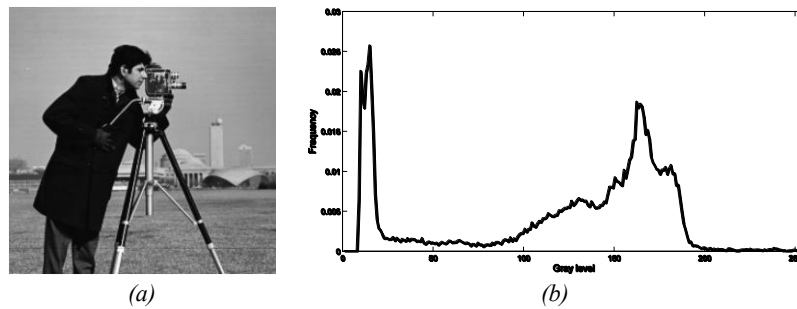


Fig. 10.1. (a) Original image “The Cameraman”, and (b) its correspondent histogram.

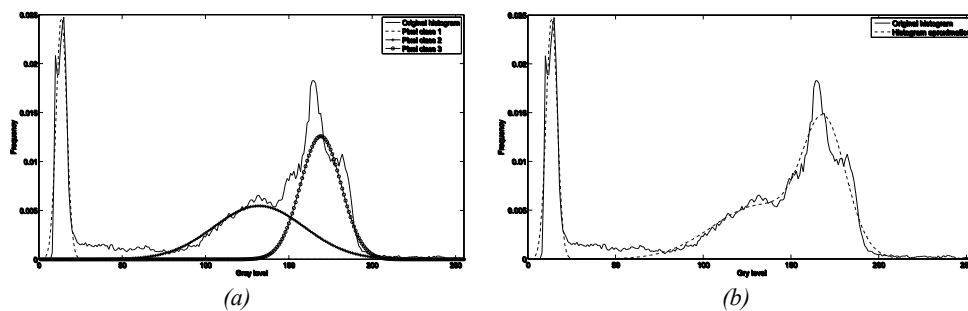


Fig. 10.2. Applying the ABC algorithm for 3 classes and its results: (a) Gaussian functions for each class, (b) Mixed Gaussian functions (approach to the original histogram).

The experiments suggest that after 200 iterations, the ABC algorithm has converged to the global minimum. Figure 10.2(a) shows the obtained Gaussian functions (pixel classes) plotted over the original histogram while Figure 10.2(b) shows the Gaussian mixture. Figure 10.3 shows the segmented image whose threshold values are calculated according to Eqs. (10.11) and (10.12).

The algorithm is tested with a greater number of Gaussian functions yielding the need of optimizing more parameters (according to Eq. 10.3). Thus, twelve parameters are now considered corresponding to the values of four Gaussian functions. One population of 20 bees and 12 dimensions are used for the test. Figure 10.4(a) shows the Gaussian functions (pixel classes) plotted over the

histogram after 200 iterations, while in Figure 10.4(b) presents the Gaussian mixture. The segmented image is depicted by Figure 10.5.



Fig. 10.3. Segmented image considering only three classes.

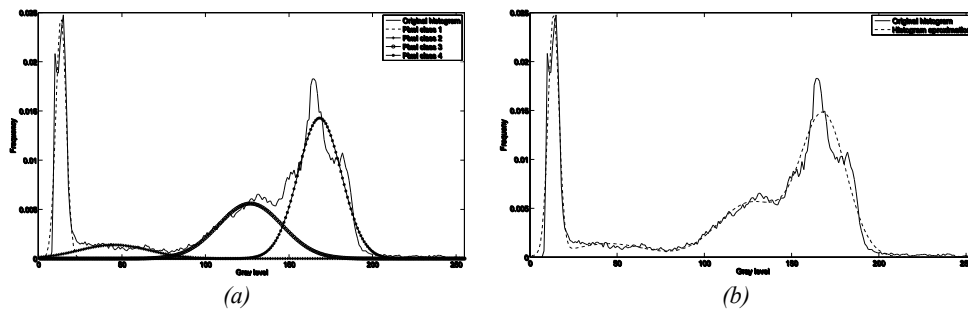


Fig. 10.4. Segmentation of the test image as it was obtained by the ABC algorithm considering 4 classes: (a) Gaussian functions for each class. (b) Mixed Gaussian functions approaching the original histogram.



Fig. 10.5. Segmented image considering four classes.

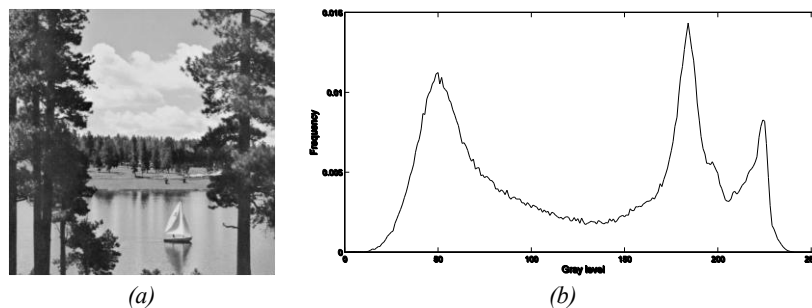


Fig. 10.6. Second experiment, (a) the original image “The scene”, and (b) its histogram.

The second experiment considers the popular benchmark image known as “The scene” (see Figure 10.6(a)). The image’s histogram is presented by Figure 10.6(b). Following the first experiment, the image is segmented considering four-pixel classes. The optimization is performed by the ABC al-

gorithm which results in the classes shown by Figure 10.7(a). In turn, Figure 10.7(b) presents the Gaussian mixture as it results from the addition of other Gaussian functions. Figure 10.8 shows the image segmentation considering four classes.

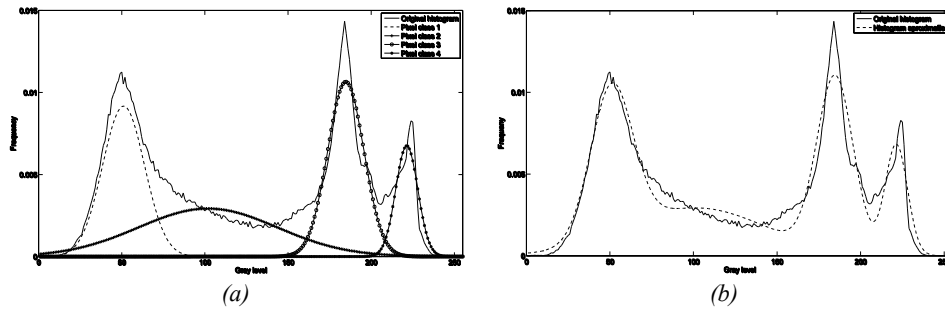


Fig. 10.7. Results obtained by the ABC algorithm for 4 classes: (a) Gaussian functions at each class, (b) Mixed Gaussian functions approaching the original histogram



Fig. 10.8. Segmented image considering four classes.

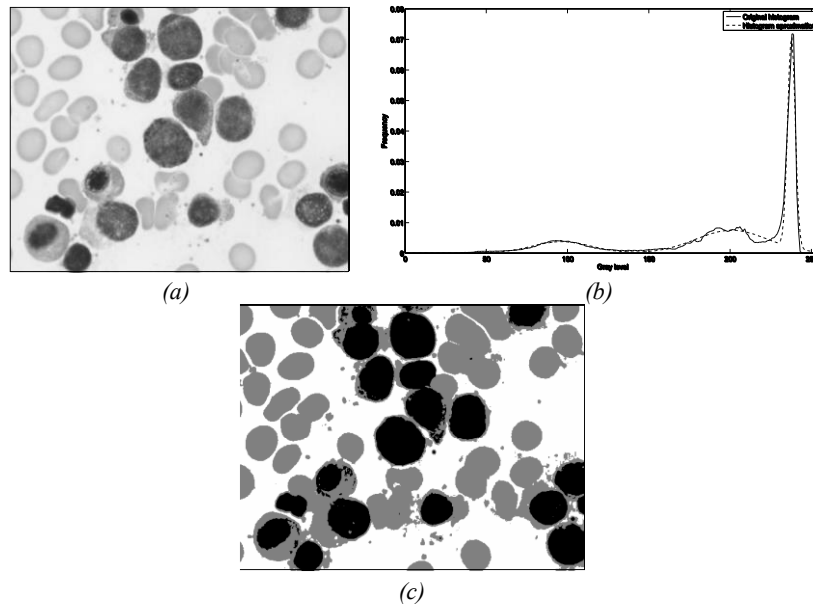


Fig. 10.9. Segmentation of a blood smear image considering three classes for the ABC algorithm: (a) Original image, (b) Comparison between the original histogram and the Gaussian approach, (c) the segmented image.

The final experiment considers a blood-smear image as it is processed by the ABC algorithm. The image shows a set of leukocytes cells in the blood smear (darker cells on image of Figure 10.9(a)). The gray blobs represent the red blood cells while the background is white yielding only three classes to be considered.

10.5.1. Comparing the ABC algorithm vs. the EM and LM methods.

This section discusses on the comparison between ABC and other algorithms such as the EM algorithm and the Levenberg-Marquardt (LM) method which are commonly employed for determining Gaussian mixtures. The discussion focuses on the following issues: sensitivity to the initial conditions, convergence and computational costs.

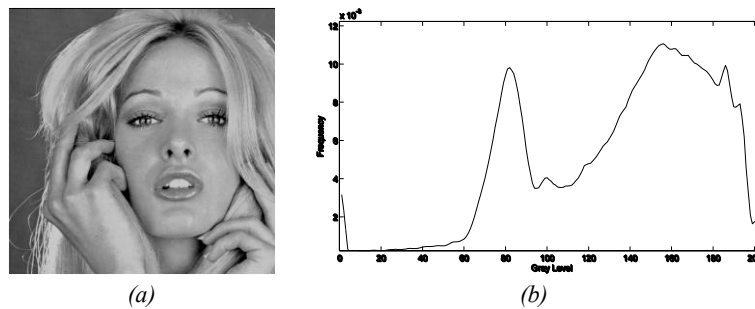


Figure 10.10. (a) Original image used for the comparison experiment and (b) its corresponding histogram.

Parameters	Initial condition 1	EM	LM	ABC	Initial condition 2	EM	LM	ABC
μ_1	40.6	33.13	32.12	32.01	10	20.90	31.80	32.50
μ_2	81.2	81.02	82.05	82.30	100	82.78	80.85	82.42
μ_3	121.8	127.52	127	127.00	138	146.67	128	127.72
μ_4	162.4	167.58	166.80	166.10	200	180.72	165.90	166.50
σ_1	15	25.90	25.50	25.30	10	18.52	20.10	25.01
σ_2	15	9.78	9.70	9.80	5	12.52	9.81	10.00
σ_3	15	17.72	17.05	17.71	8	20.5	15.15	17.57
σ_4	15	17.03	17.52	17.21	22	10.09	18.00	17.22
p_1	0.25	0.0313	0.0310	0.307	0.20	0.0225	0.0312	0.317
p_2	0.25	0.2078	0.2081	0.201	0.30	0.2446	0.2079	0.255
p_3	0.25	0.2508	0.2500	0.249	0.20	0.5232	0.2502	0.260
p_4	0.25	0.5102	0.5110	0.555	0.30	0.2098	0.5108	0.511

Table 10.1. Comparison between the EM, the LM and the ABC algorithm, considering two different initial conditions.

a) Sensitivity to initial conditions. This experiment considers different initial values for all methods assuming the same histogram in the approximation task. After convergence, only final parameters representing the Gaussian mixture are reported. Figure 10.10(a) shows the image used in the comparison while Figure 10.10(b) pictures the histogram. All experiments are conducted several times in order to assure consistency. Only two different initial states with the highest variation

are reported in Table 10.1. Likewise, Figure 10.11 shows the obtained segmented images considering two initial conditions as it is reported by Table 10.1. In the ABC case, the algorithm does not require initialization as random initial values are employed. However, in order to assure a valid comparison, same initial values are considered for the EM, the LM and the ABC method.

By analyzing the information in Table 10.1, the sensitivity of the EM algorithm to initial conditions becomes evident. Figure 10.11 shows a clear pixel misclassification in some sections of the image as a consequence of such sensitivity.

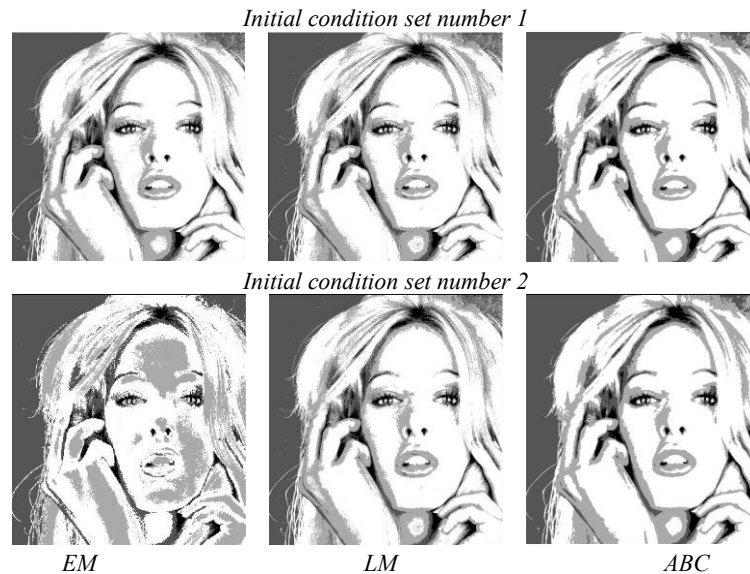


Figure 10.11. Segmented images after applying the EM, the LM and the ABC algorithm with different initial conditions.

b) Convergence and computational cost. The experiment aims to measure the number of required steps and the computing time spent by the EM, the LM and the ABC algorithm required to calculate the parameters of the Gaussian mixture in benchmark images (see Figure 10.12(a-c)). All experiments consider four classes.

Iterations	(a)	(b)	(c)
	Time elapsed		
EM	1855	1833	1870
	2.72s	2.70s	2.73s
LM	985	988	958
	4.03s	4.04s	4.98s
ABC	409	399	512
	0.78s	0.70s	0.81s

Table 10.2. Iterations and time requirements of the EM, the LM and the ABC algorithm as they are applied to segment benchmark images (see Figure 10.12).

Table 10.2 shows the averaged measurements as they are obtained from 20 experiments. It is evident that the EM is the slowest to converge (iterations) and the LM shows the highest computational cost (time elapsed) because it requires complex Hessian approximations. On the other hand, the ABC shows an acceptable trade off between its convergence time and its computational cost. Finally, Figure 10.12 below shows the segmented images as they are generated by each algorithm.

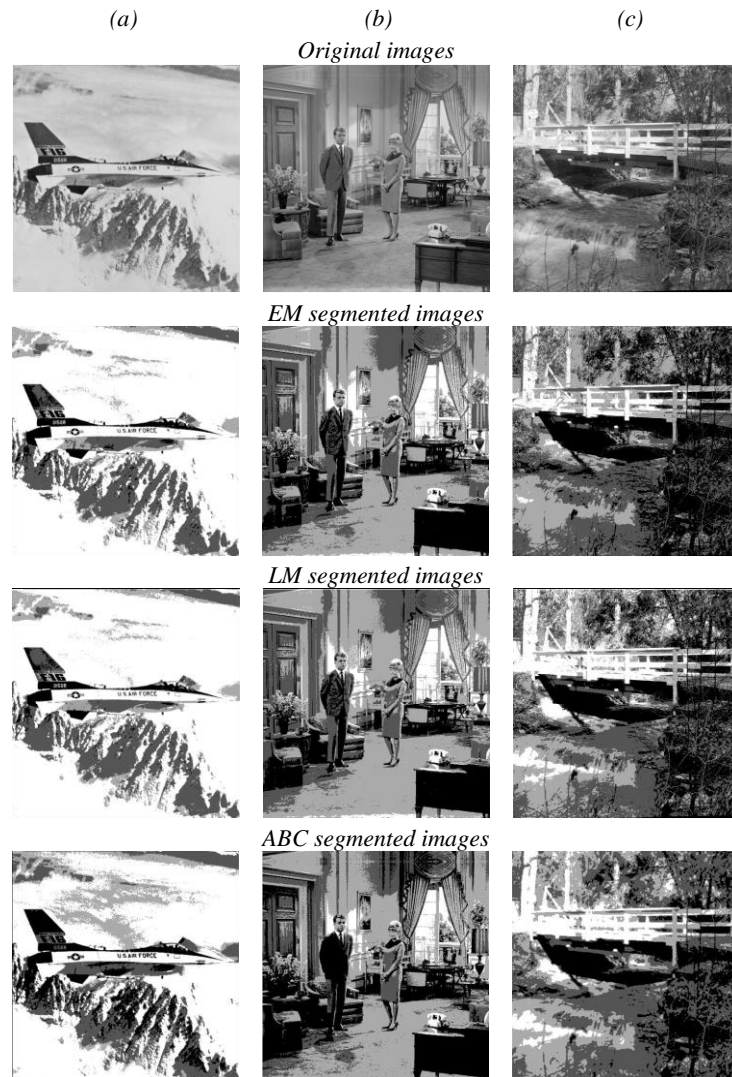


Figure 10.12. Original benchmark images (a)-(c), and segmented images obtained by the EM, the LM and the ABC algorithms.

10.6. Conclusions

In this chapter, an automatic image multi-threshold approach based on the Artificial Bee Colony (ABC) algorithm is presented. The segmentation process is considered to be similar to an optimization problem. The algorithm approximates the 1-D histogram of a given image using a Gaussian mixture model whose parameters are calculated through the ABC algorithm. Each Gaussian function approximating the histogram represents a pixel class and therefore one threshold point.

Experimental evidence shows that the ABC algorithm has an acceptable compromise between its convergence time and its computational cost when it is compared to the Expectation-Maximization (EM) method and the Levenberg-Marquardt (LM) algorithm.

Additionally, the ABC algorithm also exhibits a better performance under certain circumstances (initial conditions) on which it is well-reported in the literature (Park & Ozeki, 2009) that the EM has underperformed. Finally, the results have shown that the stochastic search accomplished by the

ABC method shows a consistent performance with no regard of the initial value and still showing a greater chance to reach the global minimum.

Chapter 11

Leukocyte Detection through an Evolutionary Method

Classical image processing methods often face great difficulties while dealing with images containing noise and distortions. Under such conditions, the use of soft computing approaches has been recently extended to address challenging real-world image processing problems. The automatic detection of Leukocytes or White Blood Cells (WBC) still remains as an unsolved issue in medical imaging. The analysis of WBC images has engaged researchers from fields of medicine and image processing alike. Since WBC can be approximated by an ellipsoid form, an ellipse detector algorithm may be successfully applied in order to recognize such elements. This chapter presents an algorithm for the automatic detection of leukocytes embedded into complicated and cluttered smear images that considers the complete process as a multi-ellipse detection problem. The approach, which is based on the Differential Evolution (DE) algorithm, transforms the detection task into an optimization problem whose individuals represent candidate ellipses. An objective function evaluates if such candidate ellipses are actually present in the edge map of the smear image. Guided by the values of such function, the set of encoded candidate ellipses (individuals) are evolved using the DE algorithm so that they can fit into the leukocytes which are enclosed within the edge map of the smear image. Experimental results from white blood cell images with a varying range of complexity are included to validate the efficiency of the proposed technique in terms of its accuracy and robustness.

11.1. Introduction

Medical image processing has become more and more important in diagnosis with the development of medical imaging and computer technique. Huge amounts of medical images are obtained by X-ray radiography, CT and MRI. They provide essential information for efficient and accurate diagnosis based on advance computer vision techniques (Scholl, et al., 2011; Zhuang & Meng, 2004).

On the other hand, White Blood Cells (WBC) also known as leukocytes play a significant role in the diagnosis of different diseases. Although computer vision techniques have successfully contributed to generate new methods for cell analysis, which in turn, have led into more accurate and reliable systems for disease diagnosis. However, high variability on cell shape, size, edge and localization, complicates the data extraction process. Moreover, the contrast between cell boundaries and the image's background may vary due to unstable lighting conditions during the capturing process.

Many works have been conducted in the area of blood cell detection. In (Wang & Chu, 2009) a method based on boundary support vectors is proposed to identify WBC. In such approach, the intensity of each pixel is used to construct feature vectors whereas a Support Vector Machine (SVM) is used for classification and segmentation. By using a different approach, Wu et al., (2007) developed an iterative Otsu method based on the circular histogram for leukocyte segmentation. According to such technique, the smear images are processed in the Hue-Saturation-Intensity (HSI) space by considering that the Hue component contains most of the WBC information. One of the latest advances in white blood cell detection research is the algorithm proposed by Wang (Shitong

Wang et al., 2007b) that is based on the fuzzy cellular neural network (FCNN). Although such method has proved successful in detecting only one leukocyte in the image, it has not been tested over images containing several white cells. Moreover, its performance commonly decays when the iteration number is not properly defined, yielding a challenging problem itself with no clear clues on how to make the best choice.

Since white blood cells can be approximated with an ellipsoid form, computer vision techniques for detecting ellipses may be used in order to recognize them. Ellipse detection in real images is an open research problem since long time ago. Several approaches have been proposed which traditionally fall under three categories: Symmetry-based, Hough transform-based (HT) and Random sampling.

In symmetry-based detection (Atherton & Kerbyson, 1993; Muammar & Nixon, 1989), the ellipse geometry is taken into account. The most common elements used in ellipse geometry are the ellipse center and axis. Using these elements and edges in the image, the ellipse parameters can be found. Ellipse detection in digital images is commonly solved through the Hough Transform (Fischler & Bolles, 1981). It works by representing the geometric shape by its set of parameters, then accumulating bins in the quantized parameter space. Peaks in the bins provide the indication of where ellipses may be. Obviously, since the parameters are quantized into discrete bins, the intervals of the bins directly affect the accuracy of the results and the computational effort. Therefore, for fine quantization of the space, the algorithm returns more accurate results, while suffering from large memory loads and expensive computation. In order to overcome such a problem, some other researchers have proposed other ellipse detectors following the Hough transform principles by using random sampling. In random sampling-based approaches (Shaked et al., 1996b; Xu et al., 1990), a bin represents a candidate shape rather than a set of quantized parameters, as in the HT. However, like the HT, random sampling approaches go through an accumulation process for the bins. The bin with the highest score represents the best approximation of an actual ellipse in the target image.

The work of Han et al., (1994a) shows that a random sampling-based approach produces improvements in accuracy and computational complexity, as well as a reduction in the number of false positives (non-existent ellipses), when compared to the original HT and the number of its improved variants.

As an alternative to traditional techniques, the problem of ellipse detection has also been handled through optimization methods. In general, they have demonstrated to give better results than those based on the HT and random sampling with respect to accuracy and robustness (Ayala-Ramirez et al., 2006). Such approaches have produced several robust ellipse detectors using different optimization algorithms such as Genetic Algorithms (GA) (Lutton & Martinez, 1994; Yao, et al., 2005) and Particle Swarm Optimization (PSO) (Cheng, et al., 2009).

Although detection algorithms based on optimization approaches present several advantages in comparison to traditional approaches, they have been scarcely applied to WBC detection. One exception is the work presented by Karkavitsas & Rangoussi, (2005) that solves the WBC detection problem through the use of GA. However, since the evaluation function, which assesses the quality of each solution, considers the number of pixels contained inside of a circle with fixed radius, the method is prone to produce misdetections particularly for images that contained overlapped or irregular WBC.

In this work, the WBC detection task is approached as an optimization problem and the differential evolution algorithm is used to build the ellipsoidal approximation. Differential Evolution (DE), introduced by Storn & Price, (1995), is a novel evolutionary algorithm, which is used to optimize complex continuous nonlinear functions. As a population-based algorithm, DE uses simple mutation and crossover operators to generate new candidate solutions and applies one-to-one compe-

tion scheme to greedily decide whether the new candidate or its parent will survive in the next generation. Due to its simplicity, ease of implementation, fast convergence, and robustness, the DE algorithm has gained much attention, reporting a wide range of successful applications in the literature (Babu & Munawar, 2007; Chiou, et al., 2005; Cuevas et al., 2010; Kannan, et al., 2005; Mayer, et al., 2005).

This chapter presents an algorithm for the automatic detection of blood cell images based on the DE algorithm. The proposed method uses the encoding of five edge points as candidate ellipses in the edge map of the smear. An objective function allows to accurately measure the resemblance of a candidate ellipse with an actual WBC on the image. Guided by the values of such objective function, the set of encoded candidate ellipses are evolved using the DE algorithm so that they can fit into actual WBC on the image. The approach generates a sub-pixel detector, which can effectively identify leukocytes in real images. Experimental evidence shows the effectiveness of such method in detecting leukocytes despite complex conditions. Comparison to the state-of-the-art WBC detectors on multiple images demonstrates a better performance of the proposed method.

The main contribution of this study is the proposal of a new WBC detector algorithm that efficiently recognize WBC under different complex conditions while considering the whole process as an ellipse detection problem. Although ellipse detectors based on optimization present several interesting properties, to the best of our knowledge, they have not yet been applied to any medical image processing up to date.

This chapter is organized as follows: Section 11.2 provides a description of the DE algorithm while in Section 11.3 the ellipse detection task is fully explained from an optimization perspective within the context of the DE approach. The complete WBC detector is presented in Section 11.4. Section 11.5 reports the obtained experimental results whereas Section 11.6 conducts a comparison between state-of-the-art WBC detectors and the proposed approach. Finally, in section 11.7, some conclusions are drawn.

11.2. Differential Evolution Algorithm

The DE algorithm is a simple and direct search algorithm, which is based on population and aims for optimizing global multi-modal functions. DE employs the mutation operator as to provide the exchange of information among several solutions.

There are various mutation base generators to define the algorithm type. The version of DE algorithm used in this work is known as rand-to-best/1/bin or “DE1” (Storn & Price, 1995). DE algorithms begin by initializing a population of N_p and D -dimensional vectors considering parameter values that are randomly distributed between the pre-specified lower initial parameter bound $x_{j,low}$ and the upper initial parameter bound $x_{j,high}$ as follows:

$$\begin{aligned} x_{j,i,t} &= x_{j,low} + \text{rand}(0,1) \cdot (x_{j,high} - x_{j,low}); \\ j &= 1, 2, \dots, D; \quad i = 1, 2, \dots, N_p; \quad t = 0. \end{aligned} \quad (11.1)$$

The subscript t is the generation index, while j and i are the parameter and particle indexes respectively. Hence, $x_{j,i,t}$ is the j th parameter of the i th particle in generation t .

In order to generate a trial solution, DE algorithm first mutates the best solution vector $\mathbf{x}_{best,t}$ from the current population by adding the scaled difference of two vectors from the current population.

$$\begin{aligned} \mathbf{v}_{i,t} &= \mathbf{x}_{best,t} + F \cdot (\mathbf{x}_{r_1,t} - \mathbf{x}_{r_2,t}); \\ r_1, r_2 &\in \{1, 2, \dots, N_p\} \end{aligned} \quad (11.2)$$

with $\mathbf{v}_{i,t}$ being the mutant vector. Indices r_1 and r_2 are randomly selected with the condition that they are different and have no relation to the particle index i whatsoever (i.e., $r_1 \neq r_2 \neq i$). The mutation scale factor F is a positive real number, typically less than one. Figure 11.1 illustrates the vector-generation process defined by Equation (11.2).

In order to increase the diversity of the parameter vector, the crossover operation is applied between the mutant vector $\mathbf{v}_{i,t}$ and the original individuals $\mathbf{x}_{i,t}$. The result is the trial vector $\mathbf{u}_{i,t}$ that is computed by considering element to element as follows:

$$\mathbf{u}_{j,i,t} = \begin{cases} v_{j,i,t}, & \text{if } \text{rand}(0,1) \leq CR \text{ or } j = j_{\text{rand}}, \\ x_{j,i,t}, & \text{otherwise.} \end{cases} \quad (11.3)$$

with $j_{\text{rand}} \in \{1, 2, \dots, D\}$. The crossover parameter ($0.0 \leq CR \leq 1.0$) controls the fraction of parameters that the mutant vector is contributing to the final trial vector. In addition, the trial vector always inherits the mutant vector parameter according to the randomly chosen index j_{rand} , assuring that the trial vector differs by at least one parameter from the vector to which it is compared ($\mathbf{x}_{i,t}$).

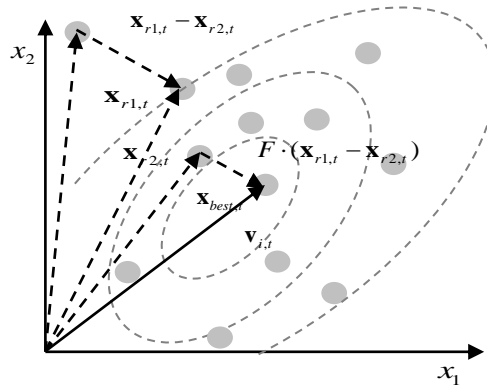


Fig. 11.1. Two-dimensional example of an objective function showing its contour lines and the process for generating \mathbf{v} in scheme DE/best/l/exp from vectors of the current generation.

Finally, a greedy selection is used to find better solutions. Thus, if the computed cost function value of the trial vector $\mathbf{u}_{i,t}$ is less or equal than the cost of the vector $\mathbf{x}_{i,t}$, then such trial vector replaces $\mathbf{x}_{i,t}$ in the next generation. Otherwise, $\mathbf{x}_{i,t}$ remains in the population for at least one more generation:

$$\mathbf{x}_{i,t+1} = \begin{cases} \mathbf{u}_{i,t}, & \text{if } f(\mathbf{u}_{i,t}) \leq f(\mathbf{x}_{i,t}), \\ \mathbf{x}_{i,t}, & \text{otherwise.} \end{cases} \quad (11.4)$$

Here, $f()$ represents the objective function. These processes are repeated until a termination criterion is attained or a predetermined generation number is reached.

11.3. Ellipse detection using DE

11.3.1 Data Preprocessing

In order to detect ellipse shapes, candidate images must be preprocessed first by an edge detection algorithm, which yields an edge map image. Then, the (x_i, y_i) coordinates for each edge pixel p_i are stored inside the edge vector $P = \{p_1, p_2, \dots, p_{N_p}\}$, with N_p being the total number of edge pixels.

11.3.2 Individual Representation

Just as a line requires two points to completely define its characteristics, an ellipse is defined by five points. Therefore, each candidate solution E (ellipse candidate) considers five edge points to represent an individual. Under such representation, edge points are selected following a random positional index within the edge array P . This procedure will encode a candidate solution as the ellipse that passes through five points p_1, p_2, p_3, p_4 and p_5 ($E = \{p_1, p_2, p_3, p_4, p_5\}$). Thus, by substituting the coordinates of each point of E into Eq. 11.5, we gather a set of five simultaneous equations which are linear in the five unknown parameters a, b, f, g and h .

$$ax^2 + 2hxy + by^2 + 2gx + 2fy + 1 = 0 \quad (11.5)$$

Considering the configuration of the edge points shown by Figure 11.2, the ellipse center (x_0, y_0) , the radius maximum (r_{\max}), the radius minimum (r_{\min}) and the ellipse orientation (θ) can be calculated as follows:

$$x_0 = \frac{hf - bg}{C}, \quad (11.6)$$

$$y_0 = \frac{gh - af}{C}, \quad (11.7)$$

$$r_{\max} = \sqrt{\frac{-2\Delta}{C(a+b-R)}}, \quad (11.8)$$

$$r_{\min} = \sqrt{\frac{-2\Delta}{C(a+b+R)}}, \quad (11.9)$$

$$\theta = \frac{1}{2} \arctan\left(\frac{2h}{a-b}\right) \quad (11.10)$$

$$\text{where } R^2 = (a-b)^2 + 4h^2, \quad C = ab - h^2 \quad \text{and} \quad \Delta = \det \begin{pmatrix} a & h & g \\ h & b & f \\ g & f & 1 \end{pmatrix} \quad (11.11)$$

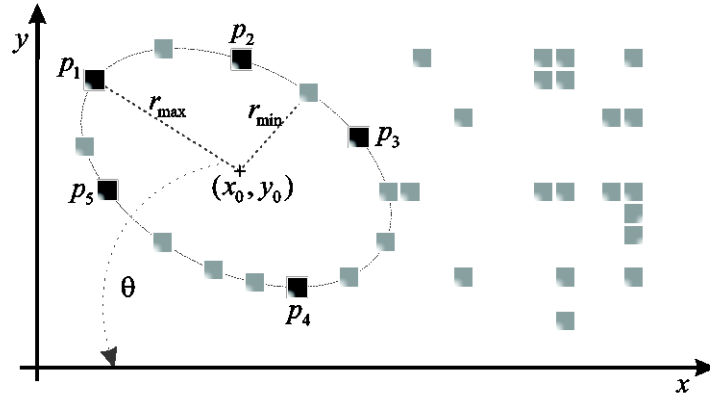


Fig. 11.2. Ellipse candidate (individual) built from the combination of points p_1, p_2, p_3, p_4 and p_5 .

11.3.3 Objective Function

Optimization refers to choosing the best element from one set of available alternatives. In the simplest case, it means to minimize an objective function or error by systematically choosing the values of variables from their valid ranges. In order to calculate the error produced by a candidate solution E , the ellipse coordinates are calculated as a virtual shape which, in turn, must also be validated, i.e. if it really exists in the edge image. The test set is represented by $S = \{s_1, s_2, \dots, s_{N_s}\}$, where N_s are the number of points over which the existence of an edge point, corresponding to E , should be tested.

The set S is generated by the Midpoint Ellipse Algorithm (MEA) (Bresenham & Jack, 1977), which is a searching method that seeks required points for drawing an ellipse. For any point (x, y) lying on the boundary of the ellipse with a, h, b, g and f , it does satisfy the equation $f_{ellipse}(x, y) \equiv r_{max}x^2 + r_{min}y^2 - r_{max}^2r_{min}^2$, where r_{max} and r_{min} represent the semi-major and semi-minor axis, respectively. However, MEA avoids computing square root calculations by comparing the pixel separation distances. A method for direct distance comparison is to test the halfway position between two pixels (sub-pixel distance) to determine if this midpoint is inside or outside the ellipse boundary. If the point is in the interior of the ellipse, the ellipse function is negative. Thus, if the point is outside the ellipse, the ellipse function is positive. Therefore, the error involved in locating pixel positions using the midpoint test is limited to one-half the pixel separation (sub-pixel precision). To summarize, the relative position of any point (x, y) can be determined by checking the sign of the ellipse function:

$$f_{ellipse}(x, y) \begin{cases} < 0 & \text{if } (x, y) \text{ is inside the ellipse boundary} \\ = 0 & \text{if } (x, y) \text{ is on the ellipse boundary} \\ > 0 & \text{if } (x, y) \text{ is outside the ellipse boundary} \end{cases} \quad (11.12)$$

The ellipse-function test in Eq. (11.12) is applied to mid-positions between pixels nearby the ellipse path at each sampling step. Figure 11.3(a) and 11.4(a) show the midpoint between the two candidate pixels at sampling position. The ellipse is used to divide the quadrants into two regions the limit of the two regions is the point at which the curve has a slope of -1 as shown in Figure 11.4.

In MEA the computation time is reduced by considering the symmetry of ellipses. Ellipses sections in adjacent octants within one quadrant are symmetric with respect to the $dy/dx=-1$ line di-

viding the two octants. These symmetry conditions are illustrated in Figure 11.4. The algorithm can be considered as the quickest providing a sub-pixel precision (Van Aken, 1984). However, in order to protect the MEA operation, it is important to assure that points lying outside the image plane must not be considered in S.

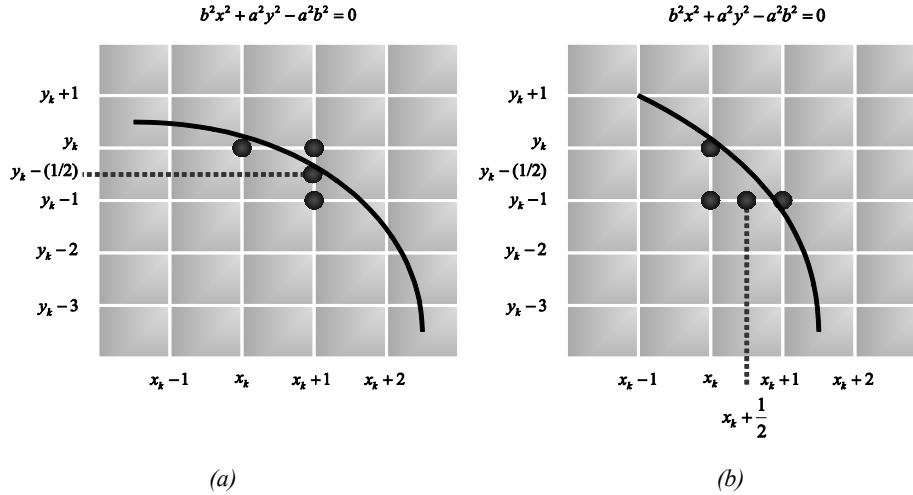


Fig. 11.3. (a) symmetry of the ellipse: an estimated one octant which belong to the first region where the slope is greater than -1, (b) In this region the slope will be less than -1 to complete the octant and continue to calculate the same so the remaining octants.

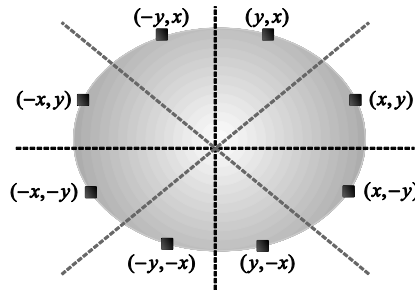


Fig. 11.4. Midpoint between candidate pixels at sampling position x_k along an elliptical path.

The objective function $J(E)$ represents the matching error produced between the pixels S of the ellipse candidate E and the pixels that actually exist in the edge image, yielding:

$$J(E) = 1 - \frac{\sum_{v=1}^{N_s} G(x_v, y_v)}{N_s} \tag{11.13}$$

where $G(x_v, y_v)$ is a function that verifies the pixel existence in (x_v, y_v) , with $(x_v, y_v) \in S$ and N_s being the number of pixels lying on the perimeter corresponding to E currently under testing.

Hence, function $G(x_v, y_v)$ is defined as:

$$G(x_v, y_v) = \begin{cases} 1 & \text{if the pixel } (x_v, y_v) \text{ is an edge point} \\ 0 & \text{otherwise} \end{cases} \tag{11.14}$$

A value of $J(E)$ near to zero implies a better response from the “ellipsoid” operator. Figure 5 shows the procedure to evaluate a candidate action E with its representation as a virtual shape S . Figure 11.5(a) shows the original edge map, while Figure 11.5(b) presents the virtual shape S representing the individual $E = \{p_1, p_2, p_3, p_4, p_5\}$. In Figure 11.5(c), the virtual shape S is compared to the original image, point by point, in order to find coincidences between virtual and edge points. The individual has been built from points p_1, p_2, p_3, p_4 and p_5 which are shown by Fig. 11.5(a).

The virtual shape S , obtained by MEA, gathers 47 points ($N_s = 47$) with only 25 of them existing

in both images (shown as darker points in Fig. 11.5(c)) and yielding: $\sum_{v=1}^{N_s} G(x_v, y_v) = 25$, therefore $J(E)=0.255$. This value indicates that the obtained virtual shape S , has a considerable number of coincidences with the edge map.

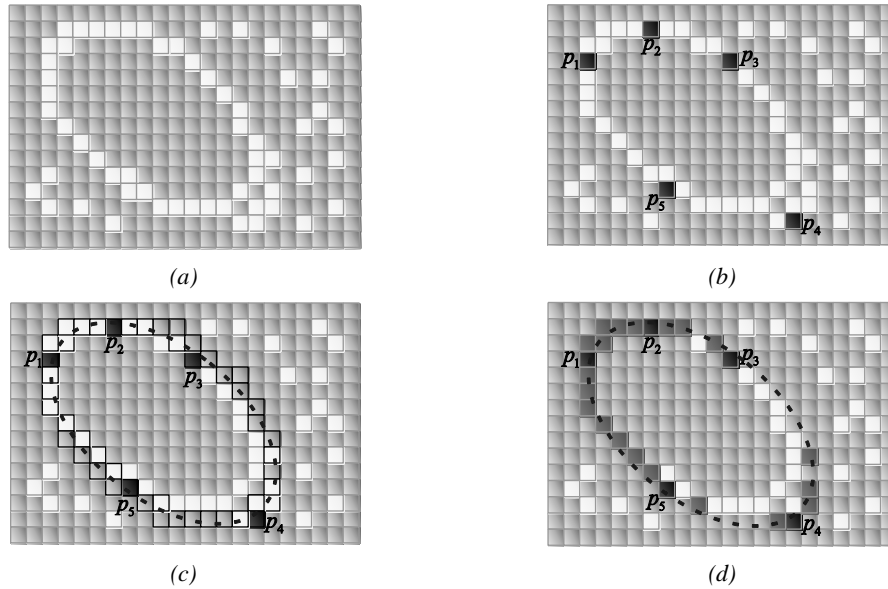


Fig. 11.5. Evaluation of a candidate solution E : the image in (a) shows the original edge map while (b) presents the individual $E = p_1, p_2, p_3, p_4$ and p_5 to be evaluated. The image (c) shows the virtual shape S and its corresponding pixels in bold line. The image in (d) shows coincidences between both images which have been marked by darker pixels while the virtual shape is also depicted through a dashed line

11.3.4 Implementation of de DE for ellipse detection

The ellipse detector algorithm based on DE can be summarized in the following steps:

Step 1	Set the DE parameters $F=0.25$ and $CR=0.8$.
Step 2	Initialize the population of m individuals $\mathbf{E}^k = \{E_1^k, E_2^k, \dots, E_m^k\}$ where each decision variable p_1, p_2, p_3, p_4 and p_5 of E_a^k is set randomly within the interval $[1, N_p]$. All values must be integers. Considering that $k=0$ and $a \in (1, 2, \dots, m)$.
Step 3	Evaluate the objective value $J(E_a^k)$ for all m individuals, and determining the $E^{best,k}$ showing the best fitness value, such that $E^{best,k} \in \{\mathbf{E}^k\} \mid J(E^{best,k}) = \min\{J(E_1^k), J(E_2^k), \dots, J(E_m^k)\}$.

Step 4	<p>Generate the trial population $\mathbf{T} = \{T_1, T_2, \dots, T_m\}$:</p> <pre> for (i=1; i<m+1; i++) do $r_1 = \text{floor}(\text{rand}(0,1) \cdot m)$; while ($r_1 = i$); do $r_2 = \text{floor}(\text{rand}(0,1) \cdot m)$; while (($r_2 = i$) or ($r_2 = r_1$)); jrand=floor($5 \cdot \text{rand}(0,1)$); for (j=1; j<6; j++) // generate a trial vector if (rand(0,1)<=CR or j=jrand) $T_{j,i} = E_j^{best,k} + F \cdot (E_{j,r_1}^k - E_{j,r_2}^k)$; else $T_{j,i} = E_{j,i}^k$; end if end for end for </pre>
Step 5	<p>Evaluate the fitness values $J(T_i)$ ($i \in \{1, 2, \dots, m\}$) of all trial individuals. Check all individuals. If a candidate parameter set is not physically plausible, i.e. out of the range $[1, N_p]$, then an exaggerated cost function value is returned. This aims to eliminate “unstable” individuals.</p>
Step 6	<p>Select the next population $\mathbf{E}^{k+1} = \{E_1^{k+1}, E_2^{k+1}, \dots, E_m^{k+1}\}$:</p> <pre> for (i=1; i<m+1; i++) if ($J(T_i) < J(E_i^k)$) $E_i^{k+1} = T_i$ else $E_i^{k+1} = E_i^k$ end if end for </pre>
Step 7	<p>If the iteration number (NI) is met, then the output $E^{best,k}$ is the solution (an actual ellipse contained in the image), otherwise go back to Step 3.</p>

11.4. The White Blood Cell Detector

In order to detect WBC, the presented detector combines a segmentation strategy with the ellipse detection approach presented in section 11.3.

11.4.1 Image Preprocessing

To employ the proposed detector, smear images must be preprocessed to obtain two new images: the segmented image and its corresponding edge map. The segmented image is produced by using a segmentation strategy whereas the edge map is generated by a border extractor algorithm. Such edge map is considered by the objective function to measure the resemblance of a candidate ellipse with an actual WBC.

The goal of the segmentation strategy is to isolate the white blood cells (WBC's) from other structures such as red blood cells and background pixels. Information of color, brightness and gradients

are commonly used within a thresholding scheme to generate the labels to classify each pixel. Although a simple histogram thresholding can be used to segment the WBC's, at this work the Diffused Expectation-Maximization (DEM) has been used to assure better results (Boccignone, et al., 2004).

DEM is an Expectation-Maximization (EM) based algorithm which has been used to segment complex medical images (Boccignone, et al., 2007). In contrast to classical EM algorithms, DEM considers the spatial correlations among pixels as a part of the minimization criteria. Such adaptation allows to segment objects in spite of noisy and complex conditions. The method models an image as a finite mixture, where each mixture component corresponds to a region class and uses a maximum likelihood approach to estimate the parameters for each class, via the expectation maximization (EM) algorithm, which is coupled to anisotropic diffusion over classes in order to account for the spatial dependencies among pixels.

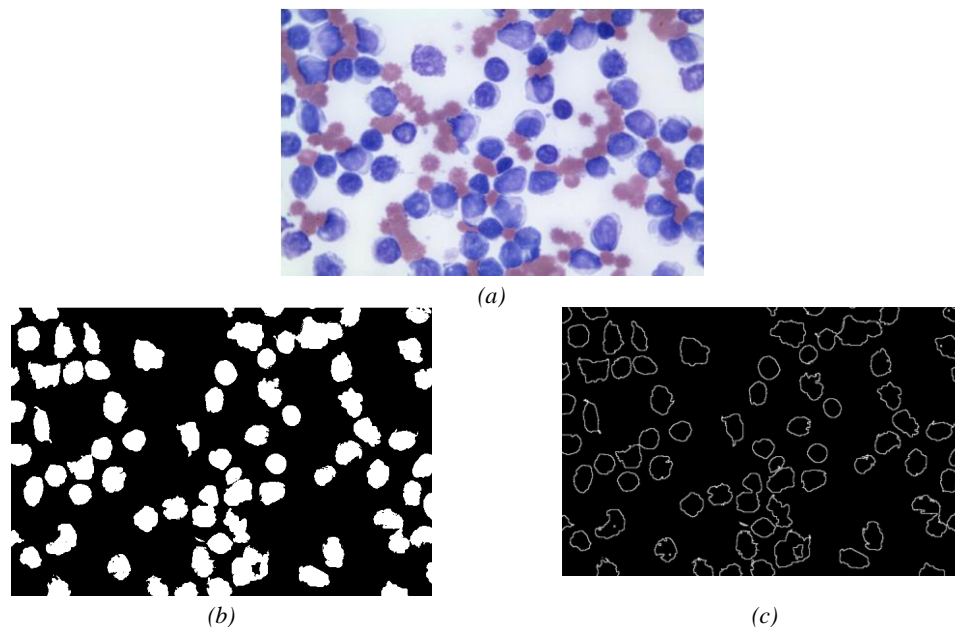


Fig. 11.6. Preprocessing process. (a) original smear image, (b) segmented image obtained by DEM and (c) the edge map obtained by using the morphological edge detection procedure.

For the WBC's segmentation, it has been used the implementation of DEM provided in (Boccignone, et al., Mathworks, 2004). Since the implementation allows to segment gray-level images and color images, it can be used for operating over all smear images with no regard about how each image has been acquired. The DEM has been configured considering three different classes ($K=3$), $g(\nabla h_{ik}) = |\nabla h_{ik}|^{-9/5}$, $\lambda=0.1$ and $m=10$ iterations. These values have been found as the best configuration set according to (Boccignone et al., 2004).

As a final result of the DEM operation, three different thresholding points are obtained: the first corresponds to the WBC's, the second to the red blood cells whereas the third represents the pixels classified as background. Figure 11.6(b) presents the segmentation results obtained by the DEM approach employed at this work considering the Figure 11.6(a) as the original image.

Once the segmented image has been produced, the edge map is computed. The purpose of the edge map is to obtain a simple image representation that preserves object structures. The DE-based detector operates directly over the edge map in order to recognize ellipsoidal shapes. Several algo-

rithms can be used to extract the edge map; however, at this work, the morphological edge detection procedure (Gonzalez & Woods, 2008) has been used to accomplish such a task. Morphological edge detection is a traditional method to extract borders from binary images in which original images (I_B) are eroded by a simple structure element (I_E) composed by a matrix-template of 3x3 with all its values equal to one. Then, the eroded image is inverted (\bar{I}_E) and compared with the original image ($\bar{I}_E \wedge I_B$) in order to detect pixels which are present in both images. Such pixels compose the computed edge map from I_B . Figure 11.6(c) shows the edge map obtained by using the morphological edge detection procedure.

11.4.2 Ellipse Detection Approach

The edge map is used as input image for the ellipse detector presented in Section 11.3. Table 11.1 presents the parameter set that has been used in this work for the DE algorithm after several calibration examples have been conducted. The final configuration matches the best possible calibration proposed in (Wang & Huang, 2010), where it has been analyzed the effect of modifying the DE-parameters for several generic optimization problems. The population-size parameter ($m=20$) has been selected considering the best possible balance between convergence and computational overload. Once it has been set, such configuration has been kept for all test images employed in the experimental study.

m	F	CR	NI
20	0.25	0.80	200

Table 11.1. DE parameters used for leukocytes detection in medical images.

Under such assumptions, the complete process to detect WBC's is implemented as follows:

Step 1	Segment the WBC's using the DEM algorithm (described in 11.4.1)
Step 2	Get the edge map from the segmented image.
Step 3	Start the ellipse detector based in DE over the edge map while saving best ellipses (Section 11.3).
Step 4	Define parameter values for each ellipse that identify the WBC's.

11.4.3. Numerical Example

In order to present the algorithm's step-by-step operation, a numerical example has been set by applying the proposed method to detect a single leukocyte lying inside of a simple image. Figure 11.7(a) shows the image used in the example. After applying the threshold operation, the WBC is located besides few other pixels which are merely noise (see Fig. 11.7(b)). Then, the edge map is subsequently computed and stored pixel by pixel inside the vector P . Fig. 11.7(c) shows the resulting image after such procedure.

The DE-based ellipse detector is executed using information of the edge map (for the sake of easiness, it only considers a population of four particles). Like all evolutionary approaches, DE is a population-based optimizer that attacks the starting point problem by sampling the search space at multiple, randomly chosen, initial particles. By taking five random pixels from vector P , four different particles are constructed. Fig. 11.7(d) depicts the initial particle distribution $\mathbf{E}^0 = \{E_1^0, E_2^0, E_3^0, E_4^0\}$. By using the DE operators, four different trial particles $\mathbf{T} = \{T_1, T_2, T_3, T_4\}$

(ellipses) are generated, their locations are shown in Fig. 11.7(e). Then, the new population \mathbf{E}^1 is selected considering the best elements obtained among the trial elements \mathbf{T} and the initial particles \mathbf{E}^0 . The final distribution of the new population is depicted in Fig. 11.7(f). Since the particles E_2^0 and E_3^0 hold (in Fig. 11.7(f)) a better fitness value ($J(E_2^0)$ and $J(E_3^0)$) than the trial elements T_2 and T_3 , they are considered as particles of the final population \mathbf{E}^1 . Figures 7(g) and 7(h) present the second iteration produced by the algorithm whereas Fig. 6(i) shows the population configuration after 25 iterations. From Fig. 11.7(i), it is clear that all particles have converged to a final position, which is able to accurately cover the WBC.

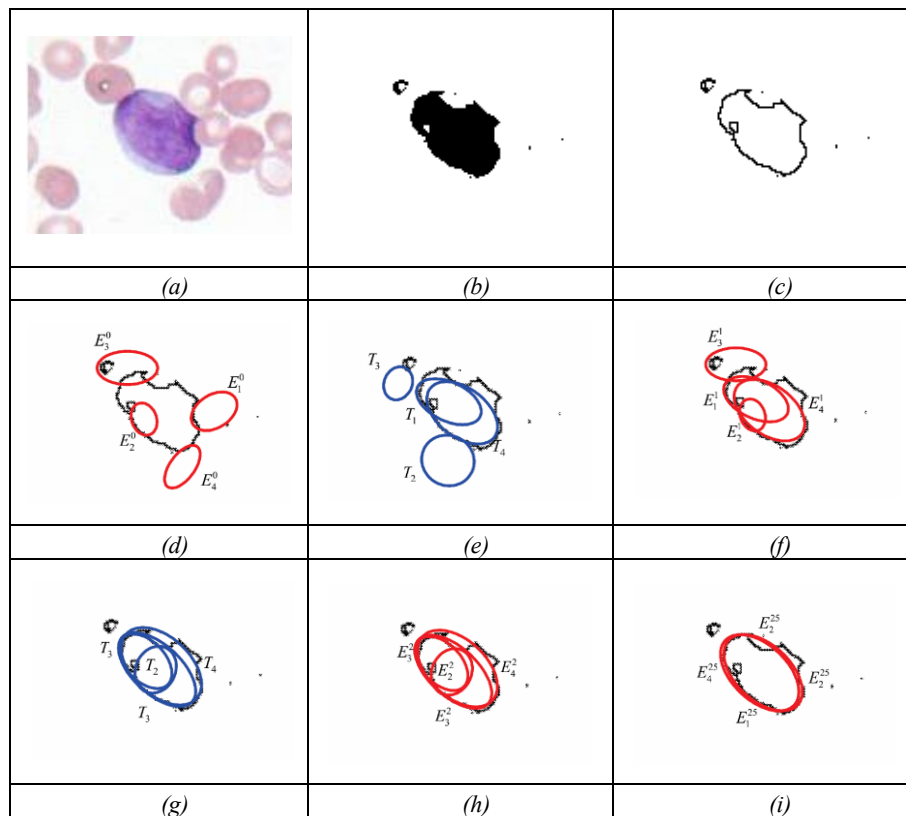


Fig. 11.7. Detection numerical example: (a) The image used as example. (b) Segmented image. (c) Edge map. (d) Initial particles \mathbf{E}^0 . (e) Trial elements \mathbf{T} produced by the DE operators. (f) New population \mathbf{E}^1 . (g) Trial elements produced considering \mathbf{E}^1 as input population. (h) New population \mathbf{E}^2 . (i) Final particle configuration after 25 iterations.

11.5. Experimental Results

Experimental tests have been developed in order to evaluate the performance of the WBC detector. It was tested over microscope images from blood-smears holding a 960 x 720 pixel resolution. They correspond to supporting images on the leukemia diagnosis. The images show several complex conditions such as deformed cells and overlapping with partial occlusions. The robustness of the algorithm has been tested under such demanding conditions. All the experiments has been developed using an Intel Core i7-2600 PC, with 8GB in RAM.

Figure 11.8(a) shows an example image employed in the test. It was used as input image for the WBC detector. Figure 11.8(b) presents the segmented WBC's obtained by the DEM algorithm. Figures 11.8(c) and 11.8(d) present the edge map and the white blood cells after detection, respectively. The results show that the proposed algorithm can effectively detect and mark blood cells despite cell occlusion, deformation or overlapping. Other parameters may also be calculated through the algorithm: the total area covered by white blood cells and relationships between several cell sizes.

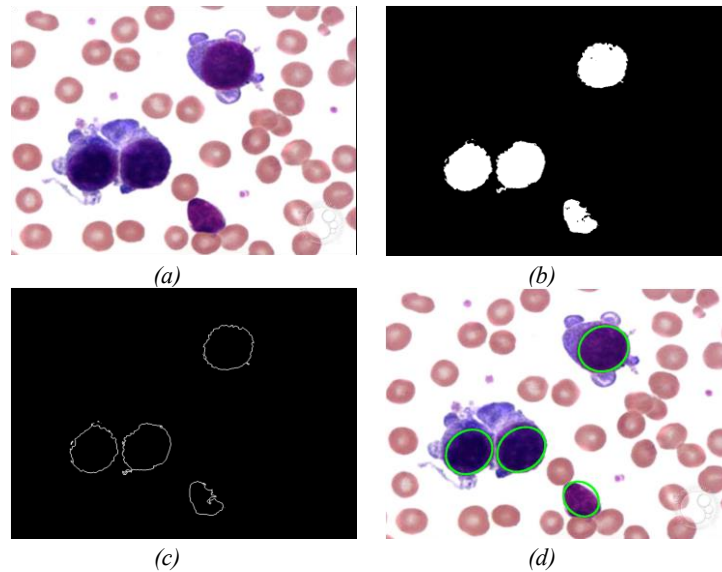


Fig. 11.8. Resulting images of the first test after applying the WBC detector: (a) Original image, (b) image segmented by the DEM algorithm, (c) edge map and (d) the white detected blood cells.

Other example is presented in Figure 11.9. It represents a complex example with an image showing seriously deformed cells. Despite such imperfections, the proposed approach can effectively detect the cells as it is shown in Figure 11.9(d).

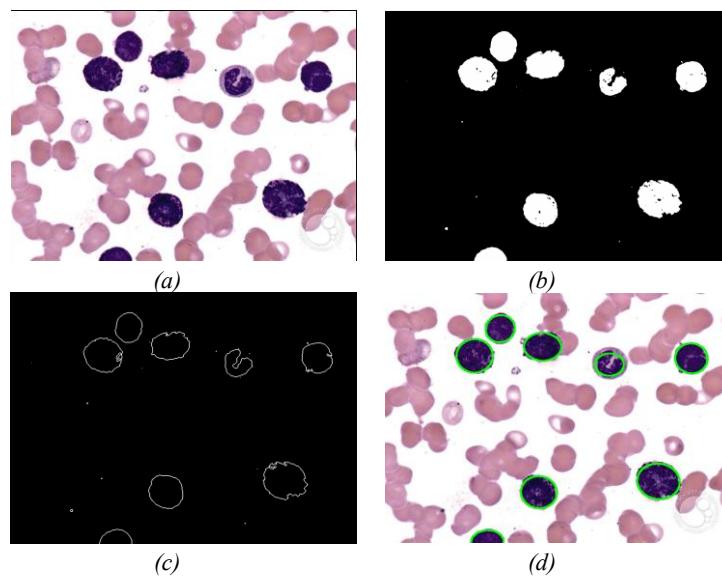


Fig. 11.9. Resulting images of the second test after applying the WBC detector: (a) Original image, (b) image segmented by the DEM algorithm, (c) edge map and (d) the white detected blood cells.

11.6. Comparisons to other Methods

A comprehensive set of smear-blood test images is used to test the performance of the proposed approach. We have applied the proposed DE-based detector to test images in order to compare its performance to other WBC detection algorithms such as the Boundary Support Vectors (BSV) approach (Wang & Chu, 2009), the Iterative Otsu (IO) method (Wu, et al., 2006), the Wang algorithm (Wang Shitong et al., 2007a) and the Genetic algorithm-based (GAB) detector (Karkavitsas & Rangoussi, 2008). In all cases, the algorithms are tuned according to the value set which is originally proposed by their own references.

11.6.1 Detection Comparison

To evaluate the detection performance of the proposed detection method, Table 11.2 tabulates the comparative leukocyte detection performance of the BSV approach, the IO method, the Wang algorithm, the BGA detector and the proposed method, in terms of detection rates and false alarms. The experimental data set includes 50 images which are collected from the ASH Image Bank (<http://imagebank.hematology.org/>). Such images contain 517 leukocytes (287 bright leukocytes and 230 dark leukocytes according to smear conditions) which have been detected and counted by a human expert. Such values act as ground truth for all the experiments. For the comparison, the detection rate (DR) is defined as the ratio between the number of leukocytes correctly detected and the number leukocytes determined by the expert. The False Alarm Rate (FAR) is defined as the ratio between the number of non-leukocyte objects that have been wrongly identified as leukocytes and the number leukocytes which have been actually determined by the expert.

Experimental results show that the proposed DE method, which achieves 98.26% leukocyte detection accuracy with 2.71% false alarm rate, is compared favorably against other WBC detection algorithms, such as the BSV approach, the IO method, the Wang algorithm and the BGA detector.

Leukocyte Type	Method	Leukocytes detected	Missing	False alarms	DR	FAR
Bright Leukocytes (287)	BSV	130	157	84	45.30%	29.27%
	IO	227	60	73	79.09%	25.43%
	Wang	231	56	60	80.49%	20.90%
	BGA	220	67	22	76.65%	7.66%
	DE-based	281	6	11	97.91%	3.83%
Dark Leukocytes (230)	BSV	105	125	59	46.65%	25.65%
	IO	183	47	61	79.56%	26.52%
	Wang	196	34	47	85.22%	20.43%
	BGA	179	51	23	77.83%	10.00%
	DE-based	227	3	3	98.70%	1.30%
Overall (517)	BSV	235	282	143	45.45%	27.66%
	IO	410	107	134	79.30%	25.92%
	Wang	427	90	107	82.59%	20.70%
	BGA	399	118	45	77.18%	8.70%
	DE-based	508	9	14	98.26%	2.71%

Table 11.2. Comparative leukocyte detection performance of the BSV approach, the IO method, the Wang algorithm, the BGA detector and the proposed DE method over the data set, which contains 30 images and 426 leukocytes

11.6.2 Robustness Comparisson

Images of blood smear are often deteriorated by noise due to various sources of interference and other phenomena that affect the measurement processes in imaging and data acquisition systems. Therefore, the detection results depend on the algorithm's ability to cope with different kinds of noises. In order to demonstrate the robustness in the WBC detection, the proposed DE approach is compared to the BSV approach, the IO method, the Wang algorithm and the BGA detector under noisy environments. In the test, two different experiments have been studied.

The first inquest explores the performance of each algorithm when the detection task is accomplished over images corrupted by Salt & Pepper noise. The second experiment considers images polluted by Gaussian noise. Salt & Pepper and Gaussian noise are selected for the robustness analysis because they represent the most compatible noise types commonly found in images of blood smear (Landi & Piccolomini, 2012). The comparison considers the complete set of 50 images presented in Section 11.6.1 containing 517 leukocytes which have been detected and counted by a human expert.

The added noise is produced by MatLab©, considering two noise levels of 5% and 10% for Salt & Pepper noise whereas $\sigma = 5$ and $\sigma = 10$ are used for the case of Gaussian noise. Such noise levels, according to (Tapiovaara & Wagner, 1993), correspond to the best trade of between detection difficulty and the real existence in medical imaging. If higher noise levels are used then the detection process would be unnecessarily complicated without representing a feasible image condition.

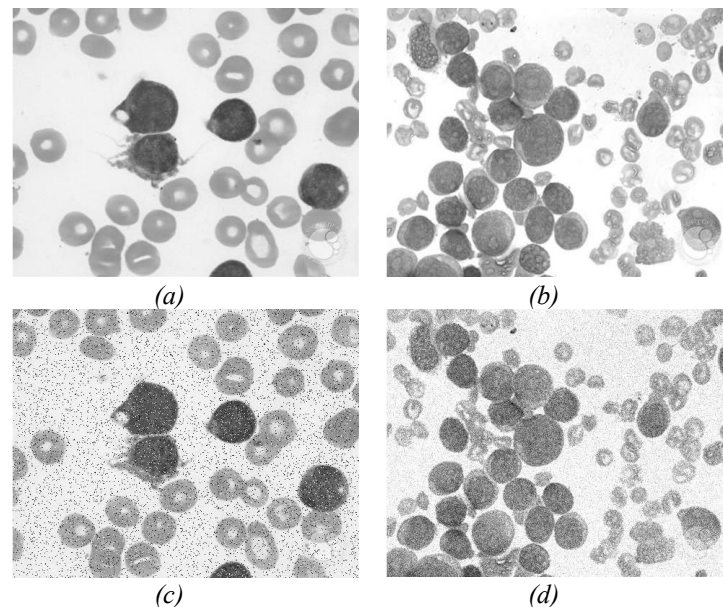


Fig. 11.10. Examples of images included in the experimental set for robustness comparison. (a)-(b) Original images. (c) Image contaminated with 10% of Salt & Pepper noise and (d) image polluted with $\sigma = 10$ of Gaussian noise.

Fig. 11.10 shows two examples of the experimental set. The outcomes in terms of the detection rate (DR) and the false alarm rate (FAR) are reported for each noise type in Table 11.3 and Table 11.4. The results show that the proposed DE algorithm presents the best detection performance, achieving in the worst case a DR of 89.55% and 91.10%, under contaminated conditions of Salt & Pepper and Gaussian noise, respectively. On the other hand, the DE detector possesses the least degradation performance presenting a FAR value of 5.99% and 6.77%.

Noise level	Method	Leukocytes detected	Missing	False alarms	DR	FAR
5% Salt & Pepper noise 517 Leukocytes	BSV	185	332	133	34.74%	26.76%
	IO	311	206	106	63.38%	24.88%
	Wang	250	176	121	58.68%	27.70%
	BGA	298	219	135	71.83%	24.18%
	DE-based	482	35	32	91.55%	7.04%
10% Salt & Pepper noise 517 Leukocytes	BSV	105	412	157	20.31%	30.37%
	IO	276	241	110	53.38%	21.28%
	Wang	214	303	168	41.39%	32.49%
	BGA	337	180	98	65.18%	18.95%
	DE-based	463	54	31	89.55%	5.99%

Table 11.3. Comparative WBC detection among methods that considers the complete data set of 30 images corrupted by different levels of Salt & Pepper noise

Noise level	Method	Leukocytes detected	Missing	False alarms	DR	FAR
$\sigma = 5$ Gaussian noise 517 Leukocytes	BSV	214	303	98	41.39%	18.95%
	IO	366	151	87	70.79%	16.83%
	Wang	358	159	84	69.25%	16.25%
	GAB	407	110	76	78.72%	14.70%
	DE-based	487	30	21	94.20%	4.06%
$\sigma = 10$ Gaussian noise 517 Leukocytes	BSV	162	355	129	31.33%	24.95%
	IO	331	186	112	64.02%	21.66%
	Wang	315	202	124	60.93%	23.98%
	GAB	363	154	113	70.21%	21.86%
	DE-based	471	46	35	91.10%	6.77%

Table 11.4. Comparative WBC detection among methods that considers the complete data set of 30 images corrupted by different levels of Gaussian noise.

11.6.3 Stability Comparison

In order to compare the stability performance of the proposed method, its results are compared to those reported by Wang Shitong et al., (2007a) which is considered as an accurate technique for the detection of WBC.

The Wang algorithm is an energy-minimizing method which is guided by internal constraint elements and influenced by external image forces, producing the segmentation of WBC's at a closed contour. As external forces, the Wang approach uses edge information which is usually represented by the gradient magnitude of the image. Therefore, the contour is attracted to pixels with large image gradients, i.e. strong edges. At each iteration, the Wang method finds a new contour configuration, which minimizes the energy that corresponds to external forces and constraint elements.

In the comparison, the net structure and its operational parameters, corresponding to the Wang algorithm, follow the configuration suggested in (Wang Shitong et al., 2007a) while the parameters for the DE-based algorithm are taken from Table 11.1.

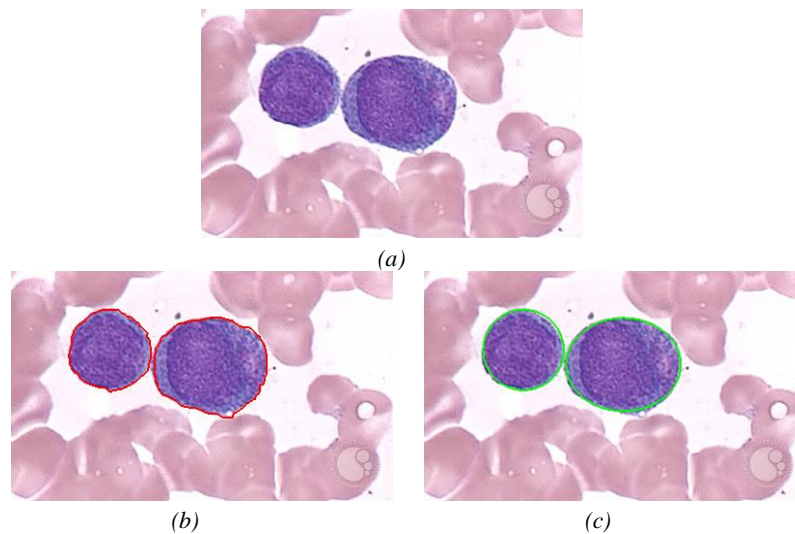


Fig. 11.11. Comparison of the DE and the Wang's method for white blood cell detection in medical images. (a) Original image. (b) Detection using the Wang's method, (c) Detection after applying the DE method.

Figure 11.11 shows the performance of both methods considering a test image with only two white blood cells. Since the Wang method uses gradient information in order to appropriately find a new contour configuration, it needs to be executed iteratively in order to detect each structure (WBC). Figure 11(b) shows the results after the Wang approach has been applied considering only 200 iterations. Furthermore, Figure 11(c) shows results after applying the DE-based method which has been proposed in this paper.

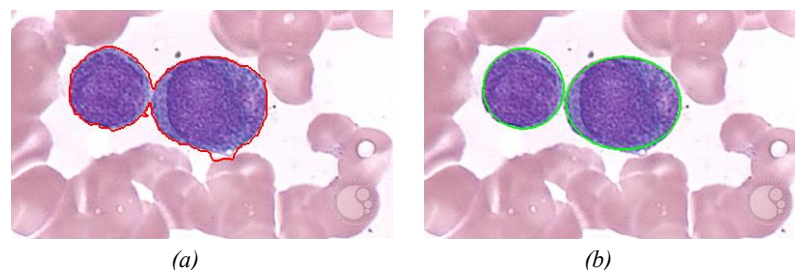


Fig. 11.12. Result comparison for the white blood cells detection showing (a) Wang's algorithm after 400 cycles and (b) DE detector method considering 1000 cycles.

The Wang algorithm uses the fuzzy cellular neural network (FCNN) as optimization approach. It employs gradient information and internal states in order to find a better contour configuration. In each iteration, the FCNN tries, as contour points, different new pixel positions which must be located nearby the original contour position. Such fact might cause the contour solution to remain trapped into a local minimum. In order to avoid such a problem, the Wang method applies a considerable number of iterations so that a near optimal contour configuration can be found. However, when the number of iterations increases the possibility to cover other structures increases too. Thus, if the image has a complex background (just as smear images do) or the WBC's are too close, the method gets confused so that finding the correct contour configuration from the gradient magnitude is not easy. Therefore, a drawback of Wang's method is related to its optimal iteration number (instability). Such number must be determined experimentally as it depends on the image context and its complexity. Figure 11.12(a) shows the result of applying 400 cycles of the Wang's

algorithm while Figure 11.12(b) presents the detection of the same cell shapes after 1000 iterations using the proposed algorithm. From Fig. 11.12(a), it can be seen that the contour produced by Wang's algorithm degenerates as the iteration process continues, wrongly covering other shapes lying nearby.

In order to compare the accuracy of both methods, the estimated WBC area which has been approximated by both approaches, is compared to the actual WBC size considering different degrees of evolution i.e. the cycle number for each algorithm. The comparison considers only one WBC because it is the only detected shape in the Wang's method. Table 11.5 shows the averaged results over twenty repetitions for each experiment. In order to enhance the analysis, Fig. 11.13 illustrates the Error-percentage vs. Iterations evolution from an extended data set which has been compiled from Table 11.5.

Algorithm	Iterations	Error%
Wang	30	88%
	60	70%
	200	1%
	400	121%
	600	157%
DE-based	30	24.30%
	60	7.17%
	200	2.25%
	400	2.25%
	600	2.25%

Table 11.5. Error in cell's size estimation after applying the DE algorithm and the Wang's method to detect one leukocyte embedded into a blood-smear image. The error is averaged over twenty experiments.

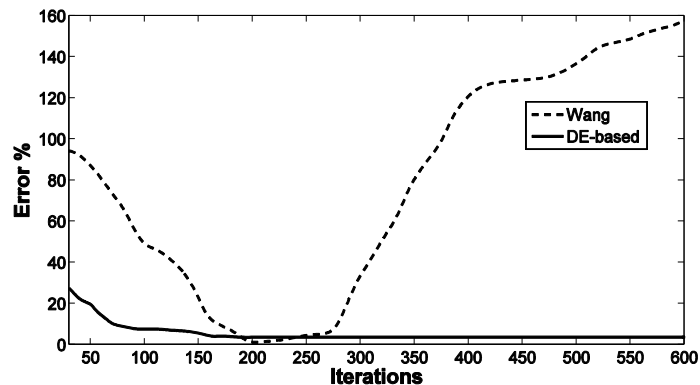


Fig. 11.13. Error percentage vs. Iterations evolution from an extended data set from Table 11.5.

11.7. Conclusions

In this chapter, an algorithm for the automatic detection of blood cell images based on the DE algorithm has been presented. The approach considers the complete process as a multiple ellipse detection problem. The proposed method uses the encoding of five edge points as candidate ellipses in the edge map of the smear. An objective function allows to accurately measure the resemblance of a candidate ellipse with an actual WBC on the image. Guided by the values of such objective function, the set of encoded candidate ellipses are evolved using the DE algorithm so that

they can fit into actual WBC on the image. The approach generates a sub-pixel detector which can effectively identify leukocytes in real images.

The performance of the DE-method has been compared to other existing WBC detectors (the Boundary Support Vectors (BSV) approach (Wang & Chu, 2009), the Iterative Otsu (IO) method (Wu et al., 2006), the Wang algorithm (Wang Shitong et al., 2007a) and the Genetic algorithm-based (BGA) detector (Karkavitsas & Rangoussi, 2008) considering several images which exhibit different complexity levels. Experimental results demonstrate the high performance of the proposed method in terms of detection accuracy, robustness and stability.

Chapter 12

A Multi-Level Thresholding method for Breast Thermograms Analysis using Dragonfly Algorithm

Breast cancer is one of the most common diseases and the second cause of death in women around the world. The presence of a cancerous tumor increase temperature in the region near to it, such heating is then transferred to the skin surface. In this sense, screening seeks to help in cancer diagnostic process before symptoms became evident in a person, different imaging techniques are employed for this purpose (Mammography, Ultrasonography, X-ray, Magnetic Resonance, etc). In the past decade, thermography has shown its major potential to early diagnosis of breast diseases. Thermographic images provide information related to vascular or physiological changes and have some advantages regarding other diagnostic methods; they are non-ionizing, non-invasive, passive, painless and real-time screening. On the other hand, thresholding has been widely used to solve several problems. It is regularly the first step in the process of image analysis that uses histograms to classify the pixels in the image. Segmentation of medical digital images has been stated as an important task for several medical applications. This chapter proposes a segmentation technique for thermographic images that consider the spatial information of the pixel contained in the image. This approach employs a novel optimization technique called the Dragonfly Algorithm to compute the best thresholds that segment the image. The experimental results exhibit a well-performance of the proposal in comparison to the other methods over a set of randomly selected thermograms retrieved from the Database for Research Mastology with Infrared Image. The presented approach could provide a highly reliable clinical decision support, which aims to help clinicians in performing a diagnosis using thermography images.

12.1. Introduction

Breast cancer is the most commonly detected cancer in females around the world. Approximately 1 in 8 women in the United States of America (Society, 2013) and 2 of 5 globally (Canadian Cancer Society, 2015) will develop breast cancer throughout their lifetime. Since 2013 breast cancer was typified as the second cause of death in women (Society, 2013). The increase of cases in the incidence of breast cancer is perhaps due to the change in lifestyle factors, the rapid growth of industries and urbanization (PA, Naik Nagappa, Udupa, & Mathew, 2015). In the same context, several studies have shown that if breast cancer is discovered in early stages increases the survival rate of women (better prognosis), and therefore, it would allow providing proper treatment (Gautherie, 1980; Lee & Chen, 2015; Walker & Kaczor, 2012). This fact motivates researchers to search novel techniques that reach an early and accurate diagnosis, thus, improving the life expectancy of patients.

Some studies reported that the presence of a cancerous tumour increase temperature in the region near to it, that is caused by the Nitric Oxide produced for cancer cells (Usuki et al., 1990). Nitric Oxide interferes with the normal blood vessel flow and produces local vasodilatation and consequently an increased blood flow (Anbar et al., 2001; Gamagami, 1996; Ignarro, et al., 1986), this vasodilatation process causes a higher temperature compared to the standard breast tissue temperature. It has been reported, that deep breast cancerous lesions can increase the temperature in the skin surface (Anbar, 1994).

It is known that objects irradiate infrared energy above absolute zero temperature. The Stefan Boltzmann Law (Boltzmann, 1884) states that the total radiation emitted by an object is directly proportional to the fourth power of its temperature in Kelvin. Therefore, it describes the relation between the energy radiated by an object and its temperature. Hence, it is possible to get the body temperature distribution by measuring its infrared radiation (Kermani, Samadzadehaghdam, & EtehadTavakol, 2015).

On the other hand, screening seeks to help in cancer diagnostic process before symptoms became evident in a person. Therefore, it may help to reduce the time for finding a diagnose and thus discover cancer at its early stage. Over the last decades, several medical imaging techniques have been intensively used with the purpose of detecting early signs of breast cancer, which include Ultrasound, Positron Emission Tomography (PET), Computed Tomography (CT), Magnetic Resonance Imaging (MRI), and Mammography. Mammography screening method is the most used diagnostic imaging technique nowadays, despite the uncomfortable, distressing and painful procedure required for its capture (Kolb, et al., 2002; Sathish, et al., 2017; Tan, et al., 2007), as well as the several false positives or negatives in women with dense breasts, because dense breasts could hide a tumor underneath the tissue (4-34% of false negative ratio) (Etehadtavakol, et al., 2013; Huynh, et al., 1998; Lee & Chen, 2015). Moreover, mammography uses X-Ray radiation, and this could increase the risks of future cancer since this type of radiation is noxious to human tissue (Huynh et al., 1998; Tan et al., 2007).

Furthermore, mammography technique presents accuracy problems regarding small tumours; several studies show that there is approximately a 4-34% of false negative ratio with this procedure (Huynh et al., 1998). In this sense, it represents a major nuisance and causes pain in patients (Kolb et al., 2002).

Another breast screening method significantly employed is ultrasound (Kuhl et al., 2005), it has been used majorly in young women with the purpose of diminishing the radiation received in a woman's life. Nonetheless, this method presents several performance issues, due to noise conditions and expertise from the technician that captures it. These problems lead to failures of this technique when it tries to detect micro-calcifications and deep breast tissue (Qi & Diakides, 2003).

There is another technique for breast screening which is mostly used as a complementary diagnostic method, which is Magnetic Resonance Imaging (MRI). Although MRI has better capabilities than mammography or ultrasound, it has numerous negative issues, such as a high rate of false-positive and a long data acquisition time (Port, Park, Borgen, Morris, & Montgomery, 2007). In the past decade, thermography has shown its major potential to early diagnosis of breast diseases. Thermography has the potential of detecting first signs of forming cancer earlier than mammography (Etehadtavakol, et al., 2010; Keyserlingk, et al., 1998; Qi & Diakides, 2003).

The temperature related to each point of breasts skin surface may be analyzed with the purpose of detecting pathologies. In thermal imaging, the body emissions are sensed by an infrared camera and display temperature distribution (Borchardt, et al., 2013; Diakides & Bronzino, 2006). Thus, results settle thermography as a complementary method for breast pathology detection. In some works (Elhoseny et al., 2018; Ng, 2009; Ng & Sud, 2001; Ng & Sudharsan, 2015), it has been stated that thermography could have the potential for breast cancer detection up to 10 years earlier than the most used method.

Thermographic images provide information related to vascular or physiological changes (Etehadtavakol, et al., 2010), in this case of breasts. Thermal images had some advantages regarding other diagnostic methods, they are non-ionizing, non-invasive, passive, painless and real-time, which makes it safe to repeat in case of monitoring. The capability to not use ionizing radiation

makes this method appropriate for pregnant or nursing women too (Etehadtavakol & NG, 2013). It is comfortable for patients because it does not require contact with the skin and presents advantages for young women, due to young women's tissues are dense, and early diagnosis is difficult and risky through X-Ray screening. Some authors have indicated that Computer-aided detection systems (CADs) might help physicians in diagnosing by providing valuable information on pathologies or abnormalities existing in the medical images (Arakeri & Reddy, 2013; Moghbel & Mashohor, 2013).

On the other hand, the segmentation technique has been widely used to solve several problems. It is regularly the first step in the process of image analysis because image segmentation is a crucial step in digital image processing applications to observe abnormal regions or to classify the elements in such images (Gharbia, et al., 2018; Kim & Seo, 2013; Manh & Lee, 2013). The primary objective of segmentation is to divide an image into different areas based on established criteria, such as color, texture, brightness or motion, to simplify the next analysis step (Chang & Teng, 2007; Dhungana, 2002; Sonali, et al., 2018). Habitually, a segmentation procedure is followed by a visualization, detection, recognition and quantification analysis. Moreover, over the last decade, thresholding has been extensively employed in the automation process for medical image analysis (Balafar, et al., 2010; Chae, et al., 2016; Huang, et al., 2017; Mishra, et al., 2015; Wahab, et al., 2017) as a mean to help physicians in the diagnostic procedure. Although there are many works for automatic and semi-automatic segmentation, the interpretation of image details and analysis is still a problem to solve nowadays, due to complicated structures with similar characteristics, noise conditions, low contrast, and unclear boundaries, which are typical circumstances in medical images (Hall et al., 1992).

Segmentation of medical digital images has been stated as an important task for several medical applications, which includes surgical and post-surgical assessment, pathology and abnormality recognition, diagnostic, and treatment planning (Huang et al., 2017; Shehab et al., 2018; Zhang, et al., 2007). The clinical applicability of thermography screening is still limited; nevertheless, considering the needs for helping in the diagnostic procedure, it is critical improving the application of such technique and the automatic detection in order to increase the reliability and high acceptance in the clinical practice (Essa, et al., 2017; Wahab et al., 2017).

In (Scales, Herry, & Frize, 2004) has been proposed a semi-automatic segmentation method for thermal images consistent of eight steps. However, here the authors have reported that only 4 of 21 images presented a pleasing detection of the region of interest, as well as most of the errors, were caused by the inframammary fold and problems with the edge detection process. Schaefer (Schaefer & Nakashima, 2007), proposed a semi-automatic segmentation method for thermography screenings, the authors here used a fuzzy rule-based system for classifying the segmented regions, yet they employed a beforehand step to segment manually the images by a human expert.

Recently, in (Motta, et al., 2010) has been presented an automatic segmentation method for thermal breast images, which combines automatic thresholding and border detection techniques. Despite this method presents promising results, the region of interest detected may not include some portions of the upper breasts. Hence, the development of an automatic technique is still open research, which should hold properties like low computational cost and robustness to support the process of diagnosing of breast cancer using thermograms. Thus, reliable thresholding of breast thermograms could act as a fast indicator of being used in further clinical diagnose assessment. In this paper is proposed a robust automatic multi-thresholding methodology for breast thermogram analysis by means of a swarm algorithm. The main goal of this work is to develop a new technique, which may assist the clinicians in the diagnostical process of breast cancer.

In the proposed approach, the multi-thresholding task is performed using a novelty swarm technique, named Dragonfly Algorithm (DA), which it was introduced in 2015 for Seyedali Mirjalili,

(2016). In the designed methodology were employed two typical segmentation techniques, the Otsu's method and the Kapur's entropy, such techniques were used as objective functions on the DA. Since in the related literature, the segmentation is performed over the histogram, there is no contextual information about the pixels. To overcome this problem, the method presented in this paper is based on the energy curve instead of the histogram.

The energy curve possesses similar features as the histogram (Oliva, et al., 2018), and it is computed using the energy function (EF) that was first introduced in (Ghosh, et al., 2007; Patra, et al., 2014). The EF computes the energy of each intensity level considering the position and vicinity of each pixel.

For image segmentation the typically used techniques are non-parametrical, the usage of a swarm algorithm to compute the multi-thresholding provides a lower computational cost and adds robustness to the method. With the purpose to get an objective performance evaluation of the proposed approach, it was compared with two classic metaheuristic algorithms, the Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995b) and the Genetic Algorithms (GA) (De Jong, 1988), as well as with two recently proposed metaheuristics, the Krill Herd algorithm (KH) (Gandomi & Alavi, 2012), and the Runner-Root Algorithm (RRA) (Merrikh-Bayat, 2015). Furthermore, as a part of the evaluation of the quality of the segmented images are presented a qualitative analysis realized by a human expert, as well as a quantitative valuation conducted using different metrics, which correspond to the Standard Deviation (STD), the Peak-Signal-to-Noise Ratio (PSNR), the Structure Similarity Index (SSIM) and the Feature Similarity Index (FSIM).

The remainder of this document is organized as follows. Section 12.2 presents the overall description of the swarm technique employed, correspondingly to the Dragonfly Algorithm. In section 12.3 are described in detail the segmentation approaches employed for this proposal.

On the other hand, in section 12.4 is explained the proposed methodology used for the multi-level thresholding using the Dragonfly Algorithm. Subsequently, in section 12.5 are exhibited the results obtained by applying the proposed approach and a fair comparison between the different methods, by describing the qualitative and quantitative evaluations. In section 12.6 is presented a brief discussion from the experimental results since both points of view for the valuation. Finally, section 12.7 draws some conclusions from this work.

12.2. Dragonfly Algorithm

The Dragonfly Algorithm (DA) was proposed in 2015 for Seyedali Mirjalili, (2016) and is based on the two different types of swarming behaviors of Dragonflies in nature. The DA algorithm balances the phases of exploration and exploitation by imitating the natural swarm interaction of dragonflies for navigating, food search and enemy avoidance. Such behavior is called dynamic or static swarming. The dynamic swarming refers to the moving phase, and the static swarming denotes the hunting phase. In the static swarming, the minimum possible number of Dragonflies form a small group and fly in all directions, meanwhile in the dynamic swarming, a significant number of Dragonflies is required to conform a big set, and afterwards they only fly in one direction. Correspondingly, in the DA there are two basic phases, exploration and exploitation, in the same way, that any other meta-heuristic algorithm.

Considering this, the exploration stage of the DA corresponds to the static swarm behavior. Consequently, the exploitation state refers to the dynamic compartment. In the exploration state, Dragonflies generate a sub-set to move (fly) over diverse areas to achieve the goal of exploring the search space, while in the exploitation phase, Dragonflies move in bigger clusters along only one direction, and with this, they reach the primary objective for exploitation.

All swarms present a behavior following the principles given by Reynolds, (1987):

Separation. This process states the static collision avoidance between individuals in the same neighborhood.

Alignment. It refers to the velocity of matching between individuals in the same neighborhood.

Cohesion. This procedure indicates the tendency of the individuals in the direction of the center of the neighborhood.

Such principles are also recreated in the Dragonfly Algorithm. Considering that the principal goal of a swarm is surviving, the entire individuals of the swarm are attracted to the food sources and in the same way, diverted away from the enemies by using the beforehand mentioned principles.

The swarm will have five different type of actions to update the position of the new individuals; these behaviors are mathematically modeled in the following manner:

Separation:

$$S_i = -\sum_{j=1}^N X - x_j \quad (12.1)$$

Where X represents the position of the current individual, x_j exhibits the j -th position neighboring individual and, N corresponds to the number of neighboring individuals.

Alignment:

$$A_i = \frac{\sum_{j=1}^N V_j}{N} \quad (12.2)$$

Where x_j is the velocity of the j -th neighboring individual.

Cohesion:

$$C_i = \frac{\sum_{j=1}^N X_j}{N} - X \quad (12.3)$$

Where X represents the position of the current individual, N corresponds to the number of neighborhoods and, x_j is the j -th position neighboring individual.

Attraction to a food source:

$$F_i = X^+ - X \quad (12.4)$$

Where X corresponds to the position of the current individual and, X^+ is the position of the food source.

Deflection from enemies:

$$E_i = X^- + X \quad (12.5)$$

Where X shows the current individual position and, X^- corresponds to the enemy's position. Therefore, the behavior of the dragonflies in the swarm is supposed to be represented by the combination of the beforehand mentioned actions.

Once the positions of the dragonflies, enemies and food source are updated, is necessary to update the radius of dragonflies' neighbors. Considering, that a dragonfly possesses one, as a minimum number of neighbors, the velocity vector is calculated as follows:

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i) + w\Delta X_t \quad (12.6)$$

Where s represents the separation weight, S_i corresponds to the separation of the individual i -th, a is the alignment weight, A_i refers to the alignment of the individual i -th, c is the cohesion weight, C_i symbolizes the cohesion of the individual i -th, f is the food factor, F_i embodies the food source of the individual i -th, e refers to the enemy factor, E_i corresponds to the position of the enemy of the individual i -th, w states the inertia weight and, t shows the iteration counter.

Once the velocity vector is calculated, the position vector is estimated by:

$$X_{t+1} = X_t + \Delta X_{t+1} \quad (12.7)$$

Where t refers to the current iteration and therefore $t + 1$ is the next iteration. By taking into consideration the parameters s , a , c , f and e (separation, alignment, cohesion, food and enemy factors), it is possible to achieve diverse types of exploration and exploitation behaviors.

12.3. Segmentation Approaches

The problem of thresholding is solved using the histogram of the image as input; therefore, the information regarding the image is provided by it. Nevertheless, the frequency of occurrence of each pixel does not provide much information. In this sense, building a curve similar to a histogram that describes the energy of an image provides another way to perform thresholding by incorporating contextual information. In other words, this new representation is calculated based on the characteristics of the energy curve. Considering this, the gray values of the pixel in a given range represent an object on the image. (Let the gray values of the pixel in the range represent an object in the image.) Providing a new curve as input with smooth and transparent behavior facilitates the discrimination of different objects in the image as compared to the histogram. Thus, the energy curve becomes more useful to detect appropriate threshold values. Thresholding methods can be directly applied to the energy curve since it has similar features to the histogram. The following three subsections briefly discuss the formulation of the energy curve and two representative approaches for image thresholding.

12.3.1. Energy Curve

The energy of the image I at gray intensity value l ($0 \leq l \leq L$) is calculated by generating a two-dimensional matrix for every intensity value as $B_l = \{b_{i,j}, 1 \leq i \leq m, 1 \leq j \leq n\}$ where $b_{i,j} = 1$ if the intensity at the current position is greater than l the intensity value ($I_{i,j} > l$), or else $b_{i,j} = -1$.

To create the energy curve, we define a digital image I for this process as a matrix $I = \{l_{ij}, 1 \leq i \leq m, 1 \leq j \leq n\}$ of size $m \times n$ where l_{ij} denotes the gray level of the image I at the pixel (i, j) . The maximum value of the gray intensity of the image I is denoted as L .

Considering the described in (Patra et al., 2014), the contextual spatial correlation between surrounding pixels is calculated; for this purpose, a neighborhood N of order d at given position (i, j) is used $N_{ij}^d = \{(i+u, j+v), (u, v) \in N^d\}$. The value of d determines the configuration that the neighborhood system takes (Ghosh et al., 2007). This paper considers the second-order neighborhood system N^2 . The system can be defined in spatial terms as $(u, v) \in \{(\pm 1, 0), (0, \pm 1), (1, \pm 1), (-1, \pm 1)\}$ and is shown in Figure 12.1.

$(i-1, j-1)$	$(i-1, j)$	$(i-1, j+1)$
$(i, j-1)$	(i, j)	$(i, j+1)$
$(i+1, j-1)$	$(i+1, j)$	$(i+1, j+1)$

Figure. 12.1. Spatial representation of the neighborhood system N^2 .

Let $C = \{c_{ij}, 1 \leq i \leq m, 1 \leq j \leq n\}$ be a constant matrix where $c_{ij} = 1, \forall (i, j)$. The energy value E_l of the image I at the gray intensity value l is computed as:

$$E_l = -\sum_{i=1}^m \sum_{j=1}^n \sum_{pq \in N_{ij}^2} b_{ij} \cdot b_{pq} + \sum_{i=1}^m \sum_{j=1}^n \sum_{pq \in N_{ij}^2} c_{ij} \cdot c_{pq} \tag{12.8}$$

The right side of the Equation (12.8) is a constant term devoted to assuring a positive energy value $E_l \geq 0$. A quick look at Eq. 12.8 shows that for a given image I at the intensity value l will be zero if all the elements of the binary image B_l are either 1 or -1. This approach determinates the energy associated to every intensity value of the image to generate a curve considering spatial contextual information of the image.

12.3.2. Otsu's between class variance

The thresholding methodology uses strategies to select a threshold value to partition a histogram into two categories, a popular technique was proposed by Otsu (1979). On the use of images, it is often difficult to detect the valleys and bottoms precisely, especially in such cases where the valley is flat and broad with noise. In this case, the information concerning neighboring pixels in the original picture can modify the histogram to make it more useful for thresholding.

For the multi-level approach, nt thresholds are necessary to divide the original image into $nt+1$ classes. Thus, the set of thresholds used for segmentation for a given image is encoded as $\mathbf{th} = [th_1, th_2, \dots, th_m]$. In this sense, the energy value E_i of each pixel of a digital image according to the frequency of its occurrence generates a probability $PE_i = E_i/NP$ where $\sum_{i=1}^{NP} PE_i = 1$ and

NP is the total number of pixels. According to the placement of every threshold value, each of the generated classes is used to compute the variance σ^2 and their means μ_k as:

$$\sigma^2 = \sum_{k=1}^{nt} \sigma_k = \sum_{k=1}^{nt} \omega_k (\mu_k - \mu_T)^2 \quad (12.9)$$

$$\mu_k = \sum_{i=th_k}^{th_{k+1}-1} \frac{kPE_i}{\omega_i(th_i)} \quad (12.10)$$

where

$$\omega_k = \sum_{i=th_k}^{th_{k+1}-1} PE_i \quad (12.11)$$

Finally, the Otsu's method maximizes the variance for the given set of threshold values:

$$f_{Otsu}(\mathbf{th}) = \max(\sigma^2(\mathbf{th})), \quad 0 \leq th_i \leq L-1, \quad i = 1, 2, \dots, nt \quad (12.12)$$

12.3.3. Kapur's entropy

The technique of Kapur tries to segment a histogram based on the probability distribution of the image's histogram using entropy as a measure (Kapur, et al., 1985). The set of thresholds at which the function returns a maximum value is considered as the optimal set of threshold values. Similar to Otsu's method, it can be applied directly to the energy curve. Kapur's method searches for the optimal set of thresholds \mathbf{th} that maximizes the overall entropy.

$$f_{Kapur}(\mathbf{th}) = \max\left(\sum_{k=1}^{nt} H_k\right) \quad (12.13)$$

where the entropy of each class is calculated as in Eq. 12.14.

$$H_k = \sum_{i=th_k}^{th_{k+1}-1} \frac{PE_i}{\omega_k} \ln\left(\frac{PE_i}{\omega_k}\right) \quad (12.14)$$

The probability distribution PE_i and ω_k are computed using the same criteria as Otsu's method.

12.4. Dragonfly algorithm for thermal image thresholding

This section explains the process for implement the Dragonfly Algorithm (DA) to segment thermal images from mammography using the energy curve. The Energy Curve defines the search space to consider the problem of multi-level thresholding from an optimization point of view. The implementation of the DA can be divided into two approaches, the first one uses the Otsu's method and the second one employs the Kapur's entropy as the objective function.

In this optimization process, the thermal image (I) represents the input, and the first step is to compute the Energy Curve, after that the DA initialize a random population of candidate solutions (second step). Considering that the image has 255 intensity levels (L), the search space is then defined between the bounds $[0, 255]$. The construction of the population and the candidate solutions are defined in Eq. (12.15).

$$\mathbf{X} = [\mathbf{th}_1, \mathbf{th}_2, \dots, \mathbf{th}_N], \quad \mathbf{th}_i = [th_{i1}, th_{i2}, \dots, th_{in}]^T, \quad (12.15)$$

subject to $th_{i1} < th_{i2} < \dots < th_{in} < L$

From Eq. (12.15), $\mathbf{th}_i \subseteq \mathbf{X}$ and it is a vector that contains the set of thresholds (th_j) that should segment the image, and T refers to the transpose operator. The amount of thresholds in \mathbf{th}_i is defined by the dimensions of the problem (nt).

12.4.1. DA Implementation

The proposed segmentation algorithm for thermal images has been implemented considering the Otsu's and Kapur's methods as the objective function. In this sense, there is a binary selector (SM) that permits to change the objective function. The DA implementation can be summarized into the following steps:

Step 1	Read the thermal image I and store it as the grayscale image I_{Gr} .
Step 2	Obtain the energy curve
Step 3	Initialize the DA control parameters and step vectors
Step 4	Initialize a population \mathbf{X} of N random particles with nt dimensions
Step 5	if $SM=1$ Evaluate \mathbf{X} in the Otsu objective function Eq. (12.12) else Evaluate \mathbf{X} in the Kapur objective function Eq. (12.13) end if
Step 6	Update the food source and enemy
Step 7	Update the parameters $w, s, a, c, f,$ and e
Step 8	Calculate $S, A, C, F,$ and E using Equations (12.1) to (12.5)
Step 9	Update the neighborhood radius
Step 10	if a dragonfly has one or more dragonflies in its vicinity Update the velocity vector using Eq. (12.6) Update the position vector using Eq. (12.7) else Update the position vector using Eq. (12.7) end
Step 11	Verify if the new positions exist in the feasible search space
Step 12	The iteration counter is increased in 1, and if the stop criteria is satisfied then go to step 13. Otherwise, jump to step 5
Step 13	Select the best thresholds and apply them to the thermal image in grayscale

12.4.2. Performance Evaluation

From an optimization point of view, the quality of the solutions is evaluated on the objective function. However, for the problem of multi-level segmentation, it is necessary to verify the accuracy of the pixel's classification.

For experimental purposes in this paper are used the following metrics that evaluate the quality of the segmented images are the Standard Deviation (STD), the Peak-Signal-to-Noise Ratio (PSNR),

the Structure Similarity Index (SSIM) and the Feature Similarity Index (FSIM). The selected metrics are widely used in the related literature to verify different aspects that exist between the input and output images (Cuevas, Osuna, et al., 2017; Oliva et al., 2017).

For example, the STD is used to verify the stability of the results obtained by the optimization (Ghamisi, et al., 2012), and it is computed as follows:

$$STD = \sqrt{\frac{\sum_{i=1}^{Iter_{max}} (\sigma_i - \mu)}{Ru}} \quad (12.16)$$

In the same way, the PSNR is used to verify the similarity that exist between the original and the segmented image. To compute the PSNR it is necessary to use the Root Mean Square Error (RMSE) pixel to pixel (Agrawal, et al., 2013; Akay, 2013; Horng & Liou, 2011; Oh, et al., 2004), the PSNR is the defined as:

$$PSNR = 20 \log_{10} \left(\frac{255}{RMSE} \right), (\text{dB}) \quad (12.17)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{ro} \sum_{j=1}^{co} (I_{in}(i, j) - I_{seg}(i, j))^2}{ro \times co}}$$

In Eq. (12.17), I_{in} is the original image, I_{seg} is the segmented image. Meanwhile, ro and co are the maximum numbers of rows and columns of the image. A comparison of the structures contained in the segmented image is performed using the SSIM (Wang, et al., 2004), and it is defined in Eq. (12.18). A higher SSIM value represents a better segmentation of the original image.

$$SSIM(I_{in}, I_{seg}) = \frac{(2\mu_{I_{in}}\mu_{I_{seg}} + C1)(2\sigma_{I_{in}I_{seg}} + C2)}{(\mu_{I_{in}}^2 + \mu_{I_{seg}}^2 + C1)(\sigma_{I_{in}}^2 + \sigma_{I_{seg}}^2 + C2)} \quad (12.18)$$

$$\sigma_{I_{in}I_{seg}} = \frac{1}{N-1} \sum_{i=1}^N (I_{in_i} + \mu_{I_{in}})(I_{seg_i} + \mu_{I_{seg}})$$

From Eq. (12.18) $\mu_{I_{in}}$ is the mean of the input (original) image and $\mu_{I_{seg}}$ is the mean of the segmented image. In the same way, for each image, the values of $\sigma_{I_{in}}$ and $\sigma_{I_{seg}}$ correspond to the standard deviation. $C1$ and $C2$ are constants used to avoid the instability when $\mu_{I_{in}}^2 + \mu_{I_{seg}}^2 \approx 0$. The values of $C1$ and $C2$ are set to 0.065 considering the experiments of (Agrawal et al., 2013).

In the same context, the FSIM (Zhang, et al., 2011), helps to verify the similarity between two images. In this paper, the FSIM employs the original grayscale image and the segmented image. As PSNR and SSIM the higher value is interpreted as better performance of the thresholding method. The FSIM is then defined as:

$$FSIM = \frac{\sum_{w \in \Omega} S_L(w) PC_m(w)}{\sum_{w \in \Omega} PC_m(w)} \quad (12.19)$$

In Eq. (12.19) the entire domain of the image is defined by Ω , and their values are computed by Eq. (12.20).

$$\begin{aligned}
 S_L(w) &= S_{PC}(w)S_G(w) \\
 S_{PC}(w) &= \frac{2PC_1(w)PC_2(w)+T_1}{PC_1^2(w)+PC_2^2(w)+T_1} \\
 S_G(w) &= \frac{2G_1(w)G_2(w)+T_2}{G_1^2(w)+G_2^2(w)+T_2}
 \end{aligned} \tag{12.20}$$

G is the gradient magnitude (GM) of a digital image and is defined, and the value of PC that is the phase congruence is defined as follows:

$$\begin{aligned}
 G &= \sqrt{G_x^2 + G_y^2} \\
 PC(w) &= \frac{E(w)}{\left(\varepsilon + \sum_n A_n(w) \right)}
 \end{aligned} \tag{12.21}$$

Where $A_n(w)$ is the local amplitude on scale n and $E(w)$ is the magnitude of the response vector in w on n . ε is a small positive number and $PC_m(w) = \max(PC_1(w), PC_2(w))$.

12.5. Experimental Results

This chapter introduces the use of the DA for thermal image segmentation. Specifically, this methodology is evaluated in the case of Breast Thermography. For this purpose, a set of 8 images retrieved from the Database for Research Mastology with Infrared Image (Silva, et al., 2014) were randomly selected from the entire database to visually analyze the performance of the proposed approach. The selected images with the corresponding histograms and the energy curves are presented in Fig. 12.2.

The images were captured considering various protocols with a FLIR SC-620 camera; images are containing artificial artifacts such as labels, bars or logos were cropped to focus the segmentation of the body. Since some images are stored as RGB images rather than encoding the intensity of the thermal radiation directly, RGB images are firstly converted to gray-scale (intensity) images to perform the thresholding process. Considering this fact and the random selection of the images, some of them are presented in grayscale and others in the RGB format.

The performance of the proposed methodology is compared with other metaheuristic approaches such as GA, PSO, KH, and RRA considering the control parameters recommended by their respective authors. Since metaheuristic algorithms involve the use of random variables, it is necessary to perform a statistical analysis of the results. Each experiment is composed of 35 independent runs of the same algorithm over a specific image, and the average and deviation are reported.

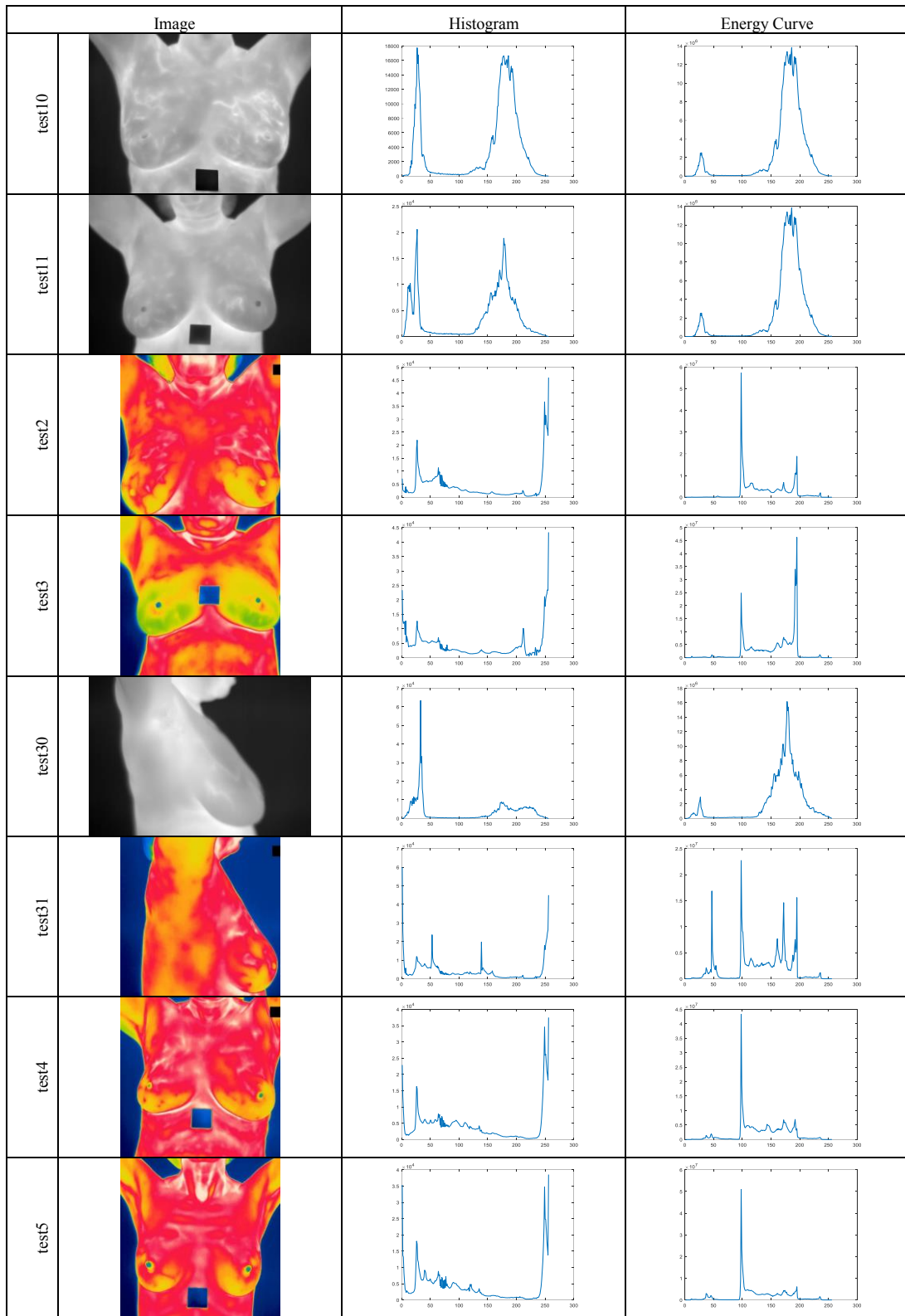


Figure 12.2. Selected images and their corresponding histogram and energy curve

Since Breast Thermography images are taken on a controlled environment, it is possible to segment them using only 2, 3, 4, and 5 thresholds. Each run of every algorithm is stopped after 500 iterations containing 50 individuals each to provide a fair comparison. All experiments were evaluated on the environment of MATLAB 8.3 on an Intel I7 700 @ 3.5 GHz with 8GB of RAM.

12.5.1. Otsu's results

The proposed methodology based on the DA for the segmentation of Breast Thermography images is compared to other metaheuristic methodologies considering the Otsu's method to identify the best possible thresholds for a given image. Table 12.1 presents the best thresholds found by each method. It can be noticed that many approaches find the same threshold values, especially during segmentation with a small number of thresholds.

Image	t	DA	GA	KH	PSO	RRA
test10	2	102,184	102,184	110,223	102,184	102,184
test10	3	92,165,192	92,165,192	102,184,186	92,165,192	92,165,192
test10	4	89,160,183,203	88,160,183,203	95,167,193,194	89,160,183,203	89,160,183,203
test10	5	79,145,171,187,206	78,143,170,186,205	78,154,179,200,236	77,142,172,187,205	78,154,172,185,236
test11	2	100,182	100,182	108,154	100,182	100,182
test11	3	91,164,194	91,164,194	100,182,249	91,164,193	91,164,194
test11	4	89,160,184,209	90,161,184,209	92,164,193,241	90,160,183,208	89,160,184,209
test11	5	59,122,163,186,210	62,123,163,185,210	91,162,188,218,223	64,122,163,186,208	64,123,162,185,210
test2	2	125,169	125,169	146,174	125,169	125,169
test2	3	115,147,181	115,147,181	125,169,211	115,147,181	115,147,181
test2	4	113,144,176,206	113,144,176,206	113,147,183,234	113,144,175,207	113,144,176,206
test2	5	108,129,154,179,207	107,130,155,179,208	114,143,176,213,233	106,129,154,177,205	105,127,150,177,215
test3	2	124,168	124,168	145,177	124,168	124,168
test3	3	77,133,173	77,133,173	123,168,215	78,133,173	77,133,173
test3	4	74,119,151,180	73,119,151,180	81,132,172,178	73,119,150,180	74,119,151,180
test3	5	73,114,141,165,184	69,114,140,164,183	72,119,147,178,181	72,113,138,162,183	70,114,138,164,183
test30	2	104,197	104,197	116,208	104,197	104,197
test30	3	100,181,210	100,181,210	104,197,209	100,181,210	100,181,210
test30	4	91,164,192,217	90,163,192,217	105,181,211,251	91,163,192,217	91,164,192,217
test30	5	82,151,179,200,221	81,150,178,199,220	79,160,188,212,241	82,152,179,199,220	82,150,179,200,224
test31	2	80,145	80,145	128,207	80,145	80,145
test31	3	77,134,176	77,134,176	80,145,158	77,134,176	77,134,176
test31	4	75,120,151,181	75,120,151,181	76,135,176,220	72,120,151,180	75,120,151,181
test31	5	75,120,151,178,207	77,120,151,178,207	75,120,151,183,215	77,120,150,179,209	
test4	2	79,142	79,142	137,252	79,142	79,142
test4	3	76,125,165	76,125,165	79,142,180	76,125,165	76,125,165
test4	4	75,120,153,183	74,120,153,183	73,126,166,206	74,120,153,182	75,120,153,183
test4	5	73,110,132,158,184	70,110,132,159,185	78,121,155,183,246	74,111,132,157,182	70,113,132,159,187
test5	2	74,144	74,144	137,151	74,144	74,144
test5	3	72,125,167	72,125,167	74,144,168	72,125,167	72,125,167
test5	4	71,118,151,185	71,118,151,185	71,126,168,248	69,118,152,185	71,118,151,185
test5	5	70,112,138,167,202	73,112,138,167,202	78,116,148,183,191	65,111,138,167,201	78,112,148,181,191

Table 12.1. Thresholds obtained by DA, GA, KH, PSO, and RRA using Otsu's method as the objective function.

The objective of image thresholding is to generate high-quality images with a given number of thresholds. As described in the previous subsection, the PSNR is a quality metric often used to an-

alyze the quality of a processed signal concerning the original. However, PSNR has been extended to analyze multi-dimensional signals, images in this case. On Table 12.2, a higher mean value of PSNR indicates better segmentation of the image considering the thresholds values of a given algorithm. Contrary to the mean value, a smaller value of STD is desired as it reflects less variation between the results generated by each approach. Typically, the STD value increases together with the number of thresholds, as the problem becomes more complex.

Image	t	DA	GA	KH	PSO	RRA
test10	2	14.6687	1.26E-14	14.6687	1.26E-14	13.5481 2.20E+00 13.1867 1.46E-01 14.6687 1.26E-14
test10	3	19.0755	1.06E-02	19.0085	1.18E-01	14.6051 3.63E-01 16.9484 4.67E-01 19.0798 1.49E-02
test10	4	20.1734	5.59E-03	20.1220	2.57E-01	18.0927 1.54E+00 18.3513 4.80E-01 20.1523 1.42E-01
test10	5	22.1036	2.94E-02	22.0340	1.14E-01	20.2068 1.12E+00 19.6899 4.03E-01 12.2507 0.00E+00
test11	2	14.5316	0.00E+00	14.5316	0.00E+00	14.4225 2.09E+00 13.0335 1.23E-01 14.5316 0.00E+00
test11	3	17.7455	3.60E-15	17.7293	8.19E-02	14.6255 4.56E-01 15.9579 2.22E-01 17.7293 3.60E-15
test11	4	18.5435	4.40E-04	18.5104	1.77E-01	17.6970 8.76E-01 17.4437 1.27E+00 18.4966 5.32E-02
test11	5	21.5254	5.48E-01	21.4719	3.01E-01	20.1041 1.20E+00 19.1363 8.27E-01 11.6602 1.08E-14
test2	2	10.1335	5.41E-15	10.1335	5.41E-15	8.7132 4.62E-02 9.0780 7.81E-02 10.1335 5.41E-15
test2	3	11.2974	7.21E-15	11.3151	1.05E-01	10.0853 1.84E-01 10.1557 3.03E-01 11.3328 6.59E-02
test2	4	12.1731	2.75E-02	11.6385	1.69E-01	11.4805 6.32E-01 11.6444 3.70E+00 11.6429 9.26E-02
test2	5	17.5442	4.04E+00	14.8925	4.23E+00	14.1203 3.97E+00 16.8609 4.27E+00 9.2574 1.80E-15
test3	2	12.1067	7.21E-15	12.1114	1.92E-02	10.2866 1.42E-01 10.9184 6.83E-02 12.1067 7.21E-15
test3	3	19.0459	1.08E-14	19.0373	1.59E-01	12.1821 2.96E-01 17.0906 5.26E-01 19.0507 1.63E-02
test3	4	20.5130	2.55E-01	20.4970	2.74E-01	19.1492 7.62E-01 18.3894 6.19E-01 20.5177 2.27E-01
test3	5	21.2057	4.08E-01	21.1571	5.01E-01	20.1831 1.00E+00 19.1823 7.00E-01 12.1362 7.21E-15
test30	2	15.1364	5.41E-15	15.1364	5.41E-15	14.4597 1.65E+00 13.5849 1.05E-01 15.1364 5.41E-15
test30	3	16.7166	0.00E+00	16.6983	9.11E-02	15.2292 5.09E-01 15.0489 2.96E-01 16.7107 2.99E-02
test30	4	19.3592	1.08E-14	19.1880	1.95E-01	17.0289 1.32E+00 17.4175 3.94E-01 19.2058 6.48E-02
test30	5	20.3050	5.46E-02	20.2358	1.42E-01	19.3267 6.15E-01 18.2120 1.69E-01 11.3916 5.41E-15
test31	2	16.6405	3.60E-15	16.6405	3.60E-15	11.5916 1.50E-01 14.9381 9.79E-02 16.6405 3.60E-15
test31	3	17.4062	3.60E-15	17.3939	1.07E-01	16.8352 2.38E-01 15.6016 3.17E-01 17.4082 1.20E-02
test31	4	18.3440	3.94E-02	18.2651	1.65E-01	17.3622 4.83E-01 16.4892 4.43E-01 18.3221 1.07E-01
test31	5	18.4365	2.17E-02	18.3613	1.83E-01	18.2791 5.66E-01 16.8223 4.08E-01 11.6463 3.60E-15
test4	2	17.1131	0.00E+00	17.1155	9.95E-03	9.8767 7.05E-02 15.1451 1.33E+00 17.1131 0.00E+00
test4	3	18.7283	3.60E-15	18.7028	1.20E-01	17.4631 3.61E-01 16.9111 4.08E-01 18.7283 3.60E-15
test4	4	19.3847	3.60E-15	19.2365	2.26E-01	18.8003 8.73E-01 17.4880 8.35E-01 19.3554 1.21E-01
test4	5	20.2407	2.95E-01	20.1937	2.46E-01	19.4868 1.05E+00 18.1292 7.31E-01 9.7554 0.00E+00
test5	2	16.7890	3.60E-15	16.7890	3.60E-15	9.8326 5.70E-02 15.1808 4.03E-01 16.7890 3.60E-15
test5	3	18.2873	7.21E-15	18.2185	1.73E-01	17.1246 2.91E-01 16.3585 5.33E-01 18.2873 7.21E-15
test5	4	18.9319	1.21E-01	18.9077	3.56E-01	18.2989 9.84E-01 17.1750 7.89E-01 18.8769 2.58E-01
test5	5	19.5870	3.11E-02	19.4124	5.47E-01	19.3108 1.22E+00 17.6162 1.05E+00 11.0702 0.00E+00

Table 12.2. Mean and STD values of the PSNR metric using Otsu by the DA, GA, KH, PSO, and RRA.

The Feature Similarity Index (FSIM) is evaluated for all approaches and reported in Table 12.3. This metric is evaluated to determinate how well the features of an image are preserved after its processing. The features can play an important role in classification systems for Breast Thermographic images. The DA provides better results than its counterparts regarding FSIM on most test scenarios.

Image	t	DA	GA	KH	PSO	RRA					
test10	2	0.7394	5.63E-16	0.7394	5.63E-16	0.7842	1.77E-02	0.6653	6.60E-04	0.7394	5.63E-16
test10	3	0.7832	3.64E-06	0.7820	2.09E-03	0.7448	6.01E-03	0.7009	5.76E-03	0.7833	3.61E-04
test10	4	0.8155	9.65E-05	0.8145	8.68E-04	0.7774	1.38E-02	0.7318	2.87E-03	0.8155	2.36E-04
test10	5	0.8425	2.44E-04	0.8420	1.70E-03	0.8129	7.38E-03	0.7531	6.31E-03	0.8196	3.38E-16
test11	2	0.7906	4.51E-16	0.7906	4.51E-16	0.8138	9.73E-03	0.7115	1.61E-03	0.7906	4.51E-16
test11	3	0.8167	4.51E-16	0.8165	2.69E-04	0.7912	4.13E-03	0.7345	1.98E-03	0.8165	4.51E-16
test11	4	0.8248	3.82E-05	0.8238	8.49E-04	0.8133	6.25E-03	0.7450	7.70E-03	0.8248	2.74E-04
test11	5	0.8541	7.30E-03	0.8521	1.23E-03	0.8314	1.01E-02	0.7625	6.67E-03	0.8188	0.00E+00
test2	2	0.6897	1.13E-16	0.6897	1.13E-16	0.6539	7.50E-03	0.6205	1.06E-03	0.6897	1.13E-16
test2	3	0.7287	5.63E-16	0.7284	5.00E-04	0.6950	7.86E-03	0.6556	1.78E-03	0.7285	3.19E-04
test2	4	0.7395	5.50E-05	0.7394	6.23E-04	0.7263	6.78E-03	0.6795	2.90E-02	0.7395	1.83E-04
test2	5	0.8030	2.86E-02	0.7850	3.00E-02	0.7586	3.48E-02	0.7273	3.32E-02	0.6546	3.38E-16
test3	2	0.7235	5.63E-16	0.7234	1.52E-04	0.7076	6.63E-03	0.6509	1.32E-03	0.7235	5.63E-16
test3	3	0.7715	4.51E-16	0.7716	1.11E-03	0.7244	8.78E-03	0.6944	4.77E-03	0.7718	1.10E-03
test3	4	0.8232	9.00E-03	0.8243	2.28E-03	0.7764	1.00E-02	0.7397	6.05E-03	0.8245	2.95E-03
test3	5	0.8449	1.09E-02	0.8456	1.06E-02	0.8171	1.15E-02	0.7633	1.18E-02	0.8364	2.25E-16
test30	2	0.8386	2.25E-16	0.8386	2.25E-16	0.8358	3.22E-03	0.7547	2.16E-04	0.8386	2.25E-16
test30	3	0.8386	3.38E-16	0.8385	2.07E-04	0.8398	1.69E-03	0.7548	7.96E-04	0.8385	9.16E-05
test30	4	0.8543	4.51E-16	0.8531	1.78E-03	0.8436	5.51E-03	0.7713	5.17E-03	0.8532	2.26E-04
test30	5	0.8703	8.71E-04	0.8693	2.56E-03	0.8587	6.56E-03	0.7826	2.51E-03	0.7437	1.13E-16
test31	2	0.7773	5.63E-16	0.7773	5.63E-16	0.7506	8.09E-03	0.6995	2.34E-04	0.7773	5.63E-16
test31	3	0.7910	3.38E-16	0.7907	4.76E-04	0.7815	3.92E-03	0.7119	1.41E-03	0.7907	5.31E-05
test31	4	0.8324	4.45E-04	0.8312	2.18E-03	0.7939	4.03E-03	0.7476	5.64E-03	0.8320	1.90E-03
test31	5	0.8390	2.39E-04	0.8373	2.21E-03	0.8254	1.21E-02	0.7592	1.25E-02	0.7610	3.38E-16
test4	2	0.7136	3.38E-16	0.7135	3.52E-04	0.6772	1.19E-02	0.6414	3.17E-03	0.7136	3.38E-16
test4	3	0.7791	3.38E-16	0.7783	1.55E-03	0.7246	1.01E-02	0.7018	3.91E-03	0.7783	3.38E-16
test4	4	0.8161	5.63E-16	0.8150	1.89E-03	0.7876	8.63E-03	0.7314	7.55E-03	0.8158	1.86E-03
test4	5	0.8521	1.29E-02	0.8489	9.13E-03	0.8121	1.31E-02	0.7599	1.72E-02	0.6825	3.38E-16
test5	2	0.7352	5.63E-16	0.7352	5.63E-16	0.6912	7.95E-03	0.6621	1.26E-03	0.7352	5.63E-16
test5	3	0.7838	3.38E-16	0.7837	1.19E-03	0.7444	6.64E-03	0.7047	3.93E-03	0.7838	3.38E-16
test5	4	0.8165	2.77E-03	0.8157	2.55E-03	0.7878	7.86E-03	0.7342	5.53E-03	0.8161	2.18E-03
test5	5	0.8392	3.64E-04	0.8373	5.92E-03	0.8166	9.47E-03	0.7517	1.14E-02	0.7727	4.51E-16

Table 12.3. Mean and STD value of FSIM metric using Otsu by the DA, GA, KH, PSO, and RRA.

In Table 12.4, the results of the SSIM metric are presented. In the context of Breast thermographic images, a high structural similarity index indicates that the visible structures of the original image are likely to be passed on to the segmented image. This index is the most important for this paper since the objective of this contribution is to enhance the images visually for a better diagnosis. Similarly, to the previous indices, DA performs better than the other approaches.

Image	t	DA	GA	KH	PSO	RRA					
test10	2	0.6625	3.38E-16	0.6625	3.38E-16	0.7210	3.39E-02	0.6621	3.33E-03	0.6625	3.38E-16
test10	3	0.7502	1.18E-04	0.7482	3.22E-03	0.6597	8.39E-03	0.7430	1.10E-02	0.7503	4.82E-04
test10	4	0.7776	1.13E-04	0.7756	2.33E-03	0.7274	3.30E-02	0.7776	5.01E-03	0.7756	1.56E-03
test10	5	0.8057	5.38E-04	0.8041	2.78E-03	0.7732	1.75E-02	0.8022	5.43E-03	0.7207	2.25E-16
test11	2	0.6845	5.63E-16	0.6845	5.63E-16	0.7302	2.59E-02	0.6838	3.42E-03	0.6845	5.63E-16
test11	3	0.7450	5.63E-16	0.7448	1.00E-03	0.6832	6.66E-03	0.7442	3.59E-03	0.7448	5.63E-16
test11	4	0.7664	1.95E-05	0.7553	1.65E-03	0.7405	1.43E-02	0.7553	1.74E-02	0.7551	7.33E-04
test11	5	0.7984	8.34E-03	0.7969	2.53E-03	0.7757	1.77E-02	0.7936	8.54E-03	0.6888	2.25E-16
test2	2	0.3558	5.63E-17	0.3558	5.63E-17	0.2443	3.56E-03	0.3524	5.79E-03	0.3558	5.63E-17
test2	3	0.4482	1.69E-16	0.4495	7.26E-03	0.3521	1.45E-02	0.4477	2.07E-02	0.4507	4.67E-03
test2	4	0.5628	1.93E-03	0.4749	1.19E-02	0.4592	4.31E-02	0.4753	1.61E-01	0.4752	6.36E-03
test2	5	0.7950	1.70E-01	0.7539	1.78E-01	0.5931	1.87E-01	0.6375	1.64E-01	0.2969	1.13E-16
test3	2	0.5400	2.25E-16	0.5391	8.84E-04	0.4493	6.34E-03	0.5389	4.18E-03	0.5389	2.25E-16
test3	3	0.8276	0.00E+00	0.8274	2.65E-03	0.5399	1.66E-02	0.8263	9.32E-03	0.8277	5.23E-04
test3	4	0.8565	5.04E-03	0.8566	4.67E-03	0.8279	1.41E-02	0.8543	1.07E-02	0.8570	4.22E-03
test3	5	0.8696	5.77E-03	0.8675	7.52E-03	0.8476	1.66E-02	0.8672	1.19E-02	0.5797	1.13E-16
test30	2	0.5842	0.00E+00	0.5842	0.00E+00	0.5879	1.91E-02	0.5835	1.55E-03	0.5842	0.00E+00
test30	3	0.5894	3.38E-16	0.5887	8.53E-04	0.5813	8.29E-03	0.5889	3.20E-03	0.5888	3.56E-04
test30	4	0.6245	1.13E-16	0.6220	2.36E-03	0.5936	1.54E-02	0.6206	6.88E-03	0.6207	3.20E-04
test30	5	0.6376	1.12E-03	0.6362	3.41E-03	0.6242	9.82E-03	0.6371	3.59E-03	0.3739	2.82E-16
test31	2	0.6520	3.38E-16	0.6520	3.38E-16	0.3776	4.31E-03	0.6513	2.30E-03	0.6520	3.38E-16
test31	3	0.6631	4.51E-16	0.6631	2.36E-03	0.6529	3.97E-03	0.6614	6.89E-03	0.6632	2.73E-04
test31	4	0.6849	1.05E-03	0.6827	4.48E-03	0.6603	1.07E-02	0.6838	1.15E-02	0.6842	3.08E-03
test31	5	0.6931	5.60E-04	0.6851	4.78E-03	0.6799	1.57E-02	0.6871	1.14E-02	0.5150	2.25E-16
test4	2	0.7591	2.25E-16	0.7591	1.44E-04	0.3161	4.65E-03	0.7407	7.69E-02	0.7591	2.25E-16
test4	3	0.8001	3.38E-16	0.7986	2.58E-03	0.7648	5.33E-03	0.7988	8.57E-03	0.7988	3.38E-16
test4	4	0.8164	3.38E-16	0.8133	4.56E-03	0.7993	1.67E-02	0.8152	1.62E-02	0.8159	2.27E-03
test4	5	0.8322	5.76E-03	0.8322	5.88E-03	0.8132	2.11E-02	0.8303	1.67E-02	0.2608	5.63E-17
test5	2	0.7334	3.38E-16	0.7321	3.38E-16	0.2639	3.02E-03	0.7321	8.16E-03	0.7321	3.38E-16
test5	3	0.7597	3.38E-16	0.7583	3.74E-03	0.7354	5.07E-03	0.7565	1.17E-02	0.7597	3.38E-16
test5	4	0.7756	1.11E-03	0.7727	7.66E-03	0.7568	2.14E-02	0.7732	1.60E-02	0.7717	6.23E-03
test5	5	0.7834	8.01E-04	0.7785	1.31E-02	0.7784	2.54E-02	0.7820	2.45E-02	0.5626	1.13E-16

Table 12.4. Mean and STD value of the SSIM metric using Otsu by the DA, GA, KH, PSO, and RRA

To perform a qualitative analysis, Table 12.5 collect segmented images from each method to visually compare them. Regarding the evaluation made for a human expert from the results displayed in Table 12.5, it is possible to mention that the overall performance of the DA algorithm for

thresholding is more accurate to extract the regions where possible cancer lesions or an increased blood flow under the skin have been presented on the breast thermograms.

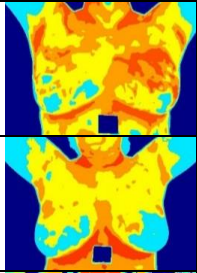
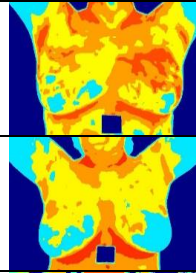
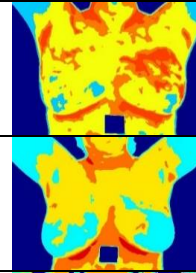
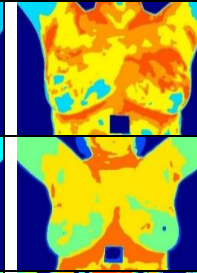
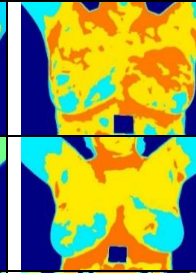
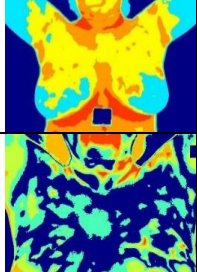
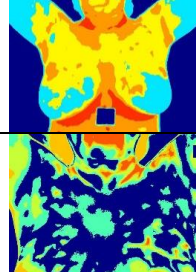
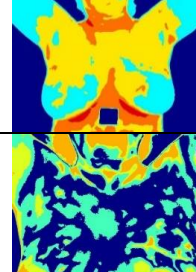
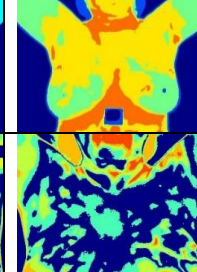
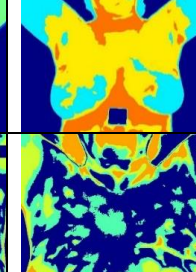
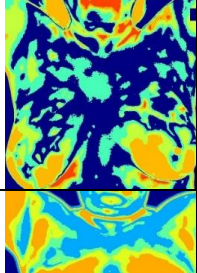
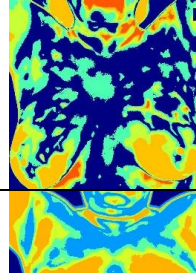
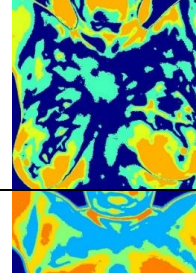
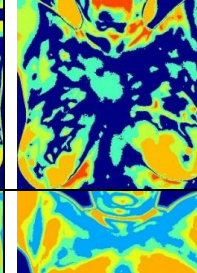
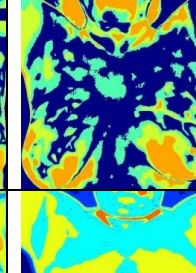
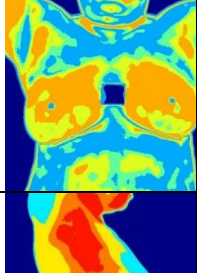
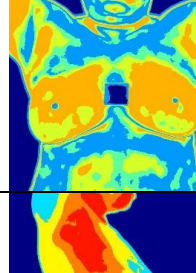
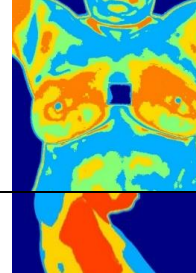
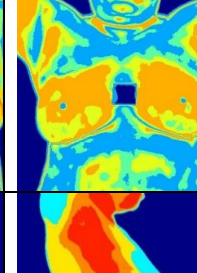
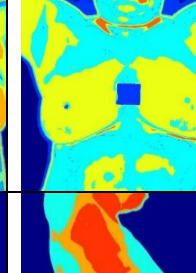
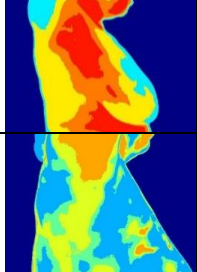
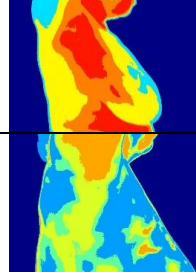
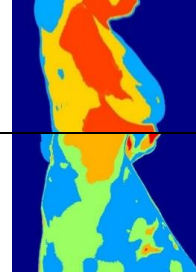
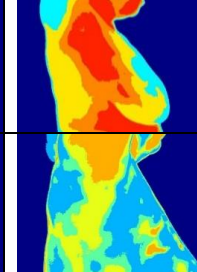
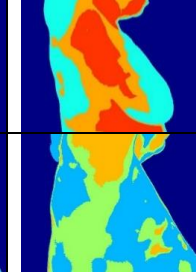
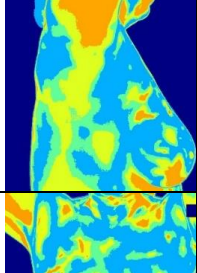
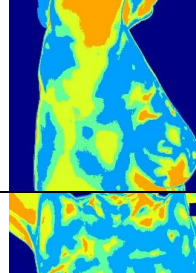
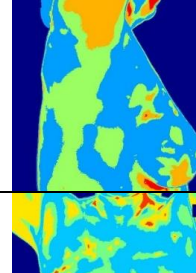
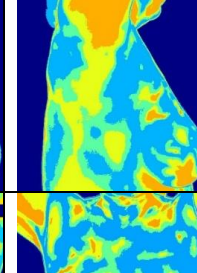
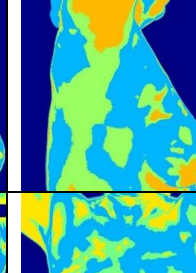
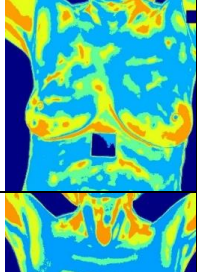
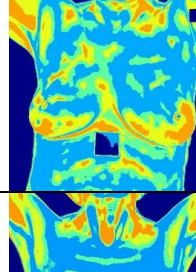
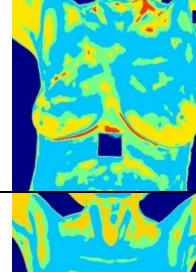
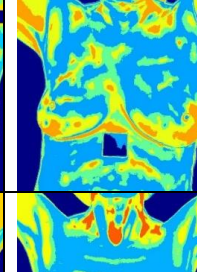
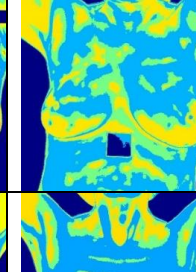
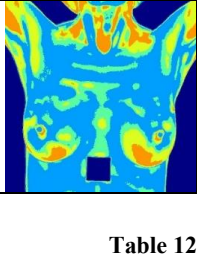
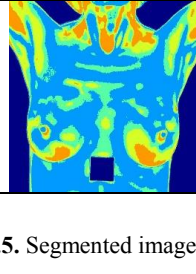
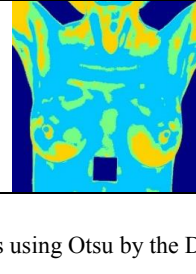
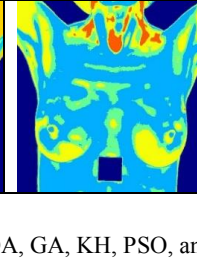
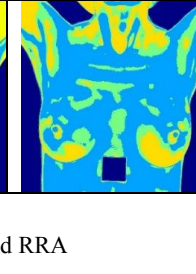
	DA	GA	KH	PSO	RRA
test10					
test11					
test2					
test3					
test30					
test31					
test4					
test5					

Table 12.5. Segmented images using Otsu by the DA, GA, KH, PSO, and RRA

Taking as an example the image “test10” in Table 12.5, the areas depicted in red, are better defined than in the results obtained for the other four methods, such areas in the surface of the

breasts and the armpits frame the regions where the existence of cancer cells are likely to appear, or where the likelihood of appearance is higher. In the case of the image “test5”, it is possible to identify more accurately the risk areas, where the breast lymphatic nodes are located, the same in which the vasodilation changes may have occurred.

Considering the visual thresholding results obtained for the Otsu’s Method by the Dragonfly Algorithm for five classes, in general terms, it can be said that this pixel classification may help the clinicians to evaluate easily the growing of a malignant lesion as well as the local vasodilation changes produced under the skin surface.

12.5.2. Kapur results

This subsection is devoted to analyzing the performance of the presented thresholding approach based on the DA applied to Breast Thermographic images using Kapur as the objective function.

Image	t	DA	GA	KH	PSO	RRA
test10	2	45,118	45,118	46,118	45,118	45,118
test10	3	45,115,152	45,115,152	44,116,151	46,114,152	45,115,152
test10	4	45,115,152,205	46,115,152,205	44,112,152,208	47,114,151,205	45,115,152,205
test10	5	43,79,116,152,205	43,79,116,152,205	48,81,116,146,198	46,79,115,148,203	43,79,116,152,205
test11	2	134,210	134,210	134,206	134,210	134,210
test11	3	52,127,206	52,127,206	59,125,210	51,127,206	52,127,206
test11	4	37,82,128,206	37,82,128,206	38,84,126,212	37,83,127,206	37,82,128,206
test11	5	37,82,127,166,212	37,80,127,166,211	38,80,121,164,204	39,81,125,164,210	37,82,127,166,212
test2	2	95,196	95,196	94,195	95,196	95,196
test2	3	95,144,196	95,144,196	95,152,195	95,149,196	95,144,196
test2	4	53,95,144,196	53,95,144,196	47,94,154,196	52,95,135,197	53,95,144,196
test2	5	53,95,144,196,239	54,92,143,196,239	43,94,126,156,197	53,91,129,160,199	53,95,144,196,239
test3	2	95,197	95,197	96,196	94,197	95,197
test3	3	95,154,196	95,153,197	93,148,199	95,153,196	95,154,197
test3	4	61,95,154,196	60,95,150,197	54,93,156,196	46,96,155,196	61,95,154,197
test3	5	52,95,153,196,241	44,96,129,158,197	39,94,130,156,196	60,92,125,159,200	61,95,126,158,197
test30	2	42,133	42,133	43,132	42,133	42,133
test30	3	42,133,197	42,133,197	42,131,202	42,133,198	42,133,197
test30	4	42,89,136,198	42,89,136,197	44,92,135,191	40,89,136,201	42,89,136,198
test30	5	40,85,128,156,202	41,85,128,156,202	44,89,118,153,199	40,80,124,156,208	40,85,128,156,202
test31	2	134,195	134,195	131,195	134,195	134,195
test31	3	59,95,195	59,95,195	30,136,195	30,141,195	59,95,195
test31	4	59,95,147,195	59,95,147,195	57,95,154,202	61,92,146,195	59,95,147,195
test31	5	29,59,95,147,195	29,59,95,147,195	57,93,133,155,195	32,62,93,139,195	29,59,95,147,195
test4	2	96,196	96,196	96,196	96,196	96,196
test4	3	95,148,196	95,148,196	95,143,199	95,147,196	95,148,196
test4	4	61,95,148,196	61,95,148,196	64,95,137,196	60,93,150,195	61,95,148,196
test4	5	31,61,95,148,196	30,61,95,148,196	27,63,94,161,201	64,95,131,158,199	31,61,95,148,196
test5	2	95,196	95,196	95,195	95,196	95,196
test5	3	58,94,196	58,94,196	61,91,196	57,94,198	58,94,196
test5	4	58,94,146,196	58,94,146,196	59,94,136,197	57,94,137,197	58,94,146,196
test5	5	58,94,146,196,240	58,94,125,159,196	62,93,118,159,203	56,90,128,159,195	58,94,146,196,240

Table 12.6. Thresholds obtained by DA, GA, KH, PSO, and RRA using Kapur’s method as the objective function.

Table 12.6 presents the threshold values found by each algorithm which are later used to segment every image on the qualitative analysis. Following the same scheme as the Otsu’s subsection, the

quality of the thresholded images using Kapur is evaluated and compared using PSNR, FSIM, and SSIM.

In Table 12.7, it can be noticed that the PSNR values reported on the DA column are better than the compared approaches. However, in this metric, it is easy to see that the RRA can provide competitive results, especially on the image ‘test3’.

Image	t	DA	GA	KH	PSO	RRA
test10	2	12.1997	3.60E-15	11.9557	3.60E-15	12.1899 8.98E-03 12.1997 3.60E-15 12.1997 3.60E-15
test10	3	16.8120	4.40E-02	16.4829	5.21E-04	16.7844 2.43E-02 16.7907 6.39E-02 16.8177 8.02E-03
test10	4	21.5439	5.62E-02	21.1196	2.75E-03	21.4279 6.60E-02 21.4767 1.03E-01 21.5427 2.22E-02
test10	5	25.9079	2.11E-03	25.3770	8.38E-03	25.5773 1.64E-01 25.7302 1.94E-01 25.8765 4.61E-02
test11	2	12.6028	9.01E-15	12.3507	9.01E-15	12.5760 2.09E-02 12.6028 9.01E-15 12.6025 1.83E-03
test11	3	17.4219	9.63E-03	17.0822	7.47E-03	17.3664 3.83E-02 17.4153 8.88E-03 17.4305 7.42E-03
test11	4	22.2531	1.33E-04	21.8072	1.10E-03	21.9598 1.05E-01 22.2189 7.91E-02 22.2328 4.93E-02
test11	5	26.6271	7.94E-02	26.5998	7.75E-03	26.2514 1.42E-01 26.4399 1.90E-01 26.1173 5.20E-02
test2	2	12.2890	3.60E-15	12.0432	3.60E-15	12.2763 2.21E-02 12.2886 2.51E-03 12.2890 3.60E-15
test2	3	17.3017	3.60E-15	16.9553	1.49E-03	17.0053 1.69E-01 17.0755 1.75E-01 17.1754 1.88E-01
test2	4	21.5762	1.07E-01	21.1592	1.10E-02	21.2460 1.91E-01 21.3550 2.07E-01 21.5753 4.52E-02
test2	5	25.7208	2.04E-01	25.3353	1.46E-02	25.2232 2.25E-01 25.2730 3.17E-01 25.7420 1.50E-01
test3	2	12.4890	6.82E-02	12.2997	9.01E-15	12.5275 2.50E-02 12.4758 6.40E-02 12.5475 1.91E-02
test3	3	17.3660	1.37E-02	17.0455	2.28E-02	17.1161 1.81E-01 17.0840 2.36E-01 17.3253 1.71E-01
test3	4	21.6751	4.22E-02	21.2736	4.08E-02	21.4249 1.42E-01 21.4423 2.42E-01 21.6875 8.33E-02
test3	5	25.8392	1.72E-01	25.4318	2.33E-02	25.4305 2.07E-01 25.3746 4.71E-01 25.8996 1.44E-01
test30	2	12.1609	5.41E-15	11.9177	5.41E-15	12.1099 3.79E-02 12.1609 5.41E-15 12.1609 5.41E-15
test30	3	17.2918	7.21E-15	16.9456	1.24E-03	17.1989 5.41E-02 17.2836 9.35E-03 17.2918 7.21E-15
test30	4	21.8704	1.13E-02	21.4355	2.77E-03	21.6944 7.89E-02 21.8262 4.66E-02 21.8726 8.14E-03
test30	5	26.3595	1.69E-02	25.8309	4.87E-03	25.9781 1.25E-01 26.1931 1.49E-01 26.3433 3.13E-02
test31	2	11.9671	5.41E-15	11.7278	5.41E-15	11.9138 7.35E-02 11.9588 3.87E-02 11.9456 6.44E-02
test31	3	16.7005	3.60E-15	16.3665	7.50E-05	16.5544 8.71E-02 16.6849 2.10E-02 16.6898 6.04E-02
test31	4	20.8479	6.43E-02	20.6158	8.92E-02	20.6704 1.03E-01 20.7150 1.12E-01 20.9162 1.02E-01
test31	5	25.2580	2.84E-01	24.9256	1.75E-02	24.6141 2.46E-01 24.6269 4.07E-01 25.3255 1.85E-01
test4	2	12.2886	5.41E-15	12.0428	5.41E-15	12.2798 1.34E-02 12.2882 1.34E-03 12.2886 5.41E-15
test4	3	17.4473	7.21E-15	17.0983	6.56E-04	17.2359 1.39E-01 17.3281 1.55E-01 17.3852 1.43E-01
test4	4	21.8937	4.15E-02	21.4643	4.51E-03	21.5335 1.53E-01 21.7453 1.50E-01 21.8770 6.11E-02
test4	5	26.0947	1.84E-01	25.6502	1.53E-02	25.5232 1.89E-01 25.7132 2.30E-01 26.0601 1.59E-01
test5	2	11.7387	1.06E-02	11.5091	5.41E-15	11.7282 1.88E-02 11.7379 1.12E-02 11.7365 1.20E-02
test5	3	16.7691	4.62E-02	16.4413	2.58E-04	16.5718 1.13E-01 16.6841 9.84E-02 16.6777 1.10E-01
test5	4	21.3107	1.00E-01	21.0129	5.81E-02	21.0801 1.02E-01 21.1130 1.45E-01 21.3197 1.01E-01
test5	5	25.9242	1.17E-01	25.4156	1.50E-02	25.3057 2.14E-01 25.3478 4.20E-01 25.8004 2.31E-01

Table 12.7. Mean and STD values of the PSNR metric using Kapur by the DA, GA, KH, PSO, and RRA.

Regarding the Feature Similarity Index Metric (FSIM), the Dragonfly Algorithm can find better threshold values that generate output results with better features; this behavior can be seen in Table 12.8. This table also indicates that many approaches can perform well with a small number of thresholds. This phenomenon suggests that as the number of thresholds increases the complexity of the search space also is significantly incremented.

Image	t	DA	GA	KH	PSO	RRA
test10	2	0.7678	3.38E-16	0.7678	3.38E-16	0.7536 2.33E-03 0.7678 3.38E-16 0.7678 3.38E-16
test10	3	0.8068	8.04E-03	0.8054	1.75E-04	0.7913 2.14E-03 0.8060 7.76E-03 0.8067 3.38E-16
test10	4	0.8127	5.95E-03	0.8111	5.90E-04	0.7973 2.94E-03 0.8114 9.00E-03 0.8124 8.22E-04
test10	5	0.8063	1.77E-04	0.8042	1.07E-03	0.7911 8.22E-03 0.8045 5.91E-03 0.8040 1.56E-03
test11	2	0.8059	2.25E-16	0.8059	2.25E-16	0.7889 2.02E-03 0.8059 2.25E-16 0.8059 2.25E-16
test11	3	0.8348	2.07E-02	0.8069	1.63E-02	0.8185 1.42E-02 0.8264 1.67E-02 0.8067 1.64E-02
test11	4	0.8359	3.07E-04	0.8353	6.54E-04	0.8131 7.54E-03 0.8354 3.73E-03 0.8343 2.61E-03
test11	5	0.8436	4.39E-03	0.8402	8.11E-04	0.8193 7.52E-03 0.8408 5.27E-03 0.8421 2.68E-03
test2	2	0.6353	0.00E+00	0.6346	0.00E+00	0.6305 1.23E-02 0.6346 4.39E-03 0.6346 0.00E+00
test2	3	0.6679	2.25E-16	0.6459	1.68E-03	0.6615 1.76E-02 0.6464 2.40E-02 0.6599 1.91E-02
test2	4	0.7058	1.95E-02	0.7115	1.61E-02	0.6874 2.07E-02 0.7054 2.54E-02 0.7093 1.88E-02
test2	5	0.7424	2.91E-02	0.7293	3.52E-03	0.7190 2.31E-02 0.7380 2.85E-02 0.7328 1.82E-02
test3	2	0.6996	3.38E-16	0.6996	3.38E-16	0.6822 3.83E-03 0.6982 2.67E-03 0.6993 2.15E-03
test3	3	0.7182	1.33E-03	0.7107	2.15E-03	0.6945 1.13E-02 0.7117 2.07E-02 0.7164 1.45E-02
test3	4	0.7546	1.81E-02	0.7455	1.65E-02	0.7170 2.51E-02 0.7442 2.28E-02 0.7525 1.69E-02
test3	5	0.7777	1.84E-02	0.7679	7.33E-03	0.7428 2.13E-02 0.7736 2.18E-02 0.7705 1.36E-02
test30	2	0.8104	0.00E+00	0.8104	0.00E+00	0.7941 3.02E-03 0.8104 0.00E+00 0.8104 0.00E+00
test30	3	0.8425	7.89E-16	0.8424	3.61E-04	0.8267 3.71E-03 0.8422 1.11E-03 0.8425 7.89E-16
test30	4	0.8494	1.78E-03	0.8483	7.03E-04	0.8365 5.81E-03 0.8513 4.44E-03 0.8489 1.25E-03
test30	5	0.8686	7.61E-04	0.8674	9.30E-04	0.8449 5.89E-03 0.8622 5.18E-03 0.8680 1.33E-03
test31	2	0.7184	1.13E-16	0.7184	1.13E-16	0.7040 7.64E-03 0.7189 3.75E-03 0.7203 6.17E-03
test31	3	0.7517	5.63E-16	0.7517	7.18E-05	0.7325 5.69E-03 0.7511 2.23E-03 0.7517 3.46E-04
test31	4	0.7658	2.25E-03	0.7590	1.83E-03	0.7446 1.04E-02 0.7584 1.51E-02 0.7610 4.93E-03
test31	5	0.7911	1.27E-02	0.7873	3.00E-03	0.7644 1.96E-02 0.7818 1.65E-02 0.7908 5.77E-03
test4	2	0.6184	0.00E+00	0.6157	0.00E+00	0.6198 1.79E-02 0.6157 9.01E-03 0.6157 0.00E+00
test4	3	0.6686	0.00E+00	0.6263	3.90E-04	0.6551 1.47E-02 0.6264 2.03E-02 0.6419 2.68E-02
test4	4	0.7250	1.56E-02	0.7275	4.98E-03	0.6946 2.66E-02 0.7192 2.57E-02 0.7285 1.27E-02
test4	5	0.7482	2.14E-02	0.7354	1.84E-03	0.7277 3.22E-02 0.7399 3.82E-02 0.7379 2.30E-02
test5	2	0.6604	6.02E-03	0.6574	3.38E-16	0.6500 8.74E-03 0.6608 6.31E-03 0.6617 6.80E-03
test5	3	0.6837	6.14E-03	0.6825	7.80E-04	0.6885 1.88E-02 0.6939 2.23E-02 0.6987 1.45E-02
test5	4	0.7325	1.69E-02	0.7076	9.05E-03	0.7068 1.98E-02 0.7322 2.34E-02 0.7260 1.89E-02
test5	5	0.7685	3.24E-03	0.7628	3.85E-03	0.7469 1.76E-02 0.7606 2.97E-02 0.7666 1.14E-02

Table 12.8. Mean and STD value of FSIM metric using Kapur by the DA, GA, KH, PSO, and RRA.

Table 12.9 presents the results of the Structural Similarity Index Measure. DA also continues outperforming on most of the images, followed by RRA and PSO. As SSIM plays an important role to determinate the quality of the structures left after the segmentation process this index can help us to decide which method is more fitted for the segmentation of Breast Thermographic images. In this case, DA can be recommended to perform this task.

Image	t	DA	GA	KH	PSO	RRA					
test10	2	0.7343	3.38E-16	0.7196	3.38E-16	0.7217	3.79E-03	0.7343	3.38E-16	0.7343	3.38E-16
test10	3	0.7492	4.98E-03	0.7342	4.91E-05	0.7335	3.52E-03	0.7472	6.39E-03	0.7484	6.76E-16
test10	4	0.7996	1.45E-02	0.7838	1.51E-04	0.7818	3.23E-03	0.7941	1.95E-02	0.7975	1.05E-03
test10	5	0.7826	4.37E-04	0.7671	1.22E-03	0.7709	1.14E-02	0.7829	8.87E-03	0.7841	2.60E-03
test11	2	0.7439	3.38E-16	0.7290	3.38E-16	0.7277	2.02E-03	0.7439	3.38E-16	0.7439	3.38E-16
test11	3	0.7728	2.28E-02	0.7274	1.80E-02	0.7583	1.62E-02	0.7421	1.82E-02	0.7639	1.80E-02
test11	4	0.7701	3.49E-04	0.7546	7.57E-04	0.7497	9.63E-03	0.7693	5.29E-03	0.7698	8.62E-04
test11	5	0.7888	3.43E-03	0.7698	1.05E-03	0.7684	9.58E-03	0.7859	7.85E-03	0.7883	2.43E-03
test2	2	0.7760	2.25E-16	0.7604	2.25E-16	0.7608	1.21E-03	0.7760	3.54E-04	0.7760	2.25E-16
test2	3	0.8064	5.63E-16	0.7716	9.91E-04	0.7718	4.10E-02	0.7976	3.59E-02	0.7876	3.18E-02
test2	4	0.8621	2.78E-02	0.8492	2.16E-02	0.8183	3.81E-02	0.8545	3.40E-02	0.8592	2.62E-02
test2	5	0.8920	1.26E-02	0.8705	2.17E-03	0.8580	1.63E-02	0.8812	1.95E-02	0.8878	8.39E-03
test3	2	0.7184	5.63E-16	0.7046	5.63E-16	0.7021	2.93E-03	0.177	2.26E-03	0.7190	3.21E-03
test3	3	0.7459	9.55E-04	0.7173	1.97E-03	0.7040	3.24E-02	0.7420	4.94E-02	0.7331	3.22E-02
test3	4	0.8160	3.45E-02	0.8032	2.49E-02	0.7663	4.07E-02	0.8059	3.68E-02	0.8054	2.72E-02
test3	5	0.8579	9.68E-03	0.8388	5.58E-03	0.8256	2.22E-02	0.8536	1.98E-02	0.8508	9.00E-03
test30	2	0.5407	1.13E-16	0.5299	1.13E-16	0.5314	2.16E-02	0.5407	1.13E-16	0.5407	1.13E-16
test30	3	0.6219	3.38E-16	0.6095	3.58E-05	0.6082	3.54E-03	0.6219	9.82E-04	0.6219	3.38E-16
test30	4	0.6303	7.06E-03	0.6162	5.76E-04	0.6250	9.34E-03	0.6348	9.02E-03	0.6324	4.96E-03
test30	5	0.6449	1.44E-03	0.6300	9.45E-04	0.6309	7.53E-03	0.6428	5.83E-03	0.6421	2.10E-03
test31	2	0.0886	1.41E-17	0.0650	1.41E-17	0.0870	7.49E-02	0.0737	4.40E-02	0.0663	7.39E-02
test31	3	0.3599	1.69E-16	0.3459	1.27E-03	0.3552	5.57E-02	0.3532	2.56E-02	0.3528	3.73E-02
test31	4	0.6447	1.31E-01	0.6149	6.08E-02	0.6132	1.07E-01	0.6054	1.09E-01	0.5277	1.02E-01
test31	5	0.7217	8.29E-02	0.6704	4.25E-03	0.7281	7.20E-02	0.7081	6.63E-02	0.6803	4.71E-02
test4	2	0.6750	1.13E-16	0.6596	1.13E-16	0.6701	1.11E-02	0.6731	6.19E-03	0.6731	1.13E-16
test4	3	0.7001	2.25E-16	0.6761	1.55E-04	0.6841	2.42E-02	0.7130	2.41E-02	0.6899	2.34E-02
test4	4	0.7823	1.38E-02	0.7668	5.69E-03	0.7354	2.76E-02	0.7713	2.27E-02	0.7806	1.11E-02
test4	5	0.8123	1.73E-02	0.7873	2.39E-03	0.7963	2.78E-02	0.8095	2.65E-02	0.8086	1.82E-02
test5	2	0.2454	2.49E-01	0.0687	1.41E-17	0.3425	3.09E-01	0.2103	2.61E-01	0.1928	2.81E-01
test5	3	0.7023	8.88E-03	0.6827	2.96E-04	0.6848	3.48E-02	0.6986	2.69E-02	0.6982	2.64E-02
test5	4	0.7487	2.38E-02	0.6993	1.24E-02	0.7200	3.01E-02	0.7394	2.62E-02	0.7483	2.74E-02
test5	5	0.7857	3.51E-03	0.7686	2.84E-03	0.7620	1.54E-02	0.7833	2.32E-02	0.7825	5.42E-03

Table 12.9. Mean and STD value of the SSIM metric using Kapur by the DA, GA, KH, PSO, and RRA

	DA	GA	KH	PSO	RRA
test10					
test11					
test2					
test3					
test30					
test31					
test4					
test5					

Table 12.10. Segmented images using Kapur by the DA, GA, KH, PSO, and RRA

After the quantitative results of segmented images using Kapur, it is time for a qualitative analysis of the segmentation results. To visually provide an example of the performance of each algorithm, four thresholds are used to segment every image of the examined sub-set of Breast Thermographic images. Table 12.10 presents the visual comparison of the five approaches. The overall execution of the proposed DA-approach outperforms the other four methods according to the expert's evaluation over the set of segmented images. Taking as an example, the results obtained for

the image “test2”, it can be observed that in the breast area, the enclosed regions depicted in yellow are better defined. These results may help the clinician to proceed with an evaluation of the possible pathological lesions under the skin surface.

Another clear example of the good performance of the DA-based proposal it can be observed in the results provided by the image “test31” in Table 12.10. In such results, it can be appreciated that the regions depicted in red near the breast area are better defined. In this case, the clinician or a subsequent system could be able to identify such areas as “potential risk points” or to conduct a treatment tracing.

Image		Otsu’s method				Kapur’s method			
		DA vs. GA	DA vs. KH	DA vs. PSO	DA vs. RRA	DA vs. GA	DA vs. KH	DA vs. PSO	DA vs. RRA
test10	2	3.59E-01	1.04E-16	3.34E-06	1.04E-16	1.18E-01	1.53E-14	2.18E-06	2.98E-02
test10	3	1.90E-09	2.16E-14	9.59E-14	1.68E-16	3.17E-02	5.91E-13	1.62E-09	1.72E-02
test10	4	2.52E-14	2.18E-14	1.06E-13	2.68E-16	2.08E-10	5.92E-13	4.49E-13	2.25E-07
test10	5	1.83E-13	8.75E-14	8.75E-14	8.41E-15	2.30E-13	4.08E-14	4.49E-14	1.60E-08
test11	2	4.06E-01	1.04E-16	2.59E-07	1.04E-16	7.23E-02	1.53E-14	2.12E-02	3.31E-01
test11	3	7.29E-10	1.51E-14	1.53E-14	1.04E-16	4.36E-03	1.05E-11	6.73E-07	8.00E-03
test11	4	2.18E-14	2.18E-14	2.18E-14	1.68E-16	9.62E-11	4.08E-14	4.92E-14	2.08E-02
test11	5	1.49E-07	1.86E-13	7.79E-11	5.14E-14	1.01E-10	1.10E-11	3.44E-07	5.75E-01
test2	2	7.82E-02	1.04E-16	8.00E-06	1.04E-16	8.20E-02	1.50E-06	3.31E-01	1.55E-02
test2	3	2.47E-10	1.52E-14	7.48E-13	1.04E-16	8.18E-02	1.53E-14	5.76E-14	3.90E-05
test2	4	1.34E-13	3.02E-14	3.02E-14	4.99E-15	2.49E-03	1.87E-12	3.74E-12	1.20E-02
test2	5	9.16E-07	2.35E-13	8.47E-09	3.03E-12	3.40E-01	1.74E-10	9.87E-09	2.11E-01
test3	2	1.60E-01	1.04E-16	7.94E-06	1.04E-16	6.38E-06	6.91E-01	1.57E-01	2.04E-05
test3	3	1.52E-08	1.53E-14	1.53E-14	1.04E-16	4.21E-07	3.02E-12	1.55E-11	4.99E-02
test3	4	7.90E-13	3.02E-14	7.90E-13	3.45E-15	2.11E-02	1.94E-12	8.86E-11	1.57E-01
test3	5	3.44E-06	3.68E-13	1.12E-08	8.19E-13	1.90E-03	7.60E-10	1.68E-07	1.74E-01
test30	2	1.06E-02	1.04E-16	6.16E-07	1.04E-16	3.02E-1	1.53E-14	1.04E-02	4.87E-01
test30	3	2.08E-09	1.53E-14	2.12E-13	1.04E-16	8.17E-02	1.53E-14	7.91E-12	5.92E-02
test30	4	1.53E-14	1.53E-14	1.53E-14	5.65E-16	4.07E-04	2.68E-13	5.48E-10	2.20E-02
test30	5	7.72E-13	8.75E-14	3.77E-13	6.40E-15	4.34E-10	1.58E-13	9.40E-13	1.05E-03
test31	2	5.91E-03	1.04E-16	1.96E-09	1.04E-16	1.53E-01	2.61E-07	8.18E-02	2.21E-02
test31	3	1.55E-08	1.43E-14	2.12E-13	1.04E-16	8.18E-02	1.53E-14	8.38E-12	2.21E-02
test31	4	1.02E-12	4.08E-14	4.08E-14	6.08E-16	3.59E-08	2.72E-10	6.29E-10	1.67E-04
test31	5	5.37E-14	5.37E-14	5.37E-14	3.57E-15	2.76E-01	1.50E-10	4.60E-09	9.85E-01
test4	2	1.60E-01	1.04E-16	5.77E-09	1.04E-16	4.11E-01	8.14E-11	8.17E-02	2.18E-02
test4	3	4.06E-08	1.27E-14	7.48E-13	1.04E-16	3.31E-01	5.80E-14	2.08E-13	3.71E-04
test4	4	1.53E-14	1.53E-14	1.53E-14	1.04E-16	3.17E-03	7.82E-14	3.32E-12	9.33E-03
test4	5	6.09E-03	4.66E-13	3.15E-08	4.07E-12	1.73E-03	5.30E-12	3.49E-09	9.12E-03
test5	2	3.34E-04	1.04E-16	1.04E-07	1.04E-16	5.82E-03	7.43E-04	6.17E-01	4.11E-02
test5	3	2.49E-10	1.21E-14	5.79E-14	1.04E-16	5.92E-01	1.78E-13	1.41E-11	4.14E-07
test5	4	7.38E-06	8.67E-14	1.38E-09	1.07E-15	1.99E-04	1.93E-11	1.22E-09	8.20E-01
test5	5	6.93E-14	6.93E-14	6.93E-14	1.57E-14	8.23E-08	6.07E-13	6.63E-13	3.76E-05

Table 12.11. p -Values of Wilcoxon’s test.

12.6. Discussion

The overall results presented in section 12.5 indicate that the DA applied to the problem of segmentation of Breast Thermographic images performs competitively on the evaluated dataset.

According to most results, DA outperforms its counterparts with a few exceptions. Since the segmentation process is carried over the energy curve rather than the image histogram, contextual information is incorporated into the process. The consideration of the surrounding of a pixel generates segmented images with smaller noise levels making it suitable for segmenting thermal images.

Since the performance of the presented approach is evaluated against four metaheuristic algorithms, the Wilcoxon's rank test was applied for a pair-wise comparison between the DA and the other four approaches (García et al., 2008). For this purpose, the values of the objective function taken from 35 independent samples are evaluated using this non-parametric significance proof. In this case, Wilcoxon's test assesses the differences between two related methods. The analysis is conducted considering a 5% (0.05) significance level over the solution with the best objective function considering both Otsu and Kapur, and presents a pair-wise comparison with all the other algorithms. The test was applied to each image considering the different number of thresholds evaluated.

In the Wilcoxon analysis, it is considered a null hypothesis that there is no notable difference between the two methods ($p > 0.05$). Conversely, it is admitted as an alternative hypothesis that there is an important difference between the two approaches ($p < 0.05$). Table 12.11 presents the p -values computed by the Wilcoxon's test. After a careful analysis over Table 12.11, it is evident that in the majority of the tests, the p -value was less against the other algorithms, which is a strong evidence of the better performance of the DA-based proposal. In Table 12.11, rejected hypothesis are marked as bold where only three experiments are not significantly different to be considered drawn from different distributions, these correspond to the images "test11" and "test5" employing Kapur's method. Such results, match with the evidenced for the human expert regarding the visual results obtained with this technique.

From the qualitative point of view, the DA-proposal performs better in most of the cases for the thresholding task using both Otsu's and Kapur's methods than the other four methods using for comparison. However, in the case of the proposed methodology employing Kapur's objective function presents a lower performance for thresholding in 5 classes compared to the proposal using Otsu's method. In general terms, the results present an effective method for framing the different skin cellular behavior. This finding may yield to a robust non-invasive tool, which could assist clinicians to improve the current breast cancer diagnostic procedure.

12.7. Conclusions

As Thermography is becoming popular in the diagnosis of several diseases where Breast Cancer is one of the most common applications, new sensors are available in the market to improve the health-care industry. However, the incorporation of low cost and portable thermal cameras to the diagnosis process typically involve low-resolution images which makes difficult for the health-care professional the visual inspection of the thermography. According to the quality of the sensor, the temperature differences between different tissues might not be significant enough to provide a clear border between two regions since the heat radiates from a warmer zone to a cooler one. To overcome this problem, this chapter presents a segmentation method based on the Dragonfly Algorithm (DA) to divide the image into homogeneous regions with clear borders. Contrary to similar approaches based on the histogram of the image, the presented methods work over the energy curve of the image to consider spatial information of each pixel and its vicinity. The energy curve shares properties with the histogram of an image as both present valleys and peaks making it suitable for thresholding.

The presented approach uses the DA to search for the best set of threshold values that separate the energy curve into a given number of classes. Each candidate solution is evaluated using a non-parametric criterion as the objective function, either Otsu's or Kapur's. Following the quality of

each solution, the DA can iterate until it reaches an optimal configuration for the segmentation of a given thermography.

To evaluate the performance of the presented methodology, four metaheuristic algorithms were implemented to perform the same task; two of them are classical methods such as Genetic Algorithms (GA) and Particle Search Optimization (PSO), while the other two are novel algorithms recently published, the Runner-Root Algorithm (RRA), and the Krill-Herd (KH) algorithm. All four methods are implemented with two variants; using Otsu, and evaluating the Kapur criterion as the objective function. The experiments were evaluated over a set of eight Breast Thermography images retrieved from the Database for Research Mastology with Infrared Image. The results were quantitatively analyzed by comparing the resemblance of the segmented image in comparison to the original using image quality metrics such as the Peak-Signal-to-Noise Ratio (PSNR), Structural Similarity Index Metric (SSIM), and the Feature Similarity Index Metric (FSIM). As metaheuristic algorithms involve stochastic operators, to validate the statistical results the Wilcoxon's test is applied to determinate if the presented DA-based method is significantly different from the other examined methods. Moreover, eight images were randomly chosen from the Thermography dataset to exanimate the segmented images qualitatively. Results indicate that the DA-Breast Thermography Thresholding (DA-BTT) outperforms the other implementations for both Otsu and Kapur variants on most metrics generating clear images with sharp borders. Although this paper is not intended to provide a method able to diagnose breast cancer by itself, the DA-BTT contributes to the enhancement of thermal images to facilitate the labor of health-professionals in the diagnosis and monitoring of Breast Cancer and other vascular conditions as thermal cameras are cheaper than MRI machines and more accessible to transport.

This work presents an intermediate process for analyzing breast thermograms by using a multi-level thresholding method, which combined with a higher system may help in the cancer diagnosis procedure. Nevertheless, for future development of the presented method, it would be necessary to add a large data set combined with a clinical study, in order to evaluate the skin cell behavior along the time and different stages of pathologies. These circumstances may lead to achieving that the DA-BTT approach could provide a highly reliable clinical decision support, which aims to help clinicians in performing a diagnosis using breast thermography images.

Chapter 13

Conclusion

This chapter aims to provide the general conclusions of the research conducted for this thesis, as well as to illuminate the particular assumptions of each chapter.

Soft Computing, as opposed to traditional computing, deals with approximate models and gives solutions to complex real-life problems. Unlike Hard Computing, Soft Computing is tolerant of imprecision, uncertainty, partial truth, and approximations. In effect, the role model for SC is the human mind.

Intelligent Systems, and hence SC techniques, are becoming more important as the power of computer processing devices increases and their cost is reduced. Intelligent systems are required to make complex decisions and choose the best outcome, using complex algorithms, from many possibilities. This requires fast processing power and the large storage space that has, in recent years, become available to many research centers, universities, and technical colleges at a very low cost.

With the power and recognition of the Internet of Things concept, the need for using SC techniques and building intelligent systems has become more important than ever. Today, most SC applications can be handled efficiently by low-cost but super-fast microcontrollers.

We already see the use of Fuzzy Logic, Artificial Neural Networks, and Expert Systems in many everyday domestic appliances, such as washing machines, cookers, and refrigerators. Many industrial and commercial applications of SC are also in everyday use, and this is only expected to increase within the next decade.

The applications presented in this work were designed with the purpose of serving as technological tools that can then be used for the development of new devices, however, such approach was not further developed. Though, they have highlighted the utility of different SC methodologies in solving problems in Optimization, Computer Vision, and medicine.

This research is composed of several individual projects, which employ both classical and new optimization algorithms.

Another accomplished objective of this research was the translation of Computer Vision and medical problems into optimization problems. The main intention for this was to open the topic for new proposals that can also present new methodologies for solving such problems.

A goal of this research was the development of new evolutionary approaches for the optimization of complex problems; these approaches are introduced in chapters 2 and 3. After testing such approaches over a standard benchmark dataset, they have demonstrated strong capabilities to solve global optimization problems, as well as low-computational cost.

The core of this manuscript is the application of Soft Computing to the domain of Computer Vision; chapters 4 through 10 present several approaches to solving problems in this field. In these chapters, it is possible to observe that the key to structuring the presented methods was the adequa-

tion of the algorithms and the translation into optimization problems. Each of these applications revealed that it is necessary first to analyze the best possible combination of optimization algorithms in order to fit them to a particular problem.

This research also strove to expand the concepts to automated detection and diagnosis for pathologies such as Leukemia and breast cancer, as presented in chapters 11 and 12. The application of SC techniques in medicine has gained great interest worldwide, due to the increase in these diseases.

Following the organization of the thesis, the particular conclusions of each chapter are presented below.

Chapter 1 introduces the field of research for this work and the specific concepts of optimization and Soft Computing.

Chapter 2 presents an Opposition-Based EMO, named OBEMO, that combines the opposition-based learning (OBL) strategy and the standard EMO technique. The OBL is a machine intelligence strategy that simultaneously considers a current estimate and its opposite value to achieve a fast approximation for a given candidate solution. The standard EMO is enhanced by using two OBL steps: the population initialization and the production of new generations. The enhanced algorithm significantly reduces the required computational effort, yet avoids any detriment to the positive search capabilities of the original EMO algorithm.

Results demonstrate that the OBEMO is as accurate as the standard EMO while requiring a shorter number of iterations. Likewise, it is as fast as other state-of-the-art EMO-based algorithms, such as HEMO (Takeuchi, 2008) and FEMO (Rocha & Fernandes, 2007), while still keeping the original accuracy.

Although the results offer evidence that the Opposition-Based EMO method can yield good results for complicated optimization problems, the chapter's aim is to show that the Opposition-Based Electromagnetism-like method can effectively be considered as an attractive alternative for solving global optimization problems.

Additionally, chapter 3 describes a methodology to implement human-knowledge-based optimization strategies. Under this approach, a conducted search strategy is modeled in the rule base of a Takagi-Sugeno Fuzzy inference system, so that the implemented fuzzy rules express the conditions under which candidate solutions are evolved during the optimization process.

All reported approaches that integrate fuzzy logic and metaheuristic techniques consider the optimization capabilities of the metaheuristic algorithms for improving the performance of fuzzy systems. In the method presented here, the approach is completely different. Under this new schema, the fuzzy system directly conducts the search strategy during the optimization process. In this chapter, the main goal is to propose a methodology for emulating human search strategies in an algorithmic structure. To the best of our knowledge, this is the first time that a fuzzy system is used as a metaheuristic algorithm.

The presented methodology exhibits three important characteristics: (1) Generation: Under the proposed methodology, fuzzy logic provides a simple and well-known method for constructing a search strategy via the use of human knowledge. (2) Transparency: It generates fully interpretable models whose content expresses the search strategy as humans can conduct it. (3) Improvement: As human experts interact with an optimization process, they obtain a better understanding of successful search strategies capable of finding optimal solutions. As a result, new rules are added so that their inclusion in the existing rule base improves the quality of the original search strategy.

Under the proposed methodology, new rules can be easily incorporated into an already existent system. The addition of such rules allows the capacities of the original system to be extended.

To demonstrate the ability and robustness of our approach, the presented fuzzy-based algorithm has been experimentally evaluated with a test suite of 19 benchmark functions. To assess the performance of the fuzzy-based algorithm, it has been compared to other popular optimization approaches based on evolutionary principles currently in use. The results, statistically validated, have confirmed that the presented algorithm outperforms its competitors for most of the test functions in terms of its solution quality and convergence.

Additionally, chapter 4 details an image segmentator approach based on LVQ networks that consider the segmentation process as a pixel classification fully based on color. The segmentator operated directly upon the image pixels using the classification properties of the LVQ networks. The algorithm was effectively applied to the segmentation of sampled images, showing its capacity to satisfactorily segment color despite remarkable illumination differences in indoor and outdoor scenes. The results demonstrated the operation of the LVQ algorithm, which in turn is capable of topologically organizing the input space, thus accomplishing the segmentation process despite a small number of neurons.

The presented system introduced two important features. First, since the LVQ algorithm works directly on the image pixels with no dynamic model or probability distribution, the execution time is faster than other approaches. Second, the algorithm exhibited interesting generalization properties, in particular considering images with changing illumination. Supplementary increases in the segmentator performance might be reached if the parameter is also adapted using some kind of optimization technique.

Further, chapter 5 described a Block-Matching algorithm that combines Harmony Search with a fitness approximation mode. The approach used as potential solutions the motion vectors belonging to the search window. To save computational time, the approach incorporated a fitness calculation strategy to decide which motion vectors can be estimated or actually evaluated. Guided by the values given by this fitness calculation strategy, the set of motion vectors are evolved using the HS operators so that the best possible motion vector can be identified.

Since the presented algorithm does not consider any fixed search pattern during the BM process or any other movement assumption, a high probability for finding the true minimum (accurate motion vector) is expected regardless of the movement complexity contained in the sequence. Therefore, the chance of being trapped in local minimum is reduced in comparison to other BM algorithms.

The performance of HS-BM has been compared to other existing BM algorithms by considering different sequences that present a great variety of formats and movement types. Experimental results demonstrate that the presented algorithm maintains the best balance between coding efficiency and computational complexity.

Although the experimental results indicate that the HS-BM method can yield better results on complicated sequences, it should be noted that the aim of this chapter is to show that the fitness approximation can effectively serve as an attractive alternative to evolutionary algorithms for solving complex optimization problems while demanding fewer function evaluations.

Chapter 6 presents an automatic image multi-threshold approach based on Learning Automata (LA). The segmentation process is considered to be similar to an optimization problem. The algorithm approximates the 1-D histogram of a given image using a Gaussian mixture model whose parameters are calculated through the LA algorithm CARLA. Each Gaussian function approximating the histogram represents a pixel class and therefore one threshold point.

Experimental evidence showed that the LA algorithm has an acceptable compromise between its convergence time and its computational cost when it is compared to the Expectation-Maximization (EM) method and the Levenberg-Marquardt (LM) algorithm. Additionally, the LA algorithm also exhibited a better performance under certain circumstances (singularities); likewise, it is well-reported in the literature (Gupta & Sortrakul, 1998; Park & Ozeki, 2009) that the EM and the LM have underperformed. Finally, the results have indicated that the stochastic search accomplished by the LA method shows a consistent performance without regard to the initial value and a greater chance to reach the global minimum.

Building on the above, chapter 7 presents a corner detection algorithm which models the structure of a potential corner in images based on a fuzzy rule set. The method is able to tolerate implicit imprecision and impulsive noise. Experimental evidence suggests that the fuzzy-based algorithm produces better results than other common methods, such as the Harris detector and the fuzzy approach proposed by Banerjee & Kundu (2008).

The presented algorithm was able to successfully identify corners on images containing different uncertainty conditions. However, it is also sensitive to blurring; specifically, a steaming up effect is produced when considering a neighborhood window wider than the one previously used for building the fuzzy model of corners (templates). Such a fact should not be considered inconvenient, because the fuzzy-based algorithm is still more capable of identifying corners amid similar blurring levels than those of conventional algorithms.

The presented detector is stable and has shown robustness to impulsive noise. This represents its major advantage over the Harris method, considering that impulsive noise is commonly found in real-time images. Moreover, the presented algorithm also exhibits a tolerance to imprecision that matches the performance of that of Banerjee & Kundu.

Furthermore, chapter 8 presents an algorithm for the automatic detection of circular shapes within complicated and noisy images with no consideration of the conventional Hough transform principles. The presented method is based on a newly developed Artificial Immune Optimization (AIO) technique, known as the Clonal Selection Algorithm (CSA). The approach detects the circle in complex images with little visual distortion despite the presence of noisy background pixels.

An important feature of this method is the consideration of the circle detection problem through an optimization approach. Such a view enables the algorithm to detect arcs or occluded circles while still matching imperfect circles. This approach demonstrates that the CSA method outperforms both the GA (as described in Ayala-Ramirez et al., 2006) and the IRHT (as described in Lu & Tan, 2008) within a statistically significant framework. The results obtained for the CSA regarding complicated and noisy images, in comparison to the GA and the IRHT methods, indicate that the Artificial Immune Systems can effectively serve as an attractive alternative to the evolutionary algorithms previously employed to extract circular shapes from images.

Chapter 9 presents a novel nature-inspired algorithm, called the States of Matter Search (SMS), for solving the pattern detection (PD) problem. The SMS algorithm is based on the simulation of the states of matter phenomenon. In SMS, individuals emulate molecules that interact with each other using evolutionary operations based on the physical principles of the thermal-energy motion mechanism. Such operations allow for the increase of the population diversity and avoid the concentration of particles within a local minimum. The presented approach combines the use of the defined operators with a control strategy that modifies the parameter setting of each operation during the evolution process. The algorithm is devised by considering each state of matter at one different exploration–exploitation rate. Thus, the evolutionary process is divided into three stages that emulate the three states of matter: gas, liquid, and solid. At each state, molecules (individuals) ex-

hibit different behaviors. Beginning from the gas state (pure exploration), the algorithm modifies the intensities of exploration and exploitation until the solid state (pure exploitation) is reached.

This method is also able to save computational time by identifying which NCC values can be merely estimated and which must be calculated instead. As a result, the approach is able not only to substantially reduce the number search positions (by using the SMS approach), but it can also do without the NCC evaluation for many of them. The presented method achieves the best balance compared to other PD algorithms, in terms of both estimation accuracy and computational cost. As a result, the approach can substantially reduce the number of function evaluations while preserving the positive search capabilities of SMS.

The performance of the proposed approach has been compared to other existing PD algorithms by considering different images presenting a great variety of formats and complexities. Experimental results demonstrate the high performance of the proposed method in terms of elapsed time and number of NCC evaluations.

In chapter 10, an automatic image multi-threshold approach based on the Artificial Bee Colony (ABC) algorithm is exhibited. The segmentation process is considered as an optimization problem. The algorithm approximates the 1-D histogram of a given image using a Gaussian mixture model whose parameters are calculated through the ABC algorithm. Each Gaussian function approximating the histogram represents a pixel class and therefore one threshold point.

Experimental evidence shows that the ABC algorithm has an acceptable compromise between its convergence time and its computational cost when compared to the Expectation-Maximization (EM) method and the Levenberg-Marquardt (LM) algorithm. Additionally, the ABC algorithm also exhibits a better performance under certain circumstances (initial conditions), regarding which it is well-reported in the literature (Park & Ozeki, 2009) that the EM has underperformed. Finally, the results have shown that the stochastic search accomplished by the ABC method shows a consistent performance without regard to the initial value while still showing a greater chance to reach the global minimum.

Chapter 11 presents an algorithm for the automatic detection of blood cell images based on the Differential Evolution algorithm. This approach considers the complete process as a multiple ellipse detection problem. It generates a sub-pixel detector which can effectively identify leukocytes in real images. The performance of the DE-method has been compared with other existing WBC detectors (the Boundary Support Vectors (BSV) approach (Wang & Chu, 2009), the Iterative Otsu (IO) method (Wu et al., 2006), the Wang algorithm (Wang Shitong et al., 2007a) and the Genetic algorithm-based (BGA) detector (Karkavitsas & Rangoussi, 2008)), considering several images exhibiting different complexity levels. Experimental results demonstrate the high performance of the proposed method in terms of detection accuracy, robustness, and stability.

Finally, Chapter 12 presents an intermediate process for analyzing breast thermograms by using a multi-level thresholding method, which, if combined with a higher system, may help in the cancer diagnosis procedure. Nevertheless, for future development of the presented method, it would be necessary to add a large data set combined with a clinical study in order to evaluate the skin cell behavior across time and the different stages of pathologies. This continuation may lead to implementation of the presented approach as a highly reliable clinical decision support, which could help clinicians in performing a diagnosis using breast thermography images.

References

- Abak, A. T., Baris, U., & Sankur, B. (n.d.-a). The performance evaluation of thresholding algorithms for optical character recognition. In *Proceedings of the Fourth International Conference on Document Analysis and Recognition* (Vol. 2, pp. 697–700). IEEE Comput. Soc. <https://doi.org/10.1109/ICDAR.1997.620597>
- Abak, A. T., Baris, U., & Sankur, B. (n.d.-b). The performance evaluation of thresholding algorithms for optical character recognition. In *Proceedings of the Fourth International Conference on Document Analysis and Recognition* (Vol. 2, pp. 697–700). IEEE Comput. Soc. <https://doi.org/10.1109/ICDAR.1997.620597>
- Ada, G. L., & Nossal, G. (1987). The Clonal-Selection Theory, *257*(2), 62–69. <https://doi.org/10.2307/24979444>
- Adeli, H., & Hung, S.-L. (1995). *Machine learning: neural networks, genetic algorithms, and fuzzy systems*. Wiley. Retrieved from <https://dl.acm.org/citation.cfm?id=184216>
- Adra, S. F., & Fleming, P. J. (2011). Diversity Management in Evolutionary Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, *15*(2), 183–195. <https://doi.org/10.1109/TEVC.2010.2058117>
- Agrawal, S., Panda, R., Bhuyan, S., & Panigrahi, B. K. (2013). Tsallis entropy based optimal multilevel thresholding using cuckoo search algorithm. *Swarm and Evolutionary Computation*, *11*, 16–30. <https://doi.org/10.1016/j.swevo.2013.02.001>
- Akay, B. (2013). A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding. *Applied Soft Computing Journal*, *13*(6), 3066–3091. <https://doi.org/10.1016/j.asoc.2012.03.072>
- Akay, B., & Karaboga, D. (2015). A survey on the applications of artificial bee colony in signal, image, and video processing. *Signal, Image and Video Processing*. <https://doi.org/10.1007/s11760-015-0758-4>
- Alcalá-Fdez, J., Alcalá, R., Gacto, M. J., & Herrera, F. (2009). Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms. *Fuzzy Sets and Systems*, *160*(7), 905–921. <https://doi.org/10.1016/J.FSS.2008.05.012>
- Alcala-Fdez, J., Alcala, R., & Herrera, F. (2011). A Fuzzy Association Rule-Based Classification Model for High-Dimensional Problems With Genetic Rule Selection and Lateral Tuning. *IEEE Transactions on Fuzzy Systems*, *19*(5), 857–872. <https://doi.org/10.1109/TFUZZ.2011.2147794>
- Alcala, R., Alcala-Fdez, J., & Herrera, F. (2007). A Proposal for the Genetic Lateral Tuning of Linguistic Fuzzy Systems and Its Interaction With Rule Selection. *IEEE Transactions on Fuzzy Systems*, *15*(4), 616–635. <https://doi.org/10.1109/TFUZZ.2006.889880>
- Alcala, R., Gacto, M. J., & Herrera, F. (2011). A Fast and Scalable Multiobjective Genetic Fuzzy System for Linguistic Fuzzy Modeling in High-Dimensional Regression Problems. *IEEE Transactions on Fuzzy Systems*, *19*(4), 666–681. <https://doi.org/10.1109/TFUZZ.2011.2131657>
- Aldwaik, M., & Adeli, H. (2014). Advances in optimization of highrise building structures. *Structural and Multidisciplinary Optimization*. <https://doi.org/10.1007/s00158-014-1148-1>
- Ali, M. I., & Shabir, M. (2014). Logic Connectives for Soft Sets and Fuzzy Soft Sets. *IEEE Transactions on Fuzzy Systems*, *22*(6), 1431–1442. <https://doi.org/10.1109/TFUZZ.2013.2294182>
- Amjad Iqbal, M., Kazim Khan, N., Mujtaba, H., & Rauf Baig, A. (2011). *Information and Control ICIC International ©2011 ISSN. International Journal of Innovative Computing* (Vol. 7). Retrieved from <http://www.ijicic.org/09-1070-1.pdf>
- Anbar, M. (1994). Hyperthermia of the cancerous breast: analysis of mechanism. *Cancer Letters*, *84*(1), 23–29. [https://doi.org/10.1016/0304-3835\(94\)90354-9](https://doi.org/10.1016/0304-3835(94)90354-9)
- Anbar, M., Milescu, L., Naumov, A., Brown, C., Button, T., Carty, C., & AIDulaimi, K. (2001). Detection of cancerous breasts by dynamic area telethermometry. *IEEE Engineering in Medicine and Biology Magazine*, *20*(5), 80–91. <https://doi.org/10.1109/51.956823>

- Arakeri, M. P., & Reddy, G. R. M. (2013). Computer-aided diagnosis system for tissue characterization of brain tumor on magnetic resonance images. *Signal, Image and Video Processing*, 9(2), 409–425. <https://doi.org/10.1007/s11760-013-0456-z>
- Atherton, T. J., & Kerbyson, D. J. (n.d.). The Coherent Circle Hough Transform. <https://doi.org/10.5244/C.7.27>
- Atherton, T. J., & Kerbyson, D. J. (1993). Using phase to represent radius in the coherent circle Hough transform. In *IEE Colloquium on Hough Transforms*. London. Retrieved from <https://ieeexplore.ieee.org/document/243198>
- Ayala-Ramirez, V., Garcia-Capulin, C. H., Perez-Garcia, A., & Sanchez-Yanez, R. E. (2006). Circle detection on images using genetic algorithms. *Pattern Recognition Letters*. <https://doi.org/10.1016/j.patrec.2005.10.003>
- Ayvaz, M. T. (2007). Simultaneous determination of aquifer parameters and zone structures with fuzzy c-means clustering and meta-heuristic harmony search algorithm. *Advances in Water Resources*, 30(11), 2326–2338. <https://doi.org/10.1016/J.ADVWATRES.2007.05.009>
- B, M. E., Essa, E., Elkhateb, A., Hassanien, A. E., & Hamad, A. (2017). Cascade Multimodal Biometric System Using Fingerprint and Iris Patterns. In A. Hassanien, K. Shaalan, T. Gaber, & M. Tolba (Eds.), *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2017. AISI 2017*. (Vol. 639). Springer Cham. <https://doi.org/10.1007/978-3-319-64861-3>
- Babu, B. V., & Munawar, S. A. (2007). Differential evolution strategies for optimal design of shell-and-tube heat exchangers. *Chemical Engineering Science*, 62(14), 3720–3739. <https://doi.org/10.1016/J.CES.2007.03.039>
- Bagis, A., & Konar, M. (2016). Comparison of Sugeno and Mamdani fuzzy models optimized by artificial bee colony algorithm for nonlinear system modelling. *Transactions of the Institute of Measurement and Control*, 38(5), 579–592. <https://doi.org/10.1177/0142331215591239>
- Balafar, M. A., Ramli, A. R., Saripan, M. I., & Mashohor, S. (2010). Review of brain MRI image segmentation methods. *Artificial Intelligence Review*, 33(3), 261–274. <https://doi.org/10.1007/s10462-010-9155-0>
- Baldick, R. (2006). *Applied optimization: formulation and algorithms for engineering systems*. Cambridge University Press.
- Banerjee, M., & Kundu, M. K. (2008). Handling of impreciseness in gray level corner detection using fuzzy set theoretic approach. *Applied Soft Computing*, 8(4), 1680–1691. <https://doi.org/10.1016/J.ASOC.2007.09.001>
- Barron, J. L., Fleet, D. J., & Beauchemin, S. S. (1994). Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1), 43–77. <https://doi.org/10.1007/BF01420984>
- Basak, J., & Pal, S. K. (2005). Theoretical quantification of shape distortion in fuzzy Hough transform. *Fuzzy Sets and Systems*, 154(2), 227–250. <https://doi.org/10.1016/J.FSS.2005.02.014>
- Baştürk, A., & Günay, E. (2009). Efficient edge detection in digital images using a cellular neural network optimized by differential evolution algorithm. *Expert Systems with Applications*, 36(2), 2645–2650. <https://doi.org/10.1016/J.ESWA.2008.01.082>
- Becker, J.-M., Grousson, S., & Coltuc, D. (n.d.). From Hough transform to integral geometry [image processing]. In *IEEE International Geoscience and Remote Sensing Symposium* (Vol. 3, pp. 1444–1446). IEEE. <https://doi.org/10.1109/IGARSS.2002.1026143>
- Beigy, H., & Meybodi, M. R. (2006). A new continuous action-set learning automaton for function optimization. *Journal of the Franklin Institute*, 343(1), 27–47. <https://doi.org/10.1016/J.JFRANKLIN.2005.07.004>
- Bhanu, B. (1986a). Automatic Target Recognition: State of the Art Survey. *IEEE Transactions on Aerospace and Electronic Systems*, AES-22(4), 364–379. <https://doi.org/10.1109/TAES.1986.310772>
- Bhanu, B. (1986b). Automatic Target Recognition: State of the Art Survey. *IEEE Transactions on Aerospace and Electronic Systems*, AES-22(4), 364–379.

- <https://doi.org/10.1109/TAES.1986.310772>
- Birbil, Ş. İ., Fang, S.-C., & Sheu, R.-L. (2004). On the Convergence of a Population-Based Global Optimization Algorithm. *Journal of Global Optimization*, 30(2–3), 301–318. <https://doi.org/10.1007/s10898-004-8270-3>
- Birbil, Ş. İ., & Fang, S. C. (2003). An electromagnetism-like mechanism for global optimization. *Journal of Global Optimization*. <https://doi.org/10.1023/A:1022452626305>
- Blum, C., & Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*. <https://doi.org/10.1007/s10479-005-3971-7>
- Boccignone, G., Ferraro, M., & Napoletano, P. (2004). Diffused expectation maximisation for image segmentation. *Electronics Letters*, 40(18), 1107. <https://doi.org/10.1049/el:20045792>
- Boccignone, G., Ferraro, M., & Napoletano P. (2004). DEM: Diffused expectation maximisation for image segmentation - File Exchange - MATLAB Central. Retrieved February 1, 2019, from <https://www.mathworks.com/matlabcentral/fileexchange/37197-dem-diffused-expectation-maximisation-for-image-segmentation>
- Boccignone, G., Napoletano, P., Caggiano, V., & Ferraro, M. (2007). A multiresolution diffused expectation–maximization algorithm for medical image segmentation. *Computers in Biology and Medicine*, 37(1), 83–96. <https://doi.org/10.1016/J.COMPBIOMED.2005.10.002>
- Bohlooli, A., & Jamshidi, K. (2012). A GPS-free method for vehicle future movement directions prediction using SOM for VANET. *Applied Intelligence*, 36(3), 685–697. <https://doi.org/10.1007/s10489-011-0289-9>
- Böhning, D., & Seidel, W. (2003). Editorial: recent developments in mixture models. *Computational Statistics & Data Analysis*, 41(3–4), 349–357. [https://doi.org/10.1016/S0167-9473\(02\)00161-5](https://doi.org/10.1016/S0167-9473(02)00161-5)
- Boltzmann, L. (1884). Ableitung des Stefan’schen Gesetzes, betreffend die Abhängigkeit der Wärmestrahlung von der Temperatur aus der electromagnetischen Lichttheorie. *Annalen Der Physik*, 258(6), 291–294. <https://doi.org/10.1002/andp.18842580616>
- Bongiovanni, G., Crescenzi, P., & Guerra, C. (1995). Parallel Simulated Annealing for Shape Detection. *Computer Vision and Image Understanding*, 61(1), 60–69. <https://doi.org/10.1006/CVIU.1995.1005>
- Borchardt, T. B., Conci, A., Lima, R. C. F., Resmini, R., & Sanchez, A. (2013). Breast thermography from an image processing viewpoint: A survey. *Signal Processing*, 93(10), 2785–2803. <https://doi.org/10.1016/j.sigpro.2012.08.012>
- Borji, A., & Hamidi, M. (2008). A NEW APPROACH TO GLOBAL OPTIMIZATION MOTIVATED BY PARLIAMENTARY POLITICAL COMPETITIONS. *International Journal of Innovative Computing, Information and Control ICIC International c* (Vol. x). Retrieved from <https://pdfs.semanticscholar.org/f8c3/bd664975f7a80982c68c353617f572695770.pdf>
- Borzabadi, A. H., Sadjadi, M. E., & Moshiri, B. (2010). A NUMERICAL SCHEME FOR APPROXIMATE OPTIMAL CONTROL OF NONLINEAR HYBRID SYSTEMS. *International Journal of Innovative Computing, Information and Control ICIC International c* (Vol. 6). Retrieved from <http://www.ijicic.org/09-0126-1.pdf>
- Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences*, 237, 82–117. <https://doi.org/10.1016/J.INS.2013.02.041>
- Brabazon, A., & O’Neill, M. (2006). *Biologically Inspired Algorithms for Financial Modelling*. Berlin/Heidelberg: Springer-Verlag. <https://doi.org/10.1007/3-540-31307-9>
- Bradski, G. R., & Bradski, G. R. (1998). Computer Vision Face Tracking For Use in a Perceptual User Interface. In *Workshop on Applications of Computer Vision* (pp. 214–219). Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.14.7673>
- Branke, J., & Schmidt, C. (2005). Faster convergence by means of fitness estimation. *Soft Computing*, 9(1), 13–20. <https://doi.org/10.1007/s00500-003-0329-4>
- Bresenham, J., & Jack. (1977). A linear algorithm for incremental digital display of circular arcs. *Communications of the ACM*, 20(2), 100–106. <https://doi.org/10.1145/359423.359432>
- Brunelli, R. (2009). *Template Matching Techniques in Computer Vision*. Chichester, UK: John

- Wiley & Sons, Ltd. <https://doi.org/10.1002/9780470744055>
- Buche, D., Schraudolph, N. N., & Koumoutsakos, P. (2005). Accelerating Evolutionary Algorithms With Gaussian Process Fitness Function Models. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 35(2), 183–194. <https://doi.org/10.1109/TSMCC.2004.841917>
- Campelo, F., Guimaraes, F. G., Igarashi, H., & Ramirez, J. A. (2005). A clonal selection algorithm for optimization in electromagnetics. *IEEE Transactions on Magnetics*, 41(5), 1736–1739. <https://doi.org/10.1109/TMAG.2005.846043>
- Canadian Cancer Society. (2015). Canadian Cancer Statistics Special topic : Predictions of the future burden of cancer in Canada. *Public Health Agency of Canada*, 1–151. <https://doi.org/Canadian Cancer Society>
- Caraveo, C., Valdez, F., & Castillo, O. (2016). Optimization of fuzzy controller design using a new bee colony algorithm with fuzzy dynamic parameter adaptation. *Applied Soft Computing*, 43, 131–142. <https://doi.org/10.1016/J.ASOC.2016.02.033>
- Carmona, C. J., González, P., del Jesus, M. J., Navío-Acosta, M., & Jiménez-Trevino, L. (2011). Evolutionary fuzzy rule extraction for subgroup discovery in a psychiatric emergency department. *Soft Computing*, 15(12), 2435–2448. <https://doi.org/10.1007/s00500-010-0670-3>
- Castillo, O., & Melin, P. (2014). A review on interval type-2 fuzzy logic applications in intelligent control. *Information Sciences*, 279, 615–631. <https://doi.org/10.1016/J.INS.2014.04.015>
- Castillo, O., Neyoy, H., Soria, J., Melin, P., & Valdez, F. (2015). A new approach for dynamic fuzzy logic parameter tuning in Ant Colony Optimization and its application in fuzzy control of a mobile robot. *Applied Soft Computing*, 28, 150–159. <https://doi.org/10.1016/J.ASOC.2014.12.002>
- Castillo, O., Ochoa, P., & Soria, J. (2016). Differential Evolution with Fuzzy Logic for Dynamic Adaptation of Parameters in Mathematical Function Optimization (pp. 361–374). Springer, Cham. https://doi.org/10.1007/978-3-319-26302-1_21
- Cengel, Y. (2014). *Thermodynamics : an engineering approach*. McGraw-Hill.
- Ceruti, M. G., & Rubin, S. H. (2007). Infodynamics: Analogical analysis of states of matter and information. *Information Sciences*, 177(4), 969–987. <https://doi.org/10.1016/J.INS.2006.07.006>
- Chae, S. H., Moon, H. M., Chung, Y., Shin, J. H., & Pan, S. B. (2016). Automatic lung segmentation for large-scale medical image management. *Multimedia Tools and Applications*, 75(23), 15347–15363. <https://doi.org/10.1007/s11042-014-2201-1>
- Chang, J.-F. (2009). A PERFORMANCE COMPARISON BETWEEN GENETIC ALGORITHMS AND PARTICLE SWARM OPTIMIZATION APPLIED IN CONSTRUCTING EQUITY PORTFOLIOS. *International Journal of Innovative Computing, Information and Control ICIC International c* (Vol. 5). Retrieved from <http://www.ijicic.org/isii08-246-1.pdf>
- Chang, P.-L., & Teng, W.-G. (2007). Exploiting the Self-Organizing Map for Medical Image Segmentation. *Twentieth IEEE International Symposium on Computer-Based Medical Systems (CBMS'07)*, 281–288. <https://doi.org/10.1109/CBMS.2007.48>
- Chaturvedi, D. K. (2008). *Soft computing: techniques and its applications in electrical engineering*. Springer.
- Chen, G., Low, C. P., & Yang, Z. (2009). Preserving and Exploiting Genetic Diversity in Evolutionary Programming Algorithms. *IEEE Transactions on Evolutionary Computation*, 13(3), 661–673. <https://doi.org/10.1109/TEVC.2008.2011742>
- Chen, S., & Wang, M. (2005). Seeking multi-thresholds directly from support vectors for image segmentation. *Neurocomputing*, 67, 335–344. <https://doi.org/10.1016/J.NEUCOM.2004.12.006>
- Chen, T.-C., & Chung, K.-L. (2001). An Efficient Randomized Algorithm for Detecting Circles. *Computer Vision and Image Understanding*, 83(2), 172–191. <https://doi.org/10.1006/CVIU.2001.0923>
- Chen, T. Q., & Lu, Y. (2002). Color image segmentation—an innovative approach. *Pattern Recognition*, 35(2), 395–405. [https://doi.org/10.1016/S0031-3203\(01\)00050-4](https://doi.org/10.1016/S0031-3203(01)00050-4)

- Chen, Z., Zhou, P., He, Y., & Chen, Y. (2002). (PDF) *Fast Integer Pel and Fractional Pel Motion Estimation for JVT*. Geneva. Retrieved from https://www.researchgate.net/publication/237128290_Fast_Integer_Pel_and_Fractional_Pel_Motion_Estimation_for_JVT
- Cheng, H. D., Guo, Y., & Zhang, Y. (2009). A novel Hough transform based on eliminating particle swarm optimization and its applications. *Pattern Recognition*, 42(9), 1959–1969. <https://doi.org/10.1016/J.PATCOG.2008.11.028>
- Cheng, H. D., Jiang, X. H., Sun, Y., & Wang, J. (2001). Color image segmentation: advances and prospects. *Pattern Recognition*, 34(12), 2259–2281. [https://doi.org/10.1016/S0031-3203\(00\)00149-7](https://doi.org/10.1016/S0031-3203(00)00149-7)
- Cheng, H. D., Jiang, X. H., & Wang, J. (2002). Color image segmentation based on homogram thresholding and region merging. *Pattern Recognition*, 35(2), 373–393. [https://doi.org/10.1016/S0031-3203\(01\)00054-1](https://doi.org/10.1016/S0031-3203(01)00054-1)
- Chiou, J.-P., Chang, C.-F., & Su, C.-T. (2005). Variable Scaling Hybrid Differential Evolution for Solving Network Reconfiguration of Distribution Systems. *IEEE Transactions on Power Systems*, 20(2), 668–674. <https://doi.org/10.1109/TPWRS.2005.846096>
- Chowdhury, D., Stauffer, D., John Wiley & Sons., & Wiley InterScience (Online service). (2000). *Principles of equilibrium statistical mechanics*. Wiley-VCH.
- Chun-Iiung Lin, & Ja-Ling Wu. (1998). A lightweight genetic block-matching algorithm for video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(4), 386–392. <https://doi.org/10.1109/76.709405>
- Cirrincone, G., & Cirrincone, M. (2003). A Novel Self-Organizing Neural Network for Motion Segmentation. *Applied Intelligence*, 18(1), 27–35. <https://doi.org/10.1023/A:1020970617241>
- Coello, C. A. C., & Cortés, N. C. (2005). Solving Multiobjective Optimization Problems Using an Artificial Immune System. *Genetic Programming and Evolvable Machines*, 6(2), 163–190. <https://doi.org/10.1007/s10710-005-6164-x>
- Comaniciu, D., & Meer, P. (n.d.). Robust analysis of feature spaces: color image segmentation. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 750–755). IEEE Comput. Soc. <https://doi.org/10.1109/CVPR.1997.609410>
- Cordón, O. (2011). A historical review of evolutionary learning methods for Mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems. *International Journal of Approximate Reasoning*, 52(6), 894–913. <https://doi.org/10.1016/J.IJAR.2011.03.004>
- Cordón, O., & Herrera, F. (1997). A three-stage evolutionary process for learning descriptive and approximate fuzzy-logic-controller knowledge bases from examples. *International Journal of Approximate Reasoning*, 17(4), 369–407. [https://doi.org/10.1016/S0888-613X\(96\)00133-8](https://doi.org/10.1016/S0888-613X(96)00133-8)
- Costa, L. da F., & Cesar, R. M. (2001). *Shape analysis and classification : theory and practice*. CRC Press. Retrieved from <https://dl.acm.org/citation.cfm?id=557875>
- Cowan, E. W. (1968). *Basic electromagnetism*. Academic Press.
- Cruz-Ramírez, M., Hervás-Martínez, C., Sánchez-Monedero, J., & Gutiérrez, P. A. (2014). Metrics to guide a multi-objective evolutionary algorithm for ordinal classification. *Neurocomputing*, 135, 21–31. <https://doi.org/10.1016/J.NEUCOM.2013.05.058>
- Cuevas, E., Cienfuegos, M., Zaldívar, D., & Pérez-Cisneros, M. (2013). A swarm optimization algorithm inspired in the behavior of the social-spider. *Expert Systems with Applications*. <https://doi.org/10.1016/j.eswa.2013.05.041>
- Cuevas, E., Díaz Cortés, M. A., & Oliva Navarro, D. A. (n.d.). *Advances of evolutionary computation : methods and operators*. Retrieved from https://www.google.com.mx/search?source=hp&ei=IxlHXI7OJ4bIaNy1n4AL&q=Advances+of+Evolutionary+Computation%3A+Methods+and+Operators%2C+Studies+in+Computational+Intelligence&btnK=Buscar+con+Google&oq=Advances+of+Evolutionary+Computation%3A+Methods+and+Operators%2C+Studies+in+Computational+Intelligence&gs_l=psy-ab.3...4422.7546..8226...3.0..0.118.231.0j2.....0....2j1..gws-wiz.....6..35i39.4mBa_Ac2Dp8

- Cuevas, E., González, M., Zaldivar, D., Pérez-Cisneros, M., & García, G. (2012). An algorithm for global optimization inspired by collective animal behavior. *Discrete Dynamics in Nature and Society*. <https://doi.org/10.1155/2012/638275>
- Cuevas, E., Oliva, D., Zaldivar, D., Pérez-Cisneros, M., & Sossa, H. (2012). Circle detection using electro-magnetism optimization. *Information Sciences*. <https://doi.org/10.1016/j.ins.2010.12.024>
- Cuevas, E., Ortega-Sánchez, N., Zaldivar, D., & Pérez-Cisneros, M. (2012). Circle detection by Harmony Search Optimization. *Journal of Intelligent and Robotic Systems: Theory and Applications*. <https://doi.org/10.1007/s10846-011-9611-3>
- Cuevas, E., Osuna, V., & Oliva, D. (2017). *Multilevel segmentation in digital images. Studies in Computational Intelligence* (Vol. 686). https://doi.org/10.1007/978-3-319-51109-2_2
- Cuevas, E., Zaldivar, D., & Pérez-Cisneros, M. (2010). A novel multi-threshold segmentation approach based on differential evolution optimization. *Expert Systems with Applications*, 37(7), 5265–5271. <https://doi.org/10.1016/J.ESWA.2010.01.013>
- Cuevas, E., Zaldivar, D., & Pérez-Cisneros, M. (2011). Seeking multi-thresholds for image segmentation with Learning Automata. *Machine Vision and Applications*, 22(5), 805–818. <https://doi.org/10.1007/s00138-010-0249-0>
- Cuevas, E., Zaldivar, D., Pérez-Cisneros, M., & Ramírez-Ortegón, M. (2011). Circle detection using discrete differential evolution optimization. *Pattern Analysis and Applications*. <https://doi.org/10.1007/s10044-010-0183-9>
- Cutello, V., Narzisi, G., Nicosia, G., & Pavone, M. (2005). Clonal Selection Algorithms: A Comparative Case Study Using Effective Mutation Potentials (pp. 13–28). Springer, Berlin, Heidelberg. https://doi.org/10.1007/11536444_2
- D. E. GOLDBERG AND J. H. HOLLAND. (1988). GUEST EDITORIAL Genetic Algorithms and Machine Learning. *Machine Learning*. <https://doi.org/10.1023/A:1022602019183>
- Dan, S. (2013). *Evolutionary Optimization Algorithm: Biologically Inspired and Population-Based Approaches to Computer Intelligence*. Wiley. <https://doi.org/10.1007/s13398-014-0173-7.2>
- Dasgupta, D. (2006). Advances in artificial immune systems. *IEEE Computational Intelligence Magazine*, 1(4), 40–49. <https://doi.org/10.1109/MCI.2006.329705>
- Dasgupta, S., Das, S., Biswas, A., & Abraham, A. (2010). Automatic circle detection on digital images with an adaptive bacterial foraging algorithm. *Soft Computing*, 14(11), 1151–1164. <https://doi.org/10.1007/s00500-009-0508-z>
- De Castro, L. N., & Timmis, J. (2002). *Artificial immune systems: a new computational intelligence approach*. Springer. Retrieved from <https://www.springer.com/us/book/9781852335946>
- de Castro, L. N., & Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, 6(3), 239–251. <https://doi.org/10.1109/TEVC.2002.1011539>
- De Castro, L. N., & Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*. <https://doi.org/10.1109/TEVC.2002.1011539>
- De Jong, K. (1988). Learning with genetic algorithms: An overview. *Machine Learning*, 3, 121–138. <https://doi.org/10.1007/BF00113894>
- Definitions | Social Determinants of Health | NCHHSTP | CDC. (n.d.). Retrieved July 5, 2019, from <https://www.cdc.gov/nchhstp/socialdeterminants/definitions.html>
- Dempster, A. P., Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1), 1--38. Retrieved from <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.133.4884>
- Dhungana, A. (2002). *Segmentation of infrared images*.
- Diakides, N. a., & Bronzino, J. D. (2006). *Medical infrared imaging* (Second). FL: CRC Press.
- Dong, G., & Xie, M. (2005). Color Clustering and Learning for Image Segmentation Based on

- Neural Networks. *IEEE Transactions on Neural Networks*, 16(4), 925–936. <https://doi.org/10.1109/TNN.2005.849822>
- Dong, N., Wu, C.-H., Ip, W.-H., Chen, Z.-Q., Chan, C.-Y., & Yung, K.-L. (2011). An improved species based genetic algorithm and its application in multiple template matching for embroidered pattern inspection. *Expert Systems with Applications*, 38(12), 15172–15182. <https://doi.org/10.1016/J.ESWA.2011.05.085>
- Dong, W., Shi, G., & Zhang, L. (2007). Immune memory clonal selection algorithms for designing stack filters. *Neurocomputing*, 70(4–6), 777–784. <https://doi.org/10.1016/J.NEUCOM.2006.10.037>
- Dorigo, M., Dorigo, M., Maniezzo, V., Colorni, A., Dorigo, M., Dorigo, M., ... Colorni, A. (1991). Positive Feedback as a Search Strategy. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.52.6342>
- Egmont-Petersen, M., de Ridder, D., & Handels, H. (2002). Image processing with neural networks—a review. *Pattern Recognition*, 35(10), 2279–2301. [https://doi.org/10.1016/S0031-3203\(01\)00178-9](https://doi.org/10.1016/S0031-3203(01)00178-9)
- Elhoseny, M., Ramirez-Gonzalez, G., Abu-Elnasr, O. M., Shawkat, S. A., N, A., & farouk, A. (2018). Secure Medical Data Transmission Model for IoT-based Healthcare Systems. *IEEE Access*, 6, 1–1. <https://doi.org/10.1109/ACCESS.2018.2817615>
- Etehadtavakol, M., Chandran, V., Ng, E. Y. K., & Kafieh, R. (2013). Breast cancer detection from thermal images using bispectral invariant features. *International Journal of Thermal Sciences*, 69, 21–36. <https://doi.org/10.1016/j.ijthermalsci.2013.03.001>
- Etehadtavakol, M., Lucas, C., Sadri, S., & Ng, E. (2010). Analysis of Breast Thermography Using Fractal Dimension to Establish Possible Difference between Malignant and Benign Patterns. *Journal of Healthcare Engineering*, 1(1), 27–44. <https://doi.org/10.1260/2040-2295.1.1.27>
- Etehadtavakol, M., & NG, E. Y. K. (2013). Breast Thermography As a Potential Non-Contact Method in the Early Detection of Cancer: a Review. *Journal of Mechanics in Medicine and Biology*, 13(02), 1330001. <https://doi.org/10.1142/S0219519413300019>
- Etehadtavakol, M., Sadri, S., & Ng, E. Y. K. (2010). Application of K- and fuzzy c-means for color segmentation of thermal infrared breast images. *Journal of Medical Systems*, 34(1), 35–42. <https://doi.org/10.1007/s10916-008-9213-1>
- Fernández, A., López, V., del Jesus, M. J., & Herrera, F. (2015). Revisiting Evolutionary Fuzzy Systems: Taxonomy, applications, new trends and challenges. *Knowledge-Based Systems*, 80, 109–121. <https://doi.org/10.1016/J.KNOSYS.2015.01.013>
- Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*. <https://doi.org/10.1145/358669.358692>
- Fischler, M. A., & Wolf, H. C. (1994). Locating perceptually salient points on planar curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2), 113–129. <https://doi.org/10.1109/34.273737>
- Freeman, & Davis. (1977). A Corner-Finding Algorithm for Chain-Coded Curves. *IEEE Transactions on Computers*, C-26(3), 297–303. <https://doi.org/10.1109/TC.1977.1674825>
- Frost, G. P., & P., G. (1998). Stochastic optimisation of vehicle suspension control systems via learning automata. Retrieved from <https://dspace.lboro.ac.uk/dspace-jspui/handle/2134/11660>
- Fullér, Robert, Canós-Darós, Lourdes, Canós-Darós, M.-J. (n.d.). TRANSPARENT FUZZY LOGIC BASED METHODS FOR SOME HUMAN RESOURCES PROBLEMS. Retrieved from http://www.revistarecta.com/n13/13_3.pdf
- Fyfe, C., & Jain, L. (2005). Teams of intelligent agents which learn using artificial immune systems. *Journal of Network and Computer Applications*, 29(2–3), 147–159. <https://doi.org/10.1016/j.jnca.2004.10.003>
- Gamagami, P. (1996). Atlas of mammography : new early signs in breast cancer, xix, 364 p.
- Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: A new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation*, 17(12), 4831–4845.

- <https://doi.org/10.1016/j.cnsns.2012.05.010>
- Gao, W., & Ren, H. (2011). *Information and Control ICIC International* ©2011 ISSN. *International Journal of Innovative Computing* (Vol. 7). Retrieved from <http://www.ijicic.org/isme09-si09-1.pdf>
- Gao, X. Z., Wang, X., & Ovaska, S. J. (2009). Fusion of clonal selection algorithm and differential evolution method in training cascade–correlation neural network. *Neurocomputing*, 72(10–12), 2483–2490. <https://doi.org/10.1016/J.NEUCOM.2008.11.004>
- García, S., Molina, D., Lozano, M., & Herrera, F. (2008). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristics*, 15(6), 617–644. <https://doi.org/10.1007/s10732-008-9080-4>
- García, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristics*, 15(6), 617–644. <https://doi.org/10.1007/s10732-008-9080-4>
- Gautherie, M. (1980). Thermopathology of Breast Cancer : Measurement and Analysis. *Annals of the New York Academy of Sciences*, 383–415. <https://doi.org/10.1111/j.1749-6632.1980.tb50764.x>
- Geem, Z. W. (2006). Optimal cost design of water distribution networks using harmony search. *Engineering Optimization*, 38(3), 259–277. <https://doi.org/10.1080/03052150500467430>
- Geem, Z. W. (2008). Novel derivative of harmony search algorithm for discrete design variables. *Applied Mathematics and Computation*, 199(1), 223–230. <https://doi.org/10.1016/J.AMC.2007.09.049>
- Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A New Heuristic Optimization Algorithm: Harmony Search. *Simulation*. <https://doi.org/10.1177/003754970107600201>
- Geem, Z. W., Lee, K. S., & Park, Y. (2005). Application of Harmony Search to Vehicle Routing. *American Journal of Applied Sciences*, 2(12), 1552–1557. Retrieved from <https://thescipub.com/pdf/10.3844/ajassp.2005.1552.1557>
- Ghamisi, P., Couceiro, M. S., Benediktsson, J. A., & Ferreira, N. M. F. (2012). An efficient method for segmentation of images based on fractional calculus and natural selection. *Expert Systems with Applications*, 39(16), 12407–12417. <https://doi.org/10.1016/j.eswa.2012.04.078>
- Gharbia, R., Hassanien, A. E., El-Baz, A. H., Elhoseny, M., & Gunasekaran, M. (2018). Multi-spectral and panchromatic image fusion approach using stationary wavelet transform and swarm flower pollination optimization for remote sensing applications. *Future Generation Computer Systems*, 88, 501–511. <https://doi.org/https://doi.org/10.1016/j.future.2018.06.022>
- Ghosh, S., Bruzzone, L., Patra, S., Bovolo, F., & Ghosh, A. (2007). A Context-Sensitive Technique for Unsupervised Change Detection Based on Hopfield-Type Neural Networks. *IEEE Transactions on Geoscience and Remote Sensing*, 45(3), 778–789. <https://doi.org/10.1109/TGRS.2006.888861>
- Ghosh, S., Razouqi, Q., Schumacher, H. J., & Celmins, A. (1998). A survey of recent advances in fuzzy logic in telecommunications networks and new challenges. *IEEE Transactions on Fuzzy Systems*. <https://doi.org/10.1109/91.705512>
- Giannakoglou, K. C., Papadimitriou, D. I., & Kampolis, I. C. (2006). Aerodynamic shape design using evolutionary algorithms and new gradient-assisted metamodels. *Computer Methods in Applied Mechanics and Engineering*, 195(44–47), 6312–6329. <https://doi.org/10.1016/J.CMA.2005.12.008>
- Goldberg, D. E. (David E. (n.d.). *Genetic algorithms in search, optimization, and machine learning*.
- Goldsby, R. A. (2005). *Immunology*. Scientific American. Retrieved from https://books.google.de/books/about/Immunology.html?id=0gloAAAACAAJ&redir_esc=y
- Gong, M., Jiao, L., Zhang, L., & Du, H. (2009). Immune secondary response and clonal selection inspired optimizers. *Progress in Natural Science*, 19(2), 237–253.

- <https://doi.org/10.1016/J.PNSC.2008.05.026>
- Gong, M., Jiao, L., & Zhang, X. (2008). A population-based artificial immune system for numerical optimization. *Neurocomputing*, 72(1–3), 149–161. <https://doi.org/10.1016/J.NEUCOM.2007.12.041>
- Gonzalez, R. C., & Woods, R. E. (Richard E. (2008). *Digital image processing*. Prentice Hall. Retrieved from https://www.google.com.mx/search?ei=coRIXPGKA4vIwQK52oDIDQ&q=Digital+Image+Processing+gonzalez+and+woods&oq=Digital+Image+Processing+gonzalez+and+woods&gs_l=psy-ab.3..0i7i30i19i7j0i8i7i10i30i19j0i8i7i30i19i2.2059.2597..2669...0.0..1.215.608.2j2j1.....0...1..gws-wiz.....0i7i1j0i13j0i13i30.3MyTABkZzNU
- Grailu, H., Lotfizad, M., & Sadoghi-Yazdi, H. (2009). An improved pattern matching technique for lossy/lossless compression of binary printed Farsi and Arabic textual images. *International Journal of Intelligent Computing and Cybernetics*, 2(1), 120–147. <https://doi.org/10.1108/17563780910939273>
- Gurrero, M., Castillo, O., & Garcia, M. (2015). Fuzzy dynamic parameters adaptation in the Cuckoo Search Algorithm using fuzzy logic. In *2015 IEEE Congress on Evolutionary Computation (CEC)* (pp. 441–448). IEEE. <https://doi.org/10.1109/CEC.2015.7256923>
- Guney, K., & Sarikaya, N. (2009). COMPARISON OF MAMDANI AND SUGENO FUZZY INFERENCE SYSTEM MODELS FOR RESONANT FREQUENCY CALCULATION OF RECTANGULAR MICROSTRIP ANTENNAS. *Progress In Electromagnetics Research B*, 12, 81–104. <https://doi.org/10.2528/PIERB08121302>
- Guo, R., & Pandit, S. M. (1998). Automatic threshold selection based on histogram modes and a discriminant criterion. *Machine Vision and Applications*, 10(5–6), 331–338. <https://doi.org/10.1007/s001380050083>
- Gupta, L., & Sortrakul, T. (1998). A gaussian-mixture-based image segmentation algorithm. *Pattern Recognition*, 31(3), 315–325. [https://doi.org/10.1016/S0031-3203\(97\)00045-9](https://doi.org/10.1016/S0031-3203(97)00045-9)
- Gurkaynak, G., Yilmaz, I., & Haksever, G. (2016). Stifling artificial intelligence: Human perils. *Computer Law & Security Review*, 32(5), 749–758. <https://doi.org/10.1016/j.clsr.2016.05.003>
- Hall, L. O., Bensaid, A. M., Clarke, L. P., Velthuizen, R. P., Silbiger, M. S., & Bezdek, J. C. (1992). A Comparison of Neural Network and Fuzzy Clustering Techniques in Segmenting Magnetic Resonance Images of the Brain. *IEEE Transactions on Neural Networks*, 3(5), 672–682. <https://doi.org/10.1109/72.159057>
- Hamet, P., & Tremblay, J. (2017). Artificial intelligence in medicine. *Metabolism: Clinical and Experimental*. <https://doi.org/10.1016/j.metabol.2017.01.011>
- Hammouche, K., Diaf, M., & Siarry, P. (2008). A multilevel automatic thresholding method based on a genetic algorithm for a fast image segmentation. *Computer Vision and Image Understanding*, 109(2), 163–175. <https://doi.org/10.1016/J.CVIU.2007.09.001>
- Han, J. H., Kóczy, L., & Poston, T. (1994a). Fuzzy Hough transform. *Pattern Recognition Letters*, 15(7), 649–658. [https://doi.org/10.1016/0167-8655\(94\)90068-X](https://doi.org/10.1016/0167-8655(94)90068-X)
- Han, J. H., Kóczy, L. T., & Poston, T. (1994b). Fuzzy Hough transform. *Pattern Recognition Letters*. [https://doi.org/10.1016/0167-8655\(94\)90068-X](https://doi.org/10.1016/0167-8655(94)90068-X)
- Han, M., Liu, C., & Xing, J. (2014). An evolutionary membrane algorithm for global numerical optimization problems. *Information Sciences*, 276, 219–241. <https://doi.org/10.1016/J.INS.2014.02.057>
- Hansen, N., Ostermeier, A., & Gawelczyk, A. (1995). On the Adaptation of Arbitrary Normal Mutation Distributions in Evolution Strategies: The Gen-erating Set Adaptation. In L. J. Eshelman (Ed.), *Proceedings of the 6th International Conference on Genetic Algorithms* (pp. 57–64). Retrieved from <http://www.cmap.polytechnique.fr/~nikolaus.hansen/GSAES.pdf>
- Harris, C., & Stephens, M. (1988). A Combined Corner and Edge Detector. In *Proceedings of the 4th Alvey Vision Conference* (pp. 147–151). <https://doi.org/10.5244/C.2.23>
- He, Y., Chen, H., He, Z., & Zhou, L. (2015). Multi-attribute decision making based on neutral

- averaging operators for intuitionistic fuzzy information. *Applied Soft Computing*, 27, 64–76. <https://doi.org/10.1016/J.ASOC.2014.10.039>
- Hecht, E., & Bueche, F. J. (n.d.). *College physics*.
- Her-Ming Jong, Liang-Gee Chen, & Tzi-Dar Chiueh. (1994). Accuracy improvement and cost reduction of 3-step search block matching algorithm for video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 4(1), 88–90. <https://doi.org/10.1109/76.276175>
- Herrera, F. (2008). Genetic fuzzy systems: taxonomy, current research trends and prospects. *Evolutionary Intelligence*, 1(1), 27–46. <https://doi.org/10.1007/s12065-007-0001-5>
- Hirche, J. (1979). Dixon, L. C. W. / Szegő, G. P. (eds.), Towards global optimisation. Proceedings of a Workshop at the University of Cagliari, Italy, October 1974. Amsterdam-Oxford. North-Holland Publ. Co. 1975. X. 472 S. *ZAMM - Zeitschrift Für Angewandte Mathematik Und Mechanik*, 59(2), 137–138. <https://doi.org/10.1002/zamm.19790590220>
- Ho, S. L., & Yang, S. (2009). An artificial bee colony algorithm for inverse problems. *International Journal of Applied Electromagnetics and Mechanics*, 31(3), 181–192. <https://doi.org/10.3233/JAE-2009-1056>
- Holland, J. H. (John H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press.
- Horng, M.-H., & Liou, R.-J. (2011). Multilevel minimum cross entropy threshold selection based on the firefly algorithm. *Expert Systems with Applications*, 38(12), 14805–14811. <https://doi.org/10.1016/j.eswa.2011.05.069>
- Howell, M. ., & Best, M. . (2000). On-line PID tuning for engine idle-speed control using continuous action reinforcement learning automata. *Control Engineering Practice*, 8(2), 147–154. [https://doi.org/10.1016/S0967-0661\(99\)00141-0](https://doi.org/10.1016/S0967-0661(99)00141-0)
- Howell, M. N., Frost, G. P., Gordon, T. J., & Wu, Q. H. (1997). Continuous action reinforcement learning applied to vehicle suspension control. *Mechatronics*, 7(3), 263–276. [https://doi.org/10.1016/S0957-4158\(97\)00003-2](https://doi.org/10.1016/S0957-4158(97)00003-2)
- Howell, M. N., & Gordon, T. J. (2001). Continuous action reinforcement learning automata and their application to adaptive digital filter design. *Engineering Applications of Artificial Intelligence*, 14(5), 549–561. [https://doi.org/10.1016/S0952-1976\(01\)00034-3](https://doi.org/10.1016/S0952-1976(01)00034-3)
- Huang, Q., Luo, Y., & Zhang, Q. (2017). Breast ultrasound image segmentation: a survey. *International Journal of Computer Assisted Radiology and Surgery*, 12(3), 493–507. <https://doi.org/10.1007/s11548-016-1513-1>
- Huang, Y.-W., Chen, C.-Y., Tsai, C.-H., Shen, C.-F., & Chen, L.-G. (2006). Survey on Block Matching Motion Estimation Algorithms and Architectures with New Results. *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology*, 42(3), 297–320. <https://doi.org/10.1007/s11265-006-4190-4>
- Hung-Kei Chow, K., & Liou, M. L. (1993). Genetic motion search algorithm for video compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 3(6), 440–445. <https://doi.org/10.1109/76.260203>
- Hung, H.-L., & Huang, Y.-F. (2011). *Information and Control ICIC International ©2011 ISSN. International Journal of Innovative Computing (Vol. 7)*. Retrieved from <http://www.ijicic.org/09-1161-1.pdf>
- Huynh, P. T., Jarolimek, A. M., & Daye, S. (1998). The false-negative mammogram. *Radiographics*, 18(5), 1137–1154. <https://doi.org/10.1148/radiographics.18.5.9747612>
- Ignarro, L. J., Harbison, R. G., Wood, K. S., & Kadowitz, P. J. (1986). Dissimilarities between methylene blue and cyanide on relaxation and cyclic GMP formation in endothelium-intact intrapulmonary artery caused by nitrogen oxide-containing vasodilators and acetylcholine. *J Pharmacol Exp Ther*, 236(1), 30–36.
- Iivarinen, J., Peura, M., Särelä, J., & Visa, A. (1997). *Comparison of Combined Shape Descriptors for Irregular Objects*. Retrieved from https://pdfs.semanticscholar.org/60b9/a74f38f7f1f36e6c4fa3e690b2c1c1c90841.pdf?_ga=2.59814061.1659232523.1549288975-443233653.1523378413
- INTERNATIONAL ORGANISATION FOR STANDARDISATION ORGANISATION

- INTERNATIONALE DE NORMALISATION ISO/IEC JTC1/SC29/WG11 CODING OF MOVING PICTURES AND AUDIO INFORMATION TECHNOLOGY-CODING OF AUDIO-VISUAL OBJECTS: VISUAL ISO/IEC 14496-2 Committee Draft. (1998). Retrieved from <http://home.mit.bme.hu/~szanto/education/mpeg/14496-2.pdf>
- Isard, M., & MacCormick, J. (n.d.). BraMBLE: a Bayesian multiple-blob tracker. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001* (Vol. 2, pp. 34–41). IEEE Comput. Soc. <https://doi.org/10.1109/ICCV.2001.937594>
- Jacquey, F., Comby, F., & Strauss, O. (2008). Fuzzy edge detection for omnidirectional images. *Fuzzy Sets and Systems*, 159(15), 1991–2010. <https://doi.org/10.1016/J.FSS.2008.02.022>
- Jain, J., & Jain, A. (1981). Displacement Measurement and Its Application in Interframe Image Coding. *IEEE Transactions on Communications*, 29(12), 1799–1808. <https://doi.org/10.1109/TCOM.1981.1094950>
- Jang, G.-J., & Kweon, I.-S. (2001). Robust Real-time Face Tracking Using Adaptive Color Model. In *International Conference on Robotics and Automation* (pp. 138–149). Retrieved from http://koasas.kaist.ac.kr/bitstream/10203/25009/1/GiJeongJang_ISIM2000.pdf
- Jepson, A. D., Fleet, D. J., & El-Maraghi, T. F. (2003). Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10), 1296–1311. <https://doi.org/10.1109/TPAMI.2003.1233903>
- Jhang, J.-Y., & Lee, K.-C. (2009). Array pattern optimization using electromagnetism-like algorithm. *AEU - International Journal of Electronics and Communications*, 63(6), 491–496. <https://doi.org/10.1016/J.AEUE.2008.04.001>
- Jianhua Lu, & Liou, M. L. (1997). A simple and efficient search algorithm for block-matching motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(2), 429–433. <https://doi.org/10.1109/76.564122>
- Jie Yao, Kharma, N., & Grogono, P. (2004). Fast robust GA-based ellipse detection. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* (p. 859–862 Vol.2). IEEE. <https://doi.org/10.1109/ICPR.2004.1334394>
- Jin, Y. (2005). A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9(1), 3–12. <https://doi.org/10.1007/s00500-003-0328-5>
- Jin, Y. (2011a). Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2), 61–70. <https://doi.org/10.1016/J.SWEVO.2011.05.001>
- Jin, Y. (2011b). Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2), 61–70. <https://doi.org/10.1016/J.SWEVO.2011.05.001>
- Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG. (2002). *JVT-E022d7*. Geneva. Retrieved from http://ip.hhi.de/imagecom_G1/assets/pdfs/JVT-E022d7ncm.pdf
- Jones, G. A., Princen, J., Illingworth, J., & Kittler, J. (n.d.). Robust Estimation of Shape Parameters. <https://doi.org/10.5244/C.4.10>
- Kamel, M., & Zhao, A. (1993). Extraction of Binary Character/Graphics Images from Grayscale Document Images. *CVGIP: Graphical Models and Image Processing*, 55(3), 203–217. <https://doi.org/10.1006/CGIP.1993.1015>
- Kang, F., Li, J., & Xu, Q. (2009). Structural inverse analysis by hybrid simplex artificial bee colony algorithms. *Computers & Structures*, 87(13–14), 861–870. <https://doi.org/10.1016/J.COMPSTRUC.2009.03.001>
- Kang, J.-G., Kim, S., An, S.-Y., & Oh, S.-Y. (2012). A new approach to simultaneous localization and map building with implicit model learning using neuro evolutionary optimization. *Applied Intelligence*, 36(1), 242–269. <https://doi.org/10.1007/s10489-010-0257-9>
- Kannan, S., Slochanal, S. M. R., & Padhy, N. P. (2005). Application and Comparison of Metaheuristic Techniques to Generation Expansion Planning Problem. *IEEE Transactions on Power Systems*, 20(1), 466–475. <https://doi.org/10.1109/TPWRS.2004.840451>
- Kapur, J. N., Sahoo, P. K., & Wong, A. K. C. (1985). A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image*

- Processing*, 29(3), 273–285.
- Karaboga, D. (2005). *An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report - TR06 / Request PDF*. Retrieved from https://www.researchgate.net/publication/255638348_An_Idea_Based_on_Honey_Bee_Swarm_for_Numerical_Optimization_Technical_Report_-_TR06
- Karaboga, D., & Akay, B. (2009). A comparative study of Artificial Bee Colony algorithm. *Applied Mathematics and Computation*, 214(1), 108–132. <https://doi.org/10.1016/J.AMC.2009.03.090>
- Karaboga, D., & Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing*, 8(1), 687–697. <https://doi.org/10.1016/J.ASOC.2007.05.007>
- Karaboga, D., & Ozturk, C. (2011). A novel clustering approach: Artificial Bee Colony (ABC) algorithm. *Applied Soft Computing*, 11(1), 652–657. <https://doi.org/10.1016/J.ASOC.2009.12.025>
- Karaboga, N. (2009). A new design method based on artificial bee colony algorithm for digital IIR filters. *Journal of the Franklin Institute*, 346(4), 328–348. <https://doi.org/10.1016/J.JFRANKLIN.2008.11.003>
- Karkavitsas, G., & Rangoussi, M. (2005). Object Localization in Medical Images Using Genetic Algorithms. *International Journal of Information and Communication Engineering*.
- Karkavitsas, G. V., & Rangoussi, M. (2008). Object localization in medical images using genetic algorithms. Retrieved from <https://www.semanticscholar.org/paper/Object-localization-in-medical-images-using-genetic-Karkavitsas-Rangoussi/17fe39623255361c75d08204dc5ee2cf5a5593ac>
- Karmakar, G. C., & Dooley, L. S. (2002). A generic fuzzy rule based image segmentation algorithm. *Pattern Recognition Letters*, 23(10), 1215–1227. [https://doi.org/10.1016/S0167-8655\(02\)00069-7](https://doi.org/10.1016/S0167-8655(02)00069-7)
- Kashki, M., Abdel-Magid, Y. L., & Abido, M. A. (2008). A Reinforcement Learning Automata Optimization Approach for Optimum Tuning of PID Controller in AVR System. In *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence* (pp. 684–692). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-85984-0_82
- Kennedy, J., & Eberhart, R. (1995a). Distribution Network Reconfiguration for Load Balancing Using Binary Particle Swarm Optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*. <https://doi.org/10.1109/ICNN.1995.488968>
- Kennedy, J., & Eberhart, R. C. (1995b). Particle swarm optimization. *Neural Networks, 1995. Proceedings., IEEE International Conference On*, 4, 1942–1948 vol.4. <https://doi.org/10.1109/ICNN.1995.488968>
- Kermani, S., Samadzadehaghdam, N., & EtehadTavakol, M. (2015). Automatic color segmentation of breast infrared images using a Gaussian mixture model. *Optik*, 126(21), 3288–3294. <https://doi.org/10.1016/j.ijleo.2015.08.007>
- Keyserlingk, J. R., Ahlgren, P. D., Yu, E., & Belliveau, N. (1998). Infrared Imaging of the Breast: Initial Reappraisal Using High-Resolution Digital Technology in 100 Successive Cases of Stage I and II Breast Cancer. *The Breast Journal*.
- Kicinger, R., Arciszewski, T., & De Jong, K. (n.d.). *Evolutionary Computation and Structural Design: a Survey of the State of the Art* *. Retrieved from <http://www.kicinger.com/publications/pdf/KicingerCAS2005.pdf>
- Kim, D.-S., Lee, W.-H., & Kweon, I.-S. (2004). Automatic edge detection using 3×3 ideal binary pixel patterns and fuzzy-based edge thresholding. *Pattern Recognition Letters*, 25(1), 101–106. <https://doi.org/10.1016/J.PATREC.2003.09.010>
- Kim, J. H., Geem, Z. W., & Kim, E. S. (2001). PARAMETER ESTIMATION OF THE NONLINEAR MUSKINGUM MODEL USING HARMONY SEARCH. *Journal of the American Water Resources Association*, 37(5), 1131–1138. <https://doi.org/10.1111/j.1752-1688.2001.tb03627.x>
- Kim, S. G., & Seo, Y. G. (2013). A TRUS prostate segmentation using Gabor texture features and

- snake-like contour. *Journal of Information Processing Systems*, 9(1), 103–116. <https://doi.org/10.3745/JIPS.2013.9.1.103>
- Kitchen, L., & Rosenfeld, A. (1982). Gray-level corner detection. *Pattern Recognition Letters*, 1(2), 95–102. [https://doi.org/10.1016/0167-8655\(82\)90020-4](https://doi.org/10.1016/0167-8655(82)90020-4)
- Kohonen, T. (1989). *Self-Organization and Associative Memory* (Vol. 8). Berlin, Heidelberg: Springer Berlin Heidelberg. <https://doi.org/10.1007/978-3-642-88163-3>
- Kolb, T. M., Lichy, J., & Newhouse, J. H. (2002). Comparison of the Performance of Screening Mammography, Physical Examination, and Breast US and Evaluation of Factors that Influence Them: An Analysis of 27,825 Patient Evaluations. *Radiology*, 225(1), 165–175. <https://doi.org/10.1148/radiol.2251011667>
- Krattenthaler, W., Mayer, K. J., & Zeiller, M. (n.d.). Point correlation: a reduced-cost template matching technique. In *Proceedings of 1st International Conference on Image Processing* (Vol. 1, pp. 208–212). IEEE Comput. Soc. Press. <https://doi.org/10.1109/ICIP.1994.413305>
- Kuhl, C. K., Schrading, S., Leutner, C. C., Morakkabati-Spitz, N., Wardelmann, E., Fimmers, R., ... Schild, H. H. (2005). Mammography, breast ultrasound, and magnetic resonance imaging for surveillance of women at high familial risk for breast cancer. *Journal of Clinical Oncology*, 23(33), 8469–8476. <https://doi.org/10.1200/JCO.2004.00.4960>
- Laboratoire de Recherche en Informatique. (n.d.). Retrieved January 23, 2019, from <https://www.lri.fr/~hansen/cmaesintro.html>
- Lai-Man Po, & Wing-Chung Ma. (1996). A novel four-step search algorithm for fast block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3), 313–317. <https://doi.org/10.1109/76.499840>
- Lai, C.-C. (2006). A Novel Image Segmentation Approach Based on Particle Swarm Optimization. *IEICE Trans. Fundamentals*, E89-A(1), 324–327. <https://doi.org/10.1093/ietfec/e89-a.1.324>
- Lai, C.-C., & Tseng, D.-C. (2001). An optimal L-filter for reducing blocking artifacts using genetic algorithms. *Signal Processing*, 81(7), 1525–1535. [https://doi.org/10.1016/S0165-1684\(01\)00050-0](https://doi.org/10.1016/S0165-1684(01)00050-0)
- Landi, G., & Piccolomini, E. L. (2012). An efficient method for nonnegatively constrained Total Variation-based denoising of medical images corrupted by Poisson noise. *Computerized Medical Imaging and Graphics*, 36(1), 38–46. <https://doi.org/10.1016/J.COMPMEIMAG.2011.07.002>
- Langdon, W. B., & Poli, R. (2002). *Foundations of Genetic Programming*. Berlin, Heidelberg: Springer Berlin Heidelberg. <https://doi.org/10.1007/978-3-662-04726-2>
- Le Hgarat-Masclé, S., Kallel, A., & Descombes, X. (2007). Ant Colony Optimization for Image Regularization Based on a Nonstationary Markov Modeling. *IEEE Transactions on Image Processing*, 16(3), 865–878. <https://doi.org/10.1109/TIP.2007.891150>
- Lee, C.-H., & Chang, F.-K. (2010). Fractional-order PID controller optimization via improved electromagnetism-like algorithm. *Expert Systems with Applications*, 37(12), 8871–8878. <https://doi.org/10.1016/J.ESWA.2010.06.009>
- Lee, C.-H., Chang, F.-K., Kuo, C.-T., & Chang, H.-H. (2012). A hybrid of electromagnetism-like mechanism and back-propagation algorithms for recurrent neural fuzzy systems design. *International Journal of Systems Science*, 43(2), 231–247. <https://doi.org/10.1080/00207721.2010.488758>
- Lee, H., & Chen, Y. P. P. (2015). Image based computer aided diagnosis system for cancer detection. *Expert Systems with Applications*, 42(12), 5356–5365. <https://doi.org/10.1016/j.eswa.2015.02.005>
- Lee, K. S., & Geem, Z. W. (2004). A new structural optimization method based on the harmony search algorithm. *Computers & Structures*, 82(9–10), 781–798. <https://doi.org/10.1016/J.COMPSTRUC.2004.01.002>
- Lee, K. S., & Geem, Z. W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering*, 194(36–38), 3902–3933. <https://doi.org/10.1016/J.CMA.2004.09.007>
- Lee, K. S., Geem, Z. W., Lee, S., & Bae, K. (2005). The harmony search heuristic algorithm for

- discrete structural optimization. *Engineering Optimization*, 37(7), 663–684. <https://doi.org/10.1080/03052150500211895>
- Lessmann, S., Caserta, M., & Arango, I. M. (2011). Tuning metaheuristics: A data mining based approach for particle swarm optimization. *Expert Systems with Applications*, 38(10), 12826–12838. <https://doi.org/10.1016/J.ESWA.2011.04.075>
- Li, M. D., Zhao, H., Weng, X. W., & Han, T. (2016). A novel nature-inspired algorithm for optimization: Virus colony search. *Advances in Engineering Software*, 92, 65–88. <https://doi.org/10.1016/J.ADVENGSOFT.2015.11.004>
- Li, X., Xiao, N., Claramunt, C., & Lin, H. (2011). Initialization strategies to enhancing the performance of genetic algorithms for the p-median problem. *Computers & Industrial Engineering*, 61(4), 1024–1034. <https://doi.org/10.1016/J.CIE.2011.06.015>
- Liang, J. J., Qu, B. Y., Suganthan, P. N., & Chen, Q. (2014). *Problem Definitions and Evaluation Criteria for the CEC 2015 Competition on Learning-based Real-Parameter Single Objective Optimization*. Retrieved from http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2015/CEC2015.htm
- Liang, L. R., & Looney, C. G. (2003). Competitive fuzzy edge detection. *Applied Soft Computing*, 3(2), 123–137. [https://doi.org/10.1016/S1568-4946\(03\)00008-5](https://doi.org/10.1016/S1568-4946(03)00008-5)
- Liao, T. W., Egbelu, P. J., Sarker, B. R., & Leu, S. S. (2011). Metaheuristics for project and construction management – A state-of-the-art review. *Automation in Construction*, 20(5), 491–505. <https://doi.org/10.1016/j.autcon.2010.12.006>
- Liaw, Y.-C., Lai, J. Z. ., & Hong, Z.-C. (2009). Fast block matching using prediction and rejection criteria. *Signal Processing*, 89(6), 1115–1120. <https://doi.org/10.1016/J.SIGPRO.2008.12.012>
- Lim, D., Yaochu Jin, Yew-Soon Ong, & Sendhoff, B. (2010a). Generalizing Surrogate-Assisted Evolutionary Computation. *IEEE Transactions on Evolutionary Computation*, 14(3), 329–355. <https://doi.org/10.1109/TEVC.2009.2027359>
- Lim, D., Yaochu Jin, Yew-Soon Ong, & Sendhoff, B. (2010b). Generalizing Surrogate-Assisted Evolutionary Computation. *IEEE Transactions on Evolutionary Computation*, 14(3), 329–355. <https://doi.org/10.1109/TEVC.2009.2027359>
- Lin Zhang, Lei Zhang, XuanqinMou, D. Z. (2011). FSIM : A Feature Similarity Index for Image. *IEEE Transactions on Image Processing*, 20(8), 2378–2386.
- Liu, F., Duan, H., & Deng, Y. (2012). A chaotic quantum-behaved particle swarm optimization based on lateral inhibition for image matching. *Optik*, 123(21), 1955–1960. <https://doi.org/10.1016/J.IJLEO.2011.09.052>
- Londhe, V. Y., & Bhasin, B. (2018). Artificial intelligence and its potential in oncology. *Drug Discovery Today*. <https://doi.org/10.1016/j.drudis.2018.10.005>
- Loupias, E., & Sebe, N. (2000). Wavelet-Based Salient Points: Applications to Image Retrieval Using Color and Texture Features (pp. 223–232). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-40053-2_20
- Lowe, D. G. (1985). *Perceptual Organization and Visual Recognition*. Springer US. Retrieved from <https://books.google.de/books?hl=es&lr=&id=h0jTBwAAQBAJ&oi=fnd&pg=PR7&dq=Perceptual+Organization+and+Visual+Recognition,+Kluwer+Academic+Publishers&ots=r87GY3I5E-&sig=LVAYQ3w1P2DrwGyBe0EpvMidMic#v=onepage&q=Perceptual+Organization+and+Visual+Recognition%2C+Kluwer+Academic+Publishers&f=false>
- Lu, H., Li, Y., Chen, M., Kim, H., & Serikawa, S. (2017). Brain Intelligence: Go Beyond Artificial Intelligence. Retrieved from <http://arxiv.org/abs/1706.01040>
- Lu, P., Chen, S., & Zheng, Y. (2012). Artificial Intelligence in Civil Engineering. *Mathematical Problems in Engineering*, 2012, 1–22. <https://doi.org/10.1155/2012/145974>
- Lu, W., & Tan, J. (2008). Detection of incomplete ellipse in images with strong noise by iterative randomized Hough transform (IRHT). *Pattern Recognition*, 41(4), 1268–1279. <https://doi.org/10.1016/J.PATCOG.2007.09.006>
- Luo, C., Zhang, S.-L., Wang, C., & Jiang, Z. (2011). A metamodel-assisted evolutionary algorithm

- for expensive optimization. *Journal of Computational and Applied Mathematics*, 236(5), 759–764. <https://doi.org/10.1016/J.CAM.2011.05.047>
- Luque, C., Valls, J. M., & Isasi, P. (2011). Time series prediction evolving Voronoi regions. *Applied Intelligence*, 34(1), 116–126. <https://doi.org/10.1007/s10489-009-0184-9>
- Lurng-Kuo Liu, & Feig, E. (1996). A block-based gradient descent search algorithm for block motion estimation in video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(4), 419–422. <https://doi.org/10.1109/76.510936>
- Lutton, E., & Martinez, P. (n.d.). A genetic algorithm for the detection of 2D geometric primitives in images. In *Proceedings of 12th International Conference on Pattern Recognition* (Vol. 1, pp. 526–528). IEEE Comput. Soc. Press. <https://doi.org/10.1109/ICPR.1994.576345>
- Ma, J., Xu, L., & Jordan, M. I. (2000). Asymptotic Convergence Rate of the EM Algorithm for Gaussian Mixtures. *Neural Computation*, 12(12), 2881–2907. <https://doi.org/10.1162/089976600300014764>
- Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation*, 188(2), 1567–1579. <https://doi.org/10.1016/J.AMC.2006.11.033>
- Mahootchi, M., Tizhoosh, H. R., & Ponnambalam, K. (2007). Opposition-Based Reinforcement Learning in the Management of Water Resources. In *2007 IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning* (pp. 217–224). IEEE. <https://doi.org/10.1109/ADPRL.2007.368191>
- MAMDANI, E. H., & ASSILIAN, S. (1999). An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. *International Journal of Human-Computer Studies*, 51(2), 135–147. <https://doi.org/10.1006/IJHC.1973.0303>
- Manh, H. T., & Lee, G. (2013). Small object segmentation based on visual saliency in natural images. *Journal of Information Processing Systems*, 9(4), 592–601. <https://doi.org/10.3745/JIPS.2013.9.4.592>
- Mardani, A., Jusoh, A., & Zavadskas, E. K. (2015). Fuzzy multiple criteria decision-making techniques and applications - Two decades review from 1994 to 2014. *Expert Systems with Applications*. <https://doi.org/10.1016/j.eswa.2015.01.003>
- Mayer, D. G., Kinghorn, B. P., & Archer, A. A. (2005). Differential evolution – an easy and efficient evolutionary algorithm for model optimisation. *Agricultural Systems*, 83(3), 315–328. <https://doi.org/10.1016/J.AGSY.2004.05.002>
- McKenna, S. J., Raja, Y., & Gong, S. (1999). Tracking colour objects using adaptive mixture models. *Image and Vision Computing*, 17(3–4), 225–231. [https://doi.org/10.1016/S0262-8856\(98\)00104-8](https://doi.org/10.1016/S0262-8856(98)00104-8)
- Meng, Z., & Pan, J.-S. (2016). Monkey King Evolution: A new memetic evolutionary algorithm and its application in vehicle fuel consumption optimization. *Knowledge-Based Systems*, 97, 144–157. <https://doi.org/10.1016/J.KNOSYS.2016.01.009>
- Merrikh-Bayat, F. (2015). The runner-root algorithm: A metaheuristic for solving unimodal and multimodal optimization problems inspired by runners and roots of plants in nature. *Applied Soft Computing Journal*, 33, 292–303. <https://doi.org/10.1016/j.asoc.2015.04.048>
- Mirjalili, S. (2016). Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications*, 27(4), 1053–1073. <https://doi.org/10.1007/s00521-015-1920-1>
- Mishra, D., Bose, I., De, U. C., & Das, M. (2015). Medical Image Thresholding Using Particle Swarm Optimization. In L. Jain, S. Patnaik, & N. Ichalkaranje (Eds.), *Advances in Intelligent Systems and Computing* (Vol. 308 AISC, pp. 379–383). New Delhi: Springer. <https://doi.org/10.1007/978-81-322-2012-1>
- Moghbel, M., & Mashohor, S. (2013). A review of computer assisted detection/diagnosis (CAD) in breast thermography for breast cancer detection. *Artificial Intelligence Review*, 39(4), 305–313. <https://doi.org/10.1007/s10462-011-9274-2>
- Mokhtarian, F., & Mohanna, F. (2006). Performance evaluation of corner detectors using consistency and accuracy measures. *Computer Vision and Image Understanding*, 102(1),

- 81–94. <https://doi.org/10.1016/J.CVIU.2005.11.001>
- Montero, E., & Riff, M.-C. (2011). On-the-fly calibrating strategies for evolutionary algorithms. *Information Sciences*, *181*(3), 552–566. <https://doi.org/10.1016/J.INS.2010.09.016>
- Moravec, H. P. (1997). Towards Automatic Visual Obstacle Avoidance. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence* (p. 584). Retrieved from <https://www.semanticscholar.org/paper/Towards-Automatic-Visual-Obstacle-Avoidance-Moravec/3b4a713d67a0a4f7099bdc40d818311b8827b8b7>
- Motta, L., Conci, A., Lima, R., Diniz, E., Luís, S. (2010). Automatic segmentation on thermograms in order to aid diagnosis and 2D modeling. *Proceedings of 10th Workshop Em Informática Médica*, 1610–1619.
- Muammar, H., & Nixon, M. (n.d.). Approaches to extending the Hough transform. In *International Conference on Acoustics, Speech, and Signal Processing* (pp. 1556–1559). IEEE. <https://doi.org/10.1109/ICASSP.1989.266739>
- Muammar, H., & Nixon, M. (1989). Approaches to extending the Hough transform. *International Conference on Acoustics, Speech, and Signal Processing*,. <https://doi.org/10.1109/ICASSP.1989.266739>
- Naderi, B., Tavakkoli-Moghaddam, R., & Khalili, M. (2010). Electromagnetism-like mechanism and simulated annealing algorithms for flowshop scheduling problems minimizing the total weighted tardiness and makespan. *Knowledge-Based Systems*, *23*(2), 77–85. <https://doi.org/10.1016/J.KNOSYS.2009.06.002>
- Najim, K., & Poznyak, A. S. (1994). *Learning automata: theory and applications*. Pergamon. Retrieved from <https://dl.acm.org/citation.cfm?id=191277>
- Nanda, S. J., & Panda, G. (2014a). A survey on nature inspired metaheuristic algorithms for partitional clustering. *Swarm and Evolutionary Computation*. <https://doi.org/10.1016/j.swevo.2013.11.003>
- Nanda, S. J., & Panda, G. (2014b). A survey on nature inspired metaheuristic algorithms for partitional clustering. *Swarm and Evolutionary Computation*, *16*, 1–18. <https://doi.org/10.1016/J.SWEVO.2013.11.003>
- Narendra, K. S., & Thathachar, M. A. L. (1989). *Learning automata: an introduction*. Prentice Hall. Retrieved from <https://dl.acm.org/citation.cfm?id=64802>
- Ng, E. Y. K. (2009). A review of thermography as promising non-invasive detection modality for breast tumor. *International Journal of Thermal Sciences*, *48*(5), 849–859. <https://doi.org/10.1016/j.ijthermalsci.2008.06.015>
- Ng, E. Y. K., & Sud, N. M. (2001). Numerical computation as a tool to aid thermographic interpretation, *25*(2), 53–60.
- Ng, E. Y., & Sudharsan, N. M. (2015). Effect of blood flow , tumour and cold stress in a female breast : a novel time-accurate computer simulation, *215*, 393–404.
- Nikolaev, D. P., & Nikolayev, P. P. (2004). Linear color segmentation and its implementation. *Computer Vision and Image Understanding*, *94*(1–3), 115–139. <https://doi.org/10.1016/J.CVIU.2003.10.012>
- Nisar, H., Malik, A. S., & Choi, T.-S. (2012). Content adaptive fast motion estimation based on spatio-temporal homogeneity analysis and motion classification. *Pattern Recognition Letters*, *33*(1), 52–61. <https://doi.org/10.1016/J.PATREC.2011.09.015>
- Novák, V., Hurtík, P., Habiballa, H., & Štepnička, M. (2015). Recognition of damaged letters based on mathematical fuzzy logic analysis. *Journal of Applied Logic*, *13*(2), 94–104. <https://doi.org/10.1016/J.JAL.2014.11.003>
- Nummiaro, K., Koller-Meier, E., & Van Gool, L. (2002). An adaptive color-based particle filter. In *1st International Workshop on Generative-Model-Based Vision* (pp. 353–358). [https://doi.org/10.1016/S0262-8856\(02\)00129-4](https://doi.org/10.1016/S0262-8856(02)00129-4)
- Oh, I.-S., Lee, J.-S., & Moon, B.-R. (2004). Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(11), 1424–1437. <https://doi.org/10.1109/TPAMI.2004.105>
- Ohlander, R., Price, K., & Reddy, D. R. (1978). Picture segmentation using a recursive region

- splitting method. *Computer Graphics and Image Processing*, 8(3), 313–333. [https://doi.org/10.1016/0146-664X\(78\)90060-6](https://doi.org/10.1016/0146-664X(78)90060-6)
- Oliva, D., Hinojosa, S., Elaziz, M. A., & Ortega-Sánchez, N. (2018). Context based image segmentation using antlion optimization and sine cosine algorithm. *Multimedia Tools and Applications*. <https://doi.org/10.1007/s11042-018-5815-x>
- Oliva, D., Hinojosa, S., Osuna-Enciso, V., Cuevas, E., Pérez-Cisneros, M., & Sanchez-Ante, G. (2017). Image segmentation by minimum cross entropy using evolutionary methods. *Soft Computing*. <https://doi.org/10.1007/s00500-017-2794-1>
- Olivas, F., Valdez, F., Castillo, O., & Melin, P. (2016). Dynamic parameter adaptation in particle swarm optimization using interval type-2 fuzzy logic. *Soft Computing*, 20(3), 1057–1070. <https://doi.org/10.1007/s00500-014-1567-3>
- Olsson, R. K., Petersen, K. B., & Lehn-Schiøler, T. (2007). State-Space Models: From the EM Algorithm to a Gradient Approach. *Neural Computation*, 19(4), 1097–1111. <https://doi.org/10.1162/neco.2007.19.4.1097>
- Omid, M., Lashgari, M., Mobli, H., Alimardani, R., Mohtasebi, S., & Hesamifard, R. (2010). Design of fuzzy logic control system incorporating human expert knowledge for combine harvester. *Expert Systems with Applications*, 37(10), 7080–7085. <https://doi.org/10.1016/J.ESWA.2010.03.010>
- Omran, M. G. H., & Mahdavi, M. (2008). Global-best harmony search. *Applied Mathematics and Computation*, 198(2), 643–656. <https://doi.org/10.1016/J.AMC.2007.09.004>
- Ong, S. ., Yeo, N. ., Lee, K. ., Venkatesh, Y. ., & Cao, D. . (2002). Segmentation of color images using a two-stage self-organizing network. *Image and Vision Computing*, 20(4), 279–289. [https://doi.org/10.1016/S0262-8856\(02\)00021-5](https://doi.org/10.1016/S0262-8856(02)00021-5)
- Ong, Y. S., Lum, K. Y., & Nair, P. B. (2008a). Hybrid evolutionary algorithm with Hermite radial basis function interpolants for computationally expensive adjoint solvers. *Computational Optimization and Applications*, 39(1), 97–119. <https://doi.org/10.1007/s10589-007-9065-5>
- Ong, Y. S., Lum, K. Y., & Nair, P. B. (2008b). Hybrid evolutionary algorithm with Hermite radial basis function interpolants for computationally expensive adjoint solvers. *Computational Optimization and Applications*, 39(1), 97–119. <https://doi.org/10.1007/s10589-007-9065-5>
- Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1), 62–66. <https://doi.org/10.1109/TSMC.1979.4310076>
- PA, S. D., Naik Nagappa, A., Udupa, N., & Mathew, S. N. (2015). Effectiveness of a Planned Teaching Program on Improving the Knowledge on Warning Signs, Risk Factors and Early Detection Methods. *Indian Journal of Pharmacy and Pharmacology*, 2(1), 6–9.
- Pal, N. R., & Pal, S. K. (1993). A review on image segmentation techniques. *Pattern Recognition*, 26(9), 1277–1294. [https://doi.org/10.1016/0031-3203\(93\)90135-J](https://doi.org/10.1016/0031-3203(93)90135-J)
- Pal, S. K., Ghosh, A., & Kundu, M. K. (Eds.). (2000). *Soft Computing for Image Processing* (Vol. 42). Heidelberg: Physica-Verlag HD. <https://doi.org/10.1007/978-3-7908-1858-1>
- Pan, Q.-K., Fatih Tasgetiren, M., Suganthan, P. N., & Chua, T. J. (2011). A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. *Information Sciences*, 181(12), 2455–2468. <https://doi.org/10.1016/J.INS.2009.12.025>
- Papakostas, G. A., Hatzimichailidis, A. G., & Kaburlasos, V. G. (2013). Distance and similarity measures between intuitionistic fuzzy sets: A comparative analysis from a pattern recognition point of view. *Pattern Recognition Letters*, 34(14), 1609–1622. <https://doi.org/10.1016/J.PATREC.2013.05.015>
- Park, H., Amari, S.-I., & Fukumizu, K. (2000). Adaptive natural gradient learning algorithms for various stochastic models. *Neural Networks*, 13(7), 755–764. [https://doi.org/10.1016/S0893-6080\(00\)00051-4](https://doi.org/10.1016/S0893-6080(00)00051-4)
- Park, H., & Ozeki, T. (2009). Singularity and Slow Convergence of the EM algorithm for Gaussian Mixtures. *Neural Processing Letters*, 29(1), 45–59. <https://doi.org/10.1007/s11063-009-9094-4>
- Park, S. H., Yun, I. D., & Lee, S. U. (1998). Color image segmentation based on 3-D clustering:

- morphological approach. *Pattern Recognition*, 31(8), 1061–1076. [https://doi.org/10.1016/S0031-3203\(97\)00116-7](https://doi.org/10.1016/S0031-3203(97)00116-7)
- Patnaik, S., & Zhong, B. (n.d.). *Soft computing techniques in engineering applications*.
- Patra, S., Gautam, R., & Singla, A. (2014). A novel context sensitive multilevel thresholding for image segmentation. *Applied Soft Computing Journal*, 23, 122–127. <https://doi.org/10.1016/j.asoc.2014.06.016>
- Pérez, P., Hue, C., Vermaak, J., & Gangnet, M. (2002). Color-Based Probabilistic Tracking. In A. Heyden (Ed.), *European Conference on Computer Vision* (pp. 661–675). Springer-Verlag. Retrieved from <https://www.irisa.fr/aspi/legland/ensta/ref/perez02a.pdf>
- Peura, M., Peura, M., & Iivarinen, J. (1997). Efficiency of Simple Shape Descriptors. *IN ASPECTS OF VISUAL FORM*, 443–451. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.30.9018>
- Pietikainen, M., Nieminen, S., Marszalec, E., & Ojala, T. (1996). Accurate color discrimination with classification based on feature distributions. In *Proceedings of 13th International Conference on Pattern Recognition* (pp. 833–838 vol.3). IEEE. <https://doi.org/10.1109/ICPR.1996.547285>
- Port, E. R., Park, A., Borgen, P. I., Morris, E., & Montgomery, L. L. (2007). Results of MRI screening for breast cancer in high-risk patients with LCIS and atypical hyperplasia. *Annals of Surgical Oncology*, 14(3), 1051–1057. <https://doi.org/10.1245/s10434-006-9195-5>
- Price, K. V., Storn, R. M., & Lampinen, J. A. (2005). *Differential evolution : a practical approach to global optimization*. Springer.
- Qi, H., & Diakides, N. a. (2003). Thermal infrared imaging in early breast cancer detection—a survey of recent research. *Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE Cat. No.03CH37439)*, 2, 1109–1112. <https://doi.org/10.1109/IEMBS.2003.1279442>
- Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. A. (2007). A novel population initialization method for accelerating evolutionary algorithms. *Computers & Mathematics with Applications*, 53(10), 1605–1614. <https://doi.org/10.1016/J.CAMWA.2006.07.013>
- Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. A. (2008). Opposition-Based Differential Evolution. *IEEE Transactions on Evolutionary Computation*, 12(1), 64–79. <https://doi.org/10.1109/TEVC.2007.894200>
- Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. A. (2008). Opposition versus randomness in soft computing techniques. *Applied Soft Computing*, 8(2), 906–918. <https://doi.org/10.1016/J.ASOC.2007.07.010>
- Raja, Y., McKenna, S. J., & Shaogang Gong. (n.d.). Tracking and segmenting people in varying lighting conditions using colour. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 228–233). IEEE Comput. Soc. <https://doi.org/10.1109/AFGR.1998.670953>
- Raju, G., & Nair, M. S. (2014). A fast and efficient color image enhancement method based on fuzzy-logic and histogram. *AEU - International Journal of Electronics and Communications*, 68(3), 237–243. <https://doi.org/10.1016/J.AEUE.2013.08.015>
- Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2011). Filter modeling using gravitational search algorithm. *Engineering Applications of Artificial Intelligence*, 24(1), 117–122. <https://doi.org/10.1016/J.ENGAPPAI.2010.05.007>
- Ratle, A. (2001). Kriging as a surrogate fitness landscape in evolutionary optimization. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 15(1), 37–49. Retrieved from <https://www.cambridge.org/core/journals/ai-edam/article/kriging-as-a-surrogate-fitness-landscape-in-evolutionary-optimization/B8F8E38290764E4E4BABA4794A0A53EE>
- Rattarangsi, A., & Chin, R. T. (1992). Scale-based detection of corners of planar curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(4), 430–449. <https://doi.org/10.1109/34.126805>
- Redner, R. A., & Walker, H. F. (1984). Mixture Densities, Maximum Likelihood and the EM Algorithm. *SIAM Review*, 26(2), 195–239. Retrieved from

- https://www.jstor.org/stable/2030064?seq=1#metadata_info_tab_contents
- Reoxiang Li, Bing Zeng, & Liou, M. L. (1994). A new three-step search algorithm for block motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, 4(4), 438–442. <https://doi.org/10.1109/76.313138>
- Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH Computer Graphics*, 21(4), 25–34. <https://doi.org/10.1145/37402.37406>
- Risinger, L., & Kaikhah, K. (2008). Motion detection and object tracking with discrete leaky integrate-and-fire neurons. *Applied Intelligence*, 29(3), 248–262. <https://doi.org/10.1007/s10489-007-0092-9>
- Rocha, A. M. A. C., & Fernandes, E. M. G. P. (n.d.). *Numerical Experiments with a Population Shrinking Strategy within a Electromagnetism-like Algorithm*. Retrieved from <https://core.ac.uk/download/pdf/55610528.pdf>
- Rocha, A. M. A. C., & Fernandes, E. M. G. P. (2009a). Hybridizing the electromagnetism-like algorithm with descent search for solving engineering design problems. *International Journal of Computer Mathematics*, 86(10–11), 1932–1946. <https://doi.org/10.1080/00207160902971533>
- Rocha, A. M. A. C., & Fernandes, E. M. G. P. (2009b). Modified movement force vector in an electromagnetism-like mechanism for global optimization. *Optimization Methods and Software*, 24(2), 253–270. <https://doi.org/10.1080/10556780802525356>
- Rocha, A. M. A. C., & Fernandes, E. M. G. P. (2011). Numerical study of augmented Lagrangian algorithms for constrained global optimization. *Optimization*, 60(10–11), 1359–1378. <https://doi.org/10.1080/02331934.2011.628671>
- Rosenfeld, A., & Johnston, E. (1973). Angle Detection on Digital Curves. *IEEE Transactions on Computers*, C-22(9), 875–878. <https://doi.org/10.1109/TC.1973.5009188>
- Rosenfeld, A., & VanderBrug, G. J. (1977). Coarse-Fine Template Matching. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(2), 104–107. <https://doi.org/10.1109/TSMC.1977.4309663>
- Rosin, P. L. (1999). Further Five-Point Fit Ellipse Fitting. *Graphical Models and Image Processing*, 61(5), 245–259. <https://doi.org/10.1006/GMIP.1999.0500>
- Roth, G., & Levine, M. D. (n.d.). Geometric primitive extraction using a genetic algorithm. In *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 640–643). IEEE Comput. Soc. Press. <https://doi.org/10.1109/CVPR.1992.223120>
- Russo, F. (1999). FIRE operators for image processing. *Fuzzy Sets and Systems*, 103(2), 265–275. [https://doi.org/10.1016/S0165-0114\(98\)00226-7](https://doi.org/10.1016/S0165-0114(98)00226-7)
- Russo, F. (2004). Impulse noise cancellation in image data using a two-output nonlinear filter. *Measurement*, 36(3–4), 205–213. <https://doi.org/10.1016/J.MEASUREMENT.2004.09.002>
- Saha, A., Mukherjee, J., & Sural, S. (2008). New pixel-decimation patterns for block matching in motion estimation. *Signal Processing: Image Communication*, 23(10), 725–738. <https://doi.org/10.1016/J.IMAGE.2008.08.004>
- Saha, A., Mukherjee, J., & Sural, S. (2011). A neighborhood elimination approach for block matching in motion estimation. *Signal Processing: Image Communication*, 26(8–9), 438–454. <https://doi.org/10.1016/J.IMAGE.2011.06.002>
- Sahoo, P. ., Soltani, S., & Wong, A. K. . (1988). A survey of thresholding techniques. *Computer Vision, Graphics, and Image Processing*, 41(2), 233–260. [https://doi.org/10.1016/0734-189X\(88\)90022-9](https://doi.org/10.1016/0734-189X(88)90022-9)
- Saka, M. P., & Geem, Z. W. (2013). Mathematical and Metaheuristic Applications in Design Optimization of Steel Frame Structures: An Extensive Review. *Mathematical Problems in Engineering*, 2013, 1–33. <https://doi.org/10.1155/2013/271031>
- Salehi, H., & Burgueño, R. (2018). Emerging artificial intelligence methods in structural engineering. <https://doi.org/10.1016/j.engstruct.2018.05.084>
- Salmond, D. J. (1990). Mixture reduction algorithms for target tracking in clutter. In *Signal and Data Processing of Small Targets 1990* (Vol. 1305, p. 37). SPIE.

- <https://doi.org/10.1117/12.2321784>
- Santamaría, J., Cerdón, O., Damas, S., García-Torres, J. M., & Quirin, A. (2009). Performance evaluation of memetic approaches in 3D reconstruction of forensic objects. *Soft Computing*, 13(8–9), 883–904. <https://doi.org/10.1007/s00500-008-0351-7>
- Sathish, D., Kamath, S., Prasad, K., Kadavigere, R., & Martis, R. J. (2017). Asymmetry analysis of breast thermograms using automated segmentation and texture features. *Signal, Image and Video Processing*, 11(4), 745–752. <https://doi.org/10.1007/s11760-016-1018-y>
- Scales, N., Herry, C., & Frize, M. (2004). Automated image segmentation for breast analysis using infrared images. *Conference Proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, 3, 1737–1740. <https://doi.org/10.1109/IEMBS.2004.1403521>
- Schaefer, G., & Nakashima, T. (2007). Breast Cancer Classification Using Statistical Features and Fuzzy Classification of Thermograms I, PLR (kC. *Breast*.
- Schmid, C., Mohr, R., & Bauckhage, C. (2000). *Evaluation of Interest Point Detectors. International Journal of Computer Vision* (Vol. 37). Retrieved from <https://link.springer.com/content/pdf/10.1023/A:1008199403446.pdf>
- Scholl, I., Aach, T., Deserno, T. M., & Kuhlen, T. (2011). Challenges of medical image processing. *Computer Science - Research and Development*, 26(1–2), 5–13. <https://doi.org/10.1007/s00450-010-0146-9>
- Sezgin, M., & Sankur, B. (n.d.). Selection of thresholding methods for nondestructive testing applications. In *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205)* (Vol. 2, pp. 764–767). IEEE. <https://doi.org/10.1109/ICIP.2001.958231>
- Sezgin, M., & Taşaltın, R. (2000). A new dichotomization technique to multilevel thresholding devoted to inspection applications. *Pattern Recognition Letters*, 21(2), 151–161. [https://doi.org/10.1016/S0167-8655\(99\)00142-7](https://doi.org/10.1016/S0167-8655(99)00142-7)
- Shahin, M. A. (2013). Artificial Intelligence in Geotechnical Engineering. In *Metaheuristics in Water, Geotechnical and Transport Engineering* (pp. 169–204). Elsevier. <https://doi.org/10.1016/B978-0-12-398296-4.00008-8>
- Shahin, M. A. (2016). State-of-the-art review of some artificial intelligence applications in pile foundations. *Geoscience Frontiers*. <https://doi.org/10.1016/j.gsf.2014.10.002>
- Shaked, D., Yaron, O., & Kiryati, N. (1996a). Deriving Stopping Rules for the Probabilistic Hough Transform by Sequential Analysis. *Computer Vision and Image Understanding*. <https://doi.org/10.1006/cviu.1996.0038>
- Shaked, D., Yaron, O., & Kiryati, N. (1996b). Deriving Stopping Rules for the Probabilistic Hough Transform by Sequential Analysis. *Computer Vision and Image Understanding*, 63(3), 512–526. <https://doi.org/10.1006/CVIU.1996.0038>
- Shan Zhu, & Kai-Kuang Ma. (n.d.). A new diamond search algorithm for fast block matching motion estimation. In *Proceedings of ICICS, 1997 International Conference on Information, Communications and Signal Processing. Theme: Trends in Information Systems Engineering and Wireless Multimedia Communications (Cat. No.97TH8237)* (Vol. 1, pp. 292–296). IEEE. <https://doi.org/10.1109/ICICS.1997.647106>
- Shaw, B., Mukherjee, V., & Ghoshal, S. P. (2012). A novel opposition-based gravitational search algorithm for combined economic and emission dispatch problems of power systems. *International Journal of Electrical Power & Energy Systems*, 35(1), 21–33. <https://doi.org/10.1016/J.IJEPES.2011.08.012>
- Shehab, A., Elhoseny, M., Muhammad, K., Sangaiah, A. K., Yang, P., Huang, H., & Hou, G. (2018). Secure and robust fragile watermarking scheme for medical images. In *IEEE Access* (Vol. 6, pp. 10269–10278). <https://doi.org/10.1109/ACCESS.2018.2799240>
- Shen Yuong Wong, Keem Siah Yap, Hwa Jen Yap, Shing Chiang Tan, & Siow Wee Chang. (2015). On Equivalence of FIS and ELM for Interpretable Rule-Based Knowledge Representation. *IEEE Transactions on Neural Networks and Learning Systems*, 26(7), 1417–1430. <https://doi.org/10.1109/TNNLS.2014.2341655>
- Shitong, W., Chung, K. F. L., & Duan, F. (2007a). Applying the improved fuzzy cellular neural

- network IFCNN to white blood cell detection. *Neurocomputing*. <https://doi.org/10.1016/j.neucom.2006.07.012>
- Shitong, W., Chung, K. F. L., & Duan, F. (2007b). Applying the improved fuzzy cellular neural network IFCNN to white blood cell detection. *Neurocomputing*, *70*(7–9), 1348–1359. <https://doi.org/10.1016/J.NEUCOM.2006.07.012>
- Shokri, M., Tizhoosh, H. R., & Kamel, M. (2006). Opposition-Based Q(λ) Algorithm. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings* (pp. 254–261). IEEE. <https://doi.org/10.1109/IJCNN.2006.246689>
- Silva, L. F.; Saade, D. C. M.; Sequeiros, G. O.; Silva, A. C.; Paiva, A. C.; Bravo, R. S.; Conci, A. (2014). A New Database for Breast Research with Infrared Image. *Journal of Medical Imaging and Health Informatics*, *4*, 92–100.
- Skowronski, J. (1999). Pel recursive motion estimation and compensation in subbands. *Signal Processing: Image Communication*, *14*(5), 389–396. [https://doi.org/10.1016/S0923-5965\(98\)00019-8](https://doi.org/10.1016/S0923-5965(98)00019-8)
- Smith, S. M., & Brady, J. M. (1997). SUSAN—A New Approach to Low Level Image Processing. *International Journal of Computer Vision*, *23*(1), 45–78. <https://doi.org/10.1023/A:1007963824710>
- Snyder, W., Bilbro, G., Logenthiran, A., & Rajala, S. (1990). Optimal thresholding—A new approach. *Pattern Recognition Letters*, *11*(12), 803–809. [https://doi.org/10.1016/0167-8655\(90\)90034-Y](https://doi.org/10.1016/0167-8655(90)90034-Y)
- Soak, S.-M., & Lee, S.-W. (2012). A memetic algorithm for the quadratic multiple container packing problem. *Applied Intelligence*, *36*(1), 119–135. <https://doi.org/10.1007/s10489-010-0248-x>
- Society, T. A. C. (2013). How Common Is Breast Cancer?
- Sonali, Sahu, S., Singh, A. K., Ghreera, S. P., & Elhoseny, M. (2018). An approach for de-noising and contrast enhancement of retinal fundus image using CLAHE. *Optics and Laser Technology*. <https://doi.org/10.1016/j.optlastec.2018.06.061>
- Sörensen, K. (2015). Metaheuristics—the metaphor exposed. *International Transactions in Operational Research*, *22*(1), 3–18. <https://doi.org/10.1111/itor.12001>
- Storn, R., & Price, K. (1995). *Differential Evolution- A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces*. Technical Report TR-95-012. <https://doi.org/10.1023/A:1008202821328>
- Tai, S.-C., Chen, Y.-R., & Chen, Y.-H. (2007). Small-diamond-based search algorithm for fast block motion estimation. *Signal Processing: Image Communication*, *22*(10), 877–890. <https://doi.org/10.1016/J.IMAGE.2007.07.004>
- Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-15*(1), 116–132. <https://doi.org/10.1109/TSMC.1985.6313399>
- Takeuchi, Y. (2008). *Information and Control ICIC International c °2008 ISSN. International Journal of Innovative Computing* (Vol. 4). Retrieved from <http://www.ijicic.org/sss06-16-1.pdf>
- Tan, K. C., Chiam, S. C., Mamun, A. A., & Goh, C. K. (2009). Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization. *European Journal of Operational Research*, *197*(2), 701–713. <https://doi.org/10.1016/J.EJOR.2008.07.025>
- Tan, S., Cheng, X., & Xu, H. (2007). *Information and Control ICIC International c °2007 ISSN. International Journal of Innovative Computing* (Vol. 3). Retrieved from <http://www.ijicic.org/06-012-1.pdf>
- Tan, T. Z., Quek, C., Ng, G. S., & Ng, E. Y. K. (2007). A novel cognitive interpretation of breast cancer thermography with complementary learning fuzzy neural memory structure. *Expert Systems with Applications*, *33*(3), 652–666. <https://doi.org/10.1016/j.eswa.2006.06.012>
- Tanimoto, S. L. (1981). Template matching in pyramids. *Computer Graphics and Image Processing*, *16*(4), 356–369. [https://doi.org/10.1016/0146-664X\(81\)90046-0](https://doi.org/10.1016/0146-664X(81)90046-0)

- Tapiovaara, M. J., & Wagner, R. F. (1993). SNR and noise measurements for medical imaging: I. A practical approach based on statistical decision theory. *Physics in Medicine and Biology*, 38(1), 71–92. <https://doi.org/10.1088/0031-9155/38/1/006>
- Taur, J. S., & Tao, C. W. (1997). Design and analysis of region-wise linear fuzzy controllers. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 27(3), 526–532. <https://doi.org/10.1109/3477.584960>
- Teh, C.-H., & Chin, R. T. (1989). On the detection of dominant points on digital curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(8), 859–872. <https://doi.org/10.1109/34.31447>
- Tenne, Y. (2012). A computational intelligence algorithm for expensive engineering optimization problems. *Engineering Applications of Artificial Intelligence*, 25(5), 1009–1021. <https://doi.org/10.1016/J.ENGAPPAI.2012.03.009>
- Thathachar, M. A. L., & Sastry, P. S. (2002). Varieties of learning automata: an overview. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 32(6), 711–722. <https://doi.org/10.1109/TSMCB.2002.1049606>
- Theofilatos, K., Pavlopoulou, N., Papasavvas, C., Likothanassis, S., Dimitrakopoulos, C., Georgopoulos, E., ... Mavroudi, S. (2015). Predicting protein complexes from weighted protein-protein interaction graphs with a novel unsupervised methodology: Evolutionary enhanced Markov clustering. *Artificial Intelligence in Medicine*. <https://doi.org/10.1016/j.artmed.2014.12.012>
- Tieying Tang, & Jiaju Qiu. (2006). An Improved Multimodal Artificial Immune Algorithm and its Convergence Analysis. In *2006 6th World Congress on Intelligent Control and Automation* (pp. 3335–3339). IEEE. <https://doi.org/10.1109/WCICA.2006.1712985>
- Tizhoosh, H. R. (n.d.). Opposition-Based Learning: A New Scheme for Machine Intelligence. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)* (Vol. 1, pp. 695–701). IEEE. <https://doi.org/10.1109/CIMCA.2005.1631345>
- Tizhoosh, H. R. (2003). Fast and Robust Fuzzy Edge Detection (pp. 178–194). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-36420-7_8
- Tourapis, A. M. (2002). Enhanced predictive zonal search for single and multiple frame motion estimation. In C.-C. J. Kuo (Ed.), *SPIE 4671, Visual Communications and Image Processing 2002, (4 January 2002)*; (p. 1069). California. <https://doi.org/10.1117/12.453031>
- Tremeau, A., & Borel, N. (1997). A region growing and merging algorithm to color segmentation. *Pattern Recognition*, 30(7), 1191–1203. [https://doi.org/10.1016/S0031-3203\(96\)00147-1](https://doi.org/10.1016/S0031-3203(96)00147-1)
- Trier, O. D., & Jain, A. K. (1995). Goal-directed evaluation of binarization methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12), 1191–1201. <https://doi.org/10.1109/34.476511>
- Tseng, D.-C., & Lai, C.-C. (1999). A genetic algorithm for MRF-based segmentation of multi-spectral textured images. *Pattern Recognition Letters*, 20(14), 1499–1510. [https://doi.org/10.1016/S0167-8655\(99\)00117-8](https://doi.org/10.1016/S0167-8655(99)00117-8)
- Tsou, C.-S., & Kao, C.-H. (2008). Multi-objective inventory control using electromagnetism-like meta-heuristic. *International Journal of Production Research*, 46(14), 3859–3874. <https://doi.org/10.1080/00207540601182278>
- Turner, R. E. (Roy E., & Betts, D. S. (David S. (1974). *Introductory statistical mechanics*. [published for] Sussex University Press [by Chatto & Windus].
- Tzovaras, D., Kompatsiaris, I., & Strintzis, M. G. (1999). 3D object articulation and motion estimation in model-based stereoscopic videoconference image sequence analysis and coding. *Signal Processing: Image Communication*, 14(10), 817–840. [https://doi.org/10.1016/S0923-5965\(98\)00046-0](https://doi.org/10.1016/S0923-5965(98)00046-0)
- Uenohara, M., & Kanade, T. (1997). Use of Fourier and Karhunen-Loeve decomposition for fast pattern matching with a large set of templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(8), 891–898. <https://doi.org/10.1109/34.608291>

- US Census Bureau Applications Development and Services Division, W. (n.d.). US Census Bureau. Retrieved January 21, 2019, from <https://www.census.gov/hhes/www/poverty/data/threshld>
- Usuki, H., Onoda, Y., Kawasaki, S., Misumi, T., Murakami, M., Komatsubara, S., & Teramoto, S. (1990). Relationship between thermographic observations of breast tumors and the DNA indices obtained by flow cytometry. *Biomedical Thermology*, *10*(4), 282–285.
- Van Aken, J. (1984). An Efficient Ellipse-Drawing Algorithm. *IEEE Computer Graphics and Applications*, *4*(9), 24–35. <https://doi.org/10.1109/MCG.1984.275994>
- Várkonyi-Kóczy, A. R. (1993). *Journal of intelligent & fuzzy systems. Journal of Intelligent & Fuzzy Systems* (Vol. 19). John Wiley & Sons. Retrieved from <https://content.iospress.com/articles/journal-of-intelligent-and-fuzzy-systems/ifs00381>
- Wahab, A. A., Irna, M., & Salim, M. (2017). Application of Infrared to Biomedical Sciences, 109–131. <https://doi.org/10.1007/978-981-10-3147-2>
- Walker, D., & Kaczor, T. (2012). Breast Thermography: History, Theory, and Use. *Natural Medicine*, *4*(7).
- Wang, H., Wu, Z., & Rahnamayan, S. (2011). Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems. *Soft Computing*, *15*(11), 2127–2140. <https://doi.org/10.1007/s00500-010-0642-7>
- Wang, H., Wu, Z., Rahnamayan, S., Liu, Y., & Ventresca, M. (2011). Enhancing particle swarm optimization using generalized opposition-based learning. *Information Sciences*, *181*(20), 4699–4714. <https://doi.org/10.1016/J.INS.2011.03.016>
- Wang, L., & Huang, F. (2010). Parameter analysis based on stochastic model for differential evolution algorithm. *Applied Mathematics and Computation*, *217*(7), 3263–3273. <https://doi.org/10.1016/J.AMC.2010.08.060>
- Wang, M., & Chu, R. (2009). A novel white blood cell detection method based on boundary support vectors. In *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*. <https://doi.org/10.1109/ICSMC.2009.5346736>
- Wang, P., Zhang, J., Xu, L., Wang, H., Feng, S., & Zhu, H. (2011). How to measure adaptation complexity in evolvable systems – A new synthetic approach of constructing fitness functions. *Expert Systems with Applications*, *38*(8), 10414–10419. <https://doi.org/10.1016/J.ESWA.2011.02.099>
- Wang, X., Fu, M., Ma, H., & Yang, Y. (2015). Lateral control of autonomous vehicles based on fuzzy logic. *Control Engineering Practice*, *34*, 1–17. <https://doi.org/10.1016/J.CONENGPRAC.2014.09.015>
- Wang, X., Gao, X. Z., & Ovaska, S. J. (n.d.). Artificial immune optimization methods and applications - a survey. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)* (pp. 3415–3420). IEEE. <https://doi.org/10.1109/ICSMC.2004.1400870>
- Wang, X., Gao, X. Z., & Ovaska, S. J. (2005). A hybrid optimization algorithm in power filter design. In *31st Annual Conference of IEEE Industrial Electronics Society, 2005. IECON 2005*. (p. 6 pp.). IEEE. <https://doi.org/10.1109/IECON.2005.1569099>
- Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*(4), 600–612. <https://doi.org/10.1109/TIP.2003.819861>
- Wilcoxon, F. (1945). *Individual Comparisons by Ranking Methods. Biometrics Bulletin* (Vol. 1). Retrieved from <http://www.jstor.org/about/terms.html>.
- Wu, A., & So, S. (2003). VLSI implementation of genetic four-step search for block matching algorithm. *IEEE Transactions on Consumer Electronics*, *49*(4), 1474–1481. <https://doi.org/10.1109/TCE.2003.1261256>
- Wu, C.-H., Wang, D.-Z., Ip, A., Wang, D.-W., Chan, C.-Y., & Wang, H.-F. (2009). A particle swarm optimization approach for components placement inspection on printed circuit boards. *Journal of Intelligent Manufacturing*, *20*(5), 551–551. <https://doi.org/10.1007/s10845-008-0235-9>

- Wu, J., Zeng, P., Zhou, Y., & Olivier, C. (2006). A novel color image segmentation method and its application to white blood cell image analysis. In *2006 8th international Conference on Signal Processing*. IEEE. <https://doi.org/10.1109/ICOSP.2006.345700>
- Wu, J., Zeng, P., Zhou, Y., & Olivier, C. (2007). A novel color image segmentation method and its application to white blood cell image analysis. In *International Conference on Signal Processing Proceedings, ICSP*. <https://doi.org/10.1109/ICOSP.2006.345700>
- Wu, P., Yang, W.-H., & Wei, N.-C. (2004a). AN ELECTROMAGNETISM ALGORITHM OF NEURAL NETWORK ANALYSIS-AN APPLICATION TO TEXTILE RETAIL OPERATION. *Journal of the Chinese Institute of Industrial Engineers*, 21(1), 59–67. <https://doi.org/10.1080/10170660409509387>
- Wu, P., Yang, W.-H., & Wei, N.-C. (2004b). AN ELECTROMAGNETISM ALGORITHM OF NEURAL NETWORK ANALYSIS—AN APPLICATION TO TEXTILE RETAIL OPERATION. *Journal of the Chinese Institute of Industrial Engineers*, 21(1), 59–67. <https://doi.org/10.1080/10170660409509387>
- Wu, Q. H. (1995). Learning coordinated control of power systems using interconnected learning automata. *International Journal of Electrical Power & Energy Systems*, 17(2), 91–99. [https://doi.org/10.1016/0142-0615\(95\)91404-8](https://doi.org/10.1016/0142-0615(95)91404-8)
- Xiao, N. (2008). A Unified Conceptual Framework for Geographical Optimization Using Evolutionary Algorithms. *Annals of the Association of American Geographers*, 98(4), 795–817. <https://doi.org/10.1080/00045600802232458>
- Xu, C., Liu, S., & Shao, S. (2010). Template matching using chaotic imperialist competitive algorithm. *Pattern Recognition Letters*, 31(13), 1868–1875. <https://doi.org/10.1016/J.PATREC.2009.12.005>
- Xu, C., Wang, J., & Shiba, N. (2006). Multistage Portfolio Optimization with Var as Risk Measure. Retrieved from <https://www.semanticscholar.org/paper/Multistage-Portfolio-Optimization-with-Var-as-Risk-Xu-Wang/50c3869e3a6f12848372fe69a54566214a482a5c>
- Xu, L., & Jordan, M. I. (1996). On Convergence Properties of the EM Algorithm for Gaussian Mixtures. *Neural Computation*, 8(1), 129–151. <https://doi.org/10.1162/neco.1996.8.1.129>
- Xu, L., Oja, E., & Kultanen, P. (1990). A new curve detection method: Randomized Hough transform (RHT). *Pattern Recognition Letters*. [https://doi.org/10.1016/0167-8655\(90\)90042-Z](https://doi.org/10.1016/0167-8655(90)90042-Z)
- Xu, X., & Zhang, J. (2007). An Improved Immune Evolutionary Algorithm for Multimodal Function Optimization. In *Third International Conference on Natural Computation (ICNC 2007)* (pp. 641–646). IEEE. <https://doi.org/10.1109/ICNC.2007.216>
- Yang, C.-N., Huang, K.-S., Yang, C.-B., & Hsu, C.-Y. (n.d.). *Error-Tolerant Minimum Finding with DNA Computing*. Retrieved from http://par.cse.nsysu.edu.tw/~cbyang/person/publish/c06dna_min.pdf
- Yang, X.-S. (2009). *Firefly Algorithms for Multimodal Optimization* (pp. 169–178). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-04944-6_14
- Yang, X.-S. (2010). A New Metaheuristic Bat-Inspired Algorithm (pp. 65–74). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-12538-6_6
- Yang, X. S. (2009). Firefly algorithms for multimodal optimization. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-642-04944-6_14
- Yang, X. S. (2010a). A new metaheuristic Bat-inspired Algorithm. In *Studies in Computational Intelligence*. https://doi.org/10.1007/978-3-642-12538-6_6
- Yang, X. S. (2010b). *Engineering Optimization: An Introduction with Metaheuristic Applications*. <https://doi.org/10.1002/9780470640425>
- Yao, J., Kharna, N., & Grogono, P. (2005). A multi-population genetic algorithm for robust and fast ellipse detection. *Pattern Analysis and Applications*, 8(1–2), 149–162. <https://doi.org/10.1007/s10044-005-0252-7>
- Yao Nie, & Kai-Kuang Ma. (2002). Adaptive rood pattern search for fast block-matching motion

- estimation. *IEEE Transactions on Image Processing*, 11(12), 1442–1449. <https://doi.org/10.1109/TIP.2002.806251>
- Yap, K. S., Wong, S. Y., & Tiong, S. K. (2013). Compressing and improving fuzzy rules using genetic algorithm and its application to fault detection. In *2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA)* (pp. 1–4). IEEE. <https://doi.org/10.1109/ETFA.2013.6648106>
- Yeo, N. C., Lee, K. H., Venkatesh, Y. V., & Ong, S. H. (2005). Colour image segmentation using the self-organizing map and adaptive resonance theory. *Image and Vision Computing*, 23(12), 1060–1079. <https://doi.org/10.1016/J.IMAVIS.2005.07.008>
- Yoo, J., & Hajela, P. (1999). Immune network simulations in multicriterion design. *Structural Optimization*, 18(2–3), 85–94. <https://doi.org/10.1007/BF01195983>
- Yu, D., Hu, Q., & Wu, C. (2007). Uncertainty measures for fuzzy relations and their applications. *Applied Soft Computing*, 7(3), 1135–1143. <https://doi.org/10.1016/J.ASOC.2006.10.004>
- Yu, J. J. Q., & Li, V. O. K. (2015). A social spider algorithm for global optimization. *Applied Soft Computing*, 30, 614–627. <https://doi.org/10.1016/J.ASOC.2015.02.014>
- Yuan, X., & Shen, X. (2008). Block Matching Algorithm Based on Particle Swarm Optimization for Motion Estimation. In *2008 International Conference on Embedded Software and Systems* (pp. 191–195). IEEE. <https://doi.org/10.1109/ICCESS.2008.35>
- Yuen, H., Princen, J., Illingworth, J., & Kittler, J. (1990). Comparative study of Hough Transform methods for circle finding. *Image and Vision Computing*, 8(1), 71–77. [https://doi.org/10.1016/0262-8856\(90\)90059-E](https://doi.org/10.1016/0262-8856(90)90059-E)
- Yüksel, M. E. (2007). Edge detection in noisy images by neuro-fuzzy processing. *AEU - International Journal of Electronics and Communications*, 61(2), 82–89. <https://doi.org/10.1016/J.AEUE.2006.02.006>
- Yurtkuran, A., & Emel, E. (2010). A new Hybrid Electromagnetism-like Algorithm for capacitated vehicle routing problems. *Expert Systems with Applications*, 37(4), 3427–3433. <https://doi.org/10.1016/J.ESWA.2009.10.005>
- Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 8(3), 338–353. [https://doi.org/10.1016/S0019-9958\(65\)90241-X](https://doi.org/10.1016/S0019-9958(65)90241-X)
- Zahiri, S.-H. (2008). Learning automata based classifier. *Pattern Recognition Letters*, 29(1), 40–48. <https://doi.org/10.1016/J.PATREC.2007.08.011>
- Zareiforush, H., Minaei, S., Alizadeh, M. R., & Banakar, A. (2015). A hybrid intelligent approach based on computer vision and fuzzy logic for quality measurement of milled rice. *Measurement*, 66, 26–34. <https://doi.org/10.1016/J.MEASUREMENT.2015.01.022>
- Zeng, X., & Liu, Z. (2005). A learning automata based algorithm for optimization of continuous complex functions. *Information Sciences*, 174(3–4), 165–175. <https://doi.org/10.1016/J.INS.2004.09.004>
- Zeng, X., Zhou, J., & Vasseur, C. (2000). A strategy for controlling nonlinear systems using a learning automaton. *Automatica*, 36(10), 1517–1524. [https://doi.org/10.1016/S0005-1098\(00\)00066-2](https://doi.org/10.1016/S0005-1098(00)00066-2)
- Zhang, C., Ouyang, D., & Ning, J. (2010). An artificial bee colony approach for clustering. *Expert Systems with Applications*, 37(7), 4761–4767. <https://doi.org/10.1016/J.ESWA.2009.11.003>
- Zhang, G. P. (2000). Neural networks for classification: A survey. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*. <https://doi.org/10.1109/5326.897072>
- Zhang, Y., Matuszewski, B. J., Shark, L., & Moore, C. J. (2007). A Novel Medical Image Segmentation Method using Dynamic Programming, (MediViz).
- Zhang, Z., Chen, C., Sun, J., & Luk Chan, K. (2003). EM algorithms for Gaussian mixtures with split-and-merge operation. *Pattern Recognition*, 36(9), 1973–1983. [https://doi.org/10.1016/S0031-3203\(03\)00059-1](https://doi.org/10.1016/S0031-3203(03)00059-1)
- Zheng, Z., Wang, H., & Khwang Teoh, E. (1999). Analysis of gray level corner detection. *Pattern Recognition Letters*, 20(2), 149–162. [https://doi.org/10.1016/S0167-8655\(98\)00134-2](https://doi.org/10.1016/S0167-8655(98)00134-2)
- Zhuang, X., & Meng, Q. (2004). Local fuzzy fractal dimension and its application in medical

- image processing. *Artificial Intelligence in Medicine*, 32(1), 29–36. <https://doi.org/10.1016/J.ARTMED.2004.01.016>
- Zongzhao Zhou, Yew Soon Ong, My Hanh Nguyen, & Dudy Lim. (n.d.). A Study on Polynomial Regression and Gaussian Process Global Surrogate Model in Hierarchical Surrogate-Assisted Evolutionary Algorithm. In *2005 IEEE Congress on Evolutionary Computation* (Vol. 3, pp. 2832–2839). IEEE. <https://doi.org/10.1109/CEC.2005.1555050>
- Zou, L., Chen, J., Zhang, J., & Dou, L. (2008). The Comparison of Two Typical Corner Detection Algorithms. In *2008 Second International Symposium on Intelligent Information Technology Application* (pp. 211–215). IEEE. <https://doi.org/10.1109/IITA.2008.275>