Amitava Chatterjee
Patrick Siarry *Editors*

Computational Intelligence in Image Processing

Springer

Computational Intelligence in Image Processing

Amitava Chatterjee · Patrick Siarry
Editors

Computational Intelligence in Image Processing

Springer

Editors
Amitava Chatterjee
Electrical Engineering Department
Jadavpur University
Kolkata
West Bengal
India

Patrick Siarry
Laboratory LiSSi
University of Paris-Est Créteil
Créteil
France

ISBN 978-3-642-30620-4 ISBN 978-3-642-30621-1 (eBook)
DOI 10.1007/978-3-642-30621-1
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012942025
ACM Code: I.4, I.2, J.2

© Springer-Verlag Berlin Heidelberg 2013
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Computational intelligence-based techniques have firmly established themselves as viable, alternate, mathematical tools for more than a decade now. These techniques have been extensively employed in many systems and application domains, e.g., signal processing, automatic control, industrial and consumer electronics, robotics, finance, manufacturing systems, electric power systems, power electronics and drives, etc. Image processing is also an extremely potent area which has attracted the attention of many researchers interested in the development of new computational intelligence-based techniques and their suitable applications, in both research problems and in real-world problems. Initially, most of the attention and, hence, research efforts, were focused on developing conventional fuzzy systems, neural networks, and genetic algorithm-based solutions. But, as time elapsed, more sophisticated and complicated variations of these systems and newer branches of stochastic optimization algorithms have been proposed for providing solutions for a wide variety of image processing algorithms. As image processing essentially deals with multidimensional nonlinear mathematical problems, these computational intelligence-based techniques lend themselves perfectly to provide a solution platform for these problems. The interest in this area among researchers and developers is increasing day by day and this is visible in the form of huge volumes of research works that get published in leading international journals and in international conference proceedings.

When the idea of this book was first conceived, the goal was to mainly expose the readers to the cutting-edge research and applications that are going on across the domain of image processing where contemporary computational intelligence techniques can be and have been successfully employed. The result of the spirit behind this original idea and its fruitful implementation in terms of contributions from leading researchers across the globe, in varied related fields, is in front of you: a book containing 15 such chapters. A wide cross-section of image processing problems is covered within the purview of this book. They include problems in the domains of image enhancement, image segmentation, image analysis, image compression, image retrieval, image classification and clustering, image registration, etc.

The book focuses on the solution of these problems using state-of-the-art fuzzy systems, neuro-fuzzy systems, fractals, and stochastic optimization techniques. Among fuzzy systems and neuro-fuzzy systems, several chapters demonstrate how type-2 neuro-fuzzy systems, fuzzy transforms, fuzzy vector quantization, the concept of fuzzy entropy, etc., can be suitably utilized for solving these problems. Several chapters are also dedicated to the solution of image processing problems using contemporary stochastic optimization techniques. These include several modern bio- and nature-inspired global optimization algorithms like bacterial foraging optimization, biogeography-based optimization, genetic programming (GP), along with other popular stochastic optimization strategies, namely, multi-objective particle swarm optimization techniques and differential evolution algorithms. It is our sincere belief that this book will serve as a unified destination where interested readers will get detailed descriptions of many of these modern computational intelligence techniques and they will also obtain fairly good exposure to the modern image processing problem domains where such techniques can be successfully applied.

This book has been divided into four parts. Part I concentrates on discussion of several image preprocessing algorithms. Part II broadly covers image compression algorithms. Part III demonstrates how computational intelligence-based techniques can be effectively utilized for image analysis purposes, and Part IV elucidates how pattern recognition, classification, and clustering-based techniques can be developed for the purpose of image inferencing.

Part I: Image Preprocessing Algorithms

This section of the book presents representative samples of how state-of-the-art computational intelligence-based techniques can be utilized for image preprocessing purposes, e.g., for image enhancement, image filtering, and image segmentation.

Chapter 1 by Yüksel and Baştürk shows how type-2 neuro-fuzzy systems can be utilized for developing image enhancement operators. Type-2 fuzzy systems are considered as improvements over the conventional type-1 fuzzy systems, where type-2 fuzzy systems utilize 'fuzzy-fuzzy sets', as opposed to the conventional 'fuzzy sets' utilized by the type-1 fuzzy systems. Type-2 fuzzy systems have specifically come into existence to handle data uncertainties in a better manner. This chapter shows how such general-purpose operators can be developed for a variety of image enhancement purposes. The chapter also specifically concentrates on the development of suitable new noise filters and noise detectors based on the above-mentioned methodology.

Chapter 2 by Kwok, Ha, Fang, Wang, and Chen focuses on the problem of contrast enhancement by employing a local intensity equalization strategy. The method shows how an image can be subdivided into sectors and each such sector can be independently equalized. The method employs a particle swarm

optimization algorithm-based technique that determines a suitable Gaussian weighting factor-based methodology for reduction of discontinuities along sector boundaries.

Chapter 3 by Boussaïd, Chatterjee, Siarry, and Ahmed-Nacer shows how intelligent hybridization of biogeography-based optimization with differential evolution can be utilized to solve multilevel thresholding problems for image segmentation purposes, utilizing the concept of fuzzy entropy. The objective here is to incorporate diversity in the biogeography-based optimization (BBO) algorithm to solve three-level thresholding problems in a more efficient manner and to provide better uniformity for the segmented image. The utility of the proposed schemes is demonstrated for a series of benchmark images, widely utilized by the researchers within this community.

Chapter 4 by Perlin and Lopes demonstrates how GP approaches can be utilized for the development of image segmentation algorithms. This chapter shows how the image segmentation problem can be viewed as a classification problem and how GP can use a set of terminals and non-terminals to arrive at the final segmented image. The method demonstrates how suitable fitness functions can be defined and how a penalty term can be utilized to obtain a fair division of an original image into its reasonable, constituent parts, in an automated manner. The performance of the algorithm has been extensively evaluated on the basis of a set of images.

Part II: Image Compression Algorithms

Image compression techniques are becoming more and more important in recent times because the race for transmission of huge volumes of image data in real time for a wide variety of applications like Internet-based transmission, mobile communication, live transmission of television events, medical imaging, etc., is well and truly on. The main objective is to simultaneously achieve two competing requirements, i.e., to achieve very high rates of compression ratio and yet there should not be any perceptible degradation in the reconstructed image at the viewer end. This section of the book presents a collection of such modern techniques which primarily aim to solve this problem as efficiently as possible.

Chapter 5 by Tsekouras and Tsolakis describes how fuzzy clustering-based vector quantization techniques can be utilized to solve these problems. This chapter first presents a systematic overview of existing fuzzy clustering-based vector quantization techniques and then it presents a new effective fuzzy clustering-based image compression algorithm that tackles two contentious issues: (i) achieving performance independent of initialization and (ii) reducing the computational cost. The method demonstrates how hybrid clusters can be formed containing crisp and fuzzy areas.

Chapter 6 by Di Martino and Sessa demonstrates how recently proposed fuzzy transforms (F-transforms) can be utilized for layer image compression and reconstruction and then proposes a new modification. The chapter discusses how

an image can be viewed as a fuzzy matrix, comprising several square submatrices, and how direct F-transforms can be suitably applied on each such image block for the compression purpose. The chapter also demonstrates how inverse F-transforms can be utilized for image reconstruction purposes at the viewer end.

Chapter 7 by Sanyal, Chatterjee, and Munshi introduces how the modified bacterial foraging optimization (BFO) algorithm can be suitably used to solve vector quantization-based image compression algorithms. This chapter shows how a nearly optimal codebook can be designed for this purpose with a high peak signal-to-noise ratio (PSNR) in the reconstructed image. The chapter also demonstrates how improvements in the chemotaxis procedure of the BFO algorithm can be useful in achieving high PSNR at the output. The utility of this algorithm is demonstrated by employing it for a variety of benchmark images.

Part III: Image Analysis Algorithms

An important research domain within the broader category of image processing is to analyze an image, captured by a suitable sensor system, for a variety of applications. Such image analysis algorithms may be solely guided by the requirement of the output of the system. In this section of the book, five chapters are included to expose the readers to five different problem domains where different aspects of image analyses are required.

Chapter 8 by Mandal, Halder, Konar, and Nagar discusses how template matching problems in a dynamic image sequence can be solved by fuzzy condition-sensitive algorithms. This chapter shows how a decision-tree-based approach can be utilized to determine the matching(s) of a given template in an entire image. A new hierarchical algorithm has been developed for this purpose and the conditions are induced with fuzzy measurements of the features. The utility of this method has been aptly demonstrated by implementing this algorithm for template matching of human eyes in facial images, under different emotional conditions.

Chapter 9 by Di Martino and Sessa presents another important application which will show how watermarking for tamper detection can be carried out for images compressed by fuzzy relation equations. This method makes use of the well-known encrypting alphabetic text Vigenère algorithm. They have used a novel, interesting method of embedding a varying binary watermark matrix in every fuzzy relation.

Chapter 10 by Bhattacharya and Das makes a detailed, systematic study on how evolutionary algorithms can be utilized for human brain registration processes, that can be useful for the purpose of brain mapping, treatment planning, image guided therapies of nervous system, etc. A new system has been developed for MR and CT image registration of human brain sections, utilizing similarity measures, for both intensity- and gradient-based images. A fuzzy c-means clustering technique has been utilized for extraction of the region of interest in each image. Any degeneracy or abnormality in human brains can be detected by utilizing this

similarity metric, utilized to test the alignment between two images. These similarity metric-based objective functions are nonconvex in nature and do not lend themselves naturally for solution by conventional optimization algorithms. Hence this problem has been solved using a genetic algorithm.

Chapter 11 by Broilo and De Natale discusses how stochastic optimization algorithms can be utilized for another important image processing-based application domain, i.e., image retrieval problems. The chapter first presents an overview of the motivations behind utilizing these methods for image retrieval and several interesting methods that have so far evolved in this domain. Detailed discussions on the setting and tuning of free parameters in traditional retrieval tools as well as direct classification of images in a dataset, based on these competing stochastic algorithms, are presented. A systematic analysis on the relative merits and demerits of these methods has been presented in the context of several application examples.

Chapter 12 by Battiato, Farinella, Guarnera, Messina, and Ravì discusses an important present-day research topic in image processing, removal of red-eye artifacts in images, caused by the flash light reflected from a human retina. While the conventional preflash approaches suffer from unacceptable power consumption problems, the software-based post-acquisition correction procedures may require substantial user interaction. Many contemporary research efforts in this problem area focus on the development of suitable red eye removal techniques with as minimum visual error as possible. This chapter discusses how boosting algorithm aided classifiers can be designed for red eye recognition utilizing the concept of gray codes feature space.

Part IV: Image Inferencing Algorithms

The last section of the book presents several chapters on how modern pattern recognition-based techniques, especially those directed toward classification and clustering objectives, can be utilized for the purpose of image inferencing.

Chapter 13 by Huang, Lee, and Lin presents how fractal analysis can be useful for the purpose of pathological prostate image classification. Very recently, the use of fractal geometry for effective analysis of pathological architecture and growth of tumors has gained prominence. This chapter demonstrates how fractal dimension can be suitably utilized along with other multicategories for feature extraction from texture features, e.g., multiwavelets, Gabor filters, gray-level co-occurrence matrix, etc. These feature extraction methodologies have been coupled with several candidate classifiers, e.g., k-NN and SVM classifiers, to evaluate their relative effectiveness in classifying such prostate images. The chapter demonstrates that, in different types of classifiers developed, each time the best correct classification rates are obtained only when the feature sets include fractal dimensions. Hence the authors have justified the importance and utility of including fractal dimension-based features in prostate image classification.

Chapter 14 by Melgani and Pasolli discusses the development of multiobjective PSO algorithms for hyperspectral image clustering problems. Hyperspectral remote sensing images are quite rich in information content and they can simultaneously capture a large number of contiguous spectra from a wide range of the electromagnetic spectrum. Development of hyperspectral image classification schemes to achieve accurate data class in an unsupervised context is widely known as a challenging research problem. This chapter demonstrates how such an unsupervised clustering problem can be solved by formulating it as a multiobjective optimization problem and how a multiobjective PSO can be suitably utilized for this purpose. The authors have implemented three different statistical criteria for this purpose, i.e., the log-likelihood function, the Bhattacharyya distance, and the minimum description length. Several experimentations clearly validate the utility of the particle swarm optimizers for automated, unsupervised analysis of hyperspectral remote sensing images.

Chapter 15 by Halder, Shaw, Orea, Bhowmik, Chakraborty, and Konar details a new computational intelligence-based approach for emotion recognition from the outer lip-contour of a subject. This approach shows how the lip region of a face image can be segmented and subsequently utilized for determining the emotion. This method demonstrates how a lip-contour model can be suitably utilized for this problem and an effective hybridization of differential evolution-based optimization and support vector machine-based classification techniques have been carried out to draw the final inference. Experimental studies on a large database of human subjects have been carried out to establish the utility of the approach.

Last but not least, we would like to take this opportunity to acknowledge the contribution made by Ilhem Boussaïd, who is a faculty member in the University of Science and Technology Houari Boumediene (USTHB), Algiers, Algeria, in preparing this book in its final form. Ilhem is pursuing her own Ph.D. at the moment, performs her regular duties in her University, is the lead author of Chap. 3 of this book, and, in addition to all these, performed all LaTeX-related activities in integrating this book. We have no words left to express our gratitude to her in this matter.

Finally, the book is in its published form in front of all the readers, worldwide. We do hope that you will find this volume interesting and thought provoking. Enjoy!

Kolkata, India, August 2011 Amitava Chatterjee
Paris, France, August 2011 Patrick Siarry

Contents

Part I Image Preprocessing Algorithms

1 Improved Digital Image Enhancement Filters Based on Type-2 Neuro-Fuzzy Techniques 3
Mehmet Emin Yüksel and Alper Baştürk

2 Locally-Equalized Image Contrast Enhancement Using PSO-Tuned Sectorized Equalization 21
N. M. Kwok, D. Wang, Q. P. Ha, G. Fang and S. Y. Chen

3 Hybrid BBO-DE Algorithms for Fuzzy Entropy-Based Thresholding 37
Ilhem Boussaïd, Amitava Chatterjee, Patrick Siarry and Mohamed Ahmed-Nacer

4 A Genetic Programming Approach for Image Segmentation 71
Hugo Alberto Perlin and Heitor Silvério Lopes

Part II Image Compression Algorithms

5 Fuzzy Clustering-Based Vector Quantization for Image Compression. 93
George E. Tsekouras and Dimitrios M. Tsolakis

6 Layers Image Compression and Reconstruction by Fuzzy Transforms 107
Ferdinando Di Martino and Salvatore Sessa

7 Modified Bacterial Foraging Optimization Technique for Vector Quantization-Based Image Compression 131
Nandita Sanyal, Amitava Chatterjee and Sugata Munshi

Part III Image Analysis Algorithms

8 A Fuzzy Condition-Sensitive Hierarchical Algorithm for Approximate Template Matching in Dynamic Image Sequence 155
Rajshree Mandal, Anisha Halder, Amit Konar and Atulya K Nagar

9 Digital Watermarking Strings with Images Compressed by Fuzzy Relation Equations. 173
Ferdinando Di Martino and Salvatore Sessa

10 Study on Human Brain Registration Process Using Mutual Information and Evolutionary Algorithms. 187
Mahua Bhattacharya and Arpita Das

11 Use of Stochastic Optimization Algorithms in Image Retrieval Problems. 201
Mattia Broilo and Francesco G. B. De Natale

12 A Cluster-Based Boosting Strategy for Red Eye Removal 217
Sebastiano Battiato, Giovanni Maria Farinella, Daniele Ravì, Mirko Guarnera and Giuseppe Messina

Part IV Image Inferencing Algorithms

13 Classifying Pathological Prostate Images by Fractal Analysis. . . . 253
Po-Whei Huang, Cheng-Hsiung Lee and Phen-Lan Lin

14 Multiobjective PSO for Hyperspectral Image Clustering 265
Farid Melgani and Edoardo Pasolli

15 A Computational Intelligence Approach to Emotion Recognition from the Lip-Contour of a Subject 281
Anisha Halder, Srishti Shaw, Kanika Orea, Pavel Bhowmik, Aruna Chakraborty and Amit Konar

Index 299

Part I
Image Preprocessing Algorithms

Chapter 1
Improved Digital Image Enhancement Filters Based on Type-2 Neuro-Fuzzy Techniques

Mehmet Emin Yüksel and Alper Baştürk

Abstract A general purpose image enhancement operator based on type-2 neuro-fuzzy networks is presented in this chapter. The operator can be used for a number of different image enhancement tasks depending on its training. Specifically, two different applications of the presented operator are considered here: (1) noise filter and (2) noise detector. Comparative evaluation of the performance of the presented operator is demonstrated by performing carefully designed filtering experiments. Some other areas of the possible application are also discussed.

1.1 Introduction

Digital image enhancement is one of the most active research areas in image restoration since images are inevitably corrupted by noise during image acquisition and/or transmission. As a consequence, a large number of methods have been developed and successfully employed for detecting and removing noise from digital images in the past few decades. Among these methods, the operators based on neuro-fuzzy techniques have been shown to exhibit superior performance over most of the competing operators.

In recent years, type-2 neuro-fuzzy systems and their applications have attracted a growing interest. Contrary to the scalar membership functions of conventional (type-1) fuzzy systems, the membership functions in type-2 systems are also

M. E. Yüksel (✉)
Department of Biomedical Engineering,
Erciyes University, Kayseri 38039, Turkey
e-mail: yuksel@erciyes.edu.tr

A. Baştürk
Department of Computer Engineering,
Erciyes University, Kayseri 38039, Turkey
e-mail: ab@erciyes.edu.tr

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_1, © Springer-Verlag Berlin Heidelberg 2013

themselves fuzzy and it is this extra degree of fuzziness that provides the designer a more efficient handling of uncertainty, which is inevitably encountered in noisy environments. Based on this observation, image enhancement operators based on type-2 neuro-fuzzy systems may be expected to exhibit much better performance than many other existing operators, provided that appropriate network structures and processing strategies are used.

In this chapter, we begin by presenting a review of the conventional as well as state-of-the-art image restoration operators available in the literature. Following this, we propose a general-purpose image enhancement operator based on type-2 neuro-fuzzy networks. Specifically, we consider two different applications of the presented operator: noise filter and noise detector. For both applications, we perform carefully designed filtering experiments and provide comparative evaluation of the performances of the presented operator and a number of competing operators selected from the literature. We complete the chapter by giving some other areas of the possible application.

1.2 Literature Review

A large number of methods for suppressing impulse noise from digital images have been proposed in the past few decades. The majority of these methods utilize order statistics filtering, which exploits the rank order information of the pixels contained in a given filtering window. The *standard median filter* [1, 2] is probably the simplest operator to remove impulse noise and operates by changing the center pixel of the filtering window with the median of the pixels within the window. Despite its simplicity, this approach provides reasonable noise removal performance but removes thin lines and blurs image details even at low noise densities. The *weighted median filter* and the *center-weighted median filter* [3–5] attempt to avoid the inherent drawbacks of the standard median filter by giving more weight to certain pixels in the filtering window and usually demonstrate better performance in preserving image details than the standard median filter at the expense of reduced noise removal performance.

A number of methods [6–24] are based on a combination of a noise filter with an *impulse detector*, which aims to classify the center pixel of a given filtering window as corrupted or not. If the impulse detector identifies the center pixel as a corrupted pixel, its restored value is obtained by processing the pixels in the filtering window by the noise filter. Otherwise, it is passed to the output unfiltered. Although this approach considerably reduces the distortion effects of the noise filter and enhances its output, its performance inherently depends on the performance of the impulse detector. As a result, many different sorts of impulse detectors exploiting median filters [6–8], center-weighted median filters [9–12], boolean filters [13], edge-detection kernels [14], homogeneity-level information [15], statistical tests [16, 17], classifier-based methods [18], rule-based methods [19], level-detection methods [20], pixel-counting methods [21] and soft computing methods [22–24] have been developed.

In addition to the median-based filters mentioned above, various types of mean filters are successfully utilized for impulse noise removal from digital images [25–33]. Finally, there are also a number of filters based on soft computing methodologies [34–43] as well as several other nonlinear filters [44–54] that combine the desired properties of the above mentioned filters. These filters are usually more complicated, but they generally provide much better noise suppression and detail-preservation performance.

Applications of type-2 fuzzy logic systems [55–65] in digital image processing have shown a steady increase in the last decade. Type-2 fuzzy logic-based image processing operators are usually more complicated than conventional and type-1 based operators. However, they usually yield better performance. Successful applications include gray-scale image thresholding [66], edge detection [67–70], noise-filtering [71–74], corner and edge detection in color images [75], deinterlacing of video signals [76] and image enhancement [77].

1.3 The Type-2 NF Operator

1.3.1 The Structure of the Operator

Figure 1.1a shows the general structure of the neuro-fuzzy image enhancement operator. The operator is constructed by combining a desired number of type-2 neuro-fuzzy (NF) blocks, defuzzifiers and a postprocessor. The operator processes the pixels contained in its filtering window (Fig. 1.1b) and generates an output based on type-2 fuzzy inference. Each NF block in the structure processes a different neighborhood relationship between the center pixel of the filtering window and two neighboring pixels. Possible neighborhood topologies are shown in Fig. 1.1c.

All NF blocks employed in the structure of the operator are identical to each other and function as suboperators. However, it should be observed that the values of the internal parameters of each of the NF blocks are different from those in the other NF blocks, even though all NF blocks have the same internal structure and the same number of internal parameters. This is because each NF block is trained for its particular neighborhood individually and independently of the others during training, which is discussed in detail later.

Each NF block accepts the center pixel and two of its appropriate neighboring pixels as input and produces an output, which is a *type-1 interval fuzzy set* representing the uncertainty interval (i.e., lower and upper bounds) for the restored value of the center pixel. The output fuzzy sets coming from the NF blocks are then fed to the corresponding defuzzifier blocks. The defuzzifier defuzzifies the input fuzzy set and converts it into a single scalar value. These scalar values are finally evaluated by the postprocessor and converted into a single output value, which is also the output value of the overall system.

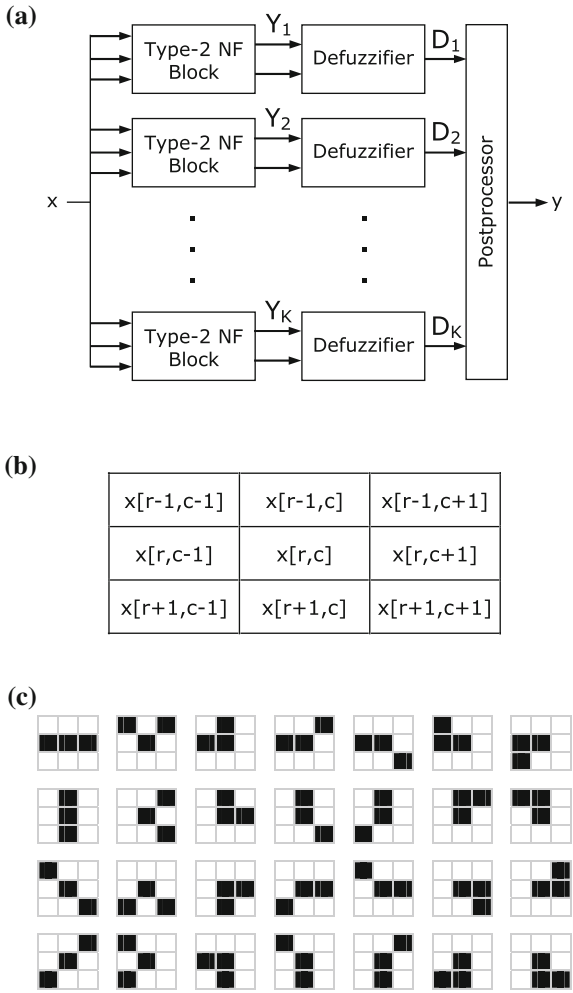


Fig. 1.1 **a** Structure of the general purpose type-2 neuro-fuzzy image enhancement operator, **b** filtering window of the operator, **c** possible pixel neighborhood topologies (Reproduced from [73] with permission from the IEEE. © 2008 IEEE.)

1.3.2 Type-2 NF Blocks

Each NF block employed in the structure of the presented image enhancement operator is a Sugeno-type first-order type-2 interval fuzzy inference system with three inputs and one output. The internal structures of the NF blocks are identical to each other. The input-output relationship of any of the NF blocks is as follows:

Let X_1^k, X_2^k, X_3^k denote the inputs of the kth NF block and Y_k denote its output. Each combination of inputs and their associated membership functions is represented by a rule in the rule-base of the kth NF block. The rule-base contains a desired number of fuzzy rules, which are as follows:

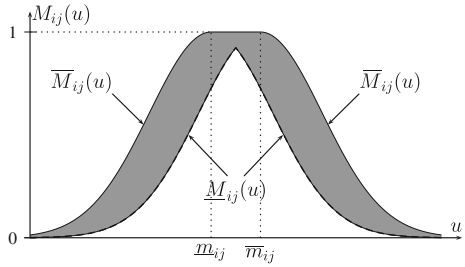


Fig. 1.2 A type-2 interval Gaussian membership function with uncertain mean. The *shaded area* is the footprint of uncertainty (FOU) (Reproduced from [73] with permission from the IEEE. © 2008 IEEE.)

1. if $(X_1^k \in M_{11}^k)$ & $(X_2^k \in M_{12}^k)$ & $(X_3^k \in M_{13}^k)$, then $R_1^k = c_{11}^k X_1^k + c_{12}^k X_2^k + c_{13}^k X_3^k + c_{14}^k$
2. if $(X_1^k \in M_{21}^k)$ & $(X_2^k \in M_{22}^k)$ & $(X_3^k \in M_{23}^k)$, then $R_2^k = c_{21}^k X_1^k + c_{22}^k X_2^k + c_{23}^k X_3^k + c_{24}^k$
3. if $(X_1^k \in M_{31}^k)$ & $(X_2^k \in M_{32}^k)$ & $(X_3^k \in M_{33}^k)$, then $R_3^k = c_{31}^k X_1^k + c_{32}^k X_2^k + c_{33}^k X_3^k + c_{34}^k$

$\vdots \qquad\qquad \vdots$

i. if $(X_1^k \in M_{i1}^k)$ & $(X_2^k \in M_{i2}^k)$ & $(X_3^k \in M_{i3}^k)$, then $R_i^k = c_{i1}^k X_1^k + c_{i2}^k X_2^k + c_{i3}^k X_3^k + c_{i4}^k$

$\vdots \qquad\qquad \vdots$

N. if $(X_1^k \in M_{N1}^k)$ & $(X_2^k \in M_{N2}^k)$ & $(X_3^k \in M_{N3}^k)$, then $R_N^k = c_{N1}^k X_1^k + c_{N2}^k X_2^k + c_{N3}^k X_3^k + c_{N4}^k$

where N is the number of fuzzy rules in the rule-base, M_{ij}^k denotes the ith membership function of the jth input and R_i^k denotes the output of the ith rule.

The antecedent membership functions are type-2 interval Gaussian membership functions with uncertain mean:

$$M_{ij}^k(u) = \exp\left[-\frac{1}{2}\left(\frac{u - m_{ij}^k}{\sigma_{ij}^k}\right)^2\right] \qquad m_{ij}^k \in [\underline{m}_{ij}^k, \overline{m}_{ij}^k] \tag{1.1}$$

with $i = 1, 2, \cdots, N$; $j = 1, 2, 3$ and $k = 1, 2, \cdots, K$. Here, the parameters m_{ij}^k and σ_{ij}^k are the *mean* and the *standard deviation* of the type-2 interval Gaussian membership function M_{ij}^k, respectively, and the interval $[\underline{m}_{ij}^k, \overline{m}_{ij}^k]$ denotes the lower and the upper bounds of the uncertainty in the mean. A sample type-2 interval Gaussian membership function and its associated *footprint of uncertainty (FOU)* are illustrated in Fig. 1.2.

Since the membership functions M_{ij}^k are *interval* membership functions, the boundaries of their FOU are characterized by their *lower* and *upper* membership

functions, which are defined as

$$\underline{M}_{ij}^{k}(u) = \begin{cases} \exp\left[-\frac{1}{2}\left(\frac{u-\underline{m}_{ij}^{k}}{\sigma_{ij}^{k}}\right)^{2}\right] & u > \frac{\underline{m}_{ij}^{k}+\overline{m}_{ij}^{k}}{2} \\ \exp\left[-\frac{1}{2}\left(\frac{u-\overline{m}_{ij}^{k}}{\sigma_{ij}^{k}}\right)^{2}\right] & u \leq \frac{\underline{m}_{ij}^{k}+\overline{m}_{ij}^{k}}{2} \end{cases} \tag{1.2}$$

and

$$\overline{M}_{ij}^{k}(u) = \begin{cases} \exp\left[-\frac{1}{2}\left(\frac{u-\underline{m}_{ij}^{k}}{\sigma_{ij}^{k}}\right)^{2}\right] & u < \underline{m}_{ij}^{k} \\ 1 & \underline{m}_{ij}^{k} \leq u \leq \overline{m}_{ij}^{k} \\ \exp\left[-\frac{1}{2}\left(\frac{u-\overline{m}_{ij}^{k}}{\sigma_{ij}^{k}}\right)^{2}\right] & u > \overline{m}_{ij}^{k} \end{cases} \tag{1.3}$$

where $\underline{M}_{ij}^{k}$ and $\overline{M}_{ij}^{k}$ are the lower and the upper membership functions of the type-2 interval membership function M_{ij}^{k}, respectively.

The output of the kth NF block is the weighted average of the individual rule outputs:

$$Y_k = \frac{\sum_{i=1}^{N} w_i^k R_i^k}{\sum_{i=1}^{N} w_i^k} \tag{1.4}$$

The weighting factor, w_i^k, of the ith rule is calculated by evaluating the membership expressions in the antecedent of the rule. This is accomplished by first converting the input values to fuzzy membership values by utilizing the antecedent membership functions M_{ij}^{k} and then applying the *and* operator to these membership values. The *and* operator corresponds to the multiplication of the antecedent membership values:

$$w_i^k = M_{i1}^{k}(X_1^k) \,.\, M_{i2}^{k}(X_2^k) \,.\, M_{i3}^{k}(X_3^k) \tag{1.5}$$

Since the membership functions M_{ij}^{k} in the antecedent of the ith rule are type-2 interval membership functions, the weighting factor w_i^k is a type-1 interval set, i.e., $w_i^k = [\underline{w}_i^k, \overline{w}_i^k]$, whose lower and upper boundaries are determined by using the lower and the upper membership functions defined before:

$$\begin{aligned} \underline{w}_i^k &= \underline{M}_{i1}^{k}(X_1^k) \,.\, \underline{M}_{i2}^{k}(X_2^k) \,.\, \underline{M}_{i3}^{k}(X_3^k) \\ \overline{w}_i^k &= \overline{M}_{i1}^{k}(X_1^k) \,.\, \overline{M}_{i2}^{k}(X_2^k) \,.\, \overline{M}_{i3}^{k}(X_3^k) \end{aligned} \tag{1.6}$$

where $\underline{w}_i^k$ and $\overline{w}_i^k$ $(i = 1, 2, \cdots, N)$ are the lower and upper boundaries of the interval weighting factor w_i^k of the ith rule, respectively.

After the weighting factors are obtained, the output Y_k of the kth NF filter can be found by calculating the weighted average of the individual rule outputs by using Eq. (1.4). The output Y_k is also a type-1 interval set, i.e., $Y_k = [\underline{Y}_k, \overline{Y}_k]$, since the w_i^ks in the above Eq. are type-1 interval sets and the R_i^ks are scalars. The lower and the upper boundaries of Y_k are determined by using the iterative procedure proposed by Karnik and Mendel [78].

1.3.3 The Defuzzifier

The defuzzifier block takes the type-1 interval fuzzy set obtained at the output of the corresponding NF block as input and converts it into a scalar value by performing centroid defuzzification. Since the input set is a type-1 interval fuzzy set, i.e., $Y_k = [\underline{Y}_k, \overline{Y}_k]$, its centroid is equal to the center of the interval:

$$D_k = \frac{\underline{Y}_k + \overline{Y}_k}{2} \tag{1.7}$$

1.3.4 The Postprocessor

The postprocessor generates the final output of the proposed operator. It processes the scalar values obtained at the outputs of the defuzzifiers and produces a single scalar output, which represents the output of the operator.

The postprocessor actually calculates the average value of the defuzzifier outputs and then suitably truncates this value to an 8-bit integer number. The input-output relationship of the postprocessor may be explained as follows:

Let $D_1, D_2, \cdots, D_K$ denote the outputs of the defuzzifiers in the structure of the proposed operator (Fig. 1.1a). The output of the postprocessor is calculated in two steps. In the first step, the average value of the individual type-2 NF block outputs is calculated:

$$D_{AV} = \frac{1}{K} \sum_{k=1}^{K} D_k \tag{1.8}$$

In the second step, this value is suitably truncated to an 8-bit integer value so that the luminance value obtained at the output of the postprocessor ranges between 0 and 255:

$$y = \begin{cases} 0 & \text{if } D_{AV} < 0 \\ 255 & \text{if } D_{AV} > 255 \\ \text{round}(D_{AV}) & \text{otherwise} \end{cases} \tag{1.9}$$

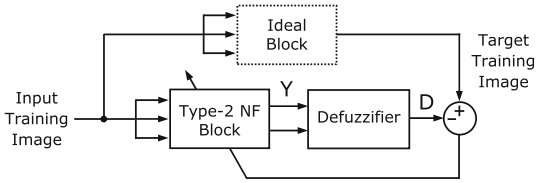


Fig. 1.3 General setup for training the type-2 NF blocks in the structure of the image enhancement operator (Adapted from [73] with permission from the IEEE. © 2008 IEEE.)

where y is the output of the postprocessor, which is also the output of the type-2 NF image enhancement operator.

1.3.5 Training the NF Blocks

The internal parameters of the proposed operator are optimized by training. Training of the proposed operator is accomplished by training the individual type-2 NF blocks in its structure. Each NF block in the structure is trained individually and independently of the others. The training setup is shown in Fig. 1.3.

The parameters of the NF block under training are iteratively adjusted in such a manner that its output converges to the output of the ideal block. The ideal block is conceptual only and does not necessarily exist in reality. It is only the output of the ideal block that is necessary for training and this is represented by a suitably chosen target training image, which varies depending on the application.

The parameters of the NF block under training are tuned by using the Levenberg Marquardt optimization algorithm [79–81] so as to minimize the learning error. Once the training of the NF blocks is completed, the internal parameters of the blocks are fixed, and the blocks are combined with the same number of defuzzifiers and a postprocessor to construct the NF operator (Fig. 1.1a).

1.3.6 Processing the Input Image

The overall procedure for processing the input image may be summarized as follows:

1. A 3×3 pixel filtering window is slid over the image one pixel at a time. The window is started from the upper-left corner of the image and moved sideways and progressively downwards in a *raster scanning* fashion.
2. For each filtering window position, the appropriate pixels of the filtering window representing the possible neighborhoods of the center pixel are fed to the corresponding NF blocks in the structure. Each NF block individually generates a type-1 interval fuzzy set as its output.

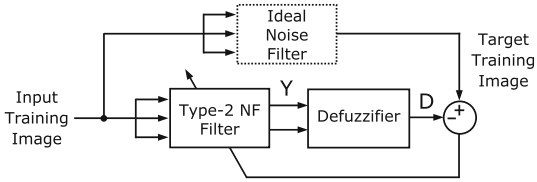


Fig. 1.4 Setup for training the type-2 NF filters in the structure of the image enhancement operator for the noise filter application (Reproduced from [73] with permission from the IEEE. © 2008 IEEE.)

3. The outputs of the NF blocks are fed to their corresponding defuzzifiers. The defuzzifiers process the input type-1 interval fuzzy sets coming from the NF blocks and output the centroid of their input sets.
4. The outputs of the defuzzifiers are fed to the postprocessor, which processes the scalar values obtained at the outputs of the defuzzifiers and produces a single scalar output. The value obtained at the output of the postprocessor is also the output value of the operator.
5. This procedure is repeated for all pixels of the noisy input image.

1.4 Applications

In this section, we demonstrate two different applications of the type-2 NF image enhancement operator presented in the previous section: noise filtering and noise detection. In both of these applications, the same general purpose type-2 NF operator shown in Fig. 1.1 are used. However, a different pair of training images is used in the training to customize the operator for each of these two applications.

1.4.1 The Type-2 NF Operator as a Noise Filter

In the first application, we demonstrate the use of the type-2 NF image enhancement operator as a noise filter. The training arrangement to customize an individual NF block in the structure of the operator as a noise filter is illustrated in Fig. 1.3. Here, the parameters of the NF block under training are iteratively tuned to minimize the difference between its output and the output of the ideal noise filter. The ideal noise filter is a conceptual filter that is capable of completely removing the noise from the image and does not necessarily exist in reality. What is necessary for training is only the output of the ideal noise filter, which is represented by the target training image.

Figure 1.4 shows the training setup for the noise filter application and Fig. 1.5 shows the images used for training. The training image shown in Fig. 1.5a is a computer-generated 40×40 pixel artificial image. Each square box in this image has a size of 4×4 pixels and the 16 pixels contained within each box have the same luminance value, which is an 8-bit integer number uniformly distributed between

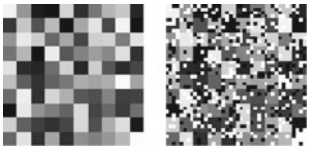


Fig. 1.5 Training images: **a** Original training image, s **b** Noisy training image (Reproduced from [73] with permission from the IEEE. © 2008 IEEE.)

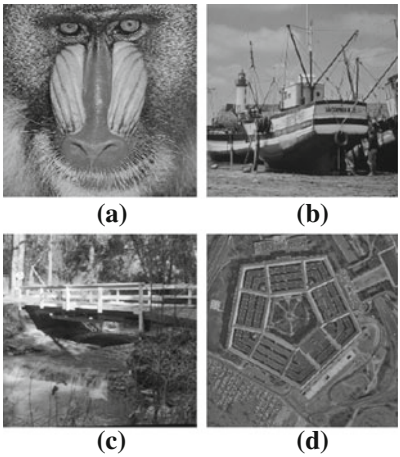


Fig. 1.6 Test images: **a** Baboon, **b** Boats, **c** Bridge, **d** Pentagon (Reproduced from [73] with permission from the IEEE. © 2008 IEEE.)

0 and 255. The image in Fig. 1.5b is obtained by corrupting the image in Fig. 1.5a by impulse noise of 30 % noise density. The images in Fig. 1.5a and b are employed as the target (desired) and the input images during training, respectively.

Several filtering experiments are performed to evaluate the filtering performance of the presented type-2 NF operator functioning as a noise filter. The experiments are especially designed to reveal the performance of the operator for different image properties and noise conditions.

Figure 1.6 shows the test images used in the experiments. Noisy experimental images are obtained by contaminating the original test images by impulse noise with an appropriate noise density depending on the experiment. For comparison, the corrupted experimental images are also restored by using a number of conventional as well as state-of-the-art impulse noise removal operators from the literature, including the standard median filter (MF) [1, 2], the switching median filter (SMF) [6], the tristate median filter (TSMF) [9], the signal-dependent rank-ordered mean filter (SDROMF) [26], the fuzzy filter (FF) [36], the progressive switching median filter (PSMF) [7], the multistate median filter (MSMF) [11], the edge-detecting median filter (EDMF) [14], the adaptive fuzzy switching filter (AFSF) [51], the alpha-trimmed mean-based filter (ATMBF) [33] and the adaptive median filter with difference-type noise detector (DNDAM) [19].

The performance of all operators is evaluated by using the mean-squared error (MSE) criterion, which is defined as

Table 1.1 Average MSE values of operators for 25, 50 and 75 % noise densities (Reproduced from [73] with permission from the IEEE. © 2008 IEEE.)

Filter	25 %	50 %	75 %	Average
MF	505	2367	8460	3777
SMF	421	2324	8454	3733
TSMF	632	3742	10996	5123
SDROMF	328	802	4067	1732
FF	265	734	3061	1353
PSMF	301	576	2640	1172
MSMF	612	3640	10317	4856
EDMF	298	1007	4955	2086
AFSF	271	547	1759	859
ATMBF	431	2333	8459	3741
DNDAM	371	800	2640	1270
Type-2 NF	145	382	980	502

$$\mathrm{MSE} = \frac{1}{RC}\sum_{r=1}^{R}\sum_{c=1}^{C}(s[r,c]-y[r,c])^2 \tag{1.10}$$

where $s[r, c]$ and $y[r, c]$ represent the luminance values of the pixels at location (r, c) of the original and the restored versions of a corrupted test image, respectively.

Table 1.1 shows the average MSE values of all operators included in the noise-filtering experiments. Here the average MSE value of a given operator for a given noise density is found by averaging the four MSE values of that operator obtained for four test images. It is seen from this Table that the proposed operator offers the best performance of all operators.

1.4.2 The Type-2 NF Operator as a Noise Detector

All image restoration filters more or less damage the uncorrupted pixels of their input image while repairing the corrupted (noisy) pixels, thus introducing undesirable blurring effects into the repaired output image. This problem can be avoided by using a special operator, called an *impulse detector*, that is capable of distinguishing the corrupted pixels of the input image from the uncorrupted ones. Hence, an impulse detector is used to guide a noise filter during its processing of the noisy input image and improve its performance. If the input pixel under concern is classified as uncorrupted, then it is passed to the output image without filtering. If it is classified as corrupted, its restored version produced by the noise filter is passed to the output image. Various different types of impulse detectors [6–24] have been shown in the

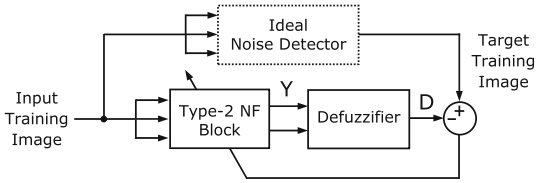


Fig. 1.7 Setup for training the type-2 NF filters in the structure of the image enhancement operator for the noise detector application. (Adapted from [73] with permission from the IEEE. © 2008 IEEE.)

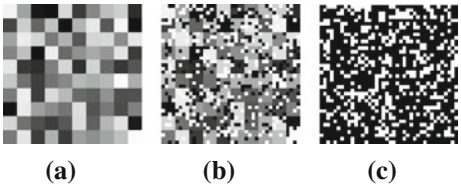


Fig. 1.8 Training images: **a** Original training image, **b** Noisy training image, **c** Noise-detection image (Reproduced and adapted from [73] with permission from the IEEE. © 2008 IEEE.)

last decade to significantly improve the performance and reduce the blurring effects of image noise removal operators.

In this section, we demonstrate the use of the presented type-2 NF operator as a noise detector. We first demonstrate how to customize the general-purpose type-2 NF image enhancement operator as a noise detector, and then we demonstrate how to use it together with a noise filter to improve the performance of that filter.

The arrangement used for training an individual type-2 NF block in the structure of the NF operator as a noise detector is illustrated in Fig. 1.7. Here, the internal parameters of the NF block under training are iteratively adjusted so that its output converges to the output of the ideal noise detector. The ideal noise detector is again a conceptual operator, and its output is represented by the noise-detection image shown in Fig. 1.8c.

Figure 1.8 shows the three training images used for the noise-detection application: the original training image, the noisy training image and the noise-detection image *from left to right*. The first two images, the original and the noisy training images, are the same as the ones used in the noise-filtering application. The third image, the noise-detection image, deserves a little explanation. It is obtained from the difference between the original training image and the noisy training image. Locations of the white pixels in this image indicate the locations of the noisy pixels. Hence, it is not difficult to see that the images in Fig. 1.8c and b are used as the target (desired) and the input images for noise detection training process, respectively.

The enhanced filtering process of a given noisy input image comprises three stages. In the first stage, the noisy input image is fed to the noise filter, which generates a repaired image at its output. In the second stage, the noisy input image is fed to the type-2 NF impulse detector, which generates a noise-detection image at its output. The noise-detection image is a black-and-white image that is similar to the target training image (Fig. 1.8c). In the third stage, the pixels of the noisy input image and the repaired output image are appropriately mixed to obtain the enhanced output image. For this purpose, those pixels of the enhanced output image that correspond

Table 1.2 MSE values for the noise-detection application

Filter	MSE without detector	MSE with detector
MF	497	195
EDMF	301	193
MMEMF	258	110

to the white pixels of the noise-detection image are copied from the repaired output image of the noise filter, while the others are copied directly from the original input image.

The validity of the method discussed above is demonstrated by using it with three different noise filters. These are the MF [1, 2], the EDMF [14] and the minimum maximum exclusive mean filter (MMEMF) [30].

Table 1.2 shows the average MSE values for the three filters for the uses "without" and "with" the detector for the baboon image corrupted by impulse noise with 25 % noise density. As can easily be seen from the Table, the detector significantly reduces the average MSE values of the filters. For a visual evaluation of the enhancement obtained by using the type-2 NF detector, the output images of the three noise filters for the uses "without" and "with" the detector for the baboon image corrupted by impulse noise with 25 % noise density are given in Fig. 1.9. For each vertical image pair in this figure, the upper image shows the direct output image of the corresponding noise filter while the lower image shows the image enhanced by using the noise filter with the type-2 NF noise detector. The undesirable blurring effects and the restoration of these distortions by means of the type-2 NF noise detector can clearly be observed by carefully examining the small details and texture in the images, such as the hair around the baboon's mouth.

1.5 Conclusions and Remarks

In this chapter, we presented an improved image enhancement operator based on type-2 NF networks. The presented operator is a general purpose operator that can be customized for a number of different tasks in image processing. We presented two specific applications here: noise filter and noise detector.

It should be pointed out, however, that other potential application areas of the general-purpose NF operator structure discussed here are not limited to the two applications presented in this chapter The presented NF operator may be used for a number of other applications in image processing provided that appropriate network topologies and training strategies are employed. In this way, it is straightforward to obtain the type-2 versions of the type-1 applications presented in [82, 83] by simply replacing the type-1 operator in the training setup by a type-2 one and appropriately choosing the training images. Other potential applications are left to the reader.

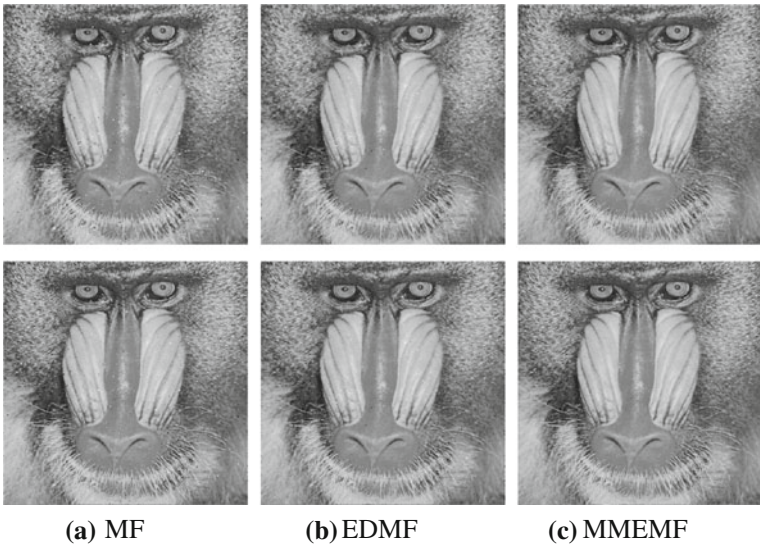


Fig. 1.9 Output images of three noise filters for the noise-detection application, comparing the noise filters with (*lower*) and without (*upper*) the type-2 NF noise detector: **a** MF, **b** EDMF, **c** MMEMF

Acknowledgments Part of the material (text, Eqs., figures, etc.) from a previously published work of the authors [73] copyrighted by the IEEE (© 2008 IEEE) has been adapted and/or reused in Sects. 1.2, 1.3 and 1.4.1 of this chapter. The authors wish to thank the IEEE for its kind permission to reuse this material.
Part of the results presented in this chapter were obtained during two research projects funded by the Erciyes University Scientific and Technological Research Center (Project code: FBT-07-12) and the Turkish Scientific and Technological Research Council (TÜBİTAK, project code: 110E051). The authors also wish to thank Erciyes University and TÜBİTAK for their kind support of this work.

References

1. Gabbouj, M., Coyle, E.J., Gallager, N.C.: An overview of median and stack filtering. Circuits Syst. Signal Process. **11**, 7–45 (1992)
2. Umbaugh, S.E.: Computer Vision and Image Processing. Prentice-Hall International Inc, Upper Saddle River (1998)
3. Yli-Harja, O., Astola, J., Neuvo, Y.: Analysis of the properties of median and weighted median filters using threshold logic and stack filter representation. IEEE Trans. on Signal Process. **39**, 395–410 (1991)
4. Ko, S.J., Lee, Y.H.: Center weighted median filters and their applications to image enhancement. IEEE Trans. Circuit Syst. **38**, 984–993 (1991)
5. Yin, L., Yang, R., Gabbouj, M.: Weighted median filters: A tutorial. IEEE Trans. Circuits Syst. **II**(43), 157–192 (1996)

6. Sun, T., Neuvo, Y.: Detail-preserving median based filters in image processing. Pattern Recognit. Lett. **15**, 341–347 (1994)
7. Wang, Z., Zhang, D.: Progressive switching median filter for the removal of impulse noise from highly corrupted images. IEEE Trans. Circuit Syst. **46**, 78–80 (1999)
8. Khryashchev, V.V., Apalkov, I.V., Priorov, A.L.: Image denoising using adaptive switching median filter. In: Proceedings of the IEEE International Conference on Image Processing (ICIP'2005), vol. 1, pp. 117–120 (2005)
9. Chen, T., Ma, K.K., Chen, L.H.: Tri-state median filter for image denoising. IEEE Trans. Image Process. **8**, 1834–1838 (1999)
10. Chen, T., Wu, H.R.: Adaptive impulse detection using center-weighted median filters. IEEE Signal Proc. Lett. **8**, 1–3 (2001)
11. Chen, T., Wu, H.R.: Space variant median filters for the restoration of impulse noise corrupted images. IEEE Trans. Circuit Syst. II, **48**, 784–789 (2001)
12. Chan, R.H., Hu, C., Nikolova, M.: An iterative procedure for removing random-valued impulse noise. IEEE Signal Proc. Lett., **11**, 921–924 (2004)
13. Aizenberg, I., Butakoff, C., Paliy, D.: Impulsive noise removal using threshold boolean filtering based on the impulse detecting functions. IEEE Signal Proc. Lett. **12**, 63–66 (2005)
14. Zhang, S., Karim, M.A.: A new impulse detector for switching median filters. IEEE Signal Proc. Lett. **9**, 360–363 (2002)
15. Pok, G., Liu, Y., Nair, A.S.: Selective removal of impulse noise based on homogeneity level information. IEEE Trans. Image Process. **12**, 85–92 (2003)
16. Beşdok, E., Yüksel, M.E.: Impulsive noise rejection from images with Jarque–Berra test based median filter. Int. J. Electron. Commun. (AEÜ) **59**, 105–110 (2005)
17. Garnett, R., Huegerich, T., Chui, C.: A universal noise removal algorithm with an impulse detector. IEEE Trans. Image Process. **14**, 1747–1754 (2005)
18. Chang, J.Y., Chen, J.L.: Classifier-augmented median filters for image restoration. IEEE Trans. Instrum. Meas. **53**, 351–356 (2004)
19. Yuan, S.Q., Tan, Y.: H.: Impulse noise removal by a global-local noise detector and adaptive median filter. Signal Process. **86**, 2123–2128 (2006)
20. Yamashita, N., Ogura, M., Lu, J.: A random-valued impulse noise detector using level detection. In: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'2005), vol. 6, pp. 6292–6295. Kobe (2005)
21. Smolka, B., Chydzinski, A.: Fast detection and impulsive noise removal in color images. Real-Time Imaging **11**, 389–402 (2005)
22. Eng, H.L., Ma, K.K.: Noise adaptive soft-switching median filter. IEEE Trans. Image Process. **10**, 242–251 (2001)
23. Yüksel, M.E., Beşdok, E.A.: simple neuro-fuzzy impulse detector for efficient blur reduction of impulse noise removal operators for digital images. IEEE Trans. Fuzzy Syst. **12**, 854–865 (2004)
24. Schulte, S., Nachtegael, M., De Witte, V., Van der Weken, D., Kerre, E.E.: A fuzzy impulse noise detection and reduction method. IEEE Trans. Image Process. **15**, 1153–1162 (2006)
25. Abreu, E., Mitra, S. K.: A signal-dependent rank-ordered mean (SD-ROM) filter—a new approach for removal of impulses from highly corrupted images. In: Proceedings of the IEEE International Conference on Acoustics, Speech and, Signal Processing (ICASSP'95), vol. 4, pp. 2371–2374 (1995)
26. Abreu, E., Lightstone, M., Mitra, S.K.: A new efficient approach for the removal of impulse noise from highly corrupted images. IEEE Trans. Image Process. **5**, 1012–1025 (1996)
27. Moore, M. S., Gabbouj, M., Mitra, S. K.: Vector SD-ROM filter for removal of impulse noise from color images. In: Proceedings of ECMCS99 EURASIP Conference on DSP for Multimedia Communications and Services. Krakow (1999)
28. Mitra, S.K., Sicuranza, G.L., Gibson, J.D.: Nonlinear Image Processing (Communications, Networking and Multimedia). Academic Press, Orlando (2001)
29. Abreu, E.: Signal-dependent rank-ordered mean (SD-ROM) filter. In: Mitra, S.K., Sicuranza, G.L., Gibson, J.D. (eds.) Nonlinear Image Processing (Communications, Networking and Multimedia), pp. 111–133. Academic Press, Orlando (2001)

30. Han, W.Y., Lin, J.C.: Minimum-maximum exclusive mean (MMEM) filter to remove impulse noise from highly corrupted images. Electron. Lett. **33**, 124–125 (1997)
31. Singh, K. M., Bora, P. K., Singh, B. S.: Rank ordered mean filter for removal of impulse noise from images. In: Proceedings of the IEEE International Conference on Industrial Technology (ICIT'02), vol. 2, pp. 980–985 (2002)
32. Zhang, D. S., Kouri, D. J.: Varying weight trimmed mean filter for the restoration of impulse noise corrupted images. In: Proceedings of the IEEE International Conference on Acoustics, Speech and, Signal Processing (ICASSP'05), vol. 4, pp. 137–140 (2005)
33. Luo, W.: An efficient detail-preserving approach for removing impulse noise in images. IEEE Signal Proc. Lett. **13**, 413–416 (2006)
34. Beşdok, E., Çivicioglu, P., Alçi, M.: Impulsive noise suppression from highly corrupted images by using resilient neural networks. Lecture Notes in Artificial Intelligence, vol. 3070, pp. 670–675 (2004)
35. Cai, N., Cheng, J., Yang, J.: Applying a wavelet neural network to impulse noise removal. In: Proceedings of the International Conference on Neural Networks and, Brain (ICNN&B'05), vol. 2, pp. 781–783 (2005)
36. Russo, F., Ramponi, G.: A fuzzy filter for images corrupted by impulse noise. IEEE Signal Process. Lett. **3**, 168–170 (1996)
37. Choi, Y.S., Krishnapuram, R.: A robust approach to image enhancement based on fuzzy logic. IEEE Trans. Image Process. **6**, 808–825 (1997)
38. Russo, F.: FIRE operators for image processing. Fuzzy Sets Syst. **103**, 265–275 (1999)
39. Van De Ville, D., Nachtegael, M., Van der Weken, D.: Noise reduction by fuzzy image filtering. IEEE Trans. Fuzzy Syst. **11**, 429–436 (2003)
40. Morillas, S., Gregori, V., Peris-Fajarne, G.: A fast impulsive noise color image filter using fuzzy metrics. Real-Time Imaging **11**, 417–428 (2005)
41. Russo, F.: Noise removal from image data using recursive neuro-fuzzy filters. IEEE Trans. Instrum. Meas. **49**, 307–314 (2000)
42. Yüksel, M.E., Baştürk, A.: Efficient removal of impulse noise from highly corrupted digital images by a simple neuro-fuzzy operator. Int. J. Electron. Commun. (AEÜ) **57**, 214–219 (2003)
43. Beşdok, E., Çivicioglu, P., Alçi, M.: Using an adaptive neuro-fuzzy inference system-based interpolant for impulsive noise suppression from highly distorted images. Fuzzy Sets Syst. **150**, 525–543 (2005)
44. Kong, H., Guan, L.: Detection and removal of impulse noise by a neural network guided adaptive median filter. In: Proceedings of the IEEE International Conference on Neural Networks, vol. 2, pp. 845–849 (1995)
45. Lee, C.S., Kuo, Y.H., Yu, P.T.: Weighted fuzzy mean filter for image processing. Fuzzy Sets Syst. **89**, 157–180 (1997)
46. Lee, C.S., Kuo, Y.H.: The important properties and applications of the adaptive weighted fuzzy mean filter. Int. J. Intell. Syst. **14**, 253–274 (1999)
47. Windyga, P.S.: Fast impulse noise removal. IEEE Trans. Image Process. **10**, 173–179 (2001)
48. Smolka, B., Plataniotis, K.N., Chydzinski, A.: Self-adaptive algorithm of impulsive noise reduction in color images. Pattern Recognit. **35**, 1771–1784 (2002)
49. Rahman, S.M.M., Hasan, M.K.: Wavelet-domain iterative center weighted median filter for image denoising. Signal Process. **83**, 1001–1012 (2003)
50. Russo, F.: Impulse noise cancellation in image data using a two-output nonlinear filter. Measurement **36**, 205–213 (2004)
51. Xu, H., Zhu, G., Peng, H.: Adaptive fuzzy switching filter for images corrupted by impulse noise. Pattern Recognit. Lett. **25**, 1657–1663 (2004)
52. Alajlan, N., Kamel, M., Jernigan, E.: Detail preserving impulsive noise removal. Signal Process. Image Commun. **19**, 993–1003 (2004)
53. Yüksel, M.E., Baştürk, A., Beşdok, E.: Detail preserving restoration of impulse noise corrupted images by a switching median filter guided by a simple neuro-fuzzy network. EURASIP J. Appl. Signal Process. **2004**, 2451–2461 (2004)

54. Yüksel, M.E.: A hybrid neuro-fuzzy filter for edge preserving restoration of images corrupted by impulse noise. IEEE Trans. Image Process. **15**, 928–936 (2006)
55. Karnik, N.N., Mendel, J.M.: Application of type-2 fuzzy logic system to forecasting of time-series. Inf. Sci. **120**, 89–111 (1999)
56. Liang, Q., Mendel, J.M.: Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters. IEEE Trans. Fuzzy Syst. **8**, 551–563 (2000)
57. John, R.I., Innocent, P.R.: Barnes MR Neuro-fuzzy clustering of radiographic tibia image data using type-2 fuzzy sets. Inf. Sci. **125**, 203–220 (2000)
58. Liang, Q.: Mendel JM MPEG VBR video traffic modeling and classification using fuzzy techniques. IEEE Trans. Fuzzy Syst. **9**, 183–193 (2001)
59. Hagras, H.: A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. IEEE Trans. Fuzzy Syst. **12**, 524–539 (2004)
60. Lynch, C., Hagras, H., Callaghan, V.: Embedded type-2 FLC for real-time speed control of marine and traction diesel engines. In: Proceedings of the FUZZ-IEEE 2005, pp. 347–352, Reno (2005)
61. Astudillo, L., Castillo, O., Melin, P.: Intelligent control of an autonomous mobile robot using type-2 fuzzy logic. J. Eng. Lett. **13**, 93–97 (2006)
62. Gu, L., Zhang, Y.Q.: Web shopping expert using new interval type-2 fuzzy reasoning. Soft Comput. **11**, 741–751 (2007)
63. Dereli, T., Baykasoglu, A., Altun, K.: Industrial applications of type-2 fuzzy sets and systems: a concise review. Comput. Ind. **62**, 125–137 (2011)
64. Mendel, J. M.: Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions. Prentice-Hall International Inc, Upper Saddle River (2001)
65. Mendel, J.M., John, R.I.B.: Type-2 fuzzy sets made simple. IEEE Trans. Fuzzy Syst. **10**, 117–127 (2002)
66. Tizhoosh, H.R.: Image thresholding using type II fuzzy sets. Pattern Recognit. **38**, 2363–2372 (2005)
67. Mendoza, O., Melin, P., Licea, G.: A new method for edge detection in image processing using interval type-2 fuzzy logic. In: Proceedings of IEEE International Conference on Granular Computing 2007, pp. 151–156. Silicon Valley (2007)
68. Bustince, H., Barrenechea, E., Pagola, M.: Interval-valued fuzzy sets constructed from matrices: application to edge detection. Fuzzy Sets Syst. **160**, 1819–1840 (2009)
69. Melin, P.: Interval type-2 fuzzy logic applications in image processing and pattern recognition. In: Proceedings of IEEE International Conference on Granular Computing 2010, pp. 728–731. Silicon Valley (2010)
70. Melin, P., Mendoza, O., Castillo, O.: An improved method for edge detection based on interval type-2 fuzzy logic. Expert Syst. Appl. **37**, 8527–8535 (2010)
71. Bansal, R., Sehgal, P., Bedi, P.: A novel framework for enhancing images corrupted by impulse noise using type-II fuzzy sets. In: Proceedings of Fifth International Conference on Fuzzy Systems and Knowledge Discovery, pp. 266–271. Shandong (2008)
72. Sun, Z., Meng, G.: An image filter for eliminating impulse noise based on type-2 fuzzy sets. In: Proceedings of International Conference on Audio, Language and Image Processing 2008, pp. 1278–1282. Shanghai (2008)
73. Yildirim, M.T., Baştürk, A., Yüksel, M.E.: Impulse noise removal from digital images by a detail-preserving filter based on type-2 fuzzy Logic. IEEE Trans. Fuzzy Syst. **16**, 920–928 (2008)
74. Murugeswari, P., Manimegalai, D.: Noise reduction in color image using interval type-2 fuzzy filter (IT2FF). Int. J. Eng. Sci. Technol. **3**, 1334–1338 (2011)
75. Madasu, H., Verma, O.P., Gangwar, P.: Fuzzy edge and corner detector for color images. In: Proceedings of Sixth International Conference on Information Technology: New Generations, pp. 1301–1306. Las Vegas (2009)
76. Jeon, G., Anisetti, M., Bellandi, V.: Designing of a type-2 fuzzy logic filter for improving edge-preserving restoration of interlaced-to-progressive conversion. Inf. Sci. **179**, 2194–2207 (2009)

77. Bansal, R., Arora, P., Gaur, M.: Fingerprint image enhancement using type-2 fuzzy sets. In: Proceedings of Sixth International Conference on Fuzzy Systems and Knowledge Discovery, pp. 412–417. Tianjin (2009)
78. Karnik, N.N., Mendel, J.M.: Centroid of a type-2 fuzzy set. Inf. Sci. **132**, 195–220 (2001)
79. Levenberg, K.: A method for the solution of certain problems in least squares. Quan. Appl. Math. **2**, 164–168 (1944)
80. Marquardt, D.W.: An algorithm for least squares estimation of nonlinear parameters. J. Soc. Ind. Appl. Math. **31**, 431–441 (1963)
81. Jang, J.S.R., Sun, C.T., Mizutani, E.: Neuro-Fuzzy and Soft Computing. Prentice-Hall International Inc, Upper Saddle River (1997)
82. Yüksel, M.E., Yildirim, M.T.: A simple neuro-fuzzy edge detector for digital images corrupted by impulse noise. Int. J. Electron. Commun. (AEÜ) **58**, 72–75 (2004)
83. Yüksel, M.E.: A simple neuro-fuzzy method for improving the performances of impulse noise filters for digital images. Int. J. Electron. Commun. (AEÜ) **59**, 463–472 (2005)

Chapter 2
Locally-Equalized Image Contrast Enhancement Using PSO-Tuned Sectorized Equalization

N. M. Kwok, D. Wang, Q. P. Ha, G. Fang and S. Y. Chen

Abstract Contrast enhancement is a fundamental procedure in applications requiring image processing. Indeed, image enhancement contributes critically to the success of subsequent operations such as feature detection, pattern recognition and other higher-level processing tasks. Of interest among methods available for contrast enhancement is the intensity modification approach, which is based on the statistics of pixels in a given image. However, due to variations in the imaging condition and the nature of the scene being captured, it turns out that global manipulation of an image may be vulnerable to a noticeable quality degradation from distortion and noise. This chapter is devoted to the development of a local intensity equalization strategy together with mechanisms to remedy artifacts produced by the enhancement while ensuring a better image for viewing. To this end, the original image is subdivided randomly into sectors, which are equalized independently. A Gaussian weighting factor is further used to remove discontinuities along sector boundaries. To achieve simultaneously the multiple objectives of contrast enhancement and viewing distortion reduction, a suitable optimization algorithm is required to determine sector locations and the associated weighting factor. For this, a particle-swarm optimization

N. M. Kwok (✉) · D. Wang
The University of New South Wales, Sydney, NSW 2052, Australia
e-mail: nmkwok@unsw.edu.au

D. Wang
e-mail: d.wang@unsw.edu.au

Q. P. Ha
University of Technology Sydney, Sydney, NSW 2007, Australia
e-mail: quangha@eng.uts.edu.au

G. Fang
University of Western Sydney, Sydney, NSW 2751, Australia
e-mail: g.fang@uws.edu.au

S. Y. Chen
Zhejiang University of Technology, Hangzhou 310023, China
e-mail: sy@ieee.org

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_2, © Springer-Verlag Berlin Heidelberg 2013

algorithm is adopted in the proposed image enhancement method. This algorithm helps optimize the Gaussian weighting parameters for discontinuity removal and determine the local region where enhancement is applied. Following comprehensive descriptions on the methodology, this chapter presents some real-life images for illustration and verification of the effectiveness of the proposed approach.

2.1 Introduction

The use of image processing technology can be found in a large number of applications including computer vision, optical classification, augmented reality, feature detection, medical and morphological signal processing. For example, in manufacturing [3], three-dimensional model construction could be facilitated by the use of properly structured illumination. In industrial automation where reliable perception of the workspace is required, a vision system can be used to detect surface defects on civil structures, enabling a maintenance [13]. Image processing techniques have been applied to restore valuable ancient paintings [16], which is an important step towards their preservation. Images from cephalic radiography could be enhanced for better diagnosis of illnesses [6]. The quality of remote sensing data could be improved using image processing techniques [14]. Numerous interesting applications can be found in the literature. One fundamental operation in image processing technique is the contrast enhancement, which critically determines the quality of its subsequent operations.

In the context of contrast enhancement, there are also a number of possible approaches. In [17], a morphological filter was used for image sharpening. The contrast could also be improved by making use of the curvelet transform [20]. In the field of soft computing [7], the image contrast could be increased by a fuzzy intensification process. In [8], image enhancement was tackled from the point of view of noise-filtering and edge boosting, where the method was applied in color images. Color image processing and enhancement is a more complicated process than its counterpart for black-and-white images [12] due to the involvement of multiple color channels and the need to preserve the color information content [15] while enhancing the contrast. Novel techniques that address these problems are in great demand. For instance, in [1] it was proposed to enhance the image quality by making use of local contrast information and fusing the morphologically enhanced image with the original.

There are other attempts to enhance an image, e.g. by color rendition [18], where a neural network is used to model the color relation from a natural scene. An approach to intensify an image using a fuzzy system was also presented in [7] where the intensity gradients of neighboring pixels are adjusted according to a rule base. Alternatively, the genetic algorithm, an evolutionary computation technique, was applied to enhance image contrast [19]. Althoughsatisfactory results could be obtained with these specific approaches, the use of histogram equalization is still a popular, effectively proven method due to its simplicity [4] and satisfactory performance. In this

class of methodology, statistics of pixel intensities collected in a histogram are constructed, and pixel intensities are modified accordingly for contrast enhancement.

Image enhancement approaches adopting histogram equalization can be broadly categorized into classes of global and local equalization implementation. The former method conducts equalization over all image pixels concurrently. In a canonical implementation, the resultant image has a histogram resembling a linear transformation or stretching from its original image histogram. In [10], spatial relationships between neighboring pixels were taken into consideration.

On the other hand, local equalization tackles image enhancement by dividing the image into multiple sectors and equalizing them independently, see [11]. In the work by Stark [21], the generation of a desired target histogram is made dependent on the characteristics of local windows. For this, a predetermined scheme can be applied to divide the image into subblocks, where each block is equalized independently. In this context, a local histogram equalization scheme was proposed in [25]. In [24], the input images were subdivided, independently equalized, and finally fused to produce a contrast-enhanced image. This approach was further developed in Kim et al. [9], where the original image is divided into overlapping subblocks and equalized according to the pixel characteristics within the block. In [21], the image histogram is matched to a distribution determined from a windowed and filtered version of the original histogram. Manipulations on the histograms were also frequently suggested by researchers. These include specific considerations in minimizing the mean brightness error between the input and output images [2]. In [22], the maximum entropy or information content criterion was invoked in contrast enhancement.

A computational intelligence optimization-based method is presented in this chapter as an alternative approach to the contrast enhancement problem for color images. The image is first randomly divided into sectors, and their contrast is increased by individual histogram equalization. The enhanced sectors are then modulated by a Gaussian mask to mitigate abrupt changes at the sector boundaries. This process is repeated, where new sectors are generated and the final output is derived from a weighted summation of the intermediate images with the weights determined via information-based weighted sum average. The performance of the approach is evaluated by using a collection of color images taken under diverse conditions. Moreover, it should be nontrivia to obtain an optimal selection of sectors, including their numbers, the boundaries and the smoothing needed to remove discontinuities along the boundaries. Here, the particle-swarm optimization (PSO) algorithm is adopted as an optimization procedure to obtain the above-mentioned settings such that the resultant image can provide the information to the viewer and for the success of subsequent processing. The PSO algorithm [5] is a multiagent-based search method mimicking the flights of bird flocks. For example, PSO is adopted to find optimal parameters for multiple-robot motion planning [23] or, more relevantly, for enhancement of an image while preserving its brightness, as reported in [12].

The Chapter is organized as follows. Section 2.2 describes the global histogram equalization process for image enhancement and its limitations. The proposed local sector-based enhancement method is developed in Sect. 2.3. Experiments conducted

using a variety of color images are described in Sect. 2.4, followed by some discussion. A conclusion is drawn in Sect. 2.5.

2.2 Global Histogram Equalization

Histogram equalization is a technique used to enhance the contrast of an image. The statistics of the image are collected and represented in a graphical representation showing the distribution of image data. Color images are frequently delivered from cameras in red green blue (RGB) signals or spaces. It is also a common strategy to enhance a color image by first converting the image to its intensity-related space, where enhancement operations are applied. The intermediate results are then converted to eventually give an enhanced color image.

Let the input or original color image be represented by

$$\mathscr{I} = \{\mathbf{I}_{uv}\}, \quad \mathbf{I}_{uv} = \left[R_{uv} G_{uv} B_{uv}\right]^T, \tag{2.1}$$

where u, v are pixel coordinates in the width and height dimensions, respectively. Since the RGB space contains three color-related signals, it is intuitive to operate on the three signal spaces simultaneously for image enhancement. Furthermore, since the human visual system is sensitive to intensity variation when accessing image contrast, the image is converted before applying enhancement. For example, the image is commonly converted to the hue saturation value (HSV) format:

$$\begin{bmatrix} H \\ S \\ V \end{bmatrix}_{uv} = \mathbf{T}\left(\begin{bmatrix} R \\ G \\ B \end{bmatrix}_{uv}\right) = \begin{bmatrix} \arctan(\sqrt{3}(G-B)/(2R-G-B)) \\ 1 - 3 \times \min(R, G, B)/(R+G+B) \\ \max(R, G, B) \end{bmatrix}, \tag{2.2}$$

where the H component represents the color tone, S denotes saturation and V corresponds to the image intensity. The restoration from HSV to RGB space is conducted using $\mathbf{T}^{-1}()$, the inverse transform of $\mathbf{T}()$.

A histogram is obtained from intensities V_{uv}, giving

$$\mathscr{H} = \{h_i\}, \quad \sum_{i=1}^{L} h_i = N, \tag{2.3}$$

where h_i is the number of pixels having the ith intensity level and N is the total number of pixels. The number of levels is taken as $L = 256$, corresponding to 8-bit ($2^8 = 256$) electronic display.

In principle, image contrast will be enhanced as long as one can make use of the whole available intensity range. A uniform histogram is therefore used, where the numbers of pixels that fall inside each intensity level are equal. That is, the desired histogram is

$$\mathscr{H}^d = \{h_j^d\}, \quad h_j^d = NL^{-1}, \quad j = 1, \ldots, L. \tag{2.4}$$

To perform enhancement, two cumulative histograms are constructed from the input and desired histograms, respectively. We have

$$\mathscr{C} = \{c_i\},\ c_i = \sum_{k=1}^{i} h_k; \text{ and } \mathscr{C}^d = \{c_j^d\},\ c_j^d = \sum_{k=1}^{j} h_k^d. \tag{2.5}$$

For a pixel with original intensity i in the cumulative histogram $\mathscr{C}$ at the $c_i th$ position, its equalized intensity value is obtained by referring to the $c_j^d th$ element in the cumulative desired histogram $\mathscr{C}^d$ and overriding. That is,

$$j = \{i : c_i = c_j^d\}. \tag{2.6}$$

The aforementioned process is referred to as global histogram equalization because all pixels in the image are used in constructing the histograms. This method is easy to implement but there are also limitations in its performance, particularly in viewing. To illustrate this remark, an image is taken for an indoor scene where the camera is being saturated from the background high-level illumination magnitude (Fig. 2.1a). The result from global histogram equalization is given in Fig. 2.1b. It is observed that some degree of enhancement is obtained for the people sighted at the bottom-right corner. Further comparison can be made with results from a canonical implementation of a local equalization scheme as well as the proposed approach, discussed in what follows, for which the results are shown respectively in Figs. 2.1c and d. It is noted that further contrast enhancements can be obtained, also illustrated by the bottom-right corner portion of the image, via the sectorized approach, while a better result is obtained from the proposed method. Histograms of the intensities of these images are plotted in Fig. 2.1e For the global equalization process, the histogram shown in cyan illustrates that there are occasions where some of the intensity ranges, with zero counts of pixel intensity, have not been utilized for conveying scene information. On the other hand, intensity ranges are more utilized in the two other sector-based equalization methods, as can be seen in Figs. 2.1c and d

2.3 Local Histogram Equalization

In order to enhance the contrast of a color image and to extract details not deliverable by global histogram equalization, a local equalization method is developed and reported in the remainder of this chapter. In brief, the proposed method consists of three major steps: (i) to independently equalize image sectors or blocks, (ii) to reduce intensity discontinuity along sector boundaries, and (iii) to aggregate an enhanced image using a weighted-sum scheme.

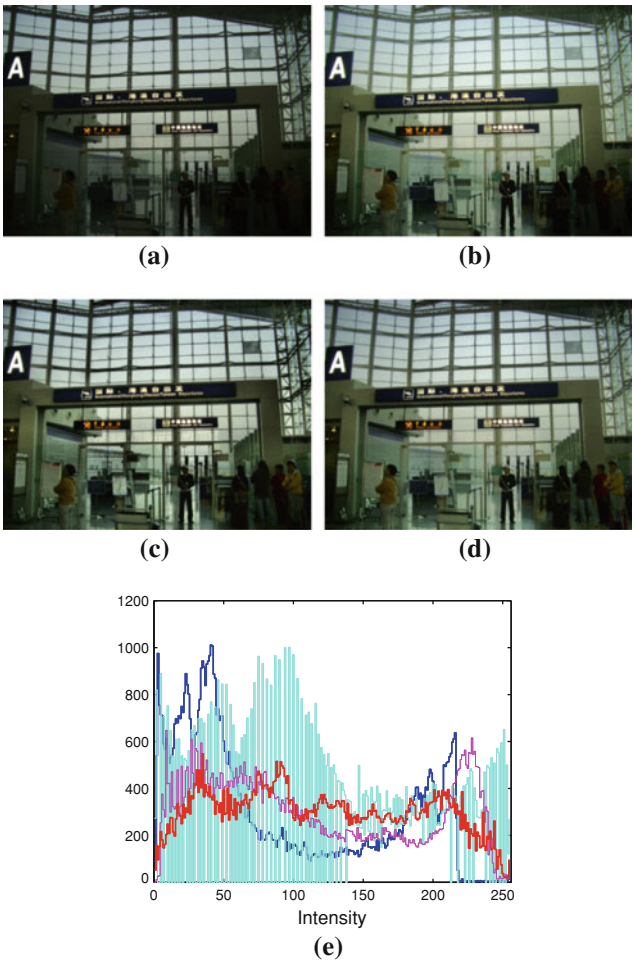


Fig. 2.1 Performance of global against local/sectorized histogram equalization: **a** original image, **b** globally equalized image by a uniform target distribution, **c** canonical sectorized equalization result, **d** proposed sectorized equalization result, **e** resulting histograms, *blue*: original **a**; *cyan*: globally equalized image **b**; *magenta*: canonical sectorized equalization **c**; *red*: proposed sectorized equalization **d**, to be discussed in Sect. 2.3

2.3.1 Sectorized Equalization

Given an image to be enhanced, the process starts first with its conversion from the RGB space to the HSV space, where the intensity component is denoted as V_{uv}. Four sectors are then generated. The center point (p, q) of dividing the sectors is determined by randomly drawing a sample in the image. That is,

Fig. 2.2 An intermediate image showing independently equalized sectors. Note intensity differences along the sector boundaries

$$p \sim \mathscr{U}(1, u_{max}),\ q \sim \mathscr{U}(1, v_{max}), \tag{2.7}$$

where u_{max}, v_{max} are the width and height of the given image in pixels, respectively; $\mathscr{U}$ is a uniform distribution; and $\sim$ stands for the sampling operation. The choice of the center point is constrained so as not to produce a too-small or too-narrow sector. In this work, the center is not allowed to lie within 10 % from the image edges. That is,

$$0.1u_{max} \leq p \leq 0.9u_{max},\ 0.1v_{max} \leq q \leq 0.9v_{max}. \tag{2.8}$$

Four sectors that are formed using the point (p, q) as the center, indexed by superscript $s = 1, \ldots, 4$, are given by

$$\mathscr{S}^s_{pq} = \begin{cases} \mathscr{I}_{1:p,1:q} \\ \mathscr{I}_{p+1:u_{max},1:q} \\ \mathscr{I}_{1:p,q+1:v_{max}} \\ \mathscr{I}_{p+1:u_{max},q+1:v_{max}} \end{cases}. \tag{2.9}$$

Each sector $\mathscr{S}^s_{pq}$ is then equalized to the desired uniform distribution using the procedure described in Sect. 2.2, giving equalized sectors as

$$\mathscr{E}^{sd}_{pq} = \{\mathscr{S}^s_{pq} : \mathscr{S}^s_{pq}(c_i) = \mathscr{E}^{sd}_{pq}(c_j)\}. \tag{2.10}$$

The result is depicted in Fig. 2.2, where it can be seen that for each individual sector, the contrast is increased. However, it is also observed that along the sector boundaries, intensity differences or discontinuities are noticeable and need to be mitigated.

2.3.2 Mitigation of Sector Discontinuities

In order to reduce the difference of intensities along sector boundaries, an arithmetic mean aggregation approach is adopted in order to combine the locally equalized

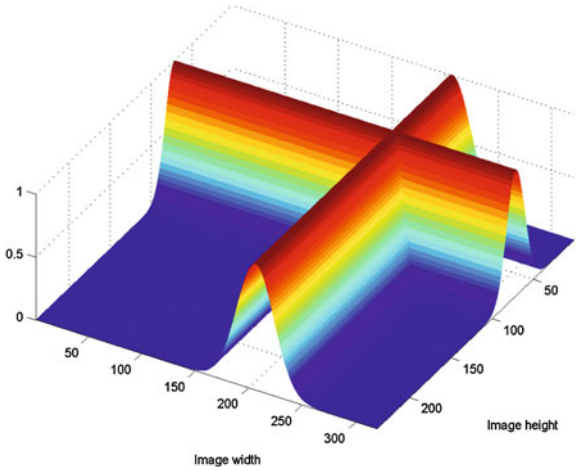


Fig. 2.3 The Gaussian weighting kernel to remove boundary discontinuities corresponding to the sectors shown in Fig. 2.2

sectors. In addition, enhancements in each sector should be retained as much as possible. Here, these requirements are satisfied by weighting the sectors with a Gaussian kernel and then integrating with the original image.

Let a normalized one-dimensional Gaussian for each boundary be given by

$$\mathbf{G}^b(\delta, \sigma) = \exp\left(\frac{-\delta^{b^2}}{2\sigma^{b^2}}\right), \tag{2.11}$$

where superscript $b \in \{u, v\}$ denotes if the Gaussian is for the height (v) or width (u) for the image dimension, δ is the distance from the boundary along the associated dimension, and σ is the Gaussian standard deviation. The overall Gaussian used to remove the boundary discontinuities is obtained from an element-wise maximization operation, that is,

$$\mathbf{G}_{uv} = \max\{\mathbf{G}^u(\delta, \sigma),\ \mathbf{G}^v(\delta, \sigma)\}. \tag{2.12}$$

The resultant Gaussian weighting kernel is shown in Fig. 2.3.

The original image $\mathscr{I}$ and the complete image $\mathscr{E}$, formed by aggregating the independently equalized sectors $\mathscr{E}^s_{pq}$, are then fused to obtain a smoothed image $\mathscr{S}_{sm}$. For this, the Gaussian weights and an element-wise operator $\odot$ defined by

$$\mathscr{S}_{sm} = \mathbf{G} \odot \mathscr{I} + (\mathbf{I} - \mathbf{G}) \odot \mathscr{E}, \tag{2.13}$$

are used, where $\mathbf{I}$ is a matrix having dimension $u \times v$ for all elements equal to unity. The smoothed image is depicted in Fig. 2.4.

Fig. 2.4 The boundary smoothed image is obtained by fusing the equalized and original images via the Gaussian weighting kernel

2.3.3 Iterated Enhancement

The smoothed image in Fig. 2.4 is obtained from a randomly selected center point (p, q). A further improvement can therefore be expected from deliberate determination of a proper center point. For the purpose of ensuring enhancement across all possible cases of scene variations, a number of center points and sectors have to be generated and their enhancement conducted iteratively using histogram equalization. To this end, a collection of smoothed images is created. Moreover, in order to produce an enhanced image from the smoothed images, a strategy for their combination using an information-based weighted-sum technique is adopted.

The quality of the smoothed intermediate image $\mathscr{S}_{sm}$ is taken as information entropy. That is,

$$H_t = -\sum_{i=0}^{L} \log(p_i)\, p_i, \tag{2.14}$$

where subscript t stands for the iteration count, $L = 255$ is the maximum intensity, p_i is the probability of pixel that takes on the ith intensity. The values of p_i are obtained as normalized histogram elements h_i.

In local and sectorized equalization, through the selection of a certain center point to sector the original image as well as repeated calculation of the quality metric for, say, τ iterations, the final output can be obtained by first normalizing the information contents as

$$\bar{H}_t = \frac{H_t}{\sum_{t=1}^{\tau} H_t}, \tag{2.15}$$

and then by combining this with a weighted-sum average of the intermediate resultant images, yielding

$$\bar{\mathscr{I}} = \sum_{t=1}^{\tau} \bar{H}_t \mathscr{S}_{sm,t}. \tag{2.16}$$

The result is depicted in Fig. 2.5. It can be seen that intensity discontinues are removed and contrast is increased in local sectors. This image then replaces the

Fig. 2.5 Resultant image obtained from fusion of sectorized equalization and smoothing

V-component in the HSV domain and is finally converted back to the RGB space as a color image.

2.3.4 PSO-Based Parameter Optimization

In the above development and illustration, it is observed that effective results are obtained by incorporating an iterative smoothing operation into the sectorized local equalization approach. A nontrivial question can then be raised as to what should be the proper sector that divides the image and how should smoothing weighting be assigned. To this end, we solve these unknowns by the use of a multiobjective optimization algorithm for which the computational effectiveness remains a requirement. For this, particle-swarm optimization (PSO) [5] is used as described in the following.

The PSO algorithm can be viewed as a stochastic search method for solving nondeterministic optimization problems. For example, in the problem at hand, the sector center point (p, q) and the standard deviation σ of the smoothing Gaussian are coded as particles:

$$\mathbf{x} = [p_1, q_1, \sigma_1, \ldots, p_\tau, q_\tau, \sigma_\tau]^T, \tag{2.17}$$

where each set of parameters or part of the particles $\{p, q, \sigma\}$ gives one smoothed image from the sectorized histogram equalization approach.

At the start of the algorithm, the particle positions are generated to cover the solution space. These positions may be deterministically or randomly distributed and the number of particles is predefined. In general, a small number reduces the computational load but at the expense of extended iterations required to obtain the optimum (but the optimal solution is not known *a priori*). The velocities $\mathbf{v}_0^i$ can also be set randomly or simply assigned as zeros. A problem-dependent fitness function is evaluated, and a fitness value is assigned to each particle. Here, the fitness function is taken from the entropy of the image given in Eq. (2.14). For the set of fitness values, the one with the highest value is taken as the global best $\mathbf{g}_{best}$ (for a maximization

problem). This set of initial fitness values is denoted as the particle-best $\mathbf{p}^i_{best}$. The velocity is then calculated using the random gain coefficients. The particle positions are updated, and the whole procedure repeats. Finally, as the satisfaction of some termination criteria , the global-best particle is reported as the optimal solution to the problem.

The essence of the algorithm can be described by the following expression,

$$\begin{aligned} \mathbf{v}^i_{k+1} &= w\mathbf{v}^i_k + \mathbf{c}_1 \otimes (\mathbf{g}_{best,k} - \mathbf{x}^i_k) + \mathbf{c}_2 \otimes (\mathbf{p}^i_{best,k} - \mathbf{x}^i_k) \\ \mathbf{x}^i_{k+1} &= \mathbf{x}^i_k + \mathbf{v}_{k+1}, \end{aligned} \tag{2.18}$$

where $\mathbf{x}$ is the particle position in the solution space, $\mathbf{v}$ is the velocity of the particle motion assuming a unity time step, w is the velocity control coefficient, $\mathbf{c}_1$, $\mathbf{c}_2$ are the gain control coefficients, $\mathbf{g}_{best}$ is the global-best position, $\mathbf{p}_{best}$ is the position of a particular particle where the best fitness is obtained so far, operator $\otimes$ denotes the external multiplication of scalars with velocities, subscript k is the iteration index, and superscript i is the particle index.

For the local equalization in the contrast enhancement problem tackled in this work, the optimization process is proposed in Algorithm 2.1 as follows:

Algorithm 2.1 PSO-tuned sector equalization

```
Input: Image 𝒥 of size v, u
Define PSO parameters: generations nG, particles nP, iterations nS
Initialize: particles P, velocities V, inertia weight w
Initialize: group best G_best, personal best P_best
for generations g = 1 : nG do
    for particles p = 1 : nP do
        for iterations s = 1 : nS do
            Get the sector center point from particle s
            Partition input image into four sectors
             Conduct uniform histogram equalization
             Get the smoothing Gaussian standard deviation σ
             Smooth sector boundaries to give smoothed image 𝒮
             Calculate entropy H for smoothed image
        end for
        Normalize entropies from each smoothed image
        Aggregate enhanced image ℰ_p for particle p
        Calculate entropy for each aggregated image
    end for
    Determine G_best and P_best from all particles
    Update particle position in solution space
end for
Output: overall contrast enhanced image ℰ
```

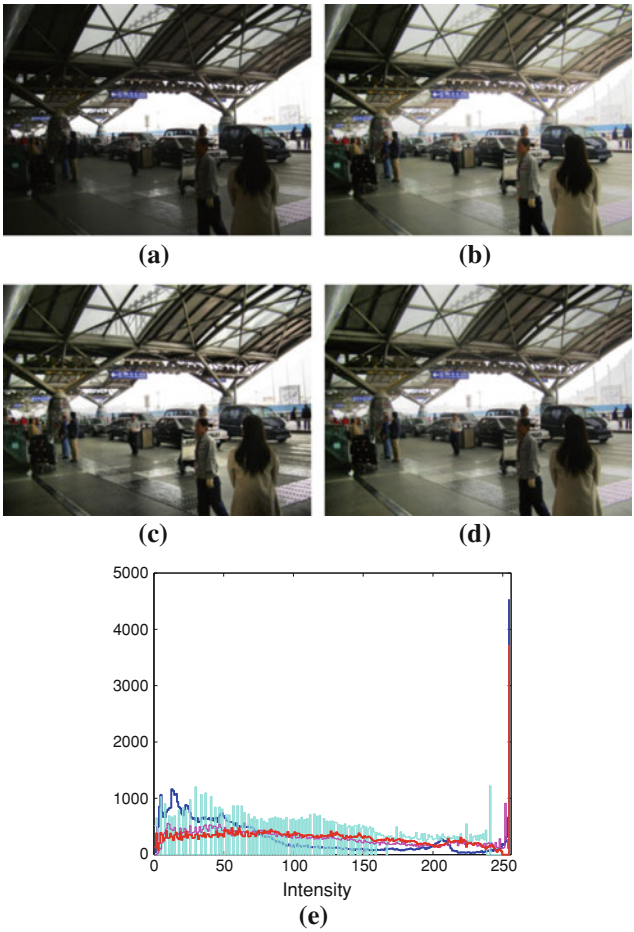


Fig. 2.6 Test results 1: **a** original image, **b** result from global equalization, **c** result from CLAHE, **d** result from proposed method, **e** plot of histograms, *blue*: original; *cyan*: globally equalized image; *magenta*: CLAHE; *red*: proposed method

2.4 Experiments and Discussion

Experiments were conducted to verify the proposed local equalization approach. A collection of 30 test images was taken under a variety of environment conditions including indoor, outdoor and cases of insufficient illuminations. The objective of enhancement includes recovering details in dark sectors that cannot be seen in the original images. In addition to objective viewing, a further assessment metric is also examined via the normalized entropy in Eq. (2.15). Here, the Shannon's entropy is adopted and calculated for all test images.

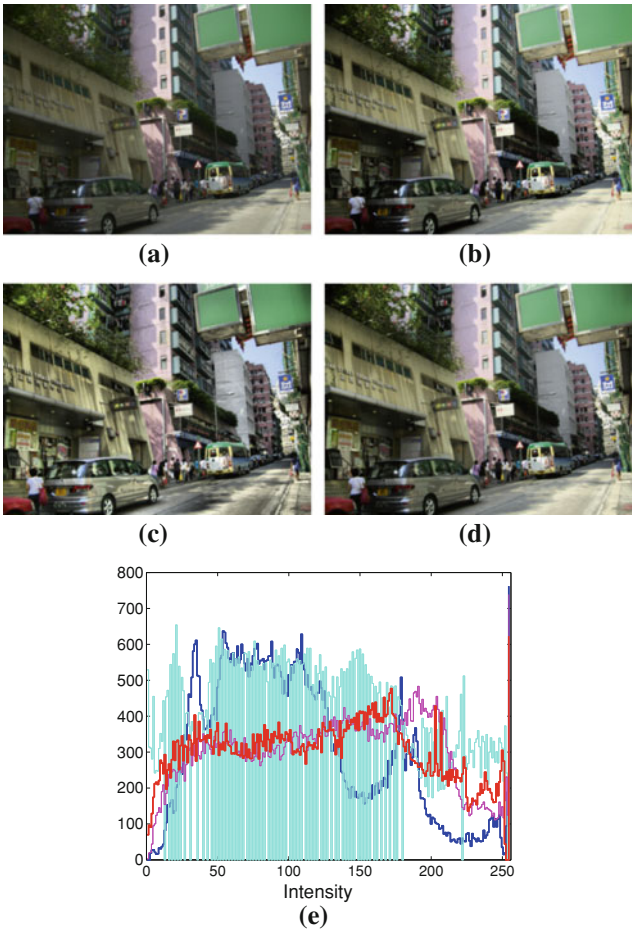


Fig. 2.7 Test results 2: **a** original image, **b** result from global equalization, **c** result from CLAHE, **d** result from proposed method, **e** plot of histograms, *blue*: original; *cyan*: globally equalized image; *magenta*: CLAHE; *red*: proposed method

Each image is captured in the RGB color space and of size 320 × 240 in width height dimensions. The PSO algorithm parameters are chosen as: ten PSO iterations, ten particles, and ten sectors encoded in each particle.

A sample of test images and their enhanced results is shown below. In the tests, the proposed method is compared to the canonical global histogram equalization method. Furthermore, results from the contrast-limited adaptive histogram equalization (CLAHE) method [26] in the Matlab implementation are also included.

As can be seen in Fig. 2.6, the performance of the CLAHE and the proposed methods both exhibit better performance than the globally equalized image in terms of contrast viewing. On the other hand, the performance of the CLAHE and the

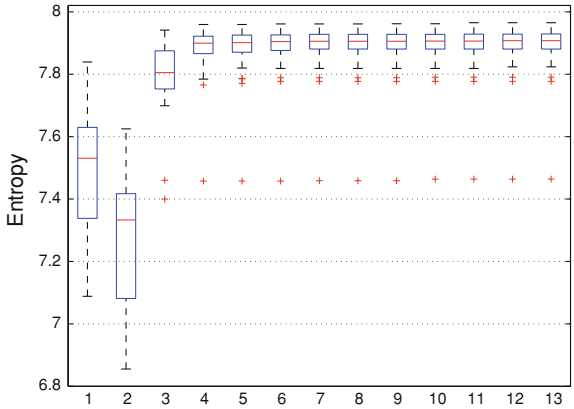


Fig. 2.8 Box plots of image entropies, original and enhanced. *First column* original image, *second column* global histogram equalization, *third column* clipped adaptive histogram equalization, fourth to 13th columns: traces of entropy over the iterations using the proposed method

local equalization method are comparable. Moreover, it is also observed subjectively that the CLAHE result contains some degree of over-equalization, particularly in the texture on the bottom-left corner of the image (Fig. 2.6c), where the CLAHE method produces a darker appearance. A careful inspection of the relevant histograms indicates that the one obtained from the proposed method is more uniform and thus contains a higher information content according to the Shannon's entropy measure.

Results from another test image are depicted in Fig. 2.7. Similar observations are also obtained for this test case. In the region around the mid-bottom of the image shown in Fig. 2.7c, the CLAHE approach also gives an over-equalization artefact that does not appear in the locally equalized image. The advantage of the proposed method over the CLAHE method is attributed to its randomly chosen sectorization and optimal tuning from the PSO algorithm.

The overall results from testing the total 30 images in separate runs are summarized in a box plot of information contents (entropies) shown in Fig. 2.8. Meritorious performance of the proposed method in comparison with others is illustrated using descriptive statistical quantities such as the median, quartiles and indications of outliers. The first column in the left represents the entropy of the original image. The second column denotes results from global histogram equalizations. It is seen that the information content with these methods generally drops down in value because part of the intensity range has not been fully utilized. This is indicated by the zero entries in the histogram as shown in Figs. 2.6e and 2.7e. The third column is obtained from results using the CLAHE approach where improvements in the information are noticeable. From column 4 to column 13, the entropies are shown with respect to the iterations performed during the proposed contrast enhancement process. It is evident that the proposed method has made an overall improvement over other methods implemented in the test. Furthermore, it is observed that the information content increases along with the iterations. This further verifies the effectiveness of the local equalization scheme proposed.

2.5 Conclusion

An image contrast enhancement method based on the concept of local histogram equalization is reported in this chapter. This method is useful for images with dark and low clarity regions due to severe limitations in the illumination condition when first captured. In this work, equalization is performed on a sectorized basis where the image is divided by a chosen division point. Each image sector is enhanced by matching to a uniformly-distributed target histogram. To reduce abrupt changes at the sector boundaries, a Gaussian mask is used with a weighted sum being obtained from information contents. The choice of the sector center and the mitigation of boundary discontinuities are determined optimally by adopting a PSO algorithm. The algorithm introduces iterative contrast enhancement, applicable to a vast variety of real-life scenes. The optimized procedure has been verified in a set of test results from real-world images by a comprehensive comparison with a number of contrast-enhancement approaches available in the literature.

References

1. Chen, Q., Xu, X., Sun, Q., Xia, D.: A solution to the deficiencies of image enhancement. Signal Process. **90**, 44–56 (2010)
2. Chen, S.D., Ramli, A.R.: Minimum mean brightness error bi-histogram equalization in contrast enhancement. IEEE Trans. Consumer Electron. **49**(4), 1310–1319 (2003)
3. Chen, S.Y., Li, Y.F., Zhang, J.W.: Vision processing for realtime 3d data acquisition based on coded structured light. IEEE Trans. Image Process. **17**(2), 167–176 (2008)
4. Cheng, H.D., Shi, X.J.: A simple and effective histogram equalization approach to image enhancement. Digit. Signal Process. **14**, 158–170 (2004)
5. Clerc, M., Kennedy, J.: The particle-swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6**(1), 58–73 (2002)
6. Frosio, J., Ferrigno, G., Borghese, N.A.: Enhancing digital cephalic radiography with mixture models and local gamma correction. IEEE Trans. Med. Imaging **25**(1), 113–121 (2006)
7. Hanmandlu, M., Jha, D., Sharma, R.: Color image enhancement by fuzzy intensification. Pattern Recognit. Lett. **24**, 81–87 (2003)
8. Kao, W.C., Chen, Y.J.: Multistage bilateral noise-filtering and edge detection for color image enhancement. IEEE Trans. Consumer Electron. **51**(4), 1346–1351 (2005)
9. Kim, J.Y., Kim, L.S., Hwang, S.H.: An advanced contrast enhancement using partially overlapped subblock histogram equalization. IEEE Trans. Circuits Syst. Video Technol. **11**(4), 475–484 (2001)
10. Kim, T.K., Park, J.K., Kang, B.S.: Contrast enhancement system using spatially adaptive histogram equalization with temporal filtering. IEEE Trans. Consumer Electron. **44**(1), 82–87 (1998)
11. Kong, N.S.P., Ibrahim, H.: Color image enhancement using brightness preserving dynamic histogram equalization. IEEE Trans. Consumer Electron. **54**(4), 1962–1967 (2008)
12. Kwok, N.M., Ha, Q.P., Liu, D.K., Fang, G.: Contrast enhancement and intensity preservation for gray-level images using multiobjective particle-swarm optimization. IEEE Trans. Autom. Sci. Eng. **6**(1), 145–155 (2009)
13. Lee, S., Chang, L.M., Skibniewski, M.: Automated recognition of surface defects using digital color image processing. Autom. Constr. **15**, 540–549 (2006)

14. Li, F., Jia, X., Fraser, D.: Superresolution reconstruction of multispectral data for improved classification. IEEE Geosci. Remote Sens. Lett. **6**(4), 689–693 (2009)
15. Naik, S.K., Murhty, C.A.: Hue-preserving color image enhancement without gamut problem. IEEE Trans. Image Process. **12**(12), 1591–1598 (2003)
16. Pei, S.C., Zeng, Y.C., Chang, C.H.: Virtual restoration of ancient chinese paintings using color contrast enhancement and lacuna texture synthesis. IEEE Trans. Image Process. **13**(3), 416–429 (2004)
17. Schavemaker, J.G.M., Reinders, M.J.T., Gerbrands, J.J., Backer, E.: Image sharpening by morphological filtering. Pattern Recognit. **33**, 997–1012 (2000)
18. Seow, M.J., Asari, V.K.: Ratio rule and homomorphic filter for enhancement of digital colour image. Neurocomputing **69**, 954–958 (2006)
19. Shyu, M.S., Leou, J.J.: A genetic algorithm approach to color image enhancement. Pattern Recognit. **31**(7), 871–880 (1998)
20. Starck, J.L., Murtagh, F., Candès, E.J., Donoho, D.L.: Gray and color image contrast enhancement by the curvelet transform. IEEE Trans. Image Process. **12**(6), 706–717 (2003)
21. Stark, J.A.: Adaptive image contrast enhancement using generalizations of histogram equalization. IEEE Trans. Image Process. **9**(5), 889–896 (2000)
22. Wang, C., Ye, Z.: Brightness preserving histogram equalization with maximum entropy: a variational perspective. IEEE Trans. Consumer Electron. **51**(4), 1326–1334 (2005)
23. Wang, D., Kwok, N.M., Liu, D.K., Ha, Q.P.: Ranked Pareto particle-swarm optimization for mobile robot motion planning. Design and Control of Intelligent Robotic Systems. In: Liu, D.K., Wang, L.F., Tan, K.C. (eds.) vol. 177/2009, pp. 97–118. Springer-Verlag, Berlin, Heidelberg (2009)
24. Wang, Y., Chen, Q., Zhang, B.: Image enhancement based on equal area dualistic subimage histogram equalization method. IEEE Trans. Consumer Electron. **45**(1), 68–75 (1999)
25. Zhu, H., Chan, H.Y., Lam, F.K.: Image contrast enhancement by constrained local histogram equalization. Comput. Vision Image Underst. **73**(2), 281–290 (1999)
26. Zuiderveld, K.: Contrast limited adaptive histogram equalization, pp. 474–485. Academic Press Professional, Inc., San Diego, CA, USA (1994).http://portal.acm.org/citation.cfm?id=180895.180940

Chapter 3
Hybrid BBO-DE Algorithms for Fuzzy Entropy-Based Thresholding

Ilhem Boussaïd, Amitava Chatterjee, Patrick Siarry and Mohamed Ahmed-Nacer

Abstract This chapter shows how a recently proposed stochastic optimization algorithm, called biogeography-based optimization (BBO), can be efficiently employed for development of three-level thresholding-based image segmentation. This technique is utilized to determine suitable thresholds utilizing a fuzzy entropy-based fitness function, which the optimization procedure attempts to maximize. The chapter demonstrates how improved BBO-based strategies, employing hybridizations with differential evolution (DE) algorithms, can be employed to incorporate diversity in the basic BBO algorithm that can help the optimization algorithm avoid getting trapped at local optima and seek the global optimum in a more efficient manner. Several such hybrid BBO-DE algorithms have been utilized for this optimum thresholding-based image segmentation procedure. A detailed implementation analysis for a popular set of well-known benchmark images has been carried out to qualitatively and quantitatively demonstrate the utility of the proposed hybrid BBO-DE optimization algorithm.

I. Boussaïd (✉) · M. Ahmed-Nacer
University of Science and Technology Houari Boumediene (USTHB),
El-Alia BP 32, Bab-Ezzouar, 16111 Algiers, Algeria
e-mail: ilhem.boussaid@univ-paris12.fr

M. Ahmed-Nacer
e-mail: anacer@cerist.dz

A. Chatterjee
Electrical Engineering Department, Jadavpur University,
Kolkata, 700 032 West Bengal, India
e-mail: cha_ami@yahoo.co.in

P. Siarry
Université de Paris-Est Créteil Val de Marne, LiSSi (EA 3956)
61 avenue du Général de Gaulle, 94010 Créteil, France
e-mail: siarry@univ-paris12.fr

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_3, © Springer-Verlag Berlin Heidelberg 2013

3.1 Introduction

Image segmentation is the process of decomposing an image into a set of regions which are visually distinct and uniform with respect to some feature. This distinguishing feature may be colour or gray-level information that is used to create histograms, or information about the pixels that indicate edges or boundaries or texture information [10]. There is a wide range of image segmentation techniques available in the literature [20, 24]. Among them, one approach of particular interest is the thresholding approach, because of its efficiency in performance and its theoretical simplicity. A comprehensive survey of image thresholding techniques is found in [26].

Thresholding techniques can be classified as bilevel and multilevel thresholding, depending on number of image segments into which an original image is decomposed. In bilevel thresholding, each image pixel is assigned to one of two brightness regions, object and background, according to whether its intensity (gray level or colour) is greater than a specified threshold T or not. In multilevel thresholding, pixels can be classified into many classes, not just foreground and background. Therefore, more than one threshold should be determined to segment the image into certain brightness regions which may correspond to one background and several objects.

Among the multitude of image thresholding techniques, entropy-based approaches have drawn a lot of attention in recent times. The principle of entropy, a well-known concept from information theory introduced by Shannon [6], is to use uncertainty as a measure to describe the information contained in a source. The maximum information is achieved when no a priori knowledge is available, in which case, it results in maximum uncertainty.

Basically, entropy thresholding considers an image histogram as a probability distribution, and determines an optimal threshold value that yields the maximum entropy. More specifically, the best entropy thresholded image is the one that preserves as much information as possible that is contained in the original unthresholded image in terms of Shannon's entropy [4].

The popular criterion for image thresholding based on maximum entropy principle was first applied by Pun [22, 23] and then improved upon in [12]. The concept was later generalized to evolve to Renyi's entropy [25] and Tsallis's entropy [33].

However, due to the possible multivalued levels of brightness in a gray-tone image, or inherent vagueness and imprecision embedded in images, the result of image thresholding is not always satisfactory. This uncertainty can be adequately analyzed through the use of fuzzy set theory [1]. This theory, proposed by Zadeh [35], is a mathematical tool to analyze vagueness and uncertainty inherent in making decisions. It has proved its efficiency and usefulness in many applications, including image processing problems. In fact, some fuzzy logic-based thresholding techniques have been proposed in the literature, where fuzzy theory is employed to select an optimal threshold by maximizing the fuzzy entropy [30, 32, 36].

Cheng et al. [5] introduced the concept of fuzzy c-partition into the maximum entropy principle to select the threshold values for gray-level images. This method was first applied for bilevel thresholding and then extended to multilevel thresholding.

Tobias and Serra [32] proposed an approach for histogram thresholding which was not based on the minimization of a threshold-dependent criterion function. The histogram threshold was determined according to the similarity between gray levels, assessed through implementation of a fuzzy measure.

In [36], Zhao et al. proposed a new technique for three-level thresholding by exploiting the relationship between the fuzzy c-partition and the probability partition. In their proposed technique, the maximum entropy principle was used to measure the compatibility between fuzzy partition and probability partition. Zhao et al. used the simplest function, which is monotonic in nature, to approximate the memberships of the bright, dark and medium fuzzy sets (defined according to pixel intensity levels) and derived a necessary condition of the entropy function arriving at a maximum. Based on the idea of Zhao et al., Tao et al. [31] designed a new three-level thresholding method for image segmentation. The authors defined a new concept of fuzzy entropy through probability analysis, fuzzy partition and entropy theory. The image is first partitioned into three parts, namely dark, gray and white, whose member functions of fuzzy region are described by a Z-function, a Π-function and a S-function, respectively. Later, Tao et al. [30] developed another system which examined the performance of their previous approach for the segmentation of infrared objects. This approach entailed the use of the ant colony optimization (ACO) method to effectively obtain the optimal combination of the free parameters of the fuzzy sets. The experimental results showed that the implementation of the proposed fuzzy entropy principle by ACO had more highly effective search performance than the genetic algorithm (GA) used in [31].

The present work aims at developing a new three-level thresholding algorithm, called DBBO-Fuzzy, based on the hybridization of biogeography-based optimization (BBO), a very recently proposed population based optimization technique, and the differential evolution (DE) algorithm, a very powerful stochastic optimizer. The approach presented in [31] is adopted as the basic support for this work.

A hybrid DE with BBO, namely DE/BBO, has been proposed in [11] where a hybrid migration operator is defined. Another combination of BBO and DE is proposed in [3], where the population is updated by applying, alternately from an iteration of the algorithm to the next, the BBO and DE updating methods. The proposed DBBO-Fuzzy algorithm incorporates the mutation procedure inherited from DE algorithm to replace the existing mutation procedure in BBO. A selection operator is also introduced in order to favor a given number of individuals for the next generation. In addition, the algorithm incorporates the features of elitism in order to prevent the best solutions from getting corrupted. The proposed algorithm is tested on a popular set of well-known benchmark images, and experimental results show that the proposed approach is reliable and efficient.

First of all, it is necessary to present some basic definitions. Section 3.2 provides the preliminaries of the fuzzy set theory and fuzzy entropy formulation, where the terminology used in [35] has been followed. The three-level thresholding problem is then formulated and the assumptions made in this paper are introduced in Sect. 3.3. Section 3.4 briefly describes the conventional BBO algorithm. The proposed algorithm is presented in Sect. 3.4, and the effectiveness of the proposed algorithm,

along with a comparison with BBO and DE-based algorithms, is demonstrated for a set of benchmark images in Sect. 3.6. Finally, Sect. 3.7 presents our conclusions.

3.2 Fuzzy Set Theory

Fuzzy set theory is a generalization of classical set theory, designed to express uncertainty and imprecision in available knowledge. The theory dates back to 1965, when Lotfi Zadeh, a professor at Berkeley, published his seminal paper on the theory of fuzzy sets and the associated logic, namely fuzzy logic [35].

Essentially, the fuzziness, a feature of imperfect information, results from the lack of crisp distinction between the elements belonging and not belonging to a set.

Definition 3.1 Let X be a universe of discourse with a generic element denoted by x_i: $X = \{x_1, x_2, \ldots, x_n\}$.

A fuzzy set A in a space X is formally defined as:

$$A = \{(x_i, \mu_A(x_i)) \,|x \in X\} \tag{3.1}$$

in which $\mu_A : X \to [0, 1]$ is the membership function or characteristic function. This function assigns to each element x_i in the set a membership grade $\mu_A(x_i) \in [0, 1]$. As opposed to classical sets, where each element must have either 0 as the membership grade if the element is completely outside the set or 1 if the element is completely in the set, the theory of fuzzy sets is based on the idea that one is uncertain about whether the element is in or out of the set. Thus, the nearer the value of $\mu_A(x_i)$ to unity, the higher the grade of membership of x_i in A. The fuzzier case and the more difficult one is when $\mu_A(x_i) = 0.5$ [15].

The fuzzy set theory approach has found interesting applications in automatic control, decision making, pattern recognition, psychology, economics, medical diagnosis, image processing and other fields. A number of aspects of digital image processing have been treated by this theory, such as image quality assessment, edge detection, image segmentation, etc.

3.2.1 Fuzzy Entropy

A very frequent measure of fuzziness is referred to as the fuzzy entropy inspired by the Shannon entropy of random variables [27] and introduced for the first time by De Luca and Termini [8]. The authors established the following four axioms for fuzzy entropy:

Definition 3.2 Let E be a set-to-point map $E : F(X) \to [0, 1]$. Hence E is a fuzzy set defined on fuzzy sets and $F(X)$ is the family of all fuzzy sets in X. E is an entropy measure if it satisfies the four De Luca and Termini axioms:

1. $E(A) = 0$ iff $A \in X$ (A nonfuzzy).
2. $E(A) = 1$ (the maximum value) iff $\mu_A(x) = 0.5, \forall x \in X$.
3. $E(A) \leq E(B)$ if A is less fuzzy than B, i.e., if $\mu_A(x) \leq \mu_B(x)$ when $\mu_B(x) \leq 0.5$, and $\mu_A(x) \geq \mu_B(x)$ when $\mu_B(x) \geq 0.5$.
4. $E(A) = E(A^c)$, where A^c is the complementary set of A.

A number of studies related to the measures of fuzzy entropy and their applications have been conducted by Kaufmann [13], Bhandari and Pal [2], Kaufmann [14], Pal and Pal [19], and Fan and Xie [9].

3.3 Problem Formulation

3.3.1 Model of an Image

A digital gray-tone image refers to a two-dimensional light intensity function defined over a spatial coordinate system. Let $G = \{0, 1, \ldots, L-1\}$ be the set of intensity values, and $D = \{(x, y) : 0 \leq x \leq M-1, 0 \leq y \leq N-1\}$ be the spatial coordinates of the pixels for an MxN image. The digital image defines a mapping $I : D \longrightarrow G$, where $0 \leq I(x, y) \leq L-1$ gives the intensity (brightness) of the image at the spatial coordinates $(x, y) \in D$ with $L = 256$ gray levels for an 8-bit image [30, 31].

Let $D_k = \{(x, y) | I(x, y) = k, (x, y) \in D\}, k \in G$. The histogram of an image, defined as $H = \{h_0, h_1, \ldots, h_{L-1}\}$, presents the frequency of occurrence of each gray level in the image and is obtained directly from the observation of the considered image. In view of this consideration, the kth gray level in the image is defined as follows:

$$h_k = \frac{n_k}{N * M}, \quad k = 0, 1, \ldots, L-1 \tag{3.2}$$

where n_k denotes the total number of pixels in D_k, and $N * M$ denotes the total number of pixels in the image. It is clear that

$$0 \leq h_k \leq 1 \quad \text{and} \quad \sum_{k=0}^{L-1} h_k = 1 \tag{3.3}$$

A probability partition (PP) of the image domain D is defined as

$$\Pi_L = \{D_0, D_1, \ldots, D_{L-1}\}$$

which is characterized by a probabilistic distribution [30, 31]:

$$p_k \equiv P(D_k) = h_k, \quad k = 0, 1, \ldots, L-1, \tag{3.4}$$

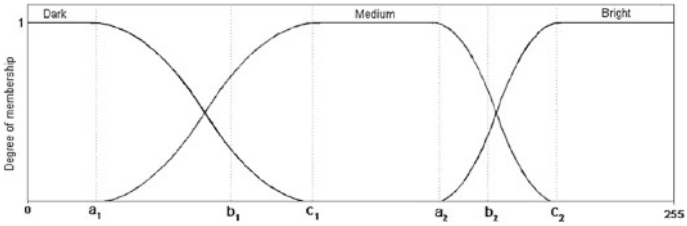


Fig. 3.1 Membership function graph

Equation (3.4) presents the relationship between the histogram H and the probability partition Π, where p_k is the probability measure of the occurrence of gray level k.

3.3.2 Three-Level Thresholding

The segmentation problem is to determine the sets $D_k \subset D$ $(k = 0, \ldots, L-1)$ whose union is the entire image D. Thus, the sets that constitute the segmentation must satisfy:

$$\bigcup_{k=0}^{L-1} D_k = D \quad \text{and} \quad D_i \cap D_j = \phi \quad (i \neq j) \tag{3.5}$$

where ϕ denotes an empty set. Ideally, a segmentation method finds those sets that correspond to distinct anatomical structures or regions of interest in the image.

In the case of three-level thresholding of an image, the aim is to separate its domain D into three parts, E_d, E_m and E_b, where E_d is composed of 'dark' pixels corresponding to the smaller gray levels, E_b is composed of those 'bright' pixels corresponding to the larger gray levels, and E_m is composed of pixels with medium gray levels.

The problem is to find the unknown probabilistic fuzzy 3-partition of D, $\Pi_3 = \{E_d, E_m, E_b\}$, which is characterized by the probability distributions [30, 31]:

$$p_d = P(E_d), \quad p_m = P(E_m), \quad p_b = P(E_b) \tag{3.6}$$

The three fuzzy partitions E_d, E_m and E_b are characterized by three membership functions μ_d, μ_m and μ_b, respectively. The membership function μ_d of dark pixels corresponds to a Z-function, the membership function μ_m of medium pixels is a Π-function, and the membership function μ_b of bright pixels of the image is a S-function [31] (see Fig. 3.1). Six free parameters $\{a_1, b_1, c_1, a_2, b_2, c_2\}$ control the shapes of the three membership functions and satisfy the conditions $0 < a_1 \leq b_1 \leq c_1 \leq a_2 \leq b_2 \leq c_2 < 255$ for an image with 256 gray levels.

The three-level thresholding involves a determination of the optimal thresholds T_1 and T_2 such that the classification of a pixel $I(x, y)$ is achieved as follows:

$$\begin{aligned} D_{kd} &= \{(x, y) : I(x, y) \leq T_1, (x, y) \in D_k\} \\ D_{km} &= \{(x, y) : T_1 < I(x, y) \leq T_2, (x, y) \in D_k\} \\ D_{kb} &= \{(x, y) : I(x, y) > T_2, (x, y) \in D_k\} \end{aligned} \tag{3.7}$$

Once the parameters a_1, b_1, c_1, a_2, b_2 and c_2 are selected then $\Pi_k = \{D_{kd}, D_{km}, D_{kb}\}$ is the PP of D_k with the probabilistic distribution:

$$\begin{aligned} p_{kd} &= P(D_{kd}) = p_k . p_{d|k} \\ p_{km} &= P(D_{km}) = p_k . p_{m|k} \\ p_{kb} &= P(D_{kb}) = p_k . p_{b|k} \end{aligned} \tag{3.8}$$

In Eq. (3.8), $p_{d|k}$ is the conditional probability of a pixel that is classified into the class "d" (dark) under the condition that the pixel belongs to D_k. Similarly, $p_{m|k}$ and $p_{b|k}$ are the conditional probabilities of a pixel belonging to classes "m" (medium) and "b" (bright), respectively.

Based on the complete probability formula, we therefore have:

$$\begin{aligned} p_d &= \sum_{k=0}^{255} p_k . p_{d|k} = \sum_{k=0}^{255} p_k . \mu_d(k) \\ p_m &= \sum_{k=0}^{255} p_k . p_{m|k} = \sum_{k=0}^{255} p_k . \mu_m(k) \\ p_b &= \sum_{k=0}^{255} = p_k . p_{b|k} = \sum_{k=0}^{255} p_k . \mu_b(k) \end{aligned} \tag{3.9}$$

Based on Eq. (3.9), it is clear that the three-level thresholding problem is reduced to finding suitable membership functions $\mu_d(k)$, $\mu_m(k)$ and $\mu_b(k)$ of a pixel with an arbitrary intensity level k. These membership functions represent the conditional probability that a pixel is classified into the dark, medium and bright regions, respectively, with respect to the variable $k \in G$ (i.e., $p_{d|k} = \mu_d(k)$, $p_{m|k} = \mu_m(k)$ and $p_{b|k} = \mu_b(k)$). It is obvious that $\mu_d(k) + \mu_m(k) + \mu_b(k) = 1, k = 0, 1, \ldots, 255$. The three membership functions are shown in Eqs. (3.10)–(3.12):

$$\mu_d(k) = \begin{cases} 1 & k \leq a_1 \\ 1 - \frac{(k-a_1)^2}{(c_1-a_1)*(b_1-a_1)} & a_1 < k \leq b_1 \\ \frac{(k-c_1)^2}{(c_1-a_1)*(c_1-b_1)} & b_1 < k \leq c_1 \\ 0 & k > c_1 \end{cases} \tag{3.10}$$

$$\mu_m(k) = \begin{cases} 0 & k \le a_1 \\ \frac{(k-a_1)^2}{(c_1-a_1)*(b_1-a_1)} & a_1 < k \le b_1 \\ 1 - \frac{(k-c_1)^2}{(c_1-a_1)*(c_1-b_1)} & b_1 < k \le c_1 \\ 1 & c_1 < k \le a_2 \\ 1 - \frac{(k-a_2)^2}{(c_2-a_2)*(b_2-a_2)} & a_2 < k \le b_2 \\ \frac{(k-c_2)^2}{(c_2-a_2)*(c_2-b_2)} & b_2 < k \le c_2 \\ 0 & k > c_2 \end{cases} \tag{3.11}$$

$$\mu_b(k) = \begin{cases} 0 & k \le a_2 \\ \frac{(k-a_2)^2}{(c_2-a_2)*(b_2-a_2)} & a_2 < k \le b_2 \\ 1 - \frac{(k-c_2)^2}{(c_2-a_2)*(c_2-b_2)} & b_2 < k \le c_2 \\ 1 & k > c_2 \end{cases} \tag{3.12}$$

The free parameters of the three membership functions can be determined by maximizing the fuzzy partition entropy [30, 31]. The total fuzzy entropy function of partition Π_3 is defined as:

$$H\left(a_1, b_1, c_1, a_2, b_2, c_2\right) = H_d + H_m + H_b \tag{3.13}$$

where,

$$\begin{aligned} H_d &= -\sum_{k=0}^{255} \frac{p_k * \mu_d\left(k\right)}{p_d} * \ln\left(\frac{p_k * \mu_d\left(k\right)}{p_d}\right) \\ H_m &= -\sum_{k=0}^{255} \frac{p_k * \mu_m\left(k\right)}{p_m} * \ln\left(\frac{p_k * \mu_m\left(k\right)}{p_m}\right) \\ H_b &= -\sum_{k=0}^{255} \frac{p_k * \mu_b\left(k\right)}{p_b} * \ln\left(\frac{p_k * \mu_b\left(k\right)}{p_b}\right) \end{aligned} \tag{3.14}$$

The best-selected set of $\{a_1, b_1, c_1, a_2, b_2, c_2\}$ is the one that corresponds to maximum entropy H.

The optimal thresholds T_1 and T_2 that segment the image into three gray levels are obtained as the intersections of the membership function curves, i.e.,

$$\mu_d\left(T_1\right) = \mu_m\left(T_1\right) = 0.5 \tag{3.15}$$

$$\mu_m\left(T_2\right) = \mu_b\left(T_2\right) = 0.5 \tag{3.16}$$

Based on Eqs. (3.10)–(3.12), it can be written [31]:

$$T_1 = \begin{cases} a_1 + \sqrt{(c_1 - a_1) * (b_1 - a_1)/2} & \text{if } (a_1 + c_1)/2 \leq b_1 \leq c_1 \\ c_1 - \sqrt{(c_1 - a_1) * (c_1 - b_1)/2} & \text{if } a_1 \leq b_1 < (a_1 + c_1)/2 \end{cases} \tag{3.17}$$

$$T_2 = \begin{cases} a_2 + \sqrt{(c_2 - a_2) * (b_2 - a_2)/2} & \text{if } (a_2 + c_2)/2 \leq b_2 \leq c_2 \\ c_2 - \sqrt{(c_2 - a_2) * (c_2 - b_2)/2} & \text{if } a_2 \leq b_2 < (a_2 + c_2)/2 \end{cases} \tag{3.18}$$

As mentioned before, to find the optimal combination of all the fuzzy parameters, we propose to use a new optimization technique based on the hybridization of BBO and DE algorithms.

3.4 Biogeography-Based Optimization Algorithm (BBO)

The theory of biogeography grew out of the works of Wallace [34] and Darwin [7] in the past and the works of McArthur and Wilson [17] more recently. Some of the key questions that this branch of biology attempts to answer are: How do organisms reach their current habitats? Do they always occupy their current distribution patterns? Why does an ecosystem have a particular number of species? The patterns of the distribution of the species across geographical areas can usually be explained through a combination of historical factors, such as speciation, extinction and migration.

The biogeography-based optimization (BBO) algorithm, developed by Dan Simon [28], is strongly influenced by the equilibrium theory of island biogeography [17]. The basic premise of this theory is that the rate of change in the number of species on an island depends critically on the balance between the immigration of new species onto the island and the emigration of species from the island.

The BBO algorithm operates upon a population of individuals called islands (or habitats). Each habitat represents a possible solution for the problem at hand. The fitness of each habitat is determined by its habitat suitability index (*HSI*), a metric which determines the goodness of a candidate solution, and each habitat feature is called a suitability index variable (*SIV*). Good solutions may have large number of species, which represent habitats with a lower *HSI* than the poor solutions.

As was mentioned before, the migration pattern is determined by the immigration rate (λ) at which new species immigrate to the habitat, and the emigration rate (μ) at which populations of established species emigrate. These parameters are functions of the number of species in a habitat.

But how might immigration and emigration work on a habitat? We make two sets of assumptions regarding these processes:

Immigration: The rate of immigration (λ) declines with the number of species (S) present on the habitat. Maximum immigration rate (I) occurs when the habitat is empty and decreases as more species are added. Once all the potential colonists are on the habitat, then one can write $S = S_{max}$ (the maximum number of species the habitat can support), and the immigration rate must be equal to zero. Generally speaking,

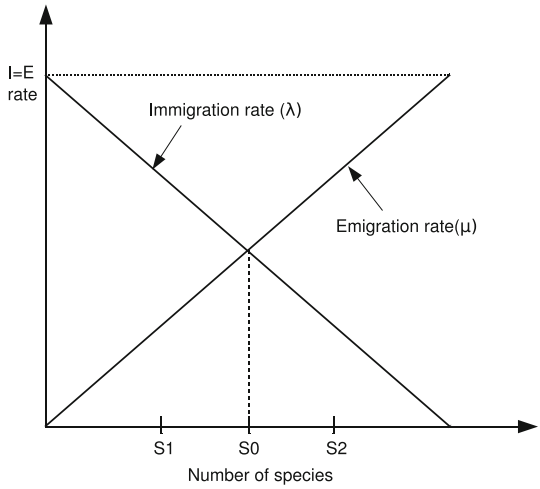


Fig. 3.2 The relationship between the fitness of habitats (number of species), emigration rate μ and immigration rate λ

the immigration rate when there are S species in the habitat is given by:

$$\lambda_S = I\left(1 - \frac{S}{S_{max}}\right) \tag{3.19}$$

Emigration: The rate of emigration (μ) for a habitat increases with the number of species (S). Maximum emigration rate (E) occurs when all possible species are present on the habitat (i.e., when $S = S_{max}$), and must be zero when no species are present. Generally speaking, the emigration rate when there are S species in the habitat is given by:

$$\mu_S = E\left(\frac{S}{S_{max}}\right) \tag{3.20}$$

Figure 3.2 graphically represents the relationships between the number of species, emigration rate μ and immigration rate λ. Over a period of time, the counteracting forces of emigration and immigration result in an equilibrium level of species richness. The equilibrium value S^* is the point at which the rate of arrival of species λ is exactly matched by the rate of emigration μ. We have assumed here that μ and λ vary following linear relationships, but different mathematical models of biogeography that include more complex variations have also been proposed [17].

We now consider the probability P_s that the habitat contains exactly S species. The number of species changes from time t to time $(t + \Delta t)$ as follows [28]:

$$P_s(t + \Delta t) = P_s(t)(1 - \lambda_s \Delta t - \mu_s \Delta t) + P_{s-1}\lambda_{s-1}\Delta t + P_{s+1}\mu_{s+1}\Delta t \tag{3.21}$$

This states that the number of species on the habitat in one time step is based on the total number of current species on the habitat, the new immigrants and the number of species that leave the habitat during this time period. We assume here that Δt is small enough so that the probability of more than one immigration or emigration can be ignored. In order to have S species at time $(t + \Delta t)$, one of the following conditions must hold:

- There were S species at time t, and no immigration or emigration occurred between t and $(t + \Delta t)$;
- One species immigrated onto a habitat already occupied by $S - 1$ species at time t.
- One species emigrated from a habitat occupied by $S + 1$ species at time t.

The limit of Eq. (3.21) as $\Delta t \to 0$ is given by Eq. (3.22).

$$\dot{P}_s = \begin{cases} -(\lambda_s + \mu_s)P_S + \mu_{s+1}P_{s+1} & \text{if } S = 0 \\ -(\lambda_s + \mu_s)P_S + \lambda_{s-1}P_{s-1} + \mu_{s+1}P_{s+1} & \text{if } 1 \le S \le S_{max} - 1 \\ -(\lambda_s + \mu_s)P_S + \lambda_{s-1}P_{s-1} & \text{if } S = S_{max} \end{cases} \tag{3.22}$$

Equation (3.22) can be arranged into a single matrix form:

$$\begin{bmatrix} \dot{P}_0 \\ \dot{P}_1 \\ \vdots \\ \vdots \\ \dot{P}_n \end{bmatrix} = \begin{bmatrix} -(\lambda_0 + \mu_0) & \mu_1 & 0 & \dots & 0 \\ \lambda_0 & -(\lambda_1 + \mu_1) & \mu_2 & \dots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \lambda_{n-2} & -(\lambda_{n-1} + \mu_{n-1}) & \mu_n \\ 0 & \dots & 0 & \lambda_{n-1} & -(\lambda_n + \mu_n) \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ \vdots \\ \vdots \\ P_n \end{bmatrix} \tag{3.23}$$

For notational brevity, we simply write $n = S_{max}$.

The BBO algorithm can be described overall by Algorithm 3.1. The two basic operators which govern the working of BBO are the migration, described in Algorithm 3.2, and the mutation, described in Algorithm 3.3, where $rand(0, 1)$ is a uniformly distributed random number in the interval [0, 1]; X_{ij} is the jth SIV of the solution X_i.

The likelihood that a given solution S is expected a priori to exist as a solution for the given problem is indicated by the species count probability P_s. In this context it should be remarked that very high *HSI* solutions and very low *HSI* solutions are both equally improbable. Medium *HSI* solutions are relatively probable. If a given

solution has a low probability, then it is likely to mutate to some other solution. Conversely, a solution with high probability is less likely to mutate. The mutation rate $m(S)$ is inversely proportional to the solution probability:

$$m(S) = m_{max}\left(1 - \frac{P_S}{P_{max}}\right) \tag{3.24}$$

where m_{max} is a user-defined parameter, and $P_{max} = \max_S P_S$, $S = 1, \ldots, S_{max}$.

Migration is used to modify existing habitats by mixing features within the population. Mutation is used to enhance diversity of the population, thereby preventing the search from stagnating. If a habitat S is selected to execute the mutation operation, then a chosen variable (SIV) is randomly modified based on its associated probability P_S. At the same time, the concept of elitism (i.e., copying some of the fittest individuals for the next generation) is also applied.

Algorithm 3.1 Biogeography-based optimization (BBO)

```
Initialize the BBO parameters: S_max, E, I, m_max, Max_gen, n_elit, ...
Initialize the generation counter : g = 0
Create a random initial population X_i, i = 1, ..., NP
Evaluate f(X_i), i = 1, ..., NP
for g = 1 to Max_gen do
    Sort the population from best fit to least fit
    for i = 1 to NP do
        Map the HSI to the number of species
        Calculate the immigration rate λ_i and the emigration rate μ_i
        Modify the nonelite members of the population probabilistically with the migration
            operator according to Algorithm 3.2
    end for
    for i = 1 to NP do
        Mutate the non-elite members of the population with the mutation operator according
            to Algorithm 3.3
    end for
    for i = 1 to NP do
        Evaluate the new habitats in the population
        Replace the habitats with their new versions
        Apply elitism to preserve n_elit best habitats
    end for
end for
```

3.5 Description of the Proposed DBBO-Fuzzy Algorithm

Motivated by the exploration capabilities of the differential evolution (DE) algorithm [18], a hybrid method combining the exploitation of BBO with the exploration of DE is proposed in this paper. The purpose of this hybridization is to benefit from the advantages of each algorithm and to compensate for each algorithm's weaknesses.

Algorithm 3.2 Migration

```
for i = 1 to NP do
    Use λ_i to probabilistically decide whether to immigrate to X_i.
    if rand(0, 1) < λ_i then
        for j = 1 to NP do
            Select the emigrating habitat X_j with probability ∝ μ_j
            if rand(0, 1) < μ_j then
                Replace a randomly selected decision variable (SIV) of X_i with its corresponding
                    variable in X_j
            end if
        end for
    end if
end for
```

Algorithm 3.3 Mutation

```
for i = 1 to NP do
    Compute the probability P_i using λ_i and μ_i
    Use the probability P_i to compute the mutation rate m_i
    for j = 1 to D do
        Select a variable (SIV) X_ij with probability ∝ P_i
        if rand(0, 1) < m_i then
            Replace X_ij with a randomly generated variable from its range
        end if
    end for
end for
```

In order to find the global solution in a better manner than the BBO algorithm, the proposed algorithm, named the DBBO-Fuzzy algorithm, replaces the BBO-based mutation by a DE mutation. In addition, a selection operation is introduced in order to favor a given number of individuals for the next generation.

The proposed algorithm can be summarized as follows:

1. Initialization The algorithm starts with an initial population of NP search variable vectors (or habitats), where the problem dimension D is the number of fuzzy parameters. For the three-level thresholding problem, six parameters $\{a_1, b_1, c_1, a_2, b_2, c_2\}$ are used (i.e., $D = 6$). Since the habitats are likely to get modified over different generations, the following notation may be adopted for representing the ith habitat of the population at the current generation g as:

$$\mathbf{X}_{i,g} = (X_{i,1,g}, X_{i,2,g}, X_{i,j,g}, \ldots, X_{i,D,g}) \tag{3.25}$$

where $i = 1, \ldots, NP$, $j = 1, \ldots, D$ and $X_{i,j}$ is the jth SIV of the habitat $\mathbf{X}_i$. Each decision variable, $X_{i,j,g}$, is randomly initialized within its corresponding lower bound (L_j) and upper bound (U_j), and it is intended to cover the entire search space uniformly in the form:

$$X_{i,j,0} = L_j + rand(0, 1) \times (U_j - L_j) \tag{3.26}$$

2. Evaluation of the objective function: The objective function values of the habitats are evaluated using the fuzzy entropy function given by Eq. (3.13). The objective function has six parameters (SIV) $a_1, b_1, c_1, a_2, b_2, c_2$, which satisfy the conditions $0 < a_1 \leq b_1 \leq c_1 \leq a_2 \leq b_2 \leq c_2 < 255$.

3. Migration: The migration operator reproduces a new population vector $\mathbf{M}_{i,g}$ as follows:

$$M_{i,j,g} = \begin{cases} X_{k,j,g} & \text{if } rand(0,1) < \lambda_i \\ X_{i,j,g} & \text{otherwise} \end{cases} \tag{3.27}$$

where $i = 1, 2, \ldots, NP$, $j = 1, \ldots, D$ and $X_{k,j,g}$ is the jth decision variable of a randomly selected habitat $\mathbf{X}_{k,g}$. $\mathbf{X}_{k,g}$ is selected with a probability based on μ_k.

4. DE Mutation: The DBBO-Fuzzy incorporates the mutation procedure inherited from DE algorithm [21, 29] to replace the existing mutation procedure in BBO.
The mutation is performed by calculating weighted vector differences between other randomly selected habitats of the same population. A differentiation constant F is used to control the amplification of the differential variation.
Next, five different mutation schemes are outlined, inspired by the suggestions of Price et al. [21]. The general convention used to name the different DE schemes is $DE/x/y$. Here DE stands for differential evolution, x represents a string denoting the type of the vector to be perturbed (whether it is randomly selected or it is the best vector in the population with respect to fitness value) and y is the number of difference vectors considered for perturbation of x.
The mutation operation constructs, for each habitat $\mathbf{M}_{i,g}$, a mutant habitat $\mathbf{V}_{i,g}$ according to one of the following mutation schemes:

- **DE/rand/1**: This mutation scheme uses a randomly selected habitat $\mathbf{M}_{r_1,g}$, and only one weighted difference vector $F.(\mathbf{M}_{r_2,g} - \mathbf{M}_{r_3,g})$ is used to perturb it.

$$\mathbf{V}_{i,g} = \mathbf{M}_{r_1,g} + F.(\mathbf{M}_{r_2,g} - \mathbf{M}_{r_3,g}) \tag{3.28}$$

- **DE/current to best/1**: Here the mutant habitat is created using any two randomly selected habitats of the population as well as the best habitat in the current generation.

$$\mathbf{V}_{i,g} = \mathbf{M}_{i,g} + F.(\mathbf{M}_{best,g} - \mathbf{M}_{i,g}) + F.(\mathbf{M}_{r_1,g} - \mathbf{M}_{r_2,g}) \tag{3.29}$$

- **DE/best/1**: Here the habitat to be perturbed is the best habitat of the current population and the perturbation is caused by using a single difference vector.

$$\mathbf{V}_{i,g} = \mathbf{M}_{best,g} + F.(\mathbf{M}_{r_1,g} - \mathbf{M}_{r_2,g}) \tag{3.30}$$

- **DE/rand/2**: to create $\mathbf{V}_{i,g}$ for each ith habitat, a total of five other distinct habitats (say the r_1, r_2, r_3, r_4, and r_5th habitats) are chosen in a random manner from the current population.

$$\mathbf{V}_{i,g} = \mathbf{M}_{r_1,g} + F.(\mathbf{M}_{r_2,g} - \mathbf{M}_{r_3,g}) + F.(\mathbf{M}_{r_4,g} - \mathbf{M}_{r_5,g}) \tag{3.31}$$

- **DE/best/2**: in this mutation scheme, the mutant habitat is formed by using two difference vectors, as shown below:

$$\mathbf{V}_{i,g} = \mathbf{M}_{best,g} + F.(\mathbf{M}_{r_1,g} - \mathbf{M}_{r_2,g}) + F.(\mathbf{M}_{r_3,g} - \mathbf{M}_{r_4,g}) \tag{3.32}$$

where the indices r_1, r_2, r_3, r_4, r_5 are randomly chosen over the interval $[1, NP]$ and should be mutually different from the running index i. F is a real constant scaling factor within the range [0, 2], usually chosen to be less than 1. $\mathbf{M}_{best,g}$ is the habitat with best fitness value in the population in generation g.

5. Selection: The values of the objective function are calculated for the updated habitats. The selection operation selects either a habitat $\mathbf{X}_{i,g}$ or its newly updated habitat $\mathbf{V}_{i,g}$ to survive as a member for the next generation, according to the fitness value. For the following generation $g+1$, new habitats $\mathbf{X}_{i,g+1}$ are selected according to the following selection rule:

$$\mathbf{X}_{i,g+1} = \begin{cases} \mathbf{V}_{i,g} \text{ if } f(\mathbf{V}_{i,g}) < f(\mathbf{X}_{i,g}) \\ \mathbf{X}_{i,g} \text{ if } f(\mathbf{V}_{i,g}) > f(\mathbf{X}_{i,g}) \end{cases} \tag{3.33}$$

The best new habitat replaces the worst corresponding one in the current population only if $\mathbf{V}_{i,g}$ is better than $\mathbf{X}_{i,g}$. This concept is similar to what happens in nature for longer-living species, where the offspring and parents are alive concurrently and have to compete.

6. Boundary constraints: If the variable value $X_{i,j,g}$ violates the boundary constraints, the corresponding violating variable value is randomly generated within the boundary constraints as follows:

$$X_{i,j,g} = L_j + rand(0, 1) \times (U_j - L_j)$$

7. Stopping criteria: If the stopping criteria are met, the vector represented by the best habitat contains the optimal combination of fuzzy parameter values that maximize the total fuzzy entropy function of partition Π_3. Otherwise, the procedure is repeated from step 3.

Table 3.1 DBBO-fuzzy parameters

Parameters	Notation	Value
Population size (number of habitats)	NP	20
Elitism parameter	n_{elit}	2
Maximum immigration rate	I	1
Maximum emigration rate	E	1
Number of generations	Max_{gen}	20
Search domain for each parameter vector (SIV)	G	[0, 255]
Number of decision variables (fuzzy parameters)	D	6
Constant of differentiation	F	0.5
DE mutation scheme	DE/rand/1	

3.6 Experimental Settings and Results

3.6.1 Test Images

The performance of the proposed DBBO-Fuzzy algorithm is compared with those of the basic BBO algorithm [28], named here as BBO-Fuzzy, and the performance of DE-Fuzzy algorithm, which proceeds exactly as the original algorithm presented in [21].

The performance of these competing three-level thresholding algorithms are tested with a set of 12 benchmark images, each with 256 gray levels. These are commonly known as Lena, Peppers, Cameraman, Airplane, Lake, Walking bridge, Mandrill, Barbara, Boat, Elaine, GoldHill and Fingerprint. All the images are of size 512×512 pixels. The original images considered are shown in Fig. 3.3.

3.6.2 Test Design

In all experiments, the same parameter values are used for each of the three aforementioned algorithms to make a fair comparison. A population of 20 individuals is used; these are evolved during 20 generations. For each test image, ten independent runs are carried out.

For the DE-Fuzzy algorithm, $F = 0.5$ and $CR = 0.9$ are chosen, as recommended in [29]. For fair performance comparison, the same mutation scheme, DE/rand/1, is adopted for both the DE-Fuzzy and the DBBO-Fuzzy algorithms. For the BBO-Fuzzy algorithm, the same parameter settings as in [28] are used. In Table 3.1, the parameter setup used in the experiments conducted is summarized.

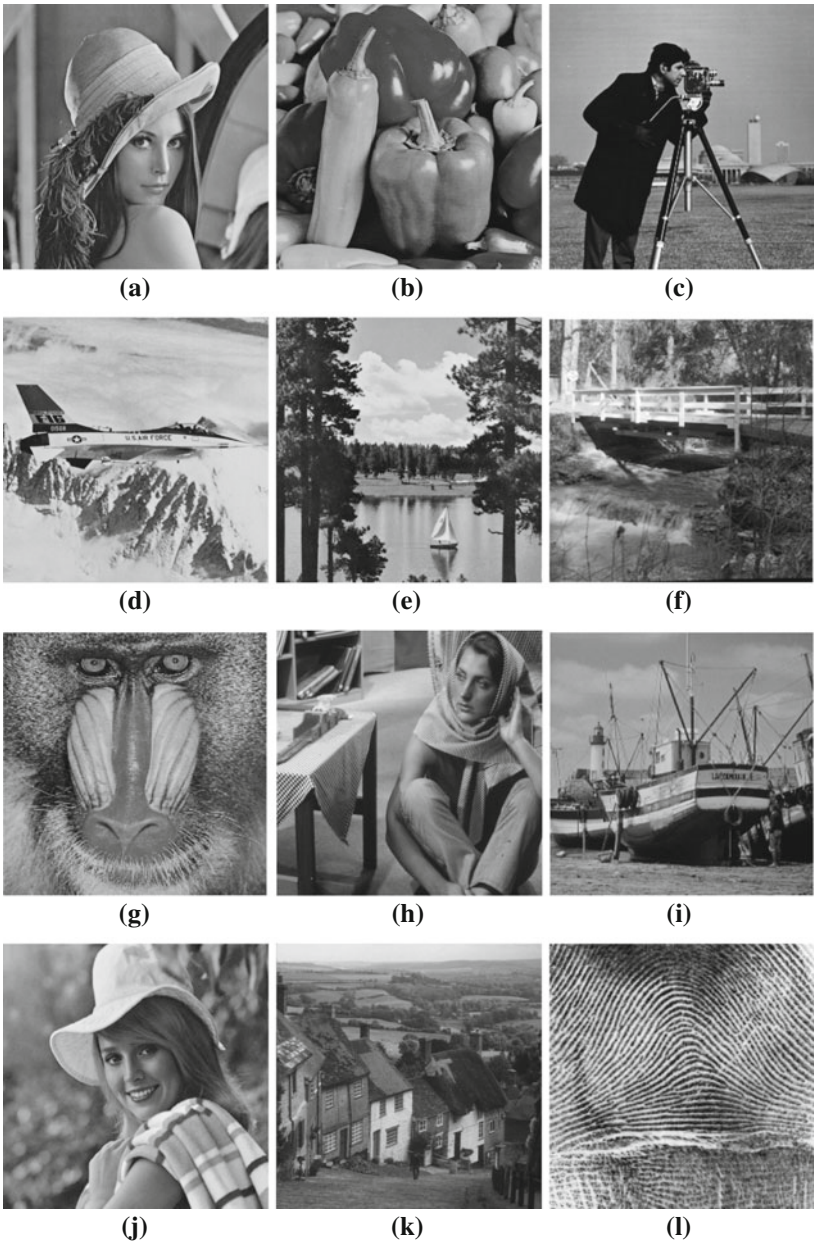


Fig. 3.3 Original test images: **a** Lena, **b** Peppers, **c** Cameraman, **d** Airplane, **e** Lake, **f** Walking bridge, **g** Mandrill, **h** Barbara, **i** Boat, **j** Elaine, **k** GoldHill, **l** Fingerprint

3.6.3 Results and Discussions

The three-level thresholding algorithms are employed to determine the optimal thresholds T_1 and T_2 that segment a given image into three gray levels while preserving the original information as much as possible after creation of this partition.

The objective is to maximize the total fuzzy entropy $H\ (b_1, c_1, a_2, b_2, c_2)$, given in Eq. (3.13), and then to find the "optimal" combination of all the fuzzy parameters $(a_1, b_1, c_1, a_2, b_2, c_2)$ that produce the maximization of fuzzy entropy. The higher value of objective function results in better segmentation.

The results are also compared based on the uniformity factor, the most commonly used measure to quantitatively judge the segmentation quality. This uniformity measure is defined as [16]:

$$u = 1 - 2 * c * \frac{\sum_{j=0}^{c} \sum_{j \in R_j} \left(f_i - \mu_j\right)^2}{N * (f_{max} - f_{min})^2} \tag{3.34}$$

where, c number of thresholds, R_j jth segmented region, f_i gray level of the pixel i, μ_j mean gray level of pixels in jth region, N total number of thresholds in the given image, f_{max} maximum gray level of pixels in the given image, f_{min}minimum gray level of pixels in the given image

The value of this uniformity measure should be a value within the interval [0, 1]. The higher the value of u, better the quality of the thresholded image.

Table 3.2 shows the average objective values (i.e., total fuzzy entropy) achieved by each algorithm for each image under test. The values in *boldface* describe the best-performing algorithm among competing algorithms. It is observed from the obtained results that the proposed DBBO-Fuzzy method obtains higher fuzzy entropy values than both BBO-Fuzzy and DE-Fuzzy in all test images.

In order to determine whether the differences between the DBBO-Fuzzy algorithm and the BBO-Fuzzy and DE-Fuzzy algorithms are statistically significant, a two-tailed t-test was conducted with $df = 10 + 10 - 2 = 18$ degrees of freedom at $\alpha = 0.05$ level of significance (i.e., at 95 % confidence level). The average objective values and the standard deviations obtained by each algorithm over ten independent runs were used to calculate the t-values. The absolute value of the computed t is found to be larger than the critical value in all test images. This suggests that, with 95 % confidence, the difference between DBBO-Fuzzy algorithm and the other two competing algorithms is statistically significant. Therefore, it is evident that the hybridization of the BBO algorithm with DE has noticeable effect on the performance of both algorithms. In these tests, 1 versus 2 means DBBO-Fuzzy algorithm versus BBO-Fuzzy algorithm, and 1 versus 3 means DBBO-Fuzzy algorithm versus DE-Fuzzy algorithm.

Table 3.3 shows the optimal thresholds obtained and the uniformity factor values attained using DBBO-Fuzzy, BBO-Fuzzy and DE-Fuzzy methods. One can observe from the results reported in Table 3.3 that the solution quality of DBBO-Fuzzy is superior to BBO-Fuzzy and DE-Fuzzy in 11 out of 12 images. The only exception

Table 3.2 Comparison of DBBO-fuzzy, BBO-fuzzy and DE-fuzzy, where *boldface* indicates the best performing algorithm

Test images	Objective values			Standard deviation			1 versus 2	1 versus 3
	DBBO-fuzzy	BBO-fuzzy	DE-fuzzy	DBBO-fuzzy	BBO-fuzzy	DE-fuzzy	t-test	t-test
Lena	**1.3851E+01**	1.3569E+01	1.3707E+01	5.3148E−02	2.8230E−02	4.1078E−02	1.4823E+01 ∗	6.7622E+00 ∗
Peppers	**1.3443E+01**	1.3089E+01	1.3314E+01	3.4273E−02	5.6806E−02	5.8473E−02	1.6883E+01 ∗	6.0430E+00 ∗
Cameraman	**1.3463E+01**	1.3090E+01	1.3235E+01	3.8302E−02	8.7798E−02	5.8679E−02	1.2307E+01 ∗	1.0301E+01 ∗
Airplane	**1.3543E+01**	1.3380E+01	1.3451E+01	2.5830E−02	6.9561E−02	5.9266E−02	6.9381E+00 ∗	4.4722E+00 ∗
Lake	**1.3817E+01**	1.3641E+01	1.3739E+01	2.2140E−02	8.0443E−02	3.2497E−02	6.6555E+00 ∗	6.2791E+00 ∗
Walk bridge	**1.4550E+01**	1.4193E+01	1.4441E+01	4.3054E−02	5.7350E−02	2.9707E−02	1.5728E+01 ∗	6.5527E+00 ∗
Mandrill	**1.3815E+01**	1.3593E+01	1.3725E+01	4.1507E−02	4.7428E−02	3.621E−02	1.1112E+01 ∗	5.1733E+00 ∗
Barbara	**1.4214E+01**	1.3845E+01	1.4097E+01	5.7206E−02	6.9849E−02	8.0261E−02	1.2925E+01 ∗	3.7587E+00 ∗
Boat	**1.3986E+01**	1.3596E+01	1.3880E+01	8.9041E−02	7.0177E−02	7.3576E−02	1.0883E+01 ∗	2.9037E+00 ∗
Elaine	**1.4089E+01**	1.3771E+01	1.3999E+01	3.5250E−02	6.0926E−02	4.9027E−02	1.4276E+01 ∗	4.7264E+00 ∗
Goldhill	**1.4069E+01**	1.3693E+01	1.3920E+01	6.7739E−02	3.7019E−02	4.5656E−02	1.5410E+01 ∗	5.7761E+00 ∗
Fingerprint	**1.2406E+01**	1.2100E+01	1.2328E+01	4.1010E−02	5.8752E−02	4.7234E−02	1.3486E+01 ∗	3.9108E+00 ∗

The results are mean best objective values and standard deviations over ten runs (∗ the difference between the two algorithms is statistically significant)

Table 3.3 Optimal threshold and uniformity factor values obtained by DBBO-fuzzy, BBO-fuzzy and DE-fuzzy, where *boldface* indicates the best performing algorithm

Test images	Optimal thresholds			Uniformity measures		
	DBBO-fuzzy	BBO-fuzzy	DE-fuzzy	DBBO-fuzzy	BBO-fuzzy	DE-fuzzy
Lena	98, 175	98, 169	83, 161	**9.8391E−01**	9.8246E−01	9.7213E−01
Peppers	85, 162	71, 205	63, 188	**9.7720E−01**	9.6821E−01	9.6464E−01
Cameraman	127, 207	132, 211	97, 198	9.6557E−**01**	9.5745E−01	**9.6767E−01**
Airplane	41, 166	46, 178	30, 153	**9.8719E−01**	9.8666E−01	9.8618E−01
Lake	96, 171	95, 165	64, 161	**9.8451E−01**	9.8420E−01	9.7778E−01
Walk Bridge	79, 154	35, 211	48, 180	**9.7816E−01**	9.6507E−01	9.6149E−01
Mandrill	74, 150	33, 143	51,167	**9.6755E−01**	9.6617E−01	9.6448E−01
Barbara	93, 170	93, 172	90, 170	**9.8233E−01**	9.7595E−01	9.7342E−01
Boat	107, 185	111, 220	75, 217	**9.8327E−01**	9.7904E−01	9.7566E−01
Elaine	98, 175	37, 208	44, 196	**9.7517E−01**	9.6634E−01	9.6595E−01
GoldHill	77, 152	52, 156	67, 181	**9.8172E−01**	9.7873E−01	9.7072E−01
Fingerprint	94, 172	41, 152	42, 179	**9.7856E−01**	9.6897E−01	9.5562E−01

Table 3.4 Representative optimal parameter sets $(a_1, b_1, c_1, a_2, b_2, c_2)$

Test images	Optimal parameters		
	DBBO-fuzzy	BBO-fuzzy	DE-fuzzy
Lena	1, 137, 140, 140, 142, 256	5, 136, 138, 141, 144, 232	1, 115, 118, 119, 124, 256
Peppers	1, 119, 122, 122, 124, 256	5, 98, 99, 104, 246, 248	1, 83, 95, 107, 194, 256
Cameraman	1, 177, 180, 180, 194, 256	28, 174, 175, 179, 215, 235	82, 94, 116, 119, 211, 256
Airplane	1, 1, 136, 138, 138, 233	5, 7, 141, 147, 149, 250	1, 1, 100, 101, 132, 240
Lake	1, 135, 136, 136, 136, 256	7, 131, 131, 132, 138, 239	20, 77, 87, 95, 143, 256
Walk bridge	1, 111, 111, 111, 112, 256	1, 3, 116, 118, 245, 254	1, 26, 131, 141, 156, 256
Mandrill	1, 103, 105, 105, 106, 256	1 18 90 94 102 250	1, 40, 120, 128, 131, 256
Barbara	1, 131, 132, 133, 137, 256	2, 124, 137, 140, 146, 242	23, 115, 120, 134, 134, 256
Boat	1, 151, 151, 151, 161, 256	24, 143, 150, 151, 245, 253	1, 100, 113, 124, 256, 256
Elaine	1, 136, 140, 141, 143, 256	2, 5, 117, 118, 244, 246	1, 1, 149, 152, 187, 256
Goldhill	1, 108, 108, 109, 110, 256	20, 27, 119, 120, 121, 241	1, 89, 100, 121, 173, 256
Fingerprint	1, 132, 132, 135, 138, 256	4, 25, 105, 106, 114, 252	1, 1, 142, 145, 149, 256

is the Cameraman image, where DBBO-Fuzzy produces a slightly worse uniformity factor than DE-Fuzzy. From this point of view also, the DBBO-Fuzzy algorithm stands out as the clear winner.

Table 3.4 presents representative optimal parameter sets $\{a_1, b_1, c_1, a_2, b_2, c_2\}$ obtained by employing DBBO-Fuzzy, BBO-Fuzzy and DE-Fuzzy algorithms.

For a visual interpretation of the three level thresholding results, the thresholded images obtained by applying DBBO-Fuzzy algorithm are presented in Fig. 3.4. After determination of the thresholds for each image, the gray levels of all pixels in a given region are changed to the average gray level of all the pixels belonging to that region.

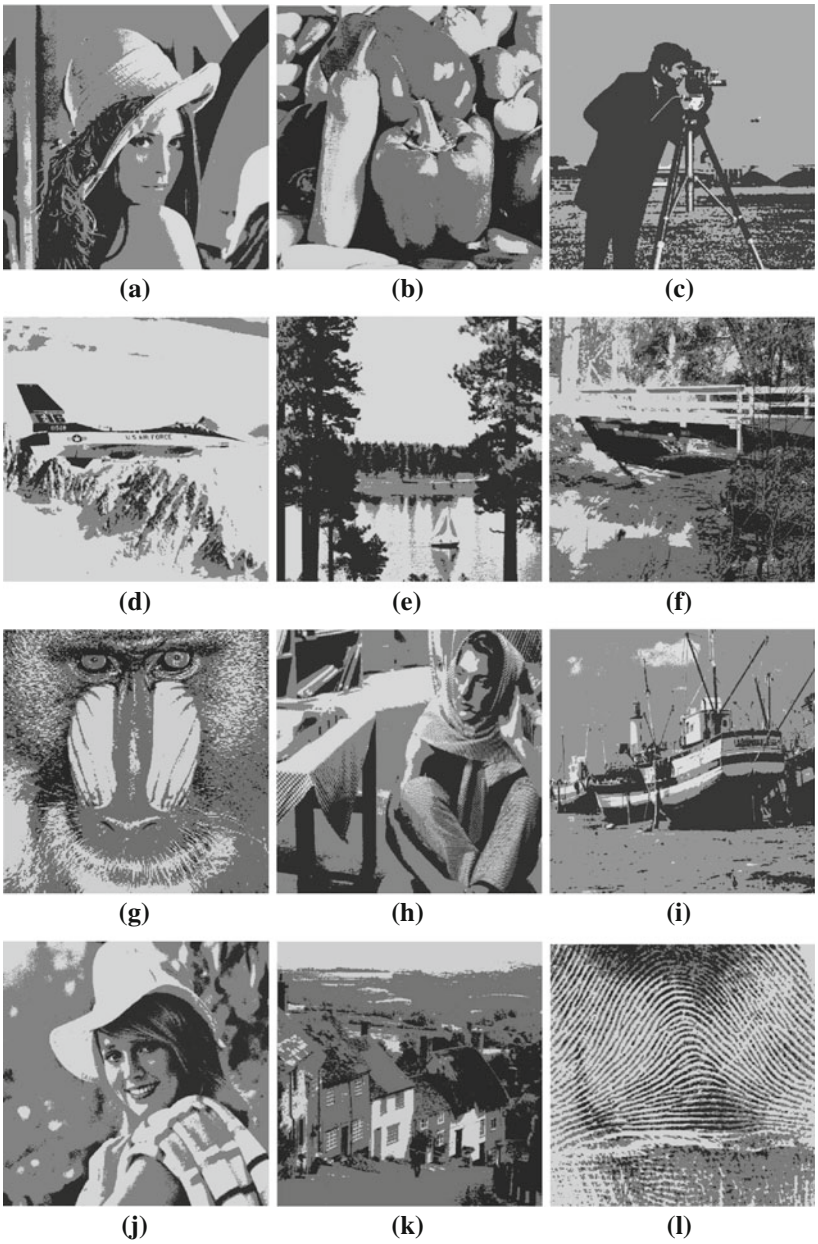


Fig. 3.4 The three-level thresholded images using DBBO-Fuzzy: **a** Lena, **b** Peppers, **c** Cameraman, **d** Airplane, **e** Lake, **f** Walking bridge, **g** Mandrill, **h** Barbara, **i** Boat, **j** Elaine, **k** GoldHill, **l** Fingerprint

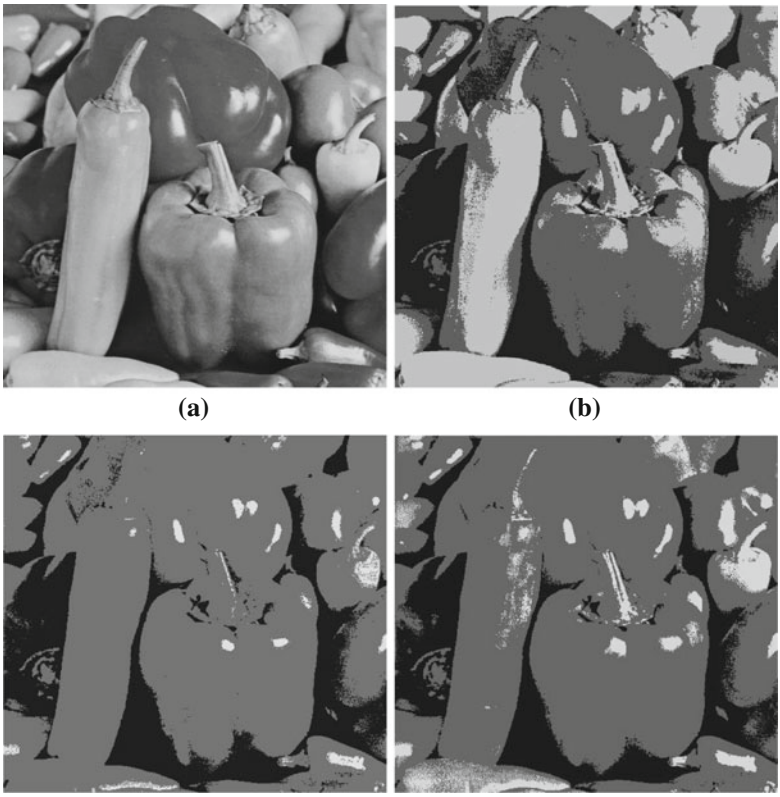


Fig. 3.5 The three-level thresholded images of peppers: **a** original image; **b** thresholded image using DBBO-Fuzzy; **c** thresholded image using BBO-Fuzzy; **d** thresholded image using DE-Fuzzy

Figures 3.5, 3.6 and 3.7 show some sample images under consideration and the resultant segmented images obtained after employing the DBBO-Fuzzy, BBO-Fuzzy and DE-Fuzzy algorithms. From the pictorial representations of the segmented images, it is clear that DBBO-Fuzzy algorithm emerges as the best performer.

3.6.3.1 Effect of the Mutation Strategy

To make a detailed, in-depth study of the DBBO-Fuzzy algorithm, it was tested employing different mutation schemes, described in Sect. 3.5, in order to investigate the effects of the mutation strategy on its performance. The results are reported in Table 3.5. To visualize the best performing scheme, the best values are given in *boldface*.

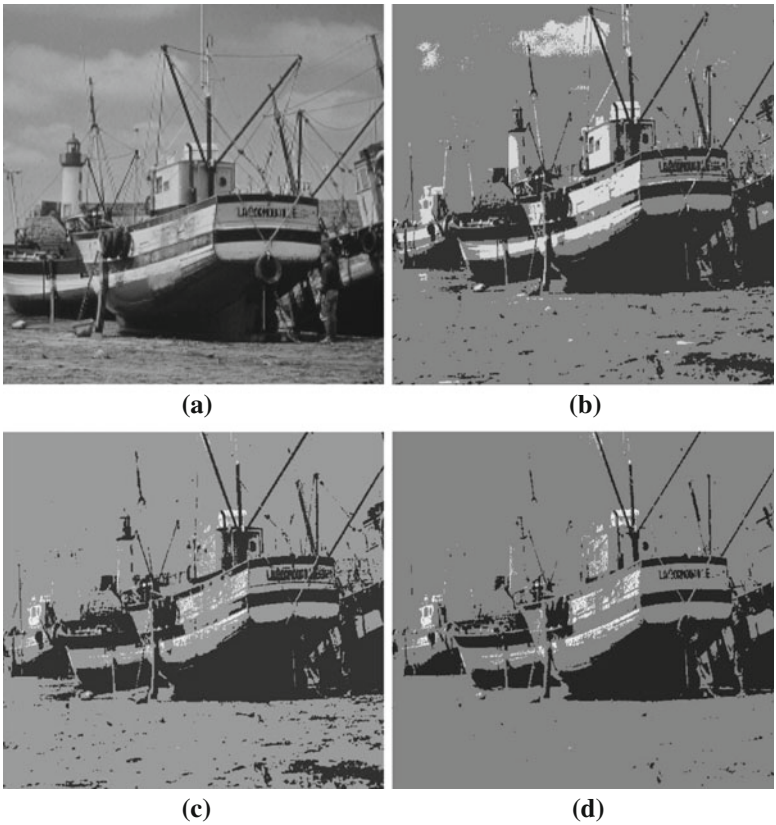


Fig. 3.6 The three-level thresholded images of boat: **a** original image; **b** thresholded image using DBBO-fuzzy; **c** thresholded image using BBO-fuzzy; **d** thresholded image using DE-fuzzy

A closer look at Table 3.5 reveals that out of the 12 test images, the DBBO-Fuzzy algorithm with DE/best/1 mutation strategy emerged as the best candidate algorithm, since it could achieve the highest values of the fuzzy entropy in eight cases (i.e., Lena, Peppers, Cameraman, Airplane, Lake, Mandrill, Elaine and Fingerprint). The DBBO-Fuzzy algorithm with DE/rand/2 proved to be the winner in only two cases (i.e., Walking bridge and Boat). The DBBO-Fuzzy algorithm with DE/rand/1 and DE/current-to-best/1 mutation schemes perform best in only one case each (i.e., GoldHill and Barbara, respectively).

In terms of the best uniformity measure value, DBBO-Fuzzy with DE/rand/1 produced the highest values in 6 test images out of 12 images (i.e., Lena, Cameraman, Walking bridge, Mandrill, Boat and Fingerprint). For the remaining six images (i.e., Peppers, Airplane, Lake, Barbara, Elaine and GoldHill), the DBBO-Fuzzy algorithm with DE/current-to-best/1 mutation scheme achieved the highest uniformity measure values.

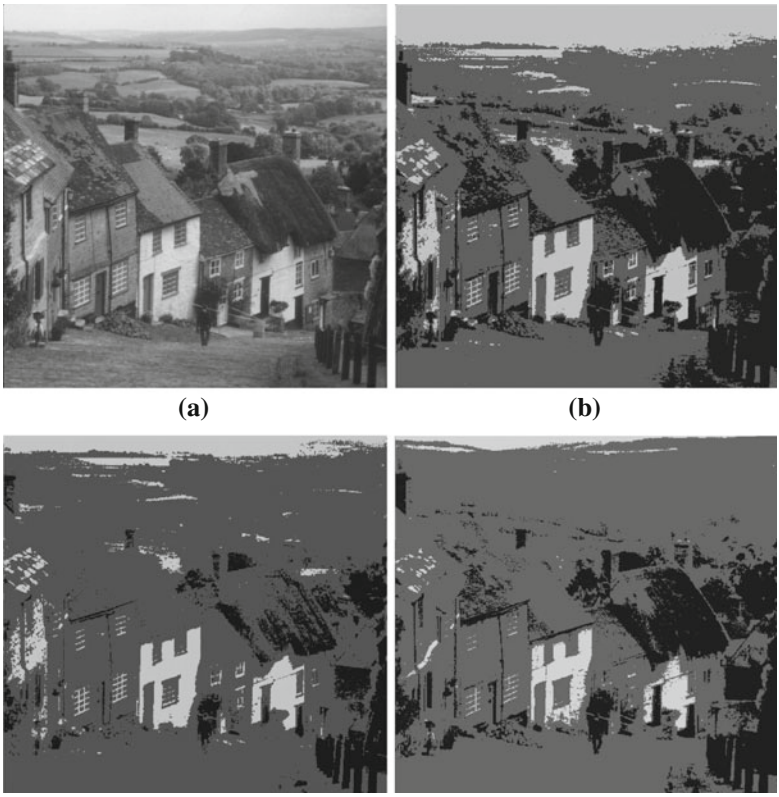


Fig. 3.7 The three-level thresholded images of goldhill: **a** original image; **b** thresholded image using DBBO-fuzzy; **c** thresholded image using BBO-fuzzy; **d** thresholded image using DE-fuzzy

All the variations of the DBBO-Fuzzy algorithm were compared to a default DBBO-Fuzzy algorithm with DE/rand/1, and differences were reported as significant if a two-tailed t-test produced a t-value larger than the critical value. The significance level α is set at 0.05.

Significant differences exist when comparing DBBO-Fuzzy with DE/rand/1 mutation strategy with the variant using DE/current-to-best/1 scheme for two test images (Airplane and Lake). The improvement in the mean objective function value obtained when using DE/best/1 mutation strategy is more significant in the case of Airplane image. For DBBO-Fuzzy with DE/rand/2 mutation, the test has provided a statistically significant difference for the Lake image. In eight out of 12 cases, DBBO-Fuzzy with DE/best/2 mutation proves to be significantly different compared to the variant using DE/rand/1 mutation.

However, in general, it can be noted that the results using different mutation schemes do not significantly affect the performance of the proposed algorithm.

Table 3.5 Comparison of DBBO-fuzzy, BBO-fuzzy and DE-fuzzy, where the best values are given in boldface

Test images	Mutation schemes	Optimal thresholds	Mean Obj ± Std Dev	Uniformity measure	Optimal parameters
Lena	DE/rand/1	98, 175	1.3851E+01 ± 5.3148E−02	**9.8391E−01**	1, 137, 140, 140, 142, 256
	DE/current-to-best/1	93, 169	1.3847E+01 ± 5.3418E−02	9.8099E−01	1, 131, 132, 132, 133, 256
	DE/best/1	96, 171	**1.3889E+01 ± 6.8730E−02**	9.8222E−01	1, 136, 136, 136, 136, 256
	DE/rand/2	99, 174	1.3815E+01 ± 7.8345E−02	9.8302E−01	1, 139, 140, 140, 141, 256
	DE/best/2	99, 178	1.3764E+01 ± 8.4071E−02 ∗	9.7977E−01	1, 138, 140, 145, 146, 256
Peppers	DE/rand/1	85, 162	1.3443E+01 ± 3.4273E−02	9.7720E−01	1, 119, 122, 122, 124, 256
	DE/current-to-best/1	86, 162	1.3444E+01 ± 3.9021E−02	**9.7728E−01**	1, 118, 118, 118, 119, 256
	DE/best/1	83, 158	**1.3462E+01 ± 4.0636E−02**	9.7576E−01	1, 117, 117, 117, 117, 256
	DE/rand/2	79, 156	1.3426E+01 ± 5.9953E−02	9.7660E−01	1, 111, 113, 114, 115, 256
	DE/best/2	88, 164	1.3392E+01 ± 5.8353E−02 ∗	9.7469E−01	1, 124, 125, 126, 126, 256
Cameraman	DE/rand/1	127, 207	1.3463E+01 ± 3.8302E−02	**9.6557E−01**	1, 177, 180, 180, 194, 256
	DE/current-to-best/1	131, 207	1.3431E+01 ± 8.2592E−02	9.6261E−01	1, 184, 185, 185, 187, 256
	DE/best/1	129, 207	**1.3484E+01 ± 7.9968E−02**	9.6267E−01	1, 182, 182, 182, 192, 256
	DE/rand/2	129, 210	1.3391E+01 ± 1.2207E−01	9.6487E−01	1, 182, 183, 184, 196, 256
	DE/best/2	124, 233	1.3365E+01 ± 1.0216E−01 ∗	9.6514E−01	1, 175, 176, 176, 256, 256
Airplane	DE/rand/1	41, 166	1.3543E+01 ± 2.5830E−02	9.8719E−01	1, 1, 136, 138, 138, 233
	DE/current-to-best/1	104, 170	1.3571E+01 ± 1.8272E−02 ∗	**9.8772E−01**	1, 146, 147, 147, 147, 227
	DE/best/1	94, 164	**1.3580E+01 ± 2.3179E−02**∗	9.8733E−01	1, 133, 133, 133, 134, 239
	DE/rand/2	46, 177	1.3524E+01 ± 3.4368E−02	9.8662E−01	1, 1, 154, 157, 157, 224
	DE/best/2	98, 172	1.3511E+01 ± 1.5145E−02 ∗	9.8743E−01	1, 138, 139, 140, 153, 234
Lake	DE/rand/1	96, 171	1.3817E+01 ± 2.2140E−02	9.8451E−01	1, 135, 136, 136, 136, 256
	DE/current-to-best/1	98, 172	1.3511E+01 ± 1.5145E−02 ∗	**9.8743E−01**	1, 138, 139, 140, 153, 234
	DE/best/1	94, 169	**1.3835E+01 ± 2.2273E−02**	9.8431E−01	1, 132, 133, 133, 133, 256
	DE/rand/2	85, 165	1.3775E+01 ± 3.1484E−02 ∗	9.8342E−01	1, 119, 122, 124, 130, 256
	DE/best/2	98, 175	1.3781E+01 ± 4.6093E−02 ∗	9.8339E−01	1, 137, 138, 138, 145, 256
Walk bridge	DE/rand/1	79, 154	1.4550E+01 ± 4.3054E−02	**9.7816E−01**	1, 111, 111, 111, 112, 256
	DE/current-to-best/1	77, 153	1.4551E+01 ± 5.9241E−02	9.7610E−01	1, 109, 109, 110, 111, 256

Table 3.5 continue

	DE/best/1	90, 165	1.4555E+01 ± 6.2526E−02	9.7631E−01	1, 127, 127, 127, 127, 256
	DE/rand/2	86, 166	**1.4571E+01 ± 4.2417E**−02	9.7803E−01	1, 118, 125, 126, 130, 256
	DE/best/2	34, 157	1.4502E+01 ± 3.9284E−02 ∗	9.7543E−01	1, 1, 113, 115, 116, 256
Test images	Mutation schemes	Optimal thresholds	Mean Obj ± Std Dev	Uniformity measure	Optimal parameters
Mandrill	DE/rand/1	74, 150	1.3815E+01 ± 4.1507E−02	**9.6755E−01**	1, 103, 105, 105, 106, 256
	DE/current-to-best/1	31, 147	1.3801E+01 ± 6.0275E−02	9.6600E−01	1, 1, 102, 102, 102, 256
	DE/best/1	29, 144	**1.3845E+01 ± 1.8303E**−02	9.6485E−01	1, 1, 97, 97, 97, 256
	DE/rand/2	30, 145	1.3808E+01 ± 3.3253E−02	9.6669E−01	1, 1, 99, 99, 100, 256
	DE/best/2	36, 161	1.3759E+01 ± 4.6532E−02 ∗	9.6713E−01	1, 1, 119, 119, 123, 256
Barbara	DE/rand/1	93, 170	1.4214E+01 ± 5.7206E−02	9.8233E−01	1, 131, 132, 133, 137, 256
	DE/current-to-best/1	96, 172	**1.4235E+01 ± 3.3796E−02**	**9.8280E−01**	1, 136, 136, 137, 137, 256
	DE/best/1	104, 178	1.4206E+01 ± 6.4360E−02	9.7831E−01	1, 146, 146, 146, 146, 256
	DE/rand/2	94, 171	1.4228E+01 ± 3.7467E−02	9.8236E−01	1, 131, 134, 135, 137, 256
	DE/best/2	96, 172	1.4154E+01 ± 7.4299E−02	9.7929E−01	1, 134, 137, 137, 138, 256
Boat	DE/rand/1	107, 185	1.3986E+01 ± 8.9041E−02	**9.8327E−01**	1, 151, 151, 151, 161, 256
	DE/current-to-best/1	99, 221	1.4017E+01 ± 6.4776E−02	9.8016E−01	1, 140, 140, 140, 252, 256
	DE/best/1	109, 184	1.4019E+01 ± 8.8207E−02	9.7937E−01	1, 154, 154, 154, 155, 256
	DE/rand/2	100, 223	**1.4022E+01 ± 6.6237E−02**	9.8178E−01	1, 137, 144, 144, 256, 256
	DE/best/2	97, 205	1.3921E+01 ± 8.5438E−02	9.8041E−01	1, 136, 137, 138, 215, 256
Elaine	DE/rand/1	98, 175	1.4089E+01 ± 3.5250E−02	9.7517E−01	1, 136, 140, 141, 143, 256
	DE/current-to-best/1	89, 164	1.4124E+01 ± 5.6691E−02	**9.7828E−01**	1, 126, 126, 126, 127, 256
	DE/best/1	91, 165	**1.4127E+01 ± 6.1915E−02**	9.7573E−01	1, 128, 128, 128, 128, 256
	DE/rand/2	100, 178	1.4107E+01 ± 4.8732E−02	9.7639E−01	1, 139, 144, 146, 146, 256
	DE/best/2	95, 171	1.4078E+01 ± 7.1696E−02	9.7630E−01	1, 133, 134, 136, 136, 256
Goldhill	DE/rand/1	77, 152	**1.4069E+01 ± 6.7739E−02**	9.8172E−01	1, 108, 108, 109, 110, 256
	DE/current-to-best/1	86, 161	1.4037E+01 ± 9.6897E−02	**9.8176E−01**	1, 121, 121, 121, 121, 256
	DE/best/1	94, 169	1.4056E+01 ± 6.5751E−02	9.8152E−01	1, 133, 133, 133, 133, 256
	DE/rand/2	87, 163	1.4027E+01 ± 9.1891E−02	9.8148E−01	1, 120, 124, 124, 125, 256
	DE/best/2	81, 159	1.3991E+01 ± 5.8453E−02 ∗	9.7927E−01	1, 110, 118, 119, 120, 256

Table 3.5 continue

Fingerprint	DE/rand/1	94, 172	1.2406E+01 ± 4.1010E−02	**9.7856E−01**	1, 132, 132, 135, 138, 256
	DE/current-to-best/1	76, 151	1.2389E+01 ± 2.3523E−02	9.7463E−01	1, 107, 108, 108, 108, 256
	DE/best/1	85, 160	**1.2416E+01 ± 3.3792E−02**	9.7398E−01	1, 119, 120, 120, 120, 256
	DE/rand/2	37, 161	1.2396E+01 ± 2.5875E−02	9.7335E−01	1, 2, 121, 121, 122, 256
	DE/best/2	34, 156	1.2336E+01 ± 5.5379E−02 ∗	9.7322E−01	1, 1, 112, 113, 117, 256

(∗ the difference with the variant of DBBO-Fuzzy using DE/rand/1 mutation scheme is statistically significant)

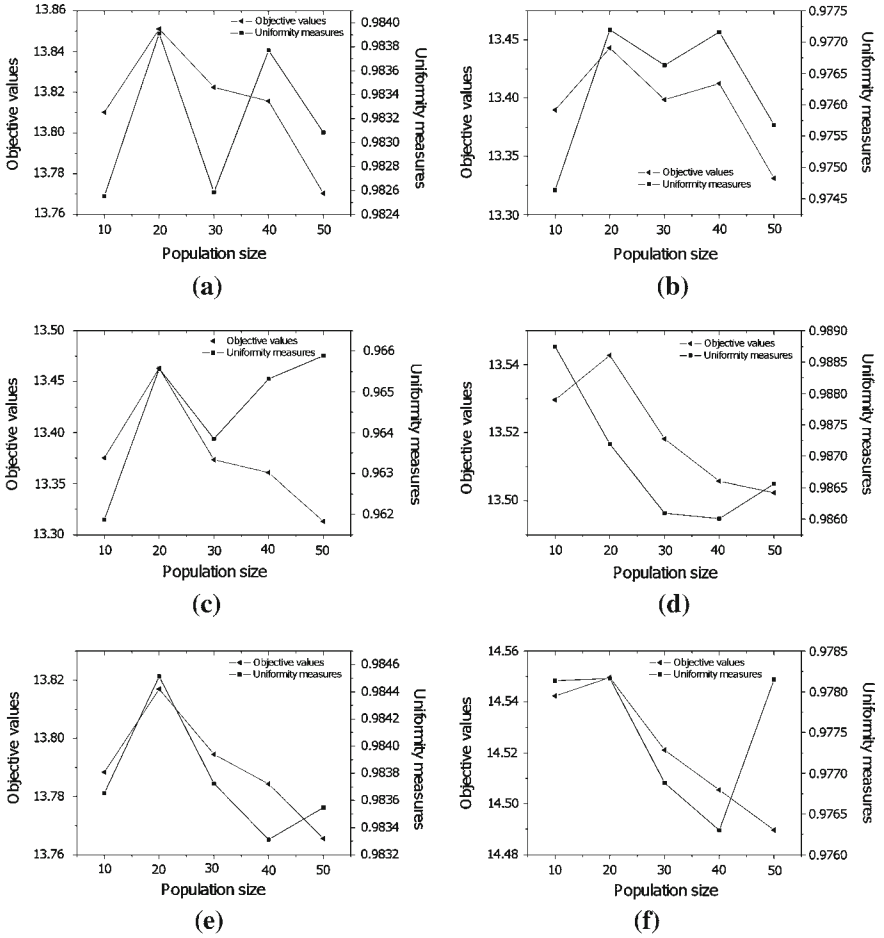


Fig. 3.8 Effect of varying the population size on the performance of DBBO-fuzzy: **a** Lena, **b** Peppers, **c** Cameraman, **d** Airplane, **e** Lake, **f** Walking bridge

3.6.3.2 Effect of the Population Size

Simulations have also been carried out for different values of population size NP, and the performance of the DBBO-Fuzzy algorithm are shown for different variations in NP in Fig. 3.8.

In general, it can be inferred that the DBBO-Fuzzy algorithm with a population size $NP = 20$ outperforms the other variants, since it could achieve the highest value of fuzzy entropy in 11 cases out of 12. Only in one case was DBBO-Fuzzy with $NP = 30$ reveals to be the best in terms of fuzzy entropy value. With $NP = 20$, the DBBO-Fuzzy variant could achieve the highest value of uniformity factor in eight cases. On the other hand, with $NP = 10$, the DBBO-Fuzzy algorithm could achieve

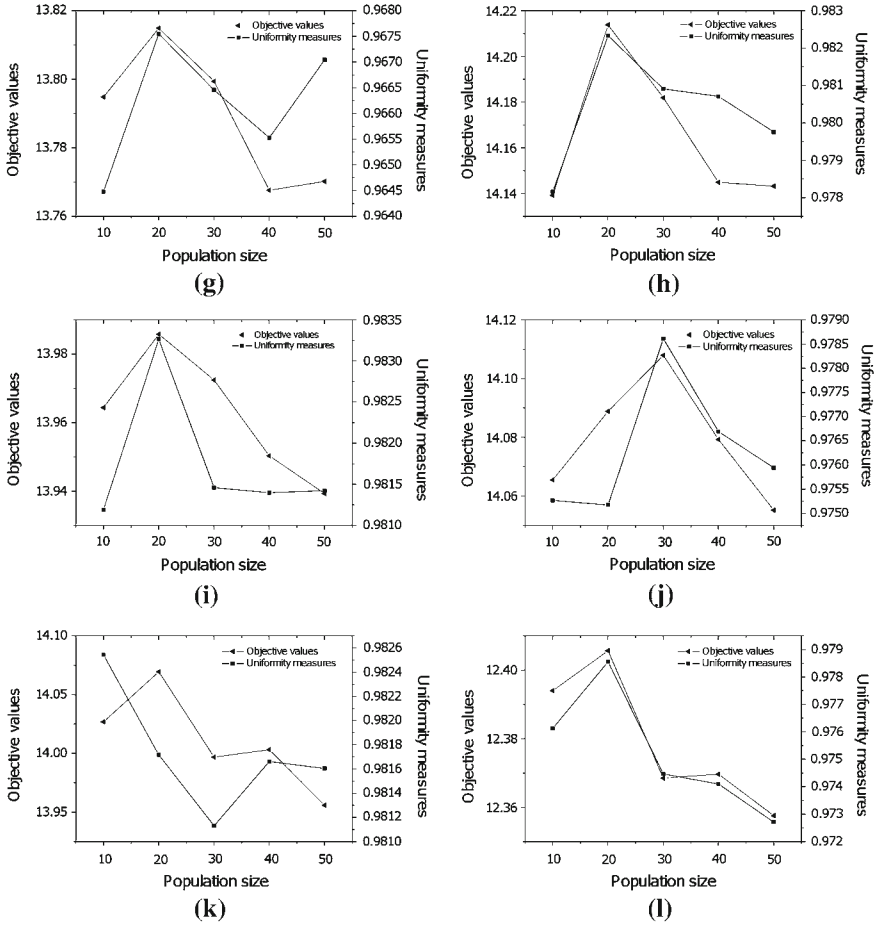


Fig. 3.8 (Cont) Effect of varying the population size on the performance of DBBO-fuzzy: **g** Mandrill, **h** Barbara, **i** Boat, **j** Elaine, **k** GoldHill, **l** Fingerprint

the highest value of uniformity factor in two cases (Airplane and GoldHill) and in one case each with $NP = 50$ (Cameraman) and $NP = 30$ (Elaine).

These results suggest that blindly increasing the population size may not have a relevant positive effect on the performance of the DBBO-Fuzzy algorithm.

3.6.3.3 Effect of the Elitism Parameter

The performance of the proposed algorithm is also evaluated in detail by varying the elitism parameter n_{elit}. In general, one can observe from Fig. 3.9 that the average objective function values tend to increase with the number of elites, although the

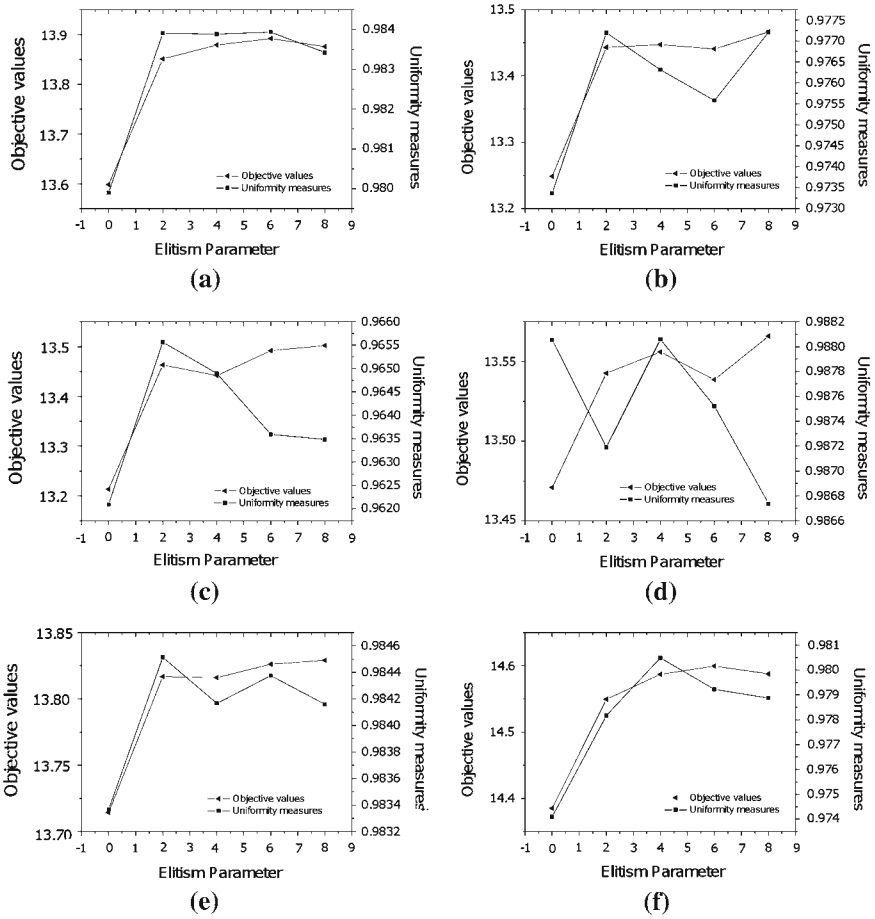


Fig. 3.9 Effect of varying the elitism parameter on the performance of DBBO-fuzzy: **a** Lena, **b** Peppers, **c** Cameraman, **d** Airplane, **e** Lake, **f** Walking bridge

uniformity measure does not follow the same trend in all test images. In nine cases out of 12, DBBO-Fuzzy with $n_{elit} = 8$ produced highest values of the fuzzy entropy. Only in two cases, the DBBO-Fuzzy variant with $n_{elit} = 6$ and in one case the DBBO-Fuzzy variant with $n_{elit} = 4$ achieved the highest values of fuzzy entropy. On the other hand, DBBO-Fuzzy without elitism ($n_{elit} = 0$) could never achieve the highest value for fuzzy entropy but produced good uniformity factor values in three cases out of 12 (i.e., Airplane, Mandrill and Boat).

Finally, an important general remark should be made here that if there is a conflicting choice between a higher fuzzy entropy value and a higher uniformity factor value, higher priority should be given to the solution having higher uniformity factor,

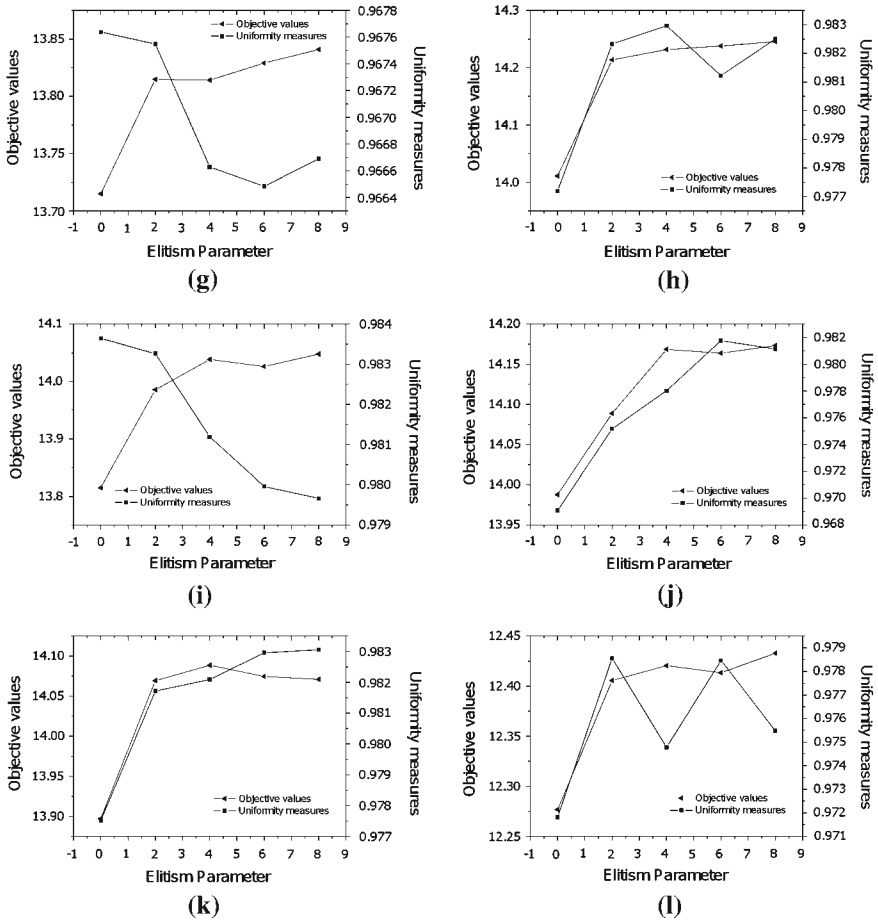


Fig. 3.9 (Cont) Effect of varying the elitism parameter on the performance of DBBO-fuzzy: **g** Mandrill, **h** Barbara, **i** Boat, **j** Elaine, **k** GoldHill, **l** Fingerprint

as it quantitatively reflects the direct impact of the quality of the output segmented image.

3.7 Conclusion

Image segmentation is a process of partitioning an image space into several homogeneous regions. Thresholding is one of the most widely used techniques in image segmentation because of its fast and easy application. However, it has been proven that thresholding often fails to produce satisfactory segmentation results due

to grayness and spatial ambiguity in images. The use of fuzzy set theory could be recognized as an adequate mathematical tool that can be used to model the inherent image vagueness. A new three-level thresholding algorithm based on the hybridization of BBO and the DE algorithms, called the DBBO-Fuzzy algorithm, has been described in this chapter. The DBBO-Fuzzy uses the DE mutation strategy to improve the global search capability and escape from local optima. The experimental results manifest that the proposed algorithm outperforms both BBO and DE algorithms and achieves a high quality of the thresholded images.

The future work will mainly focus on employing a multiobjective approach for such image segmentation problems. Instead of considering a single objective function, a biobjective model could be adopted for the three-level thresholding problem, in which one seeks to optimize simultaneously the total fuzzy entropy and the uniformity factor.

References

1. Acharya, T., Ray, A.K.: Image Processing—Principles and Applications. Wiley-Interscience, New Jersey (2005)
2. Bhandari, D., Pal, N.R.: Some new information measures for fuzzy sets. Inf. Sci. **67**, 209–228 (1993)
3. Boussaïd, I., Chatterjee, A., Siarry, P., Ahmed-Nacer, M.: Two-stage update biogeography-based optimization using differential evolution algorithm (dbbo). Comput. Oper. Res. **38**, 1188–1198 (2011)
4. Chang, C.I., Du, Y., Wang, J., Guo, S.M., Thouin, P.D.: Survey and comparative analysis of entropy and relative entropy thresholding techniques. IEE Proc. Vis. Image Signal Process. **153**(6), 837–850 (2006)
5. Cheng, H., Chen, J., Li, J.: Threshold selection based on fuzzy c-partition entropy approach. Pattern Recogn. **31**(7), 857–870 (1998)
6. Cover, T.M., Thomas, J.A.: Elements of Information Theory. Wiley-Interscience, New York (1991)
7. Darwin, C.: Origin of Species. Gramercy, New York (1995)
8. De Luca, A., Termini, S.: A definition of a non-probabilistic entropy in the setting of fuzzy sets theory. Inf. Control **20**, 301–312 (1972)
9. Fan, J., Xie, W.: Distance measure and induced fuzzy entropy. Fuzzy Sets Syst. **104**, 305–314 (1999)
10. Freixenet, J., Muñoz, X., Raba, D., Martí, J., Cufí, X.: Yet another survey on image segmentation: region and boundary information integration. In: Proceedings of the 7th European Conference on Computer Vision-Part III, ECCV '02, pp. 408–422. Springer-Verlag, London, UK (2002)
11. Gong, W., Cai, Z., Ling, C.: DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization. Soft Comput. Fusion Found. Methodol. Appl. (2010)
12. Kapur, J., Sahoo, P., Wong, A.: A new method for gray-level picture thresholding using the entropy of the histogram. Comput. Vision Graph. Image Process. **29**(3), 273–285 (1985)
13. Kaufmann, A.: Introduction to the Theory of Fuzzy Subsets. Academic Press, New York (1975)
14. Kaufmann, A.: Measures of Fuzzy Information. Mathematical Sciences Trust Society, New Delhi (1997)
15. Klir, G.J., St. Clair, U., Yuan, B.: Fuzzy Set Theory: Foundations and Applications. Prentice-Hall, Inc., Upper Saddle River (1997)

16. Levine, M., Nazif, A.: Dynamic measurement of computer generated image segmentations. IEEE Trans. Pattern Anal. Mach. Intell. **7**, 155–164 (1985)
17. MacArthur, R., Wilson, E.: The Theory of Biogeography. Princeton University Press, Princeton (1967)
18. Noman, N., Iba, I.: Accelerating differential evolution using an adaptive local search. IEEE Trans. Evol. Comput. **12**(1), 107–125 (2008)
19. Pal, N.R., Pal, S.K.: Higher order fuzzy entropy and hybrid entropy of a set. Inf. Sci. **61**, 211–231 (1992)
20. Pal, N.R., Pal, S.K.: A review on image segmentation techniques. Pattern Recognit. **26**(9), 1277–1294 (1993)
21. Price, K.V., Storn, R.M., Lampinen, J.A.: Differential Evolution: A Practical Approach to Global Optimization. Natural Computing Series. Springer, Berlin (2005)
22. Pun, T.: A new method for gray-level picture thresholding using the entropy of histogram. Signal Process. **2**(3), 223–237 (1980)
23. Pun, T.: Entropic thresholding, a new approach. Comput. Graph. Image Process. **16**(3), 210–239 (1981)
24. Sahoo, P.K., Soltani, S., Wong, A.K., Chen, Y.C.: A survey of thresholding techniques. Comput. Vision Graph. Image Process. **41**, 233–260 (1988)
25. Sahoo, P.K., Wilkins, C., Yeager, J.: Threshold selection using Renyi's entropy. Pattern Recognit. **30**(i1), 71–84 (1997)
26. Sezgin, M., Sankur, B.: Survey over image thresholding techniques and quantitative performance evaluation. J. Electron. Imaging **13**(1), 146–168 (2004)
27. Shannon, C.E.: A Mathematical theory of communication. CSLI Publications (1948)
28. Simon, D.: Biogeography-based optimization. IEEE Trans. Evol. Comput. **12**, 702–713 (2008)
29. Storn, R.M., Price, K.V.: Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. **11**, 341–359 (1997)
30. Tao, W., Jin, H., Liu, L.: Object segmentation using ant colony optimization algorithm and fuzzy. entropy **28**(7), 788–796 (2007)
31. Tao, W., Tian, J., Liu, J.: Image segmentation by three-level thresholding based on maximum fuzzy entropy and genetic algorithm. Pattern Recogn. Lett. **24**(16), 3069–3078 (2003)
32. Tobias, O., Seara, R.: Image segmentation by histogram thresholding using fuzzy sets. IEEE Trans. Image Process. **11**(12), 1457–1465 (2002)
33. Tsallis, C.: Possible generalization of Boltzmann-Gibbs statistics. J. Stat. Phys. **52**, 479–487 (1988)
34. Wallace, A.R.: The Geographical Distribution of Animals (two volumes). Adamant Media Corporation, Boston (2005)
35. Zadeh, L.A.: Fuzzy sets. Inf. Control **8**, 338–353 (1965)
36. Zhao, M., Fu, A., Yan, H.: A technique of three-level thresholding based on probability partition and fuzzy 3-partition. IEEE Trans. Fuzzy Syst. **9**(3), 469–479 (2001)

Chapter 4
A Genetic Programming Approach for Image Segmentation

Hugo Alberto Perlin and Heitor Silvério Lopes

Abstract This work presents a methodology for using genetic programming (GP) for image segmentation. The image segmentation process is seen as a classification problem where some regions of an image are labeled as foreground (object of interest) or background. GP uses a set of terminals and nonterminals, composed by algebraic operations and convolution filters. A function fitness is defined as the difference between the desired segmented image and that obtained by the application of the mask evolved by GP. A penalty term is used to decrease the number of nodes of the tree, minimally affecting the quality of solutions. The proposed approach was applied to five sets of images, each one with different features and objects of interest. Results show that GP was able to evolve solutions of high quality for the problem. Thanks to the penalty term of the fitness function, the solutions found are simple enough to be used and understood by a human user.

4.1 Introduction

The automatic recognition of objects in images is the extraction of visual information directly from the environment they are in. Such a computation is an important task in computer vision as well as in automatic decision-making.

A methodology for object recognition can contain several steps, including preprocessing of the raw image, many different image processing algorithms and

H. A. Perlin (✉)
Federal Institute of Education, Science and Technology of Paraná,
Campus Paranaguá, Paranaguá, Brazil
e-mail: hugo.perlin@ifpr.edu.br

H. S. Lopes
Federal University of Technology Paraná,
Campus Curitiba, Curitiba, Brazil
e-mail: hslopes@utfpr.edu.br

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_4, © Springer-Verlag Berlin Heidelberg 2013

extraction of characteristic vectors. One of these steps can be the image segmentation; that is, the process by which a digital image is partitioned into multiple sets of pixels (also known as segments). Each segment is labeled in such a way as to have a meaning to the user, for example, boundaries, curves, textures or objects [14].

Specifically for the recognition of objects in images, the image segmentation process can have a very important role, for instance, segmenting the image into two disjoint sets of pixels, labeled as background and object. This can be obtained by a segmentation method trained to perform a classification of pixels into two classes, as mentioned. From this point of view, the segmentation process can be interpreted as a classification procedure. In other words, the segmentation process accepts as input a digital image, and gives as output the pixels of this image classified as object and background.

Genetic programming (GP) [8] is an evolutionary computation method that searches for solutions for a problem in the form of tree-structured programs. Such programs are built by using a set of terminals and a set of functions, and GP searches the space of all possible combinations of terminals and functions to find those most suitable for solving the problem at hand. GP has been applied to a wide range of problems, including classification problems [2].

Since image segmentation can be viewed as a classification process, and GP can provide interesting solutions for this type of problems, in this work we investigate the use of GP for the segmentation of digital images, in such a way to detect a given object of interest from the background.

This work is structured as follows. Section 4.2 presents the basic aspects of image segmentation. GP is presented in Sect. 4.3. The methodology developed in this work is shown in Sect. 4.4. The computational experiments and results are presented in Sect. 4.5. Finally, Sect. 4.6 presents the conclusions and discusses future work.

4.2 Image Segmentation

There are many general-purpose algorithms for image segmentation, such as thresholding, clustering, histogram-based, model-based, compression-based and neural networks [14, 17]. The simplest method for image segmentation is the binary threshold. It consists in classifying the pixels of an image into two classes: foreground and background. This is accomplished by comparing the gray level of a pixel with a threshold value (or several values, when the classification is multilevel). If the gray level of a given pixel is above the threshold, it is classified as foreground, and otherwise it is classified as background. After the segmentation, the resulting image becomes binary with white pixels as foreground and black pixels as background [5].

The key issue of this simple segmentation method is how to select an appropriate threshold level. For this purpose, there are several methods, such as the Otsu method [12, 16] and the Kwon method [9]. The objective of these methods is to find a good threshold level for a given purpose or class of images. Examples of the binarization

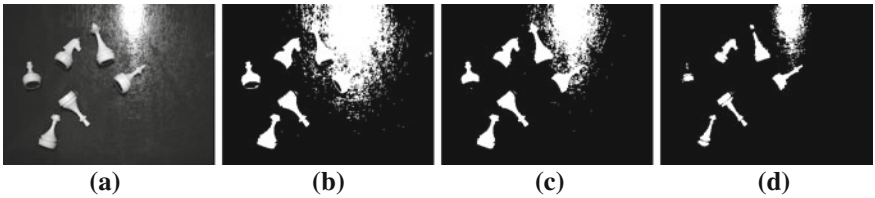


Fig. 4.1 **a** Original image, from [15] ; **b** Binary segmentation with threshold = 92 ; **c** Binary segmentation with threshold = 127 ; **d** Binary segmentation with threshold = 201

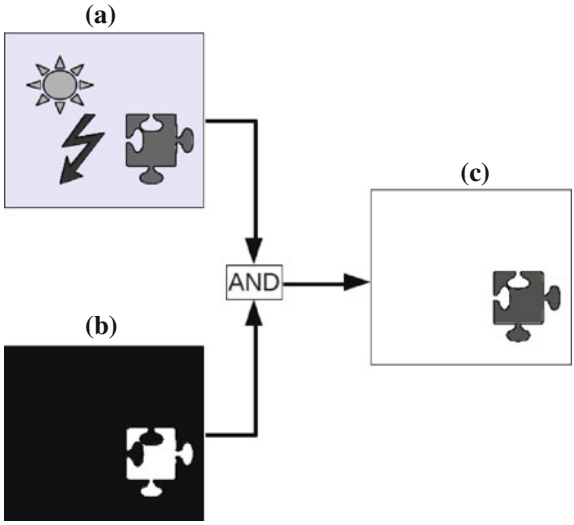


Fig. 4.2 Example of a mask operation: **a** is the original image, **b** is the mask, and **c** is the resulting segmented image

procedure with different thresholds are shown in Fig. 4.1, where one can observe the influence of the threshold level on the final result.

Another way to accomplish binarization of an image is by means of a mask where white pixels represent the object of interest and black pixels represent the background. Such mask is created by means of supervised learning. Once the mask is obtained, a logical operation is done between the mask and the original image (usually a logical AND). The result of this operation is an image with white background and a highlighted object. See an example of a mask operation in Fig. 4.2.

Image processing provides a large range of algebraic operations and convolution filters that can be applied to modify images [7]. A segmentation mask can be defined by the sequential application of these operations and filters. Therefore, the segmentation process, by using a mask, can be viewed as the application of a program. Such a program accepts an image as input, applies filters and operations in a predefined sequence, and yields another image as output.

Not only are the elements of the mask important (operations and filters), but also their internal parameters and the order of application are important for the final result.

Depending on the set of filters and operations, and the order of their application, the number of different possibilities grows exponentially. Consequently, it is necessary to use an efficient method to search the space of possible masks for a given segmentation problem.

4.3 Genetic Programming

GP is a method for the automatic evolution of programs, which, in turn are candidate solutions for a given problem [8]. GP evolved from genetic algorithms (GAs) [6] and share with them the same Darwinian principles. The key idea is the principle of natural selection, where individuals that are better adapted to the environment have a larger probability of surviving and generating descendants. In this case, individuals are the candidate solutions for a problem, adaptability is a measure of quality of solutions, and the environment is the problem instances to which solutions are tested.

Possible solutions for a problem are represented as programs in the form of trees. The internal nodes of trees are functions (operators), and the leaf nodes are terminals (inputs to the functions). Therefore, the evolving set of structures includes functions and terminals, to be defined by the user.

The quality of solutions is evaluated by means of a fitness function. Since GP employs supervised learning to train programs, this function is computed over a set of instances of the problem. Each instance is composed by inputs and a desired output. The inputs are applied to the program that, when executed, generates an output. This output is compared with the desired output, and a error value is computed. The summation of all errors over all training instances is a good measure of quality. It tends to zero when the program is able to reproduce, as expected, the outputs for all the given inputs.

During the evolutionary process, GP applies genetic operators over the population of solutions so as to create new generations of solutions. The natural selection principle is present in the selection procedure, by which individuals with good fitness have larger probability to be selected to be submitted to the reproduction and/or crossover operators. The reproduction operator simply copies an individual from the current to the next generation. The crossover operator cuts randomly selected branches into two individuals and recombines these parts.

To apply GP to a real-world problem, five definitions are necessary, as follows:

1. the terminals set;
2. the nonterminals (functions) set;
3. a measure of fitness;
4. the value of the control parameters;
5. a stop criterion.

The terminals set includes the operands that will be used as input data to the functions of the nonterminals sets. Both terminals and nonterminals must be carefully chosen, since the final solution of the problem will be composed of those elements.

If these sets do not contain all the necessary elements, GP will not be able to produce good solutions. On the other hand, if those sets include a large number of unnecessary elements, GP will face a difficult challenge in selecting the most appropriate ones. Therefore, the user has an important role in the definition of the terminals and nonterminals set, usually based on his/her knowledge of the problem.

The definition of the terminals and nonterminals sets must satisfy two criteria: closure and sufficiency. The first states that each function has to accept as input any value or kind of data that can be generated by any combination of terminals of outputs of functions. The latter states that the superset of terminals and nonterminals must have all the elements needed for a satisfactory solution to the problem.

In the same way as all evolutionary computation methods, GP also has a number of control parameters that the user is in charge of setting at each run. Amongst all those parameters, two have the strongest impact on the final results: the population size and the number of generations. Reaching a predefined number of generations is the most usual stopping criterion, although a quality criterion is also used. A more detailed description of GP is found in [8].

4.4 Methodology

In [1], GP was applied as a solution to the problem of object recognition. The methodology of this paper is similar, but here some different strategies were used to control the GP tree solution. In that paper, the authors restricted the training phase to carefully selected regions in the images, thereby introducing more difficulty in the process of creating filters. In this work, any kind of restriction was used, i.e, the entire image is used in the process of GP training.

For the image segmentation problem discussed in this work, the terminals set is formed by the input image and 15 other images, all of them obtained directly from the application of convolution filters on the original input image. Table 4.1 shows the 16 terminals, their symbols and respective definitions.

The nonterminals set is composed of 27 algebraic operations and convolution filters, all of them named here as operators. Attention should be paid to the fact that these operators are not the same as the genetic operators. These nonterminals take as arguments one or two images and produce another image. The set of nonterminals is presented in Table 4.2.

By inspection, both terminals and nonterminals sets satisfy the required closure criterion. Regarding the sufficiency criterion, the experiments later reported will show their adequacy.

The measure of quality of a solution is given by Eq. (4.1), where M is the image generated by GP, P is the standard mask that it attempts to find. To limit the interval of the function to the range [0..1], a normalization is done, dividing by the largest possible value that the function can achieve, in this case, $Max = 255 * width * height$, where width and height are the size of the images, and 255 is the number of gray levels.

Table 4.1 Terminals set used by the proposed GP

#	Symbol	Description
0	IM_0	Original image
1	IM_1	3 × 3 mean filter
2	IM_2	5 × 5 mean filter
3	IM_3	7 × 7 mean filter
4	IM_4	3 × 3 standard-deviation filter
5	IM_5	5 × 5 standard-deviation filter
6	IM_6	7 × 7 standard-deviation filter
7	IM_7	3 × 3 maximum filter
8	IM_8	5 × 5 maximum filter
9	IM_9	7 × 7 maximum filter
10	IM_10	3 × 3 minimum filter
11	IM_11	5 × 5 minimum filter
12	IM_12	7 × 7 minimum filter
13	IM_13	3 × 3 median filter
14	IM_14	5 × 5 median filter
15	IM_15	7 × 7 median filter

$$Dif_x = \frac{\sum_{i=1}^{L} \sum_{j=1}^{A} |M_{i,j} - P_{i,j}|}{Max}. \tag{4.1}$$

During the evolution of solutions by the GP, trees tend to grow in size (total number of nodes and/or depth). Consequently, an increasing complexity of solutions is observed as generations increase. The increase in the complexity of trees is not necessarily accompanied by an increase in the quality of solutions due to the proliferation of introns. Introns are elements present in the trees (combinations of terminals and nonterminals) that do not affect (or have an insignificant effect on) the quality of the solution represented by the tree. However, since the final solution will be interpreted by a human, it is desirable that a good solution be as simple as possible. This can only be accomplished by enforcing the trees to be as small as possible, but, at the same time, to be as good as possible. An efficient strategy that can do this job was proposed by [2]. Equation (4.2) shows a penalty measure:

$$Pen_x = \frac{maxNodes - 0.5 * nodes_x - 0.5}{maxNodes - 1} \tag{4.2}$$

where *maxNodes* is the maximum number of nodes of a given solution (in our work an empirical limit of 65 was used), and *nodes* is the number of nodes of the current solution. This function gives values in the range [0.5..1.0]. The lower bound is reached when the worst possible case occurs: number of nodes is 65. On the other hand, the upper bound will be reached when the solution is as simple as a single node.

Therefore, the actual fitness function for a given individual x takes into account both the quality of solution and the penalty due to the number of nodes. Equation (4.3) shows the fitness function used in this work:

Table 4.2 Function set (nonterminals) used by the proposed GP

#	Operator	Description
1	ADD(A,B)	Add images A and B
2	SUB(A,B)	Subtract image B from A
3	MUL(A,B)	Multiply image A by B
4	DIV(A,B)	Divide image A by B
5	MAX2(A,B)	Maximum, pixel-by-pixel, of images A and B
6	MIN2(A,B)	Minimum, pixel-by-pixel, of images A and B
7	ADDC(A)	ADD a constant C to all pixels of image A
8	SUBC(A)	Subtract a constant C from all pixels of image A
9	MULC(A)	Multiply all pixels of image A by a constant C
10	DIVC(A)	Divide all pixels of image A by a constant C
11	SQRT(A)	Square root of image A
12	LOG(A)	Natural logarithm of image A
13	MAX_3 × 3(A)	3 × 3 maximum filter of image A
14	MAX_5 × 5(A)	5 × 5 maximum filter of image A
15	MAX_7 × 7(A)	7 × 7 maximum filter of image A
16	MIN_3 × 3(A)	3 × 3 minimum filter of image A
17	MIN_5 × 5(A)	5 × 5 minimum filter of image A
18	MIN_7 × 7(A)	7 × 7 minimum filter of image A
19	MEDN_3 × 3(A)	3 × 3 median filter of image A
20	MEDN_5 × 5(A)	5 × 5 median filter of image A
21	MEDN_7 × 7(A)	7 × 7 median filter of image A
22	MED_3 × 3(A)	3 × 3 mean filter of image A
23	MED_5 × 5(A)	5 × 5 mean filter of image A
24	MED_7 × 7(A)	7 × 7 mean filter of image A
25	STDV_3 × 3(A)	3 × 3 standard-deviation filter of image A
26	STDV_5 × 5(A)	5 × 5 standard-deviation filter of image A
27	STDV_7 × 7(A)	7 × 7 standard-deviation filter of image A

$$fitness_x = Dif_x * Pen_x \tag{4.3}$$

The development of this work was based on public-domain software. For the GP we used a modified version of *Lilgp* [13], version 1.1. For the image processing algorithms, we used the *Opencv* library, version 2.0 [3].

4.4.1 Test Set

To evaluate GP as a tool for mask generation for image segmentation problems, five sets of different images were used. Each set of images is composed of a number of similar images. However, there are significant differences between the sets.

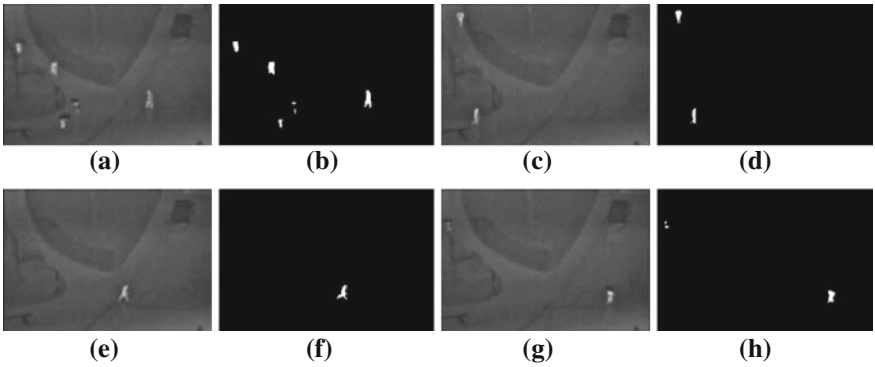


Fig. 4.3 Test set 1 (TS1) composed of four infrared images of people walking in a beach

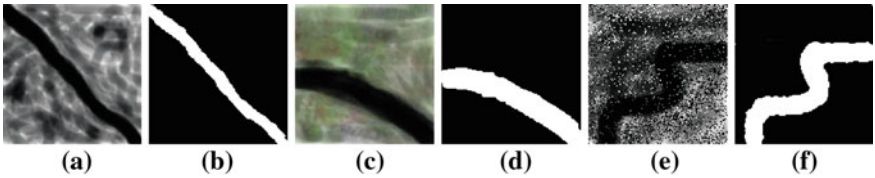


Fig. 4.4 Test set 2 (TS2) composed of three synthetic images

The first test set (TS1) is composed of four infrared images formerly published in [4]. The size of these images was 144 × 96 pixels. For this test set the intended binary masks were created by using an image editor software, and the objective is to segment images separating people from the background. Figure 4.3 shows the images of this test set (images [a], [c], [e] and [g]) together with the corresponding masks (images [b], [d], [f] and [h]).

The second test set (TS2) is composed of synthetic images created by the authors specifically for this work. It is composed of three images of 128 × 128 pixels, as shown in Fig. 4.4. As before, the corresponding mask for each image is presented beside the image.

The third test set (TS3) is formed by three aerial images of a river, extracted from Google maps.[1] The river is the object of interest, and it is surrounded by different types of terrain. The approximate location is at coordinates [−25.581156, −48.49494]. Images of this test set were 128 × 128 pixels, and the desired masks were constructed using image editor software. The objective is to highlight the river from the background. Figure 4.5 shows the test set and the corresponding masks.

The fourth test set (TS4) is composed by three images of cars circulating on a highway. These images had 128 × 96 pixels and were used by [11]. Figure 4.6 shows

[1] http://maps.google.com/

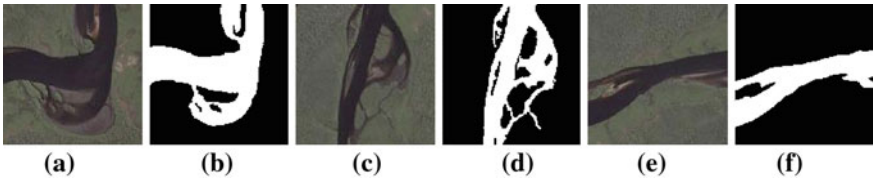


Fig. 4.5 Test set 3 (TS3) composed of three aerial images of a river

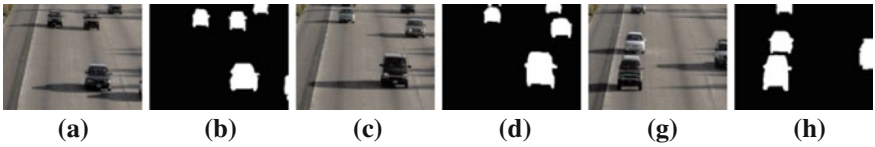


Fig. 4.6 Test set 4 (TS4) composed of images of cars in a highway

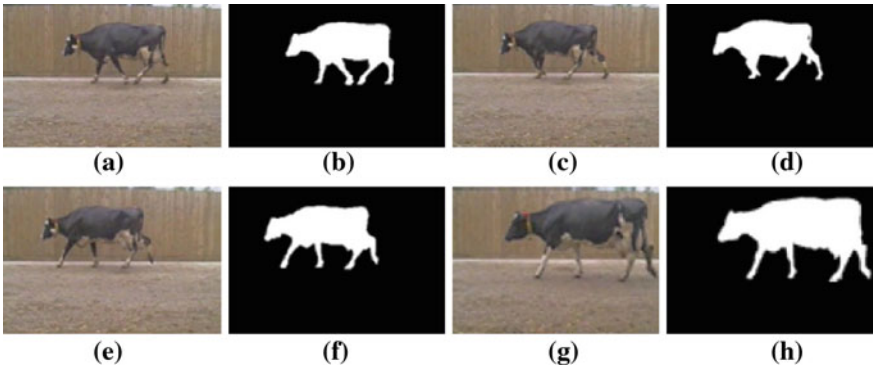


Fig. 4.7 Test set 5 (TS5) composed of images of walking cows

the images and corresponding masks, where it is possible to observe that the objective is to detect the cars from the background.

Finally, the last test set (TS5), shown in Fig. 4.7, is formed by four images of the benchmark provided by [10]. These images show a cow walking in different positions. These images have 150×100 pixels, and the objective of segmentation is to detect the cow in each image.

4.5 Computational Experiments and Results

Several experiments were done to evaluate the performance of the proposed GP method to find an appropriate sequence of filters capable of segmenting the images of the test sets according to the expected masks. The test sets previously presented

Table 4.3 Control parameters of GP

Parameter	Value
Population size	500
Number of generations	50
Initialization method	Ramped half and half
Initial depth of trees	[2..6]
Maximum depth of trees	10
Crossover probability	0.9
Reproduction probability	0.1

Table 4.4 Results of training for all test sets

	Nodes avg. ± std.dev.	Depth avg. ± std.dev.	Fitness avg. ± std.dev.
TS1	27.4333 ± 22.6619	7.5000 ± 2.9682	0.0368 ± 0.0169
TS2	24.9333 ± 10.5991	9.3000 ± 1.5120	0.0425 ± 0.0098
TS3	23.4666 ± 16.3596	8.7000 ± 2.1995	0.1407 ± 0.0411
TS4	26.4667 ± 18.8766	9.3333 ± 1.5829	0.1324 ± 0.0095
TS5	28.4333 ± 17.0368	9.1333 ± 2.0466	0.2607 ± 0.0254

were first used in a training section so as to obtain the desired mask. Later, other images, different from those used for training, were used for testing the robustness and generality of the solution found.

Since GP is a stochastic method, for all experiments a number of independent runs were done with the same inputs but with different initial random seeds, and the average results are presented. Experiments were run in a cluster of networked PCs with quad-core processors.

The control parameters of GP used in all experiments are shown in Table 4.3. These parameters, except those related to the size of the trees, are the default parameters for GP defined by [8]. No effort was done to fine-tune these parameters.

To select individuals for the crossover operator, we used the stochastic tournament selection method with 10 % of the population. For the reproduction operator, selection was done by the roulette-wheel method.

4.5.1 Training

Training was done using the images shown before. For each test set all but one images were used for training. The remaining image was used for testing (see next section). For each image of each test set, 30 independent runs were done with different initial random seeds. The results shown in Table 4.4 consider the average over all images of each test set, over all runs. Specifically for this experiment, the penalty term of Eq. (4.3) was set to 1, meaning that the size of the tree had no influence on the fitness of the solution. Later, this issue will be revisited. Since the fitness value is a function of

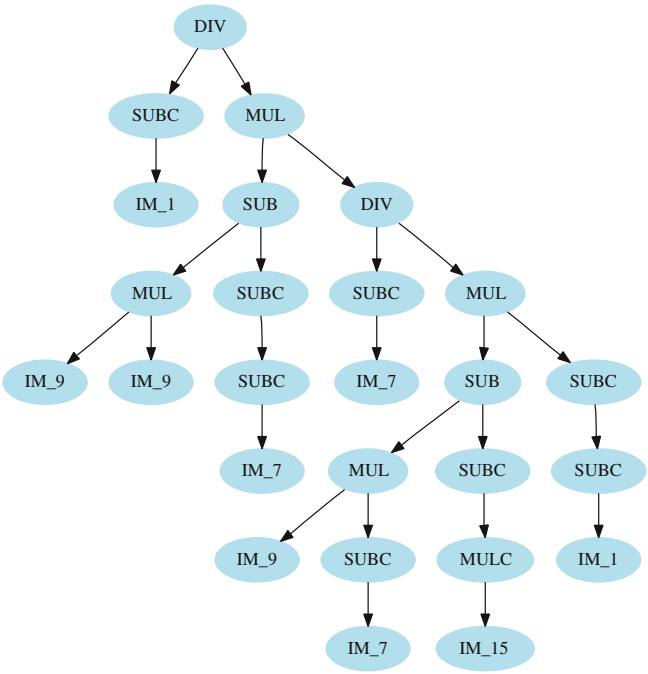


Fig. 4.8 Tree of best solution for TS1

the difference between the solution obtained by GP and that obtained by a handmade mask, it is possible to verify that the proposed method achieved solutions of good quality.

In Figs. 4.8, 4.9, 4.10, 4.11, 4.12 the best solutions found (amongst the 30 runs) are shown for each test set.

4.5.2 Testing

By using the image filters evolved by the GP approach, one image of each test set was used for testing. The image used for testing was not used for training. Results are shown in Fig. 4.13, including the original image, the expected segmentation (using the hand-made mask), and the actual image segmented by the application of the filter evolved by GP.

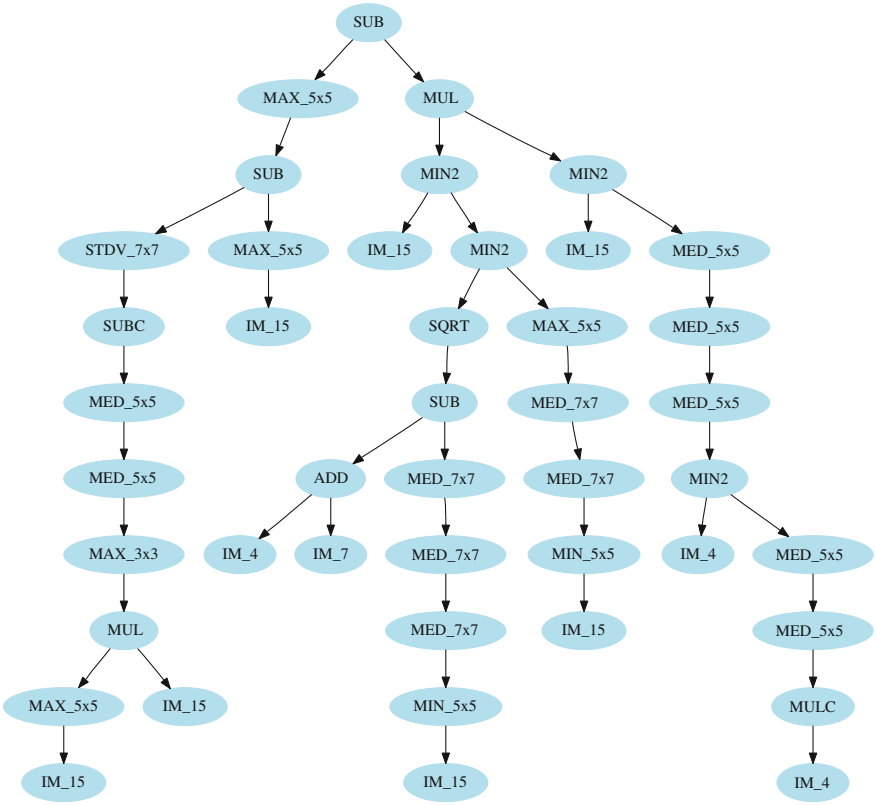


Fig. 4.9 Tree of best solution for TS2

4.5.3 Tree Size Control

The number of nodes of a tree evolved by GP has a direct relationship to the number of filters and operations necessary to accomplish the segmentation of an image. It is well-known that GP does not necessarily generate simple solutions [8]. It is possible that some of the terminals and nonterminals of the tree are useless for the final result. Such elements are known as introns. Although they may not affect the final segmentation, they represent a waste of processing time. Also, it is desirable that the evolved filter be understood by a human and, of course, understandability is inversely proportional to the number of nodes.

In order to reduce the number of nodes of the trees evolved by GP, the penalty measure shown in Eq. (4.2) was effectively used as part of the fitness function [Eq. (4.3)].

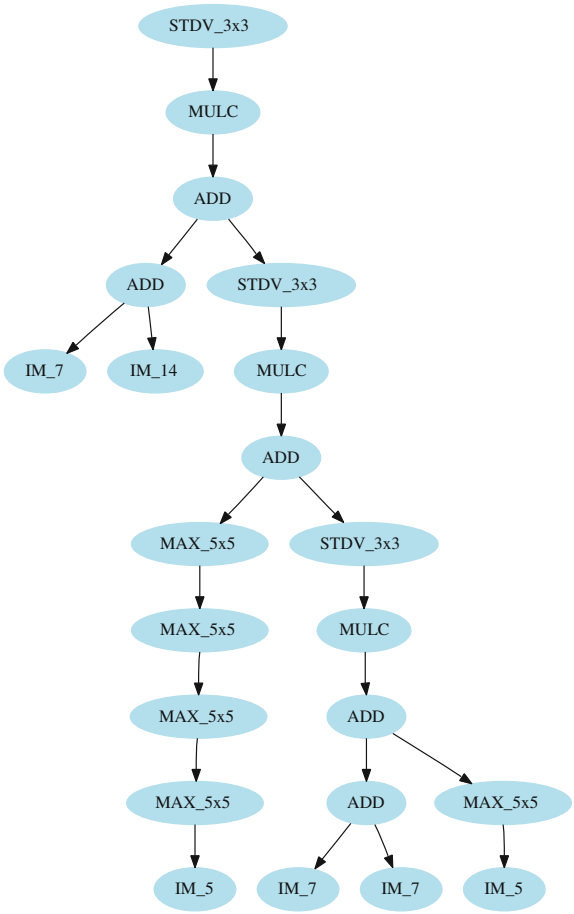


Fig. 4.10 Tree of best solution for TS3

In Table 4.5, results for the five test sets are shown. Values correspond to the average and standard deviation of the number of nodes, tree depth and fitness value, all referred to the best tree found by GP in 30 independent runs.

The average number of nodes of the solutions when using the fitness function with the penalty term was significantly lower than when ignoring the size of the trees. Although this method is very efficient for restricting the number of nodes of evolved trees, when comparing the corresponding values of Tables 4.4 and 4.5 it is possible to observe a decrement of quality between 5–20 %. Although the values of fitness are still high in the second table, restricting the size of the trees led to solutions of lower quality.

To facilitate the comparison of solutions with and without the penalty (restricting the number of nodes), we used a Pareto plot, as shown in Fig. 4.14. In this plot, the x axis is the complexity (number of nodes of the trees), and the y axis is the quality (fitness value). Recall that the fitness value is the difference between the expected

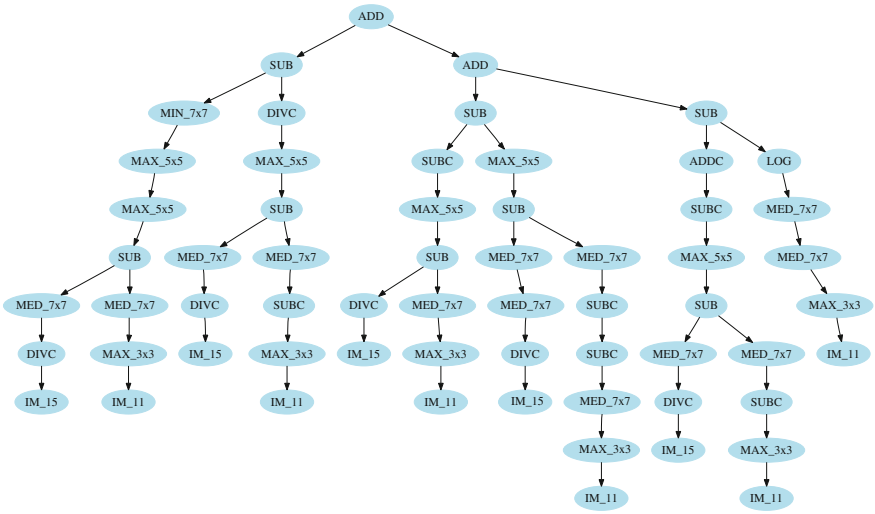


Fig. 4.11 Tree of best solution for TS4

segmentation and the one actually obtained, and therefore this is a minimization problem. As a result, the theoretical best possible value would be fitness = 0 and the number of nodes as close as possible to zero. In the Pareto plot the best solutions are found close to the origin of the coordinate system. It can be observed in the plot that the solutions found using the penalty term (restricting the number of nodes) dominate the other solutions without restriction.

To show the efficiency of the penalty term for pruning the evolved trees, Figs. 4.15, 4.16, 4.17, 4.18, 4.19 show the best solutions found for each of the five test sets, among 30 runs.

A direct comparison between Figs. 4.15, 4.16, 4.17, 4.18, 4.19 and Figs. 4.8, 4.9, 4.10, 4.11, 4.12 shows a large reduction in the complexity of the filters created by GP, but still keeping about the same quality of solutions.

4.5.4 Frequency of Use of Nonterminals

A relatively large set of nonterminals were used for the experiments here described. Such nonterminals were selected without any previous knowledge about their utility for the problem. As a matter of fact, the search space has a direct relationship with the size of the nonterminals set. Here we find a dilemma: if a large nonterminals set is used, the search space increases and the efficiency of GP may not be satisfactory. On the other hand, if a small set is used, the sufficiency criterion may not be satisfied. As shown before, the use of the penalty term decreases the number of nodes of the trees, at the expense of a small decrease in the quality of solutions. We investigated

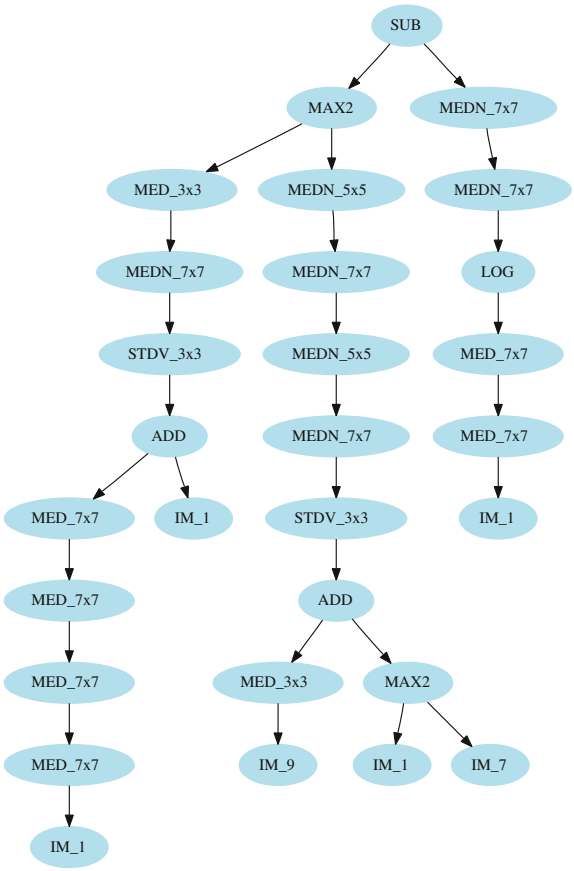


Fig. 4.12 Tree of best solution for TS5

the frequency of nonterminals (functions) in the solutions evolved by GP, so as to evaluate how parsimonious one can be regarding the nonterminals set.

The number of each nonterminal was counted during the whole evolution of the solutions (all individuals in all generations). The average over 30 runs was computed (25,000 fitness evaluations), and a frequency plot was constructed for all test sets. Figure 4.20 shows the results for the experiments without the penalty term, and Fig. 4.21 shows the results for the experiments using the penalty term.

It is possible to observe in the frequency plots that the use of the penalty term leads to a significant decrease in the frequency of use of nonterminals. Without the use of the penalty term, the average number of nonterminals used during the whole evolution (considering all test sets, all runs and all generations) was 36,788. On the other hand, when we used the penalty term, the average number significantly decreased to 11,053. If we analyze this fact together with the small decrease in quality mentioned before, it is possible to infer that some nonterminals do not make

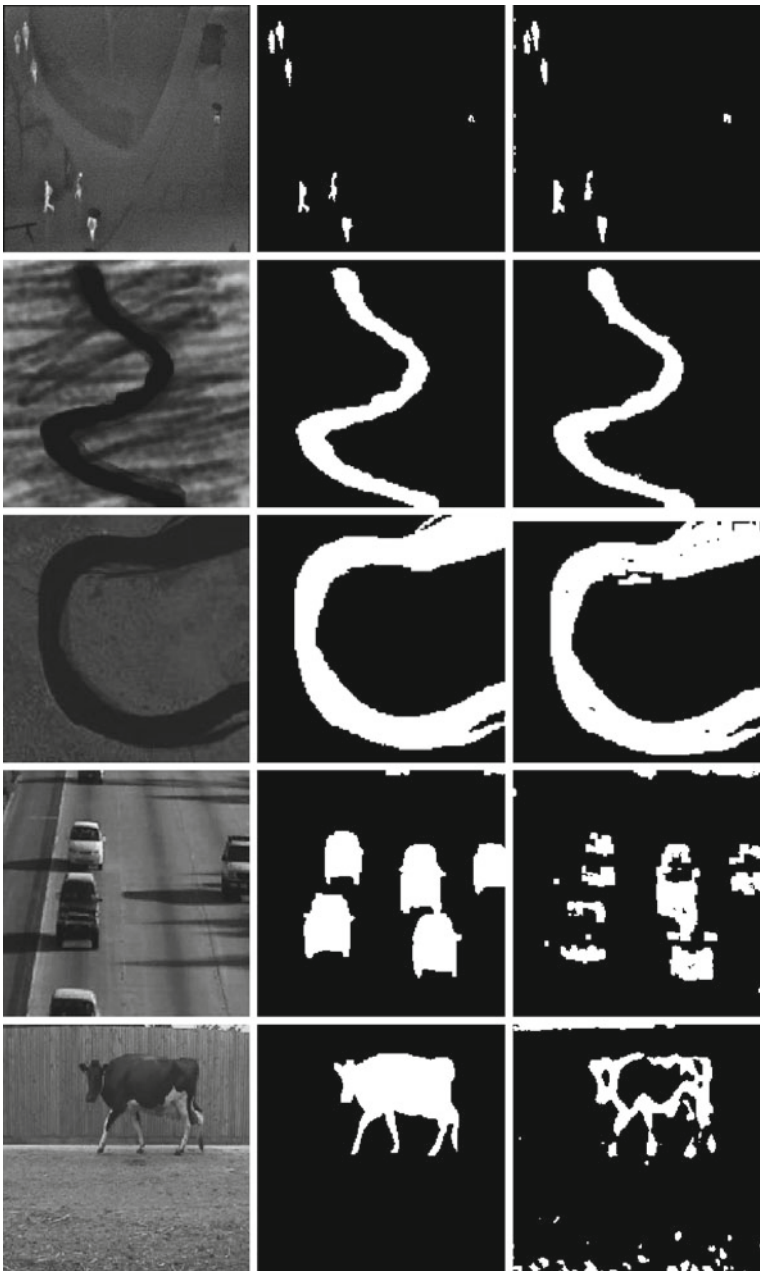


Fig. 4.13 Test of the filters evolved by GP

Table 4.5 Results obtained running GP with a penalty for increasing number of nodes

	Nodes avg. ± std.dev.	Depth avg. ± std.dev.	Fitness avg. ± std.dev.
TS1	3.9333 ± 1.0807	2.4667 ± 1.1666	0.0930 ± 0.0150
TS2	5.8000 ± 1.1567	3.5667 ± 0.9353	0.1005 ± 0.0193
TS3	6.0333 ± 1.3515	4.3667 ± 1.3515	0.1835 ± 0.0629
TS4	4.4667 ± 1.0743	2.4000 ± 1.0699	0.1865 ± 0.0120
TS5	7.1000 ± 2.1711	4.7667 ± 1.7749	0.3472 ± 0.0377

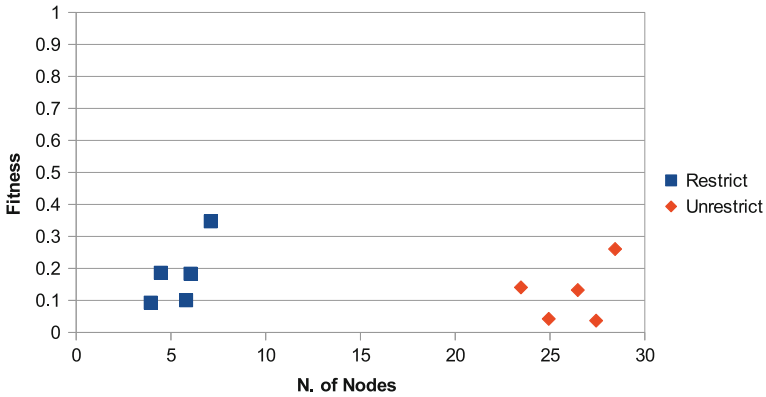


Fig. 4.14 Pareto plot for evaluating quality and complexity of solutions

important contributions to the quality of solutions and, possibly, they could be deleted from the nonterminals set.

In fact, without the penalty term, there is no clear pattern of use of nonterminals. However, using the penalty term, it is possible to observe a clear preference in using some of the nonterminals over others, thus supporting the assertion of the previous paragraph.

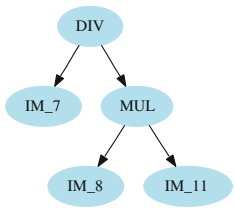


Fig. 4.15 Tree of the best solution using the penalty method, for TS1

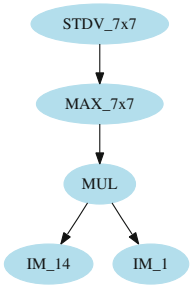


Fig. 4.16 Tree of the best solution using the penalty method, for TS2

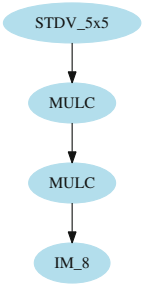


Fig. 4.17 Tree of the best solution using the penalty method, for TS3

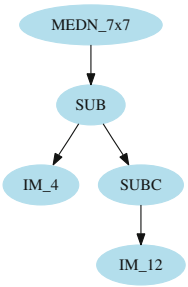


Fig. 4.18 Tree of the best solution using the penalty method, for TS4

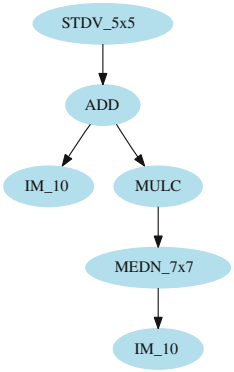


Fig. 4.19 Tree of the best solution using the penalty method, for TS5

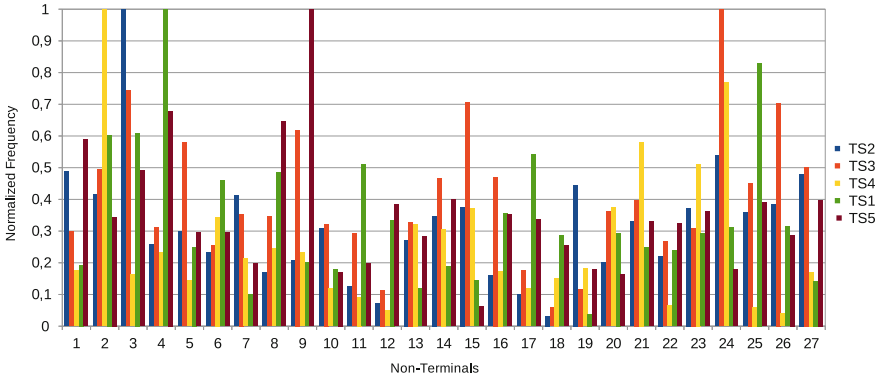


Fig. 4.20 Frequency histogram of the nonterminals for runs not using the penalty term

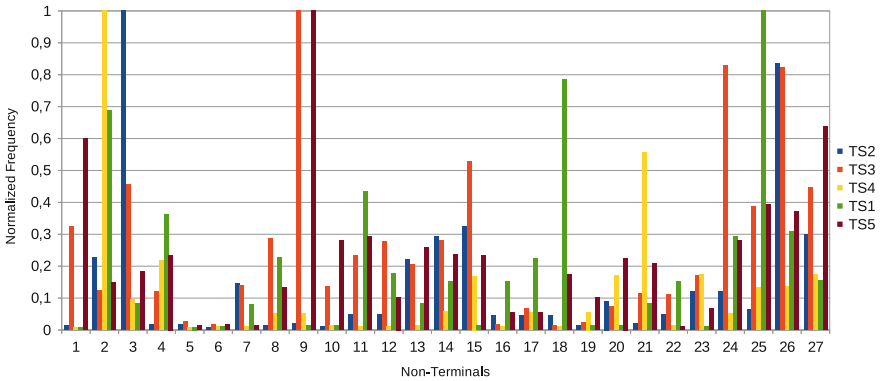


Fig. 4.21 Frequency histogram of the nonterminals for runs using the penalty term

4.6 Conclusions

Image segmentation is a very important procedure in image processing and computer vision. Although there are specialized methods for this purpose, that is, methods for specific classes of objects of interest, it is important to investigate general-purpose methods, such as GP, for this task. In particular, when the final objective is object recognition in images, this is still an open problem.

The results of the experiments described in this work strongly suggest the utility of GP for the image segmentation problem. After a training procedure, filters evolved with GP were satisfactorily applied to segment similar images, giving results of good quality.

We also verified that solutions found by GP tend to be complex, including many terminals and nonterminals. From the point of view of image segmentation, this fact may be interesting, since a human could hardly imagine the combined use of those elements for a given segmentation. However, on the other hand, complex trees are

quite difficult for a human to understand, and the corresponding filters, although efficient, can be computationally expensive.

In this work we proposed the use of a penalty term in the fitness function that was able to decrease significantly the complexity of the evolved trees, at the expense of a small reduction in quality. This study suggests that the original set of nonterminals could be significantly decreased, leading to better understanding of the evolved solutions and improvement in processing time. Overall results were very promising for all test sets, and future work will include the investigation of other sets of terminals and nonterminals, as well as the application of this methodology to other classes of images and problems, such as multilevel segmentation.

References

1. Bhanu, B., Lin, Y.: Object detection in multimodal images using genetic programming. Appl. Soft Comput. **4**(2), 175–201 (2004)
2. Bojarczuk, C., Lopes, H., Freitas, A., Michalkiewicz, E.: A constrained-syntax genetic programming system for discovering classification rules: application to medical data sets. Artif. Intell. Med. **30**(1), 27–48 (2004)
3. Daves, B.: Open computer vision library. http://sourceforge.net/projects/opencvlibrary/ (2010)
4. Davis, J.W., Keck, M.A.: A two-stage template approach to person detection in thermal imagery. In: Proceedings of the Seventh IEEE Workshops on Application of Computer Vision, vol. 1, pp. 364–369 (2005)
5. Frucci, M., Baja, G.S.: From segmentation to binarization of Gray-level images. J. Pattern Recognit. Res. **3**(1), 1–13 (2008)
6. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
7. Gonzalez, R.C., Woods, R.E.: Digital Image Processing, 3rd edn. Prentice-Hall, Upper Saddle River (2006)
8. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. The MIT Press, Cambridge (1992)
9. Kwon, S.: Threshold selection based on cluster analysis. Pattern Recognit. Lett. **25**(9), 1045–1050 (2004)
10. Leibe, B., Leonardis, A., Schiele, B.: Robust object detection with interleaved categorization and segmentation. Int. J. Comput. Vis. **77**(1–3), 259–289 (2008)
11. Martel-Brisson, N., Zaccarin, A.: Kernel-based learning of cast shadows from a physical model of light sources and surfaces for low-level segmentation. In: IEEE Conference on Computer Vision and, Pattern Recognition, pp. 1–8 (2008)
12. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)
13. Punch, B., Zongker, D.: Lil-gp genetic programming system. http://garage.cse.msu.edu/software/lil-gp/ (2010)
14. Shapiro, L.G., Stockman, G.C.: Computer Vision. Prentice-Hall, New Jersey (2001)
15. Silva, R., Erig Lima, C., Lopes, H.: Template matching in digital images using a compact genetic algorithm with elitism and mutation. J. Circuits Syst. Comput. **19**(1), 91–106 (2010)
16. Xu, X., Xu, S., Jin, L., Song, E.: Characteristic analysis of Otsu threshold and its applications. Pattern Recognit. Lett. **32**(7), 956–961 (2011)
17. Zhang, H., Fritts, J., Goldman, S.: Image segmentation evaluation: a survey of unsupervised methods. Comput. Vis. Image Underst. **110**(2), 260–280 (2008)

Part II
Image Compression Algorithms

Chapter 5
Fuzzy Clustering-Based Vector Quantization for Image Compression

George E. Tsekouras and Dimitrios M. Tsolakis

Abstract The implementation of fuzzy clustering-based vector quantization (VQ) algorithms in image compression is related to three difficulties: (a) the dependence on initialization, (b) the reduction of the computational cost, and (c) the quality of the reconstructed image. In this paper, first we briefly review the existing fuzzy clustering techniques used in VQ. Second, we present a novel algorithm that utilizes two stages to deal with the aforementioned problems. In the first stage, we develop a specialized objective function that incorporates the c-means and the fuzzy c-means in a uniform fashion. This strategy provides a tradeoff between the speed and the efficiency of the algorithm. The joint effect is the creation of hybrid clusters that possess crisp and fuzzy areas. In the second stage, we use a utility measure to quantify the contributions of the resulting clusters. Clusters with small utilities are relocated (i.e., migrated) to fuzzy areas of large clusters so that they can increase their utility and obtain a better local minimum. The algorithm is implemented in gray-scale image compression, where its efficiency is tested and verified.

5.1 Introduction

Image compression deals with the reduction of the number of bits required to store and transmit digital images. The methods developed so far to perform image compression can be classified in two categories: lossless and lossy compression. Lossless compression is an error-free procedure according to which, the pixel intensities of the

G. E. Tsekouras (✉) · D. M. Tsolakis
Laboratory of Intelligent Multimedia,
Department of Cultural Technology and Communication,
University of the Aegean, 81100 Mytilini, Lesvos Island, Greece
e-mail: gtsek@ct.aegean.gr

D. M. Tsolakis
e-mail: ctmb05018@ct.aegean.gr

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_5, © Springer-Verlag Berlin Heidelberg 2013

original image are fully recovered in the compressed image representation. Although the quality of the resulting image is excellent, this approach is computationally complex and yields low compression rates. By contrast, lossy compression attempts to compromise the quality of the recovered image in exchange for higher compression rates. Thus, although the original image cannot be perfectly recovered, the computational demands are significantly reduced.

Lossy compression is based on the decomposition of the image into a number of rectangular blocks that form a set of multidimensional training vectors $X = \{x_1, x_2, ..., x_N\}$, where N is the total number of training vectors and $x_k \in \Re^p$ $(1 \leq k \leq N)$. A common procedure to perform image compression is vector quantization (VQ). The basic undertaking of VQ is to set up a partition of X into a number of disjoint classes (clusters). Each cluster is represented by its center element called codeword. The set of codewords is referred to as the codebook and is symbolized as $V = \{v_1, v_2, ..., v_c\}$, where $v_i \in \Re^p$ is the ith codeword. To obtain a partition of X, VQ minimizes the following distortion measure:

$$D = \frac{1}{N} \sum_{k=1}^{N} \min_{1 \leq i \leq c} \left\{ \|x_k - v_i\|^2 \right\} \tag{5.1}$$

The image is reconstructed by replacing each training vector by its closest codeword.

A wide spectrum of methods has been developed to implement VQ in image compression. Many of these approaches are based on adaptive VQ [14, 15], fast VQ [8, 13, 18], reinforced learning [3] or special transformations [1, 10]. VQ methods can be classified into two categories, namely crisp and fuzzy. Crisp VQ is based on hard decision making processes and appears to be sensitive in codebook initialization. The most representative algorithm of this category is the c-means. To improve the behavior of c-means, Linde et al. [11] introduced the LBG algorithm, which begins with the smallest codebook size and gradually increases it using a splitting procedure. Later publications further improved the performance of LBG by embedding in the learning process special functions called utility measures [2, 12]. The basic idea was to detect codewords with small utilities and relocate them close to codewords with large utilities.

On the other hand, fuzzy VQ (FVQ) is carried out in terms of fuzzy cluster analysis. The most representative algorithm of this category is the fuzzy c-means [4, 5]. The fuzzy c-means assumes that each training vector belongs to multiple clusters with different participation degrees (i.e., membership degrees).Therefore, the learning is a soft decision making process.

In this paper we develop a new FVQ algorithm that deals with certain problems related to the implementation of fuzzy clustering in image compression such as the speed, the effect of initialization and the quality of the reconstructed image. The central idea is to combine the c-means and the fuzzy c-means in a uniform fashion. In addition, to significantly reduce the dependence on initialization, we equip the algorithm with a specialized codeword migration approach according to

which, small clusters are migrated to locations close to large clusters so that their individual contributions to the final partition increase.

The paper is organized as follows. Section 5.2 presents a brief review of the existing FVQ approaches. In Sect. 5.3 we analytically describe the proposed algorithm. The simulation experiments are illustrated in Sect. 5.4. Finally, we present our conclusions in Sect. 5.5.

5.2 Fuzzy Clustering-Based Vector Quantization

The theoretical background of VQ assumes that each training vector is assigned to one and only one codeword (i.e., clusters are disjoint sets). This is the main reason for using VQ quantization in image compression, because it keeps the transmitted information minimal. However, fuzzy c-means obtains a partition where each training vector belongs to more than one cluster. Therefore, we are required to assign each training vector to the codeword that has the maximum membership degree. Such a crisp interpretation of the fuzzy c-means imposes serious problems in the quality of the codebook and, eventually, in the quality of the reconstructed image [6].

A solution to this problem is to equip the learning process with a transition strategy from fuzzy conditions, where each training vector is assigned to multiple codewords, to crisp (or near-to-crisp) conditions, where each training vector is clearly assigned to only one codeword. In [16] Tsao et al. introduced the fuzzy learning vector quantization (FLVQ) algorithm, which gives a near-to-crisp partition by manipulating the fuzziness parameter from large (fuzzy conditions) to small (crisp conditions) values. However, FLVQ is computationally demanding. In [5] Karayiannis and Bezdek proved that FLVQ and fuzzy c-means can be integrated in a unique algorithmic platform. In [7], Kong et al. proposed the concept of resolution in order to produce a class of clustering algorithms that were generalizations of the c-means and fuzzy c-means. But this method also resulted in high computational cost. In [6], Karayiannis and Pai presented three improved versions of fuzzy c-means that were able to avoid the crisp interpretation of the fuzzy c-means and reduce the computational cost as well. Specifically, they developed special mechanisms to reduce the number of distance calculations and to fully transfer all training vectors in crisp conditions. However, this transition strategy is based on experimental observations, and it lacks mathematical preciseness. In [17] the transition from fuzzy to crisp conditions was guided by analytical conditions that were extracted by the optimization of a specialized objective function. The basic idea was to utilize the c-means and the fuzzy c-means under the assumption that both of them contribute equally to the training process. However, this strategy provides a restricted behaviour, since the number of degrees of freedom is limited.

In [18, 19] the above objective function was generalized so that the weight of significance between c-means and fuzzy c-means varies. This procedure provides the ability to quantitatively measure the tradeoff between the speed and the efficiency of the algorithm.

5.3 The Proposed Algorithm

In this section, we develop an algorithmic scheme that starts with fuzzy conditions and gradually is transferred in crisp conditions. The basic assumption is that codewords far away from a specific training vector should not be affected by that vector. This claim provides a reasonable conclusion according to which, if we are in the position to control the number of codewords affected by a training vector, then we can reduce this number and eventually reduce the number of distance calculations, meaning that the computational cost is reduced, also. To achieve this target, we define the set Q_k as the collection of the codewords affected by x_k. Using the concept of membership degree, Q_k is,

$$Q_k = \{ v_i \in V : \ u_{ik} > 0\} \tag{5.2}$$

where $u_{ik} \in [0, 1]$ is the fuzzy membership degree of the kth training vector to the ith cluster. Notice that the definition of Q_k in Eq. (5.2) implies that a codeword is affected by x_k if and only if the corresponding membership degree is positive, which is a natural and obvious hypothesis. When the vector x_k is in fuzzy mode then $\aleph(Q_k) > 1$; otherwise it is in crisp mode and $\aleph(Q_k) = 1$, where $\aleph(\cdot)$ stands for the set cardinality.

Figure 5.1 depicts the transition from fuzzy to crisp mode for a specific training vector.

As soon as the vector x_k has been transferred in crisp mode we calculate its membership degrees as

$$u_{ik} = \begin{cases} 1, & if \ ||x_k - v_i||^2 = \min\limits_{1 \le j \le c} \left\{ ||x_k - v_j||^2 \right\} \\ 0, & \text{otherwise} \end{cases} \tag{5.3}$$

In order to estimate the membership degrees, in the case where x_k is in fuzzy mode, we have to provide a mechanism according to which the codewords located in distant points from x_k must be removed from Q_k. To deal with this problem, we combine the c-means and the fuzzy c-means so that we are able to switch from fuzzy to crisp conditions as follows,

$$J = \theta \sum_{k=1}^{N} \sum_{i: v_i \in Q_k} u_{ik} \|x_k - v_i\|^2 + (1-\theta) \sum_{k=1}^{N} \sum_{i: v_i \in Q_k} (u_{ik})^2 \|x_k - v_i\|^2 \tag{5.4}$$

Thus, in each iteration, we minimize the function J under the next constraint,

$$\sum_{i: v_i \in Q_k} u_{ik} = 1, \qquad \forall k \tag{5.5}$$

The parameter θ determines the relative influence of c-means and fuzzy c-means. If θ takes a small value then fuzzy conditions govern the VQ design, while large

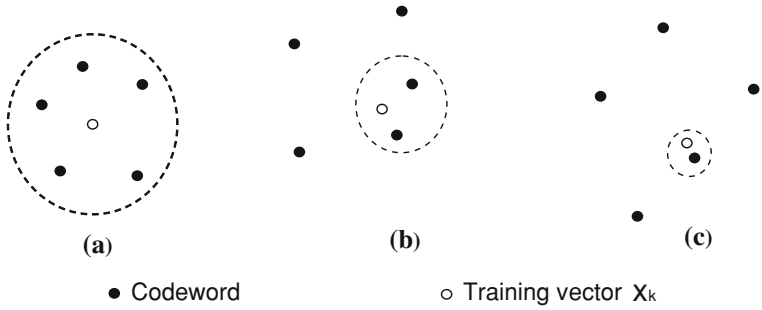


Fig. 5.1 Transition from fuzzy to crisp mode for the training vector x_k: **a** $\aleph(Q_k) = 5$, **b** $\aleph(Q_k) = 2$, and **c** $\aleph(Q_k) = 1$

values of θ force many training vectors to be transferred in crisp mode. In this paper we select $\theta \in [0.3,\ 0.7]$. Notice that throughout the VQ design the parameter θ remains constant. Taking the Lagrange multipliers, the membership degrees that solve the above optimization problem are calculated as

$$u_{ik} = \frac{2 + (\aleph(Q_k) - 2)\,\theta}{2(1-\theta)} \frac{1}{\sum_{j:\, v_j \in Q_k} \left(\frac{\|x_k - v_i\|}{\|x_k - v_j\|} \right)^2} - \frac{\theta}{2(1-\theta)} \tag{5.6}$$

while if $v_i \notin Q_k$ then $u_{ik} = 0$. In view of Eq. (5.6), u_{ik} can be negative or zero. Setting $u_{ik} \leq 0$ we obtain

$$\|x_k - v_i\|^2 \geq \frac{2 + (\aleph(Q_k) - 2)\theta}{\theta} \frac{1}{\sum_{j:\ v_j \in Q_k} \left(\frac{1}{\|x_k - v_j\|} \right)^2} \tag{5.7}$$

where $v_i \in Q_k$. It thus appears that whenever the condition (5.7) holds, x_k appears negative or zero membership degree with respect to the codeword v_i. In this case we set $u_{ik} = 0$ and remove the v_i from Q_k. Therefore, in the tth iteration, the set Q_k is updated as,

$$Q_k^{(t)} = \left\{ v_i \in Q_k^{(t-1)} :\quad u_{ik} > 0 \right\} \tag{5.8}$$

However, if we set all negative membership degrees equal to zero then the condition in Eq. (5.5) is not satisfied. To solve this problem we normalize the membership degrees as indicated next,

$$\tilde{u}_{ik} = \frac{u_{ik}}{\sum_{j:\, v_j \in Q_k} u_{jk}}, \forall\, v_i \in Q_k \tag{5.9}$$

The codewords that minimize J are estimated by setting the partial derivatives of J equal to zero and solving for the codeword v_i,

$$v_i = \frac{\sum_{k:\, v_i \in Q_k} \left[\theta\, \tilde{u}_{ik} \,+\, (1-\theta)\, (\tilde{u}_{ik})^2\right]\, x_k}{\sum_{k:\, v_i \in Q_k} \left[\theta\, \tilde{u}_{ik} \,+\, (1-\theta)\, (\tilde{u}_{ik})^2\right]} \tag{5.10}$$

According to the analysis so far, a specific cluster includes crisp and fuzzy areas. Initially, the cluster's area is fuzzy and as the learning process proceeds the crisp area gradually expands from inside to outside. To quantitatively describe these two potentially different cases we define the crisp area of the ith cluster as

$$CA_i = \{x_k \in X :\ v_i \in Q_k \quad \text{and} \quad \aleph\,(Q_k) = 1\} \tag{5.11}$$

and the respective fuzzy area as

$$FA_i = \{x_k \in X :\ v_i \in Q_k \quad \text{and} \quad \aleph\,(Q_k) > 1\} \tag{5.12}$$

Although the utilization of the fuzzy c-means ensures to some degree that the algorithm is not too sensitive to initialization, our intention is to enhance the learning performance with a codeword migration process. Small clusters contribute less, while large clusters contribute most. To quantify these contributions we use the fuzzy partial distortion for each cluster,

$$D_i = \sum_{k=1}^{N} \tilde{u}_{ik}\, \|x_k - v_i\|^2 \tag{5.13}$$

Notice that in Eq. (5.13) we account for all training vectors, since those that are assigned zero membership degrees with respect to v_i do not influence the final result. Then, we employ the concept of utility measure developed by Patane and Russo in [12],

$$U_i = \frac{D_i}{\sum_{j=1}^{c} D_j} \tag{5.14}$$

As stated in [12], an optimal partition corresponds to the situation where all utilities are close to unity. Our scope is to detect clusters with low utilities and then migrate the respective codewords to positions close to clusters with large utilities. The criterion to select a small cluster is: $U_i \leq \gamma$, where $\gamma \in (0,\ 1)$. We have experimentally found that a trustworthy choice is $\gamma = 0.5$. In each iteration, we migrate all codewords that satisfy the above inequality. Relationally, the criterion to detect a big cluster is: $U_i > 1$. Then, in each iteration, we randomly select a big cluster that satisfies the previous condition. We denote the small cluster as C_s and its codeword as v_s. The large cluster at the neighborhood of which the v_s will be relocated is symbolized as C_l and its codeword as v_l.

The training vectors assigned to C_s before the migration must change their status after the migration. We distinguish two cases: (a) the C_s does not contain any crisp area, and (b) the C_s does contain a crisp area. Considering the first case, every

training vector that belongs to C_s also belongs to at least one of the neighboring clusters, which implies that its quantization procedure will be normally continued after the migration. Therefore, we just remove the v_s from the sets Q_k for all $x_k \in C_s$ by setting $Q_k = Q_k - \{v_s\}$. As far as the neighboring codewords are concerned, we leave their status unchanged. In the second case, we note that for the training samples in the set FA_s we simply perform the previously stated procedure. Contrary to the first case, the training samples that belong to CA_s are assigned only to v_s and we have to ensure that their quantization is established after the migration. To deal with this task, we detect the training vector x_{s_0} that satisfies the next condition,

$$\left\| x_{s_0} - v_s \right\|^2 = \arg\min \left\{ \|x_k - v_s\|^2 : \ x_k \in FA_s \right\} \tag{5.15}$$

We then change the status for all vectors $x_k \in CA_s$ as follows:

$$Q_k = \left\{ v_i : \ v_i \in Q_{s_0}, \ \forall x_k \in CA_s \right\} \tag{5.16}$$

We turn our discussion to study the steps needed to relocate the codeword v_s in the new position. To minimize the influence of this movement, the crisp areas of the neighboring clusters to C_l must not change, while the respective fuzzy areas must be affected as little as possible. In addition, the new position of v_s must be located to some distant point (not too far and not too close) from v_l, so that they will not block each other. Upon the assumption that a big cluster has a considerable crisp area (a fact that is implied by the learning process), we migrate the v_s to the position of a random training vector that belongs to FA_l. We then change the status of all samples that belong to C_l as $Q_k = Q_k \bigcup \{ v_s\} \ \forall x_k \in C_l$. Notice that we do not affect the crisp areas of the neighboring clusters. Therefore, in the worst case scenario, the neighboring clusters are forced to compete with v_s and v_l for the samples that they have in common.

The question that has to be answered is when the migration shall be confirmed. Based on the analysis so far, the clusters that belong to the old neighborhood of the small cluster can only increase their size, and eventually they do not play any role in confirmation decisions. The clusters close to C_l are not significantly influenced, because the migration focuses on samples that belong to the C_l only. Therefore, taking into account the need for a fast scheme, we decide to confirm the migration when both the updated utilities U_l and U_s are greater than γ. To accomplish this, we run the algorithm just once using all $x_k \in C_l$ as the training set and the set $\{v_s, \ v_l\}$ as the codebook. A major constraint is that a small cluster can be migrated only one time throughout the implementation of algorithm, while a big cluster can be taken into account only once within a specific iteration. Finally, the implementation of the migration process is based on relocating in parallel all the small clusters. To summarize, the proposed algorithm is described next.

The Proposed Algorithm

Step 1. Randomly initialize the codebook $V = \{v_1, v_2, ..., v_c\}$ and select values for θ and γ. Set $Q_k = \{v_1, v_2, ..., v_c\}\ \forall k$
Step 2. Calculate the membership degrees using Eq. (5.6) if the training vector is in fuzzy mode; otherwise use Eq. (5.3).
Step 3. Update the codewords as indicated in Eq. (5.10).
Step 4. Apply the migration process for all small clusters.
Step 5. If the distortion in Eq. (5.1) does not change significantly go to step 6; else go to step 2.
Step 6. Put all training vectors in crisp mode and run the algorithm using Eq. (5.3) to determine the membership degrees and Eq. (5.10) to update the codewords until the distortion is stabilized.

Notice that in step 6 we run the algorithm in crisp conditions for all training vectors. It was experimentally found that this choice guides the learning process to avoid undesirable local minima.

5.4 Experimental Study

To test the efficiency of the proposed method we compared it with four different algorithms, namely the LBG, the fuzzy learning vector quantization (FLVQ) [16], the fuzzy vector quantization (FVQ) developed in [6], and the improved fuzzy learning vector quantization (IFLVQ) [19].

The experimental data consisted of the well-known Lena and Airplane gray-scale images of size 512×512 pixels, which are shown in Fig. 5.2.

For each simulation we run all algorithms using the same initial conditions for each codebook size and for each image. We used ten different initializations for each codebook size and for each image. To carry out the experiments, each image was divided in 4×4 blocks, resulting in 16384 training vectors in the 16-dimensional feature space. The performances of the algorithms were evaluated in terms of the distortion measure given in Eq. (5.1) and the peak signal-to-noise ratio (PSNR),

$$\mathrm{PSNR} = 10\ \log_{10}\left(\left(512^2\ 255^2\right) \Big/ \sum_{i=1}^{512}\sum_{j=1}^{512}(I_{ij} - \tilde{I}_{ij})^2\right) \tag{5.17}$$

where 255 is the peak gray-scale signal value, I_{ij} denotes the pixels of the original image and $\tilde{I}_{ij}$ the pixels of the reconstructed image. To assess the impact of parameter θ, we ran all experimental cases for $\theta = 0.3$, $\theta = 0.5$, and $\theta = 0.7$.

The first experiment concerns the resulting distortion measure.

Table 5.1 depicts the mean values of the distortion measures for the two images and various codebook sizes. Notice that the distortion mean values achieved by different values of θ are all very close to each other in every experimental case. This

Fig. 5.2 Test images, Lena and Airplane

Table 5.1 Distortion mean values for different codebook sizes

Method	Airplane			Lena		
	c=128	c=256	c=512	c=128	c=256	c=512
LBG	1563.04	1381..85	1273.27	1162.49	1014.63	943.49
FLVQ	1212.51	1286.90	796.92	1093.41	959.24	878.79
FVQ	1200.98	1040.57	894.52	758.47	642.31	562.82
IFLVQ	1231.56	947.06	724.75	809.27	622.23	483.11
Proposed (θ=0.3)	1159.30	922.06	712.26	733.53	578.49	463.48
Proposed (θ=0.5)	1155.39	924.00	712.79	732.15	579.93	464.08
Proposed (θ=0.7)	1157.16	923.88	715.95	732.28	577.38	465.12

Table 5.2 PSNR mean values (in dB) using various codebook sizes

Method	Airplane			Lena		
	c=128	c=256	c=512	c=128	c=256	c=512
LBG	28.2362	28.7655	29.1215	29.5300	30.1113	30.4241
FLVQ	28.3537	29.0724	29.4241	29.7794	30.3485	30.7276
FVQ	29.3721	29.9941	30.6498	31.3654	32.0859	32.6581
IFLVQ	29.2632	30.4022	31.5619	31.0843	32.2231	33.3195
Proposed (θ=0.3)	29.5249	30.5182	31.6364	31.5099	32.5391	33.4992
Proposed (θ=0.5)	29.5399	30.5091	31.6341	31.5181	32.5284	33.4937
Proposed (θ=0.7)	29.5331	30.5097	31.6142	31.5174	32.5475	33.4842

observation directly implies a nonsensitive behavior of the algorithm as far as the selection of the parameter θ is concerned. In addition, in all cases, our algorithm obtains the smallest distortion value.

The second experiment compares the five algorithms in terms of the PSNR values. We generated codebooks of sizes $c = 2^{qb}$ $(qb = 7,\ 8,\ 9)$. Since each feature vector represents a block of 16 pixels, the resulting compression rate was equal to $qb/16$ bits per pixel (bpp). Table 5.2 summarizes the mean PSNR values obtained in this experiment. The results reported in this table are highly convincing, since in all cases the proposed method outperformed the others.

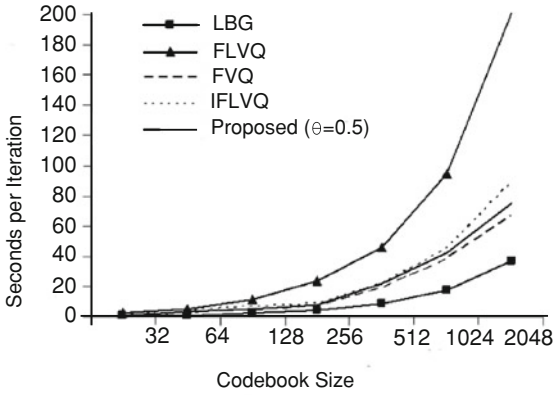


Fig. 5.3 Computational time in seconds per iteration as a function of the codebook size for the Lena image

In the third experiment, we quantified the computational demands. We used the Lena image to generate codebooks of sizes $c = 2^{qb}(qb = 5, 6, ..., 11)$ and measured the time needed by the CPU in seconds per iteration. We ran all algorithms using a computer with a dual-core CPU at 2.13 GHz and Matlab software.

The results are illustrated in Fig. 5.3. Notice that the FLVQ exhibits an exponential growth of the computational time, while, as expected, the LBG obtains the fastest performance. On the other hand the FVQ, the IFLVQ and the proposed algorithm maintain similar computational time.

The fourth experiment analyzes the cluster utility measures obtained by the FLVQ, the FVQ, the IFLVQ and the proposed algorithm. We ran the experiment for the Lena image and for a codebook of size c=256.

Figure 5.4 displays the utility distributions obtained by the FLVQ, the FVQ, the IFLVQ and the proposed method ($\theta = 0.5$). This figure clearly indicates the superiority of our method, since a large percentage of clusters maintain their utilities close to unity. Notice that the distribution obtained by the FLVQ shows the largest variance, allowing clusters with utilities greater than 3, while the largest amount of clusters are assigned utilities much less than unity (close to zero). Although IFLVQ and the FVQ managed to reduce this variance, they created a large number of clusters that are assigned very small utilities, something that is not desirable.

Finally, Table 5.3 shows a comparison between our algorithm and other methods existing in the literature. To perform the comparison, we used the Lena image because it is the most-used testing image. As far as this table indicates, the proposed algorithm obtains the best results in all cases.

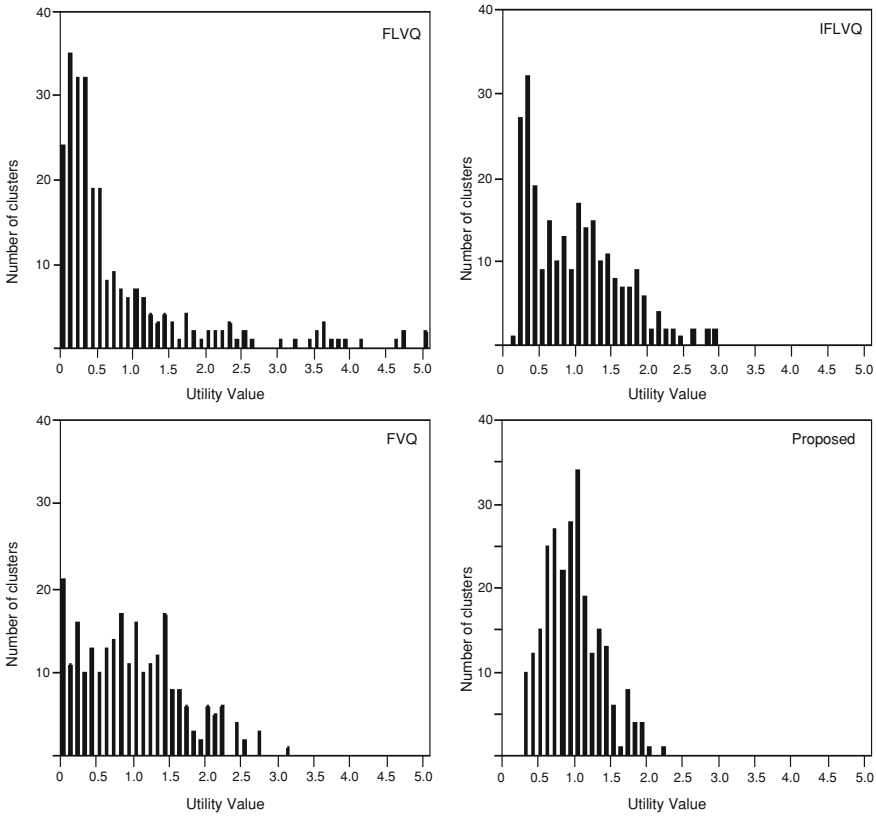


Fig. 5.4 Utility measure distributions obtained by the FLVQ, IFLVQ, FVQ and the proposed algorithm

Table 5.3 Literature comparison results

Method	c=256	c=512
Modified c-Means [9]	31.920	33.090
Enhanced LBG [12]	31.940	33.140
Adaptive Incremental LBG [14]	32.013	33.222
Proposed ($\theta = 0.5$)	32.528	33.494

5.5 Conclusions

We have developed a FVQ algorithm that attempts to resolve certain problems related to the implementation of fuzzy clustering in image compression. These problems are summarized as: (a) the computational complexity, (b) the quality of the reconstructed image, and (c) the dependence on initialization. To deal with the first two problems,

we propose to minimize an objective function that combines the merits of c-means and fuzzy c-means as well. The minimization of this function is controlled by special mechanisms that reduce the number of codewords affected by a specific training vector. The result is the creation of clusters that include crisp and fuzzy areas. To this end, the algorithm manages to reduce the computational demands and to avoid the crisp interpretation of the fuzzy c-means. The later has a straightforward influence on the quality of the reconstructed image. The third problem is resolved by utilizing a codeword migration strategy that is based on a cluster utility measure. Specifically, we detect clusters with small utilities and relocate the corresponding codewords to the fuzzy areas of big clusters. This strategy enhances the competition between clusters yielding better local minima. Finally, the efficiency of the algorithm is verified through a number of simulation experiments.

References

1. Chu, S.C., Lu, Z.M., Pan, J.S.: Hadamard transform based fast codeword search algorithm for high-dimensional VQ encoding. Inf. Sci. **177**, 734–746 (2007)
2. Fritzke, B.: The LBG-U method for vector quantization: an improvement over LBG inspired from neural networks. Neural Process. Lett. **5**, 35–45 (1997)
3. Ji, Z., Yang, T., Jiang, L., Xu, W.: A novel fuzzy reinforced learning strategy in vector quantization. IEEE International Conference on Fuzzy Systems, pp. 1306–1310 (2008)
4. Karayiannis, N.B.: An axiomatic approach to soft learning vector quantization and clustering. IEEE Trans. Neural Netw. **10**(5), 1153–1165 (1999)
5. Karayiannis, N.B., Bezdek, J.C.: An integrated approach to fuzzy learning vector quantization and fuzzy c-means clustering. IEEE Trans. Fuzzy Syst. **5**(4), 622–628 (1997)
6. Karayiannis, N.B., Pai, P.I.: Fuzzy vector quantization algorithms and their application in image compression. IEEE Trans. Image Process. **4**(9), 1193–1201 (1995)
7. Kong, X., Wang, R., Li, G.: Fuzzy clustering algorithms based on resolution and their application in image compression. Pattern Recogn. **35**, 2439–2444 (2002)
8. Lai, J.Z.C., Liaw, Y.-C.: Fast-searching algorithm for vector quantization using projection and triangular inequality. IEEE Trans. Image Process. **13**(12), 1554–2004 (2004)
9. Lee, D., Baek, S., Sung, K.: Modified k-means algorithm for vector quantizer design. IEEE Signal Process. Lett. **4**(1), 2–4 (1997)
10. Li, R.Y., Kim, J., Shamakhi, N.A.: Image compression using transformed vector quantization. Image Vis. Comput. **20**, 37–45 (2002)
11. Linde, Y., Buzo, A., Gray, R.M.: An algorithm for vector quantizer design. IEEE Trans. Commun. **28**(1), 84–95 (1980)
12. Patane, G., Russo, M.: The enhanced LBG. Neural Netw. **14**, 1219–1237 (2001)
13. Qian, S.-E.: Fast vector quantization algorithms based on nearest partition set search. IEEE Trans. Image Process. **15**(8), 2422–2430 (2006)
14. Shen, F., Hasegawa, O.: An adaptive incremental LBG for vector quantization. Neural Netw. **19**, 694–704 (2006)
15. Shen, G., Zeng, B., Liou, M.L.: Adaptive vector quantization with codebook updating based on locality and history. IEEE Trans. Image Process. **12**(3), 283–295 (2003)
16. Tsao, E.C.K., Bezdek, J.C., Pal, N.R.: Fuzzy Kohonen clustering networks. Pattern Recogn. **27**(5), 757–764 (1994)
17. Tsekouras, G.E.: A fuzzy vector quantization approach to image compression. Appli. Math. Comput. **167**, 539–560 (2005)

18. Tsekouras, G.E., Dartzentas, D., Drakoulaki, I., Niros, A.D.: Fast fuzzy vector quantization. In: Proceedings of IEEE International Conference on Fuzzy Systems, Barcelona, Spain (2010)
19. Tsekouras, G.E., Mamalis, A., Anagnostopoulos, C., Gavalas, D., Economou, D.: Improved batch fuzzy learning vector quantization for image compression. Info. Sci. **178**, 3895–3907 (2008)

Chapter 6
Layers Image Compression and Reconstruction by Fuzzy Transforms

Ferdinando Di Martino and Salvatore Sessa

Abstract Recently we proved that fuzzy transforms (F-transforms) are useful in coding/decoding images, showing that the resulting peak-signal-to-noise-ratio (PSNR) is better than the one obtained using fuzzy relation equations and comparable with that obtained using the JPEG method. Recently some authors have explored a new image compression/reconstruction technique: the range interval [0,1] is partitioned in a finite number of subintervals of equal width in such a way that each subinterval corresponds to a image-layer of pixels. Each image-layer is coded using the direct F-transform, and afterwards all the inverse F-transforms are put together to reconstruct the whole initial image. We modify slightly this process: the pixels of the original image are normalized [15] with respect to the length of the gray scale, and thus are seen as a fuzzy matrix R, which we divide into (possibly square) submatrices R_B, called blocks. Hence we divide [0,1] into subintervals by adopting the quantile method, so that each subinterval contains the same number of normalized pixels of every block R_B, then we apply the F-transforms to each block-layer. In terms of quality of the reconstructed image, our method is better than that one based on the standard F-transforms.

6.1 Introduction

A direct fuzzy transform (F-transform) [20, 23, 24, 32] is an operator which transforms a continuous function into a n-dimensional vector. An inverse F-transform converts an n-dimensional vector into a continuous function which approximates the

F. Di Martino (✉) · S. Sessa
Dipartimento di Costruzioni e Metodi Matematici in Architettura,
Università degli Studi di Napoli Federico II, Via Monteoliveto 3,
80134 Napoli, Italy
e-mail: fdimarti@unina.it

S. Sessa
e-mail: sessa@unina.it

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_6, © Springer-Verlag Berlin Heidelberg 2013

original function up to a small arbitrary quantity ε. The F-transforms have been used in literature, especially in data analysis [8, 10, 22, 25–27, 33] and image analysis [5–7, 9, 11, 22, 24, 28–31]. The F-transforms are also useful for coding/decoding images; indeed, we have obtained the best results with respect to fuzzy relation equations [1, 2, 4, 12–14, 16–21] and JPEG compression method [34].

In [30], the authors investigate properties like the monotonicity and Lipschitz conditions of functions, which are invariant with respect to the F-transforms, and moreover, they propose a new technique of coding/decoding images. The authors find a partition of an image R of sizes $N \times M$ in image-layers having the same sizes. That is the interval [0,1], seen as range of the normalized pixel values of R, is partitioned in L closed subintervals with the same width $1/L$, and every image-layer has its pixel values belonging to the corresponding subinterval. Each image-layer is compressed using the direct F-transform; the reconstructed image is obtained combining the inverse F-transforms, resulting in the decoding process of all image-layers. We propose a slight modification of the method of [30], and we show that our method, called Layers Fuzzy Transforms (abbreviated as LF-transforms), gives better results than those obtained with the standard F-transforms in terms of PSNR of the reconstructed image. We point out that in this comparison the total compression rate is given from the sum of the compression rates of all image-layers; indeed, if the original image is partitioned in L layers and ρ is the compression rate of each layer, then $L \cdot \rho$ is the total compression rate. Thus the comparison of the reconstructed images by using the standard F-transforms and LF-transforms is made by using several compression rates. As we already discussed in our previous papers [1–11], any original image is seen as a fuzzy matrix R and is divided into submatrices called blocks R_B of sizes $N(B) \times M(B)$. Moreover, in our method every block is partitioned in layers, called block-layers. We partition the interval [0,1] with the quantile method, so the same number of normalized pixel values of R_B belongs to each subinterval. Indeed, in the equal interval method (which is used in [30]), we could have blocks in which all the normalized pixel values belong to only one image-layer. For example, we consider the following image of sizes 4×4 (any pixel assumes values between 0 and 255):

$$\begin{pmatrix} 165 & 165 & 165 & 166 \\ 165 & 165 & 165 & 166 \\ 165 & 165 & 165 & 166 \\ 165 & 165 & 163 & 162 \end{pmatrix}$$

which is normalized in [0,1], and we get the following fuzzy relation (for simplicity, assume that it is already a block):

$$R_B = \begin{pmatrix} 0.647059 & 0.647059 & 0.647059 & 0.650980 \\ 0.647059 & 0.647059 & 0.647059 & 0.650980 \\ 0.647059 & 0.647059 & 0.647059 & 0.650980 \\ 0.647059 & 0.647059 & 0.639216 & 0.635294 \end{pmatrix}$$

Then we divide [0,1] into two subintervals [0,0.5] and [0.5,1], thus we decompose the normalized image R_B in the following two block-layers:

$$R_B^1 = \begin{pmatrix} 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ 0.000000 & 0.000000 & 0.000000 & 0.000000 \end{pmatrix}$$

and $R_B^1 = R_B$. Then the contribution of R_B^1 in the compression process is null. Thus, if we compress the two block-layers with a compression rate greater than that used over R_B^2, we obtain a bad quality reconstructed image with respect to one obtained coding/decoding the block R_B using the F-transforms. For this reason, we prefer to determine the domain of the block-layers partitioning [0,1] with the quantile method, so that [0,1] is divided into ℓ subintervals such that the same number of pixels belongs to each subinterval. The pixel values are disposed in increasing order; then the first $(N(B) \times M(B))/\ell$ pixels are associated with the first block-layer. The upper bound of each subinterval is given from the value of the greatest pixel, and this value is the lower bound of the successive interval. In our example [0,1] is partitioned in the two intervals [0, 0.647059] and [0.647059,1]. Each interval contains eight pixels, and the two block-layers are the following:

$$R_B^1 = \begin{pmatrix} 0.647059 & 0.647059 & 0.647059 & 0.000000 \\ 0.647059 & 0.647059 & 0.647059 & 0.000000 \\ 0.000000 & 0.000000 & 0.000000 & 0.000000 \\ 0.000000 & 0.000000 & 0.639216 & 0.635294 \end{pmatrix}$$

$$R_B^2 = \begin{pmatrix} 0.647059 & 0.647059 & 0.647059 & 0.650980 \\ 0.647059 & 0.647059 & 0.647059 & 0.650980 \\ 0.647059 & 0.647059 & 0.647059 & 0.650980 \\ 0.647059 & 0.647059 & 0.647059 & 0.647059 \end{pmatrix}$$

In our experiments we compare the results obtained with the standard F-transforms and the LF-transforms for many compression rates by using a partition of the blocks into two block-layers. This paper is organized as follows: in Sect. 6.2 we give the essential concepts and an approximation theorem concerning the F-transforms of a discrete function in two variables. We also show how the F-transforms work in the coding/decoding processes of gray images. In Sect. 6.3 we recall the main results from [30], and we present our method. In Sect. 6.4 we compare our results with those obtained using the standard F-transforms. Section 6.5 presents our conclusions.

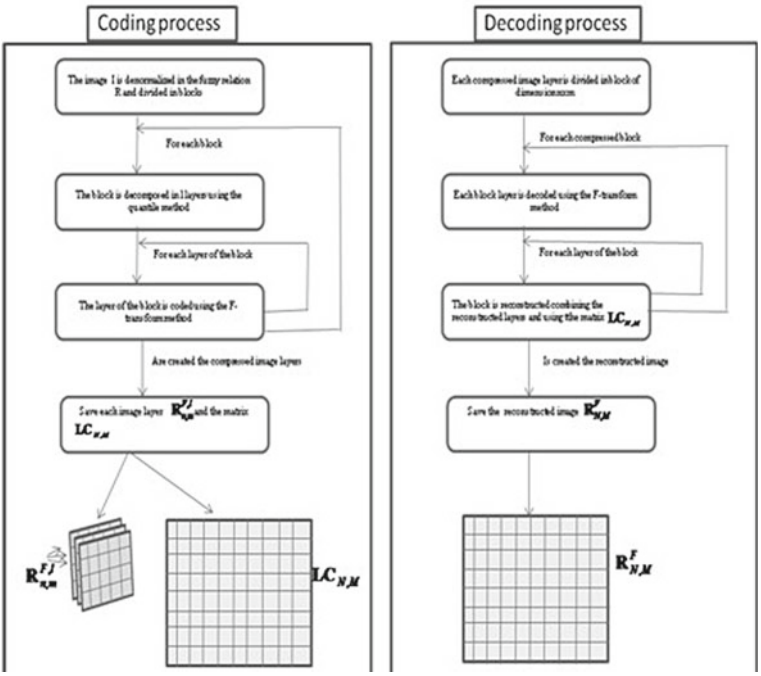


Fig. 6.1 Processes of coding/decoding images with LF-transforms

6.2 Discrete F-Transforms in Two Variables

Following the definitions and notations of [24], let $n \geq 2$ and $x_1, x_2, \ldots, x_n$ be points of $[a, b]$, called nodes, such that $x_1 = a < x_2 < \cdots < x_n = b$. We say that the fuzzy sets $A_1, \ldots, A_n : [a, b] \to [0, 1]$ form a fuzzy partition of $[a, b]$ if the following holds:

1. $A_i(x_i) = 1$ for every $i = 1, 2, \ldots, n$;
2. $A_i(x) = 0$ if $x \notin (x_{i-1}, x_{i+1})$ for $i = 2, \ldots, n-1$;
3. $A_i(x)$ is a continuous function on $[a, b]$;
4. $A_i(x)$ strictly increases on $[x_{i-1}, x_i]$ for $i = 2, \ldots, n$ and strictly decreases on $[x_i, x_{i+1}]$ for $i = 1, \ldots, n-1$;
5. $\sum_{i=1}^{n} A_i(x) = 1$ for every $x \in [a, b]$.
   $A_1, \ldots, A_n$ are called basic functions and they form "uniform fuzzy partition" if $n \geq 3$ and $x_i = a + h \cdot (i-1)$, where $h = (b-a)/(n-1)$ and $i = 1, 2, \ldots, n$ (that is, the nodes are equidistant);
6. $A_i(x_i - x) = A_i(x_i + x)$ for every $x \in [0, h]$ and $i = 2, \ldots, n-1$;
7. $A_{i+1}(x) = A_i(x - h)$ for every $x \in [x_i, x_i + 1]$ and $i = 1, 2, \ldots, n-1$.

Let $[a, b] \times [c, d]$ and $n, m \geq 2$, $x_1, x_2, \ldots, x_n \in [a, b]$ and $y_1, y_2, \ldots, y_m \in [c, d]$ be $n + m$ nodes such that $x_1 = a < x_2 < \ldots < x_n = b$ and $y_1 =$

$c < \ldots < y_m = d$. Furthermore, let $A_1, \ldots, A_n : [a, b] \to [0, 1]$ and $B_1, \ldots, B_m : [c, d] \to [0, 1]$ be fuzzy partitions of $[a, b]$ and $[c, d]$, respectively, $f : P \times Q \to$ reals be an assigned function, $P \times Q \subseteq [a, b] \times [c, d]$ where $P = \{p_1, \ldots, p_N\}$ and $Q = \{q_1, \ldots, q_M\}$ are "sufficiently dense" sets with respect to the chosen partitions. That is, for each $i = 1, \ldots, N$ (resp., $j = 1, \ldots, M$) there exists an index $k \in \{1, \ldots, n\}$ (resp., $l \in \{1, \ldots, m\}$) such that $A_k(p_i) > 0$ (resp. $B_l(q_j) > 0$). Then the matrix $[F_{kl}]$ is the (discrete) direct F-transform of f with respect to $\{A_1, \ldots, A_n\}$ and $\{B_1, \ldots, B_m\}$ if we have for each $k = 1, \ldots, n$ and $l = 1, \ldots, m$:

$$F_{kl} = \frac{\sum_{j=1}^{M} \sum_{i=1}^{N} f(p_i, q_j) A_k(p_i) B_l(q_j)}{\sum_{j=1}^{M} \sum_{i=1}^{N} A_k(p_i) B_l(q_j)} \tag{6.1}$$

We define the (discrete) inverse F-transform of f with respect to $\{A_1, A_2, \ldots, A_n\}$ and $\{B_1, \ldots, B_m\}$ to be the following function $f_{nm}^F(p_i, q_i) : P \times Q \to$ reals defined as

$$f_{nm}^F(p_i, q_j) = \sum_{k=1}^{n} \sum_{l=1}^{m} F_{kl} A_k(p_i) B_l(q_j) \tag{6.2}$$

The following approximation theorem holds [32]:

Theorem 6.1 *Let $f : P \times Q \to$ reals be an assigned function, $P \times Q \subseteq [a, b] \times [c, d]$, being $P = \{p_1, \ldots, p_N\}$ and $Q = \{q_1, \ldots, q_M\}$. Then for every $\varepsilon > 0$, there exist two integers $n(\varepsilon)$, $m(\varepsilon)$ and related fuzzy partitions $\{A_1, A_2, \ldots, A_{n(\varepsilon)}\}$ of $[a, b]$ and $\{B_1, B_2, \ldots, B_{m(\varepsilon)}\}$ of $[c, d]$ such that the sets of points P and Q are sufficiently dense with respect to such partitions and the inequality $|f(p_i, q_j) - f_{n(\varepsilon)m(\varepsilon)}^F(p_i, q_j)| < \varepsilon$ holds for every $i \in \{1, \ldots, N\}$ and $j \in \{1, \ldots, M\}$.*

Now we show how the F-transforms work for coding and decoding gray images. Let R be a gray image divided in $N \times M$ pixels interpreted as a fuzzy relation $R : (i, j) \in \{1, \ldots, N\} \times \{1, \ldots, M\} \to [0, 1]$, $R(i, j)$ being the normalized value of the pixel $P(i, j)$, that is, $R(i, j) = P(i, j)/255$ if the length of the gray scale, for instance, has 256 levels. In [6] the image R is compressed by using an F-transform in two variables $[F_{kl}]$ defined for each $k = 1, \ldots, n$ and $l = 1, \ldots, m$, as

$$F_{kl} = \frac{\sum_{j=1}^{M} \sum_{i=1}^{N} R(i, j) A_k(i) B_l(j)}{\sum_{j=1}^{M} \sum_{i=1}^{N} A_k(i) B_l(j)} \tag{6.3}$$

where we assume $p_i = i$, $q_j = j$, $a = c = 1$, $b = N$, $d = M$ and $A_1, \ldots, A_n$ (resp., $B_1, \ldots, B_m$) with $n \ll N$ (resp., $m \ll M$), form a fuzzy partition of $[1, N]$ (resp., $[1, M]$). By decoding with the inverse F-transform, we have the following fuzzy relation defined as

$$R_{nm}^F(i, j) = \sum_{k=1}^{n} \sum_{l=1}^{m} F_{kl} A_k(i) B_l(j) \tag{6.4}$$

Table 6.1 Various compression rates ($L = 2$)

F-transforms				LF-transforms				Compression rates		
Rows	Columns	Coded rows	Coded columns	Rows	Columns	Coded rows	Coded columns	ρ	$L\rho$	$L\rho_{tot}$
4	4	3	3	4	4	2	2	0.562	0.250	0.500
12	12	6	6	8	8	3	3	0.250	0.140	0.281
6	6	2	2	8	8	2	2	0.111	0.062	0.125
8	8	2	2	12	12	2	2	0.062	0.028	0.056
16	16	3	3	16	16	2	2	0.035	0.015	0.031

for every $(i, j) \in \{1, \ldots, N\} \times \{1, \ldots, M\}$. We have subdivided the image R of $N \times M$ pixels in submatrices R_B of sizes $N(B) \times M(B)$, called blocks (cf., e.g., [1, 2]), each compressed to a block F_B of sizes $n(B) \times m(B)(3 \leq n(B) < N(B), 3 \leq m(B) < M(B))$ via the direct F-transform $[F_{kl}^B]$ defined for each $k = 1, \ldots, n(B)$ and $l = 1, \ldots, m(B)$ as

$$F_{kl}^B = \frac{\sum_{j=1}^{M(B)} \sum_{i=1}^{N(B)} R_B(i, j) A_k(i) B_l(j)}{\sum_{j=1}^{M(B)} \sum_{i=1}^{N(B)} A_k(i) B_l(j)} \tag{6.5}$$

The following basic functions $A_1, \ldots, A_{n(B)}$ (resp., $B_1, \ldots, B_{m(B)}$) form a uniform fuzzy partition of $[1, N(B)]$ (resp., $[1, M(B)]$):

$$\begin{aligned} A_1(x) &= \begin{cases} 0.5(1 + \cos \frac{\pi}{h}(x - x_1)) & \text{if } x \in [x_1, x_2] \\ 0 & \text{otherwise} \end{cases} \\ A_l(x) &= \begin{cases} 0.5(1 + \cos \frac{\pi}{h}(x - x_k)) & \text{if } x \in [x_{k-1}, x_{k+1}] \\ 0 & \text{otherwise} \end{cases} \\ A_n(x) &= \begin{cases} 0.5(1 + \cos \frac{\pi}{h}(x - x_n)) & \text{if } x \in [x_{n-1}, x_n] \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{6.6}$$

where $n = n(B)$, $k = 2, \ldots, n$, $h = (N(B) - 1)/(n - 1)$, $x_k = 1 + h \cdot (k - 1)$ and

$$\begin{aligned} B_1(y) &= \begin{cases} 0.5(1 + \cos \frac{\pi}{s}(y - y_1)) & \text{if } y \in [y_1, y_2] \\ 0 & \text{otherwise} \end{cases} \\ B_t(y) &= \begin{cases} 0.5(1 + \cos \frac{\pi}{s}(y - y_t)) & \text{if } y \in [y_{t-1}, y_{t+1}] \\ 0 & \text{otherwise} \end{cases} \\ B_m(y) &= \begin{cases} 0.5(1 + \cos \frac{\pi}{s}(y - y_m)) & \text{if } y \in [y_{m-1}, y_m] \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{6.7}$$

where $m = m(B)$, $t = 2, \ldots, m$, $s = (M(B) - 1)/(m - 1)$, $y_k = 1 + s \cdot (t - 1)$.

Fig. 6.2 "Bird"

Fig. 6.3 "Bridge"

$$R^F_{n(B)m(B)}(i,j) = \sum_{k=1}^{n(B)} \sum_{l=1}^{m(B)} F^B_{kl} A_k(i) B_l(j) \tag{6.8}$$

which approximates R_B up to an arbitrary quantity ε in the sense of Theorem 6.1. Unfortunately, this theorem does not give a method for finding two integers $n(B)$ and $m(B)$ such that $|R_B(p_i, q_j) - R^F_{n(B)m(B)}(p_i, q_j)| < \varepsilon$, therefore we are obliged to prove several values of $n(B) = n(B, \varepsilon)$ and $m(B) = m(B, \varepsilon)$ (with $n(B) < N(B)$ and $m(B) < M(B)$) and hence to have various compression rates $\rho(B) = (n(B) \cdot m(B))/(N(B) \cdot M(B))$. For every compression rate, we evaluate the quality of the reconstructed image via the peak-signal-to-noise-ratio (PSNR) given by:

$$\text{PSNR} = 20 \log_{10} \frac{255}{\text{RMSE}} \tag{6.9}$$

where RMSE stands for Root Mean Square Error, and it is defined as:

$$\text{PSNR} = \sqrt{\frac{\sum_{i=1}^{N} \sum_{j=j}^{M} R^F_{NM}(i,j)^2}{N \times M}} \tag{6.10}$$

Here R^F_{NM} is the reconstructed image obtained by recomposing the $R_{n(B)m(B)}s$.

Fig. 6.4 "Camera"

Fig. 6.5 "Couple"

Fig. 6.6 "House"

Fig. 6.7 "Lena"

6.3 LF-Transform in Two Variables

In [30], the authors prefer to partition [0,1], seen as range of the normalized pixel values of R, in a set of L intervals of equal width, so that each interval includes the pixel of a layer. Each image-layer is coded with the direct F-transform and is decoded

Table 6.2 PSNR obtained in different compression rates for the image "Bird"

ρ	ρ in F-transforms	Total ρ in F-transforms	PSNR	LPSNR	% Gain
0.562	0.250	0.500	38.318	39.295	2.550
0.250	0.140	0.281	34.436	35.309	2.535
0.111	0.062	0.125	30.582	31.311	2.382
0.062	0.028	0.056	27.891	28.650	2.721
0.035	0.015	0.031	26.499	27.301	3.027

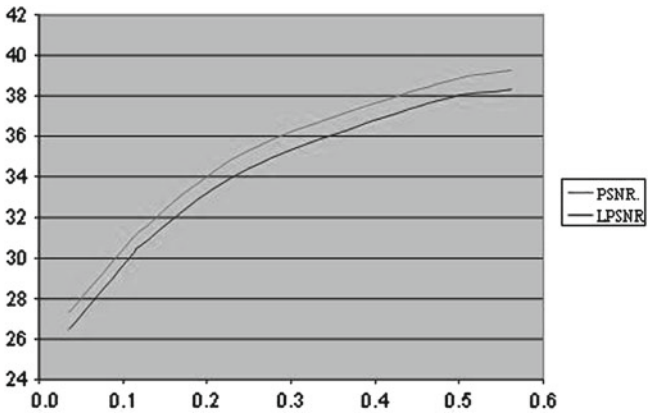


Fig. 6.8 Trend of the PSNR with the two methods for the image "Bird"

by using the inverse F-transforms. All the inverse F-transforms are combined to form the final reconstructed image. Below we report the steps of the method used in [30], based on the two layers [0,0.5], and [0.5,1].

1. [0, 1] is divided into two intervals, [0,0.5] and [0.5,1].
2. The fuzzy relation R is divided into two layers R^l, $l = 1, 2$, defined as

$$R^l(i, j) = \begin{cases} R(i, j) & \text{if } 1 - 0.5l \leq R(i, j) \leq 1 - 0.5(l - 1) \\ 1 - 0.5l & \text{otherwise} \end{cases}$$

3. Each layer R^l, $l = 1, 2$ is coded/decoded using the F-transforms.
4. The inverse F-transforms R^{1F}_{nm} and R^{2F}_{nm} are combined together into the function R^F_{nm}, which represents the reconstructed image of R.

In [30], the authors show with an example that the quality of the images reconstructed using the previous process is better than that obtained coding/decoding the original image with the standard F-transforms under the same compression rate. In terms of compression rate, we must consider that dividing the image in l layers, we store l compressed image-layers (in our case, $l = 2$).

Fig. 6.9 F-transforms, $\rho = 0.562$

Fig. 6.10 LF-transforms, $\rho = 0.500$

Fig. 6.11 F-transforms, $\rho = 0.035$

Indeed, we suppose to use a compression rate $\rho = 0.25$ and an original image of sizes 256×256 with 256 gray levels (hence 2^{16} pixels). Applying the F-transforms on the original image, then we obtain a compressed image of size 128×128 (2^{14} pixels), using the previous method with two layers, but we have stored 2^{15} pixels coming from two compressed layers of sizes 128×128. In other words we have used a final compression rate $\rho = 0.5$. Then a correct comparison is necessary between the classical F-transforms and the method of [30], in such a way that the spatial dimension of all the compressed layers is fully considered as well.

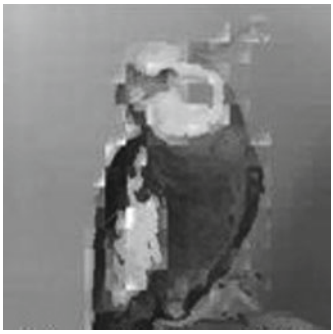


Fig. 6.12 LF-transforms, $\rho = 0.031$

Here we slightly modify the method of [30] comparing the quality of the images reconstructed with the standard F-transforms (resp. LF-transforms) under a specific (resp., similar total) compression rate ρ. The original image is divided into blocks; and each block is partitioned in 2 layers. We don't use the equal interval method to divide the [0,1] interval, but instead the quantile method so that each block-layer contains the same number of pixels. All the steps of our method are reported in what follows:

1. The pixels of the image of sizes $N \times M$ are normalized, and the resulting fuzzy relation is divided into square blocks of sizes $N(B) \times M(B)$.
2. For each block we create L layers by partitioning the set [0,1] with the quantile method. The pixels of each block are ordered in increasing sense; the lower Inf_B^l and upper Sup_B^l bound of each interval are determined, so that each layer contains the same number $N(B) \times M(B)/l$ of pixels. In the entry $M(i, j)$ of a matrix LC of sizes $N \times M$ is stored the index of the layer to which the pixel $R(i, j)$ belongs.
3. Each block-layer is coded using (6.5).
4. All the coded block-layers are stored.
5. Each block-layer is decoded using (6.8).
6. Each block is reconstructed using the formula:

$$R^F_{n(B)m(B)}(i, j) = \sum_{l=1}^{L} R^{*F,l}_{n(B)m(B)}(i, j) \tag{6.11}$$

where

$$R^{*F,l}_{n(B)m(B)}(i, j) = \begin{cases} R^{F,l}_{n(B)m(B)}(i, j) & \text{if } M(i, j) = l \\ R^{F,l}_{n(B)m(B)}(i, j) - \mathrm{Inf}_B^l & \text{otherwise} \end{cases}$$

7. Afterwards all blocks are recomposed and denormalized to give the reconstructed image.

Table 6.3 PSNR in different compression rates for the image "Bridge"

ρ	ρ in F-transforms	Total ρ in F-transforms	PSNR	LPSNR	% Gain
0.562	0.250	0.500	28.220	29.110	3.057
0.250	0.140	0.281	25.201	26.197	3.802
0.111	0.062	0.125	23.092	23.992	3.751
0.062	0.028	0.056	21.789	22.891	4.814
0.035	0.015	0.031	20.722	22.164	6.506

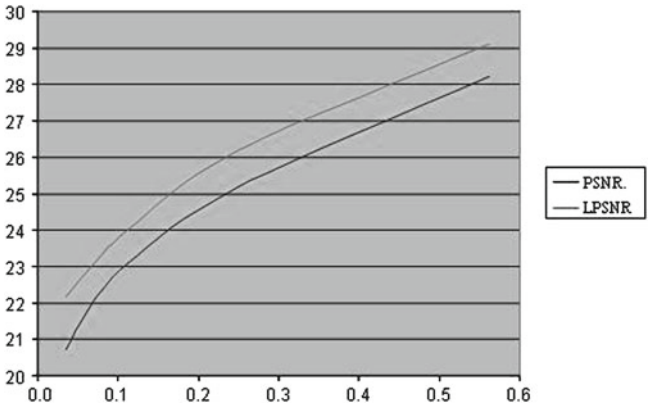


Fig. 6.13 Trend of the PSNR in the two methods for the image "Bridge"

The coding and decoding processes are schematized in Fig. 6.1.

For simplicity, we assume $M = N$, $M(B) = N(B)$, $m(B) = n(B)$ for every block B.

In Table 6.1 we report examples of compression parameters used comparing the two methods; the first (resp., second) four columns of Table 6.1 represent the sizes in rows and columns of the original block and the sizes of the compressed block by applying the standard F-transforms (resp., LF-transforms). In the last three columns of Table 6.1 we report the compression rate ρ used in the classical F-transforms, the compression rate L_ρ used on the single layer and the total compression rate $L_{\rho_{tot}}$ in LF-transforms.

In our experiments we have considered the image dataset extracted from the Image Database of the University of Southern California (http://sipi.usc.edu/database/) with $M = N = 256$ and 256 gray levels. For all the experiments we assume $L = 2$.

Fig. 6.14 F-transforms, $\rho = 0.562$

Fig. 6.15 LF-transforms, $\rho = 0.500$

6.4 Simulation Results

Here we present our results for six gray-level images "Bird" (Fig. 6.2), "Bridge" (Fig. 6.3), "Camera" (Fig. 6.4) "Couple" (Fig. 6.5), "House" (Fig. 6.6) and "Lena" (Fig. 6.7).

In Table 6.2 we report the PSNR obtained in the classical F-transforms and the PSNR related to the LF-transforms denoted with the acronym LPSNR for the image "Bird". Furthermore we indicate the percent of gain (denoted with the symbol "% Gain") parameter defined as

$$(\%\text{Gain LPSNR over PSNR}) = (\text{LPSNR} - \text{PSNR}) \cdot 100/\text{PSNR}$$

Figure 6.8 shows that LPSNR is always better than PSNR obtained using the classical F-transforms with a percent of gain between 0.25 and 0.3.

In Figs. 6.9, 6.10, 6.11 and 6.12 we show the reconstructed images with the F-transforms (resp., LF-transforms) with $\rho = 0.562$ (resp., 0.5) and $\rho = 0.035$ (resp., 0.031).

In Table 6.3 we report the results obtained with the source image "Bridge".

Figure 6.13 shows that LPSNR is always better than PSNR obtained using the classical F-transforms with a percent of gain between 0.33 and 0.65.

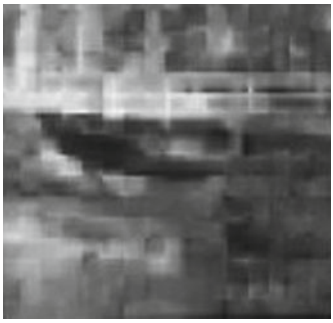


Fig. 6.16 F-transforms, $\rho = 0.035$

Fig. 6.17 LF-transforms, $\rho = 0.031$

Table 6.4 PSNR in different compression rates for the image "Camera"

ρ	ρ in F-transforms	Total ρ in F-transforms	PSNR	LPSNR	% Gain
0.035	0.015	0.031	20.627	22.466	8.919
0.062	0.028	0.056	21.848	22.911	4.865
0.111	0.062	0.125	23.080	24.006	4.009
0.250	0.140	0.281	25.427	26.429	3.940
0.562	0.250	0.050	28.488	29.498	3.543

In Figs. 6.14, 6.15, 6.16 and 6.17 we show the reconstructed images with the F-transforms (resp., LF-transforms) with $\rho = 0.562$ (resp., 0.5) and $\rho = 0.035$ (resp., 0.031).

In Table 6.4 we report the results obtained with the source image "Camera".

Figure 6.18 shows that LPSNR is always better than PSNR obtained using the classical F-transforms with a percent of gain between 0.35 and 0.89.

In Figs. 6.19, 6.20, 6.21 and 6.22 we show the reconstructed images with the F-transforms (resp., LF-transforms) with $\rho = 0.562$ (resp., 0.5) and $\rho = 0.035$ (resp., 0.031).

In Table 6.5 we report the results obtained with the source image "Couple".

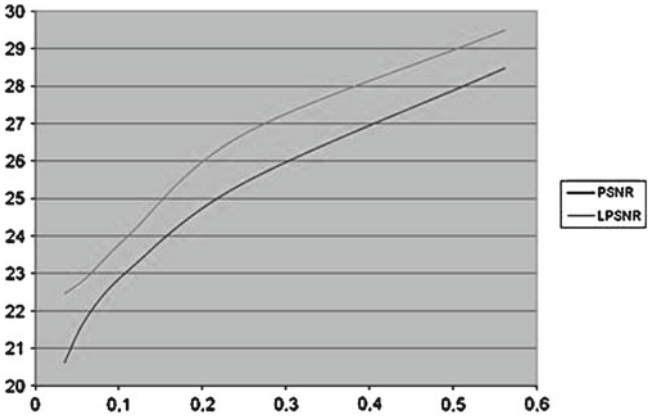


Fig. 6.18 Trend of the PSNR in the two methods for the image "Camera"

Fig. 6.19 F-transforms, $\rho = 0.562$

Fig. 6.20 LF-transforms, $\rho = 0.500$

Figure 6.23 shows that LPSNR is always better than PSNR obtained using the classical F-transforms with a percent of gain between 0.35 and 0.89.

Fig. 6.21 F-transforms, $\rho = 0.035$

Fig. 6.22 LF-transforms, $\rho = 0.031$

Table 6.5 PSNR in different compression rates for the image "Couple"

ρ	ρ in F-transforms	Total ρ in F-transforms	PSNR	LPSNR	% Gain
0.035	0.015	0.031	20.562	21.678	5.426
0.062	0.028	0.056	21.614	22.711	5.075
0.111	0.062	0.125	22.914	23.981	4.654
0.250	0.140	0.281	25.096	26.110	4.042
0.562	0.250	0.050	27.918	29.638	2.576

In Figs. 6.24, 6.25, 6.26 and 6.27 we show the reconstructed images with the F-transforms (resp., LF-transforms) with $\rho = 0.562$ (resp., 0.5) and $\rho = 0.035$ (resp., 0.031).

In Table 6.6 we report the results obtained with the source image "House".

Figure 6.28 shows that LPSNR is always better than PSNR obtained using the classical F-transforms with a percent of gain between 0.26 and 0.49.

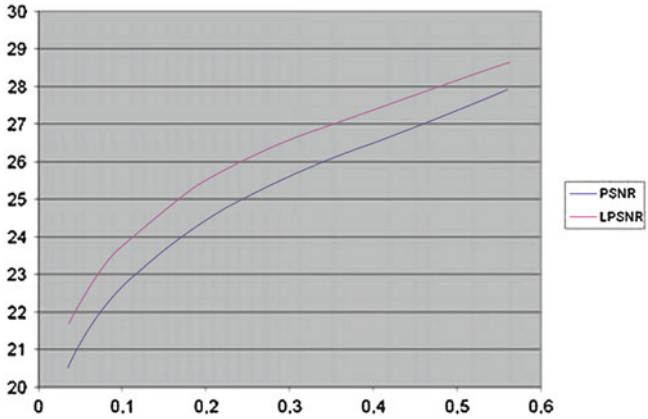


Fig. 6.23 Trend of the PSNR in the two methods for the image "Couple"

Fig. 6.24 F-transforms, $\rho = 0.562$

Fig. 6.25 LF-transforms, $\rho = 0.500$

In Figs. 6.29, 6.30, 6.31 and 6.32 we show the reconstructed images with the F-transforms (resp., LF-transforms) with $\rho = 0.562$ (resp., 0.5) and $\rho = 0.035$ (resp., 0.031).

In Table 6.7 we report the results obtained with the source image "Lena".

Fig. 6.26 F-transforms, $\rho = 0.035$

Fig. 6.27 LF-transforms, $\rho = 0.031$

Table 6.6 PSNR in different compression rates for the image "House"

ρ	ρ in F-transforms	Total ρ in F-transforms	PSNR	LPSNR	% Gain
0.035	0.015	0.031	20.562	21.578	4.940
0.062	0.028	0.056	21.363	22.271	4.250
0.111	0.062	0.125	22.914	23.781	3.781
0.250	0.140	0.281	25.094	26.010	3.649
0.562	0.250	0.050	27.918	28.648	2.576

Figure 6.33 shows that LPSNR is always better than PSNR obtained using the classical F-transforms with a percent of gain between 0.26 and 0.49.

In Figs. 6.34, 6.35, 6.36 and 6.37 we show the reconstructed images with the F-transforms (resp., LF-transforms) with $\rho = 0.562$ (resp., 0.5) and $\rho = 0.035$ (resp., 0.031).

All the results show that the percent of gain obtained with the LF-transforms with respect to the F-transforms is always greater than 0.015; this value increases by diminishing the compression rate. For strong compressions ($\rho = 0.03$), the

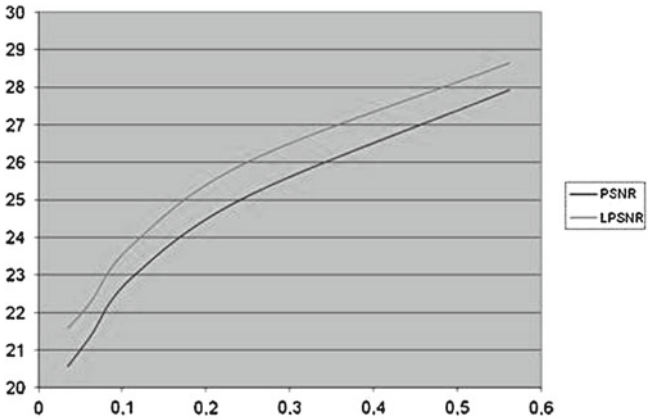


Fig. 6.28 Trend of the PSNR in the two methods for the image "House"

Fig. 6.29 F-transforms, $\rho = 0.562$

Fig. 6.30 LF-transforms, $\rho = 0.500$

reconstructed images obtained using the LF-transforms are not blurred, even if they contain spot pixels on the boundary of the blocks.

Fig. 6.31 F-transforms, $\rho = 0.035$

Fig. 6.32 LF-transforms, $\rho = 0.031$

Table 6.7 PSNR in different compression rates for the image "Lena"

ρ	ρ in F-transforms	Total ρ in F-transforms	PSNR	LPSNR	% Gain
0.035	0.015	0.031	22.790	23.616	3.623
0.062	0.028	0.056	23.891	24.686	3.335
0.111	0.062	0.125	25.477	26.261	3.082
0.250	0.140	0.281	28.103	28.784	2.424
0.562	0.250	0.500	31.142	31.692	1.766

Figure 6.38 shows the trend of the mean percent of gain with respect to the compression rate obtained for all images of the sample.

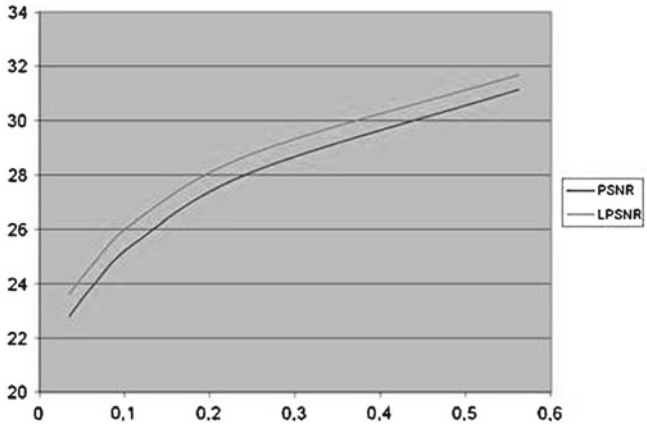


Fig. 6.33 Trend of the PSNR in the two methods for the image "Lena"

Fig. 6.34 F-transform, $\rho = 0.562$

Fig. 6.35 LF-transforms, $\rho = 0.500$

Fig. 6.36 F-transforms, $\rho = 0.035$

Fig. 6.37 LF-transforms, $\rho = 0.031$

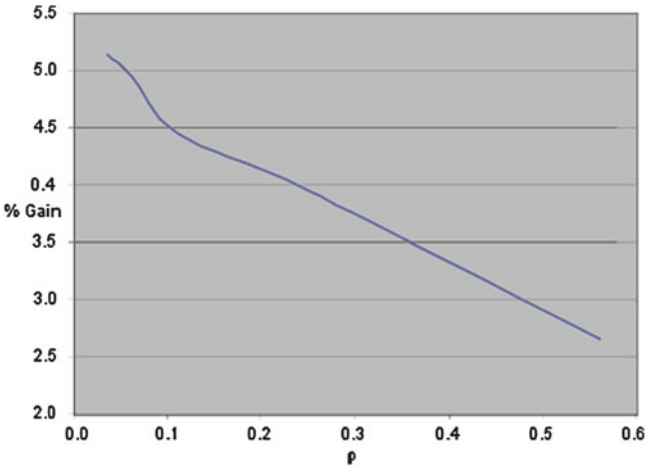


Fig. 6.38 Trend of the mean % gain with respect to ρ obtained for all the images

6.5 Conclusion

The results of our tests on the image dataset show that the images reconstructed by the LF-transforms are better than those ones obtained with the standard F-transforms, and moreover they are not blurred for low compression rates. This fact allows one to use LF-transforms in processes in which it is necessary to code images and videos with strong compression (e.g., WEB and video conferences, video-surveillance applications), without losing precious information present in the images. In other processes, like image segmentation [9] and image video compression [11, 17], the users need strongly coded images for their application; in these cases the LF-transforms can also be used since a further reduction of the storage space is possible.

References

1. Di Martino, F., Loia, V., Sessa, S.: A method for coding/decoding images by using fuzzy relation equations. In: Fuzzy Sets and Systems (IFSA 2003). Lecture Notes in Artificial Intelligence, vol. 2715, pp. 436–441. Springer, Berlin (2003)
2. Di Martino, F., Loia, V., Sessa, S.: A method in the compression/decompression of images using fuzzy equations and fuzzy similarities. In: Proceedings of the 10th IFSA World Congress, Istanbul, pp. 524–527 (2003)
3. Di Martino, F., Nobuhara, H., Sessa, S.: Eigen fuzzy sets and image information retrieval. In: Proceedings of the International Conference on Fuzzy Information Systems, Budapest, vol. 3, pp. 1385–1390 (2004)
4. Di Martino, F., Sessa, S.: Digital watermarking in coding/decoding processes with fuzzy relation equations. Soft Comput. **10**, 238–243 (2006)
5. Di Martino, F., Sessa, S.: Compression and decompression of images with discrete fuzzy transforms. Inf. Sci. **177**, 2349–2362 (2007)
6. Di Martino, F., Loia, V., Perfilieva, I., Sessa, S.: An image coding/decoding method based on direct and inverse fuzzy transforms. Int. J. Approx. Reason. **48**(1), 110–131 (2008)
7. Di Martino, F., Loia, V., Sessa, S.: Direct and inverse fuzzy transforms for coding/ decoding color images in YUV space. J. Uncertain Syst. **2**(1), 11–30 (2009)
8. Di Martino, F., Loia, V., Sessa, S.: Multidimensional fuzzy transforms for attribute dependencies. In: Proceedings of IFSA/EUSFLAT 2009, Lisbon, pp. 53–57 (2009)
9. Di Martino, F., Loia, V., Sessa, S.: A segmentation method for images compressed by fuzzy transforms. Fuzzy Sets Syst. **161**, 56–74 (2010)
10. Di Martino, F., Loia, V., Sessa, S.: Fuzzy transform method and attribute dependency in data analysis. Inf. Sci. **180**, 493–505 (2010)
11. Di Martino, F., Loia, V., Sessa, S.: Fuzzy transforms for compression and decompression of color videos. Inf. Sci. **180**, 3914–3931 (2010)
12. Di Nola, A., Pedrycz, W., Sanchez, E., Sessa, S.: Fuzzy Relation Equations and Their Applications to Knowledge Engineering. Kluwer Academic Publishers, Dordrecht (1989)
13. Gronscurth, H.M.: Fuzzy data compression for energy optimization models. Energy **23**(1), 1–9 (1998)
14. Hirota, K., Pedrycz, W.: Data compression with fuzzy relational equations. Fuzzy Sets Syst. **126**, 325–335 (2002)
15. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Kluwer Academic Publishers, Dordrecht (2000)

16. Loia, V., Pedrycz, W., Sessa, S.: Fuzzy relation calculus in the compression and decompression of fuzzy relations. Int. J. Image Graph. **2**, 1–15 (2002)
17. Loia, V., Sessa, S.: Fuzzy relation equations for coding/decoding processes of images and videos. Inf. Sci. **171**, 145–172 (2005)
18. Nobuhara, H., Pedrycz, W., Hirota, K.: Fast solving method of fuzzy relational equation and its application to lossy image compression. IEEE Trans. Fuzzy Syst. **8**(3), 325–334 (2000)
19. Nobuhara, H., Pedrycz, W., Hirota, K.: Relational image compression: optimizations through the design of fuzzy coders and YUV color space. Soft Comput. **9**(6), 471–479 (2005)
20. Nobuhara, H., Hirota, K., Di Martino, F., Pedrycz, W., Sessa, S.: Fuzzy relation equations for compression/decompression processes of colour images in the RGB and YUV colour spaces. Fuzzy Optim. Decis. Mak. **4**(3), 235–246 (2005)
21. Nobuhara, H., Hirota, K., Pedrycz, W., Sessa, W.: A motion compression/ reconstruction method based on max t-norm composite fuzzy relational equations. Inf. Sci. **176**, 2526–2552 (2006)
22. Novak, V., Perfilieva, I.: Fuzzy transform in the analysis of data. Int. J. Approx. Reason. **48**(1), 36–46 (2008)
23. Perfilieva, I.: Fuzzy transforms: application to reef growth problem. In: Demicco, R.B., Klir, G.J. (eds.) Fuzzy Logic in Geology, pp. 275–300. Academic Press, Amsterdam (2003)
24. Perfilieva, I.: Fuzzy transforms. Fuzzy Sets Syst. **157**, 993–1023 (2006)
25. Perfilieva, I., Chaldeeva, E.: Fuzzy transformation. In: Proceedings of the 9th IFSA, World Congress and 20th NAFIPS International Conference, pp. 1946–1948. Vancouver (2001)
26. Perfilieva, I., Novak, V., Pavliska, V., Dvork, A., Stepnicka, M.: Analysis and prediction of time series using fuzzy transform. In: Proceedings World Congress on Computational Intelligence/FUZZ-IEEE, pp. 3875–3879. Hong Kong (2008)
27. Perfilieva, I., Valek, R.: Fuzzy approach to data compression. In: Proceedings 8th Czech-Japan Seminar on Data Analysis and Decision Making under Uncertainty, pp. 91–100. Praga (2005)
28. Perfilieva, I., Pavliska, V., Vajgl, M., De Baets, B.: Advanced image compression on the basis of fuzzy transforms. In: Proceedings of IPMU, pp. 1167–1174. Malaga (2008)
29. Perfilieva, I.: Fuzzy transforms and their applications to image compression. In: Bloch, I., Petrosino, A.,Tettamanzi, A. (eds.) Fuzzy Logic and Applications. LNAI, vol. 3849, pp. 19–31. Springer, Heidelberg (2006)
30. Perfilieva, I., De Baets, B.: Fuzzy transforms of monotone functions with application to image compression. Inf. Sci. **180**, 3304–3315 (2010)
31. Perfilieva, I.: Fuzzy transform in image compression and fusion. Acta Mathematica Universitatis Ostraviensis **15**, 27–37 (2007)
32. Stepnicka, M., Valasek, R.: Fuzzy transforms and their application to wave equation. J. Electr. Eng. **55**(12), 7–10 (2004)
33. Stepnicka, M., Pavliska, V., Novak, V., Perfilieva, I.: Time series analysis and prediction based on fuzzy rules and the fuzzy transform. In: Proceedings of IFSA/EUS- FLAT, pp. 1601–1605. Lisbon (2009)
34. The International Telegraph And Telephone Consultative Committee. Information Technology—Digital Compression And Coding of Continuous-Tone Still Images-Requirements and Guidelines, Recommendation T81 (1992)

Chapter 7
Modified Bacterial Foraging Optimization Technique for Vector Quantization-Based Image Compression

Nandita Sanyal, Amitava Chatterjee and Sugata Munshi

Abstract Vector quantization (VQ) techniques are well-known methodologies that have attracted the attention of research communities all over the world to provide solutions for image compression problems. Generation of a near optimal codebook that can simultaneously achieve a very high compression ratio and yet maintain required quality in the reconstructed image (by achieving a high peak-signal-to-noise-ratio (PSNR)), to provide high fidelity, poses a real research challenge. This chapter demonstrates how such efficient VQ schemes can be developed where the near optimal codebooks can be designed by employing a contemporary stochastic optimization technique, namely bacterial foraging optimization (BFO), that mimics the foraging behavior of a common type of bacteria, *Escherichia coli*, popularly known as *E. coli*. An improved methodology is proposed here, over the basic BFO scheme, to perform the chemotaxis procedure within the BFO algorithm in a more efficient manner, which is utilized to solve this image compression problem. The codebook design procedure has been implemented using a fuzzy membership-based method, and the optimization procedure attempts to determine suitable free parameters of these fuzzy sets. The usefulness of the proposed adaptive BFO algorithm, along with the basic BFO algorithm, has been demonstrated by implementing them for a number of benchmark images, and their performances have been compared with other contemporary methods, used to solve similar problems.

N. Sanyal (✉)
Department of Electrical Engineering, B. P. Poddar Institute of Management and Technology, 137 VIP Road, Kolkata 700052, India
e-mail: nandita.juee93@gmail.com

A. Chatterjee · S. Munshi
Department of Electrical Engineering, Jadavpur University, Kolkata 700032 India
e-mail: cha_ami@yahoo.co.in

S. Munshi
e-mail: sugatamunshi@yahoo.com

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_7, © Springer-Verlag Berlin Heidelberg 2013

7.1 Introduction

With the advent of the Internet, teleconferencing, multimedia, high-definition television technology, etc., the amount of data (information) handled by a computer is getting increased enormously day by day. Storage and transmission of digital images has become one of the most challenging problems for the researchers working in this domain. Image compression is an efficient way by which an image can be transmitted and stored in a compact manner [1, 2]. It basically reduces the amount of data handled in an image. During transmission, a reduced volume of information is transmitted that represents a large volume of information pertaining to the original image. After transmission, the compressed image is reconstructed to represent the original [1]. Broadly speaking, there are two types of image compression schemes: lossless compression and lossy compression. In medical imaging, lossless compression scheme is mostly necessary, but in multimedia applications, the lossy image compression scheme is usually preferred. For lossless compression, the compression ratio achievable is less than that in lossy image compression. Along with the objective of obtaining the highest compression rate, as far as possible, an efficient image compression skill is also desired to ensure a better quality of compressed image. This means that the reconstructed image should possess higher fidelity [1, 3].

Nowadays, vector quantization (VQ) finds popular application in image processing [4, 5]. For the past few years, VQ techniques have been widely used as powerful data compression algorithms. A vector quantizer essentially generates a codebook to represent an image, and this codebook is generated using training data sets. Hence the generation of this codebook plays a very significant role in vector quantization. A VQ scheme performs two specific tasks, essentially known as encoding and decoding. Usually, an image to be encoded is first sectionalized into a set of image blocks or vectors, which are nonoverlapping in nature [5]. Each image vector, thus formed, is then compared with each codebook vector, which is also called a codeword, to find out that image vector-codebook vector combination for which distortion is minimum. During transmission, the codebook is transmitted along with a series of index numbers, where each index number represents the index of the codebook vector that should represent an image vector in the reconstructed image at the receiving end. This series of index numbers, along with the codebook, received by the receiver, is then decoded to form the image blocks, and the image is thus reconstructed. Normally the codebook size is much smaller than the size of the original image data set. The traditional codebook vector quantizer is generated by Linde–Buzo–Gray (LBG) algorithm, where one codebook contains M number of randomly initialized vectors that undergo modification using an iterative procedure [6, 7]. It is very simple to implement, but its performance primarily depends on the initialization of the codebook. Traditional LBG algorithms use Euclidean norm-based formulation of the objective function and, in the quest to determine the optimal codebook, there is a chance that the solution may get trapped in local minima of the average distortion measure. The LBG algorithm suffers from the deficiency that it is a hard decision-making scheme and ignores the possibility of a training vector belonging

to more than one codebook vector. In the LBG scheme, it is also well known that the computation volume is much higher. Since the advent of fuzzy inferencing system, a soft methodology has been developed to determine the similarity between two data sets [8]. This concept of fuzzy inferencing is also used for designing VQ algorithms [9] in many image processing applications. Fuzzy clustering techniques have been efficiently used in vector quantization [10–12]. This method has been proven to be successful in modeling the uncertainty involved in training datasets and also reducing the constraint of proper initialization [13]. The fuzzy method of VQ is a soft decision-making scheme where each training vector can belong to multiple clusters. Here the aim of the researchers is always focused on two aspects: first, to design a codebook that gives a globally optimal solution and second, to reduce the computational complexity involved [14]. In most cases, to obtain an optimum codebook, there is a need for a fuzzy to crisp mode transition. Many efficient techniques for smooth transition strategies from fuzzy modes to crisp modes have been already studied. Fuzzy vector quantization (FVQ) [11], Fuzzy learning vector quantization (FLVQ) [15], Fuzzy particle swarm optimized vector quantization (FPSOVQ) [9], improved batch fuzzy learning vector quantization [16] are several such special strategies developed so far, where the fuzziness parameter is reduced to small values and each training vector is assigned, finally, to a definite codebook vector. A Gaussian-type fuzzy membership function has been mostly successfully used as a soft estimator for determining the similarity between the codebook vectors and the image patterns [9] to obtain the maximum fidelity. The design of such codebook vectors, utilizing a fuzzy-based soft scheme, can be developed using a stochastic optimization-based scheme, where an optimal codebook can be generated as a global solution based on minimum average distortion measure. Such derivative-free optimization methods are usually based on a population of candidate solutions that undergo modification in an iterative manner, and the final solution is expected to reach the global optimum, thus avoiding the possibility of getting stuck in local optima. Among the stochastic optimization techniques, simulated annealing (SA) was one of the first strategies employed for optimum codebook design in image compression problems [14]. However, the performance of SA largely depended on the selection of its parameters. Later, genetic algorithm (GA) and particle swarm optimization (PSO) based methods were also proposed for optimal codebook design problems [9, 17, 18].

The present work proposes the development of a new fuzzy VQ scheme employing a bacterial foraging-based stochastic optimization strategy. The bacterial foraging optimization algorithm (BFOA), which was proposed in the last decade [19], appeared as a major breakthrough in the area of stochastic optimization. Researchers have so far successfully used it in many global stochastic optimization problems, e.g. in harmonic estimation [20], transmission loss reduction [21], active power filter design for load compensation [22], etc. In the light of the foraging behavior of *E. coli* bacteria present in the human gut, a bacterial foraging algorithm was formulated where energy intake by a group of bacteria per unit time is maximized [19, 23]. Due to the poor convergence behavior of BFOA observed in some complex engineering problems, self-adaptation schemes were proposed to improve the convergence behavior of classical BFOA [24–26]. These self-adaptive BFOA schemes

employed several strategies to adapt the concerned parameter that controls the movement of each bacterium. The present work proposes a new variant of adaptive BFOA that controls the run length unit of the BFOA, but in a different manner. This self-adaptive BFOA and the classical BFOA have been employed for fuzzy VQ algorithms in image compression for a number of benchmark images. Their performance has been compared to several other competing algorithms, and it has been satisfactorily demonstrated that the BFOAs show overall superior performance and self-adaptive BFOA can further improve the performance of classical BFOA. In the subsequent sections, first fuzzy-based VQ scheme is detailed. Then classical bacterial foraging algorithm is described in Sect. 7.3. In Sect. 7.4, the adaptive bacterial foraging algorithm (ABFOA) proposed in this work is formulated. Section 7.5 presents the simulation results, and the chapter is concluded in Sect. 7.6.

7.2 Fuzzy Vector Quantization for Image Compression

In the whole image processing domain, VQ possibly finds its application mostly in image compression problems [13, 15, 27–29]. One of the main reasons for that is the implementation speed of the VQ schemes, which is quite fast. Here, first an image is decomposed into a number of rectangular blocks, where each block constitutes a training vector in the p-dimensional space where

$$p = n_r * n_c \tag{7.1}$$

where n_r = number of pixels along a row of the image block, and n_c = number of pixels along a column of the image block. Hence, one can write

$$X = \{x_1, x_2, \ldots, x_n\} \subset \Re^p \tag{7.2}$$

where X gives the set of all training vectors in which the image is divided, and x_i is the ith such training vector. In VQ, this set is represented by a smaller set of vectors denoted as

$$V = \{v_1, v_2, \ldots, v_c\} \subset \Re^p \tag{7.3}$$

where every v_j $(1 \leq j \leq c)$ is called a codebook vector or codeword, and the set V is known as the codebook [16]. A vector quantizer is designed by assigning each training vector to a suitable codebook vector by minimizing some measure of discrepancy between training vectors and codebook vectors. The discrepancy that is to be minimized in image compression problems is the average distortion measure given as [16]:

$$D = \frac{1}{n} \sum_{k=1}^{n} \min_{1 \leq j \leq c} \{ \| x_k - v_j \|^2 \} \tag{7.4}$$

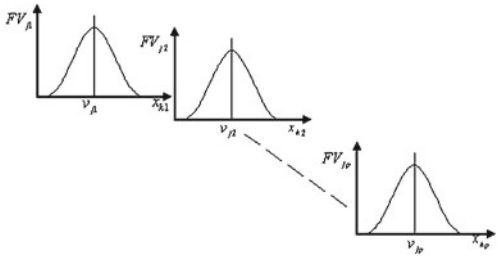


Fig. 7.1 Fuzzy sets or membership functions chosen

This method is a crisp decision-making scheme and does not take into account the possibility that one training vector may have resemblance with two or more codebook vectors. Fuzzy set theory can easily address the issue of the degree of such resemblance [8, 30]. As mentioned before, the LBG algorithm based vector quantization method is a popular iterative method by which an optimal codebook is designed. This uses a similar Euclidean norm-type measure and minimizes average distortion. However, in fuzzy inference analysis, a soft and flexible decision measure is considered. As mentioned earlier, each image is divided into a number of training vectors and a codebook V of size c is designed. A Gaussian-type fuzzy membership function is considered to process the fuzzy inferencing system [9]. Each training vector x_i and each codebook vector v_j is of dimension p. Hence the codebook vector v_j can be given as:

$$v_j = (v_{j_1}, v_{j_2}, \ldots, v_{j_p}); \qquad j \in \{1, 2, \ldots, c\} \tag{7.5}$$

A Gaussian membership function (MF) is created for each fuzzy set associated with the ith dimension of the jth codebook vector, i.e., v_{ji}. Then the degree of resemblance for representing the training data vector x_k in terms of codebook vector j is given by the activation of a fuzzy IF –THEN rule given as [9, 28] :

Rule j : IF (x_{k_1} is FV_{j_1}) and (x_{k_2} is FV_{j_2}) and …and (x_{k_p} is FV_{j_p}) THEN x_k belongs to v_j with

$$MV = MV_{(j_1, j_2, \ldots, jp)} \qquad j \in \{1, 2, \ldots c\} \tag{7.6}$$

Hence there is a total of c rules. Each training vector is given as $x_k = (x_{k1}, x_{k2}, \ldots, x_{kp})$, $k \in \{1, 2, \ldots, n\}$. FV_{ji} represents the fuzzy set associated with the ith dimension of the jth codebook vector v_j. $MV_{(j_1, j_2, \ldots, jp)}$ determines the activation strength or membership value of the fuzzy rule j. The higher this activation strength, the greater the confidence that can be placed on the codebook vector v_j, representing the training vector x_k. The fuzzy sets FV_{ji} associated with each v_j are shown in Fig. 7.1.

The membership function of the fuzzy set FV_{ji} for the jth fuzzy rule is given as

$$FV_{ji} = \exp\left[-\left(\frac{x_{ki} - v_{ji}}{\gamma}\right)^2\right] \tag{7.7}$$

where

$$\gamma = \sqrt{\frac{1}{N_t}\sum_{i=1}^{N_t}\|x_i - \bar{x}\|^2} \quad \text{and} \quad \overline{x} = \frac{1}{N_t}\sum_{i=1}^{N_t} x_i \tag{7.8}$$

Here N_t is the number of training vectors, and γ is the spread of each Gaussian MF. The maximum value of MV is obtained when the training image pattern x_k is closest to v_j in p-dimensions. When a particular training image pattern x_k^t is chosen, where $x_k^t = (x_{k_1}^t, x_{k_2}^t, \ldots, x_{k_p}^t)$, it is used as an input to each fuzzy rule j, and the degree of resemblance (DR) of x_k^t with different v_js is determined. This is given in terms of the activation strength $MV_{(j_1, j_2, \ldots, jp)}$, which is calculated as:

$$DR_j^t = \mu_j^t = FV_{j1}(x_{k1}^t) * FV_{j2}(x_{k2}^t) * \ldots * FV_{jp}(x_{kp}^t) \tag{7.9}$$

Hence for each x_k^t, $DR_j^t j = (1, 2, \ldots, c)$ is calculated. Then the training pattern $x_k^t$is assigned to that codebook vector q for which rule q produced the highest MV or DR_j^t. This can be represented as:

$$DR_q^t = \max(DR_1^t, DR_2^t, \ldots, DR_c^t) \tag{7.10}$$

In this way all the training image patterns are mapped by suitable codebook vectors, one by one. Then the performance of the coding operation is evaluated with the help of the fitness function:

$$\sum_{i=1}^{N_t}\sum_{j=1}^{c} S_{ij} \left\|x_i - v_j\right\|^2 \tag{7.11}$$

where

$$S_{ij} = \begin{cases} 1 & DR_j^i = \max(DR_1^i, DR_2^i, \ldots, DR_c^i) \\ 0 & \text{otherwise} \end{cases} \tag{7.12}$$

When the fitness function becomes smaller and smaller in value, it means that the actual error between the image patterns and the corresponding codebook vectors is getting lower and lower, i.e., the average distortion is becoming less and less in value. The physical interpretation of this mathematical operation is that, for the same compression ratio achieved, the reconstructed image represents the original image more and more faithfully. Therefore the design of a globally optimum codebook is thus reduced to a problem of minimization of average distortion measure. In this present work, the proposed fuzzy BFOA-based VQ (FBFOVQ) learning scheme is

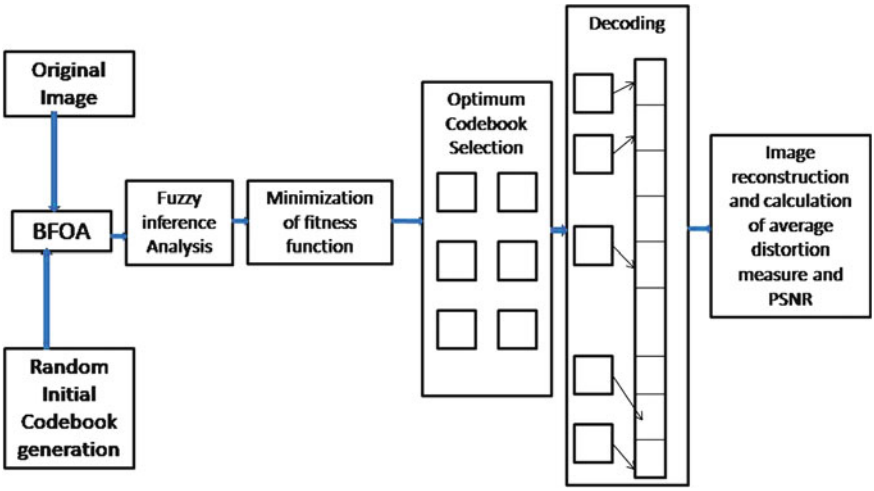


Fig. 7.2 Fuzzy bacterial foraging optimization vector quantization designed for image compression

utilized to determine the suitable codebook to represent the original training image dataset. Figure 7.2 shows the overall method in schematic form.

7.3 Bacterial Foraging Optimization Algorithm

In present-day research activities on stochastic optimization techniques, development of evolutionary principles based on real-life functioning of living organisms is a major emphasis. Amongst these techniques, the foraging behavior by which a living organism locates, handles and ingests food is found to be well understood. The foraging animal always behaves in such a way that energy intake $\frac{E}{T}$ per unit time is maximized. The animals with better foraging strategies always enjoy reproductive success, while animals with poor foraging behavior are always eliminated after many generations or are shaped into good ones. All these behaviors are carefully observed and a new stochastic optimization technique is formulated which is known as the bacterial foraging optimization algorithm (BFOA) [19, 23]. BFOA when formulated on the basis of biomimicing *E. coli* bacteria present in the human gut has the following steps: chemotaxis, swarming, reproduction and elimination-dispersal. Algorithm 7.1 shows the classical BFOA whose different steps are detailed now.

Chemotaxis. The *E. coli* bacteria move in two different ways: swim or run and tumble, with the help of flagella - actuators. In its whole life time, each bacterium alternates between swim and tumble. *E. coli* bacteria have an inherent decision-making system which always enables the bacteria to be able to search food and avoid noxious substances. This is known as chemotaxis. The bacteria are always in search of more neutral regions and try to move toward greater nutrient concentration.

Let us consider θ as the position of the bacteria in p-dimensional space, i.e.,$\theta \in \Re^p$ and $J(\theta)$ be the cost function. Then $J(\theta) \leq 0$ represents a nutrient-rich environment, $J(\theta) = 0$ represents neutral, and $J(\theta) > 0$ represents a noxious environment. The principal aim of the bacterium is to minimize the cost function. For the ith bacterium, the position after jth chemotactic step is given by:

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\phi(j) \tag{7.13}$$

where $\phi(j)$ is a unit vector in a random direction to describe tumble. $\theta^i(j,k,l)$ represents ith bacterium at jth chemotactic, kth reproductive and lth elimination dispersal step. $C(i)$ is the size of the step taken in the random direction termed as run length unit. If $J\left(\theta^i(j+1,k,l)\right)$ assumes a value lower than $J\left(\theta^i(j,k,l)\right)$, then the bacterium will take further steps in this same particular direction, up to a maximum number of permissible steps, called N_s.

Swarming. In presence of a semisolid medium with a single nutrient chemoeffecter (sensor), *E. coli* and other bacteria try to move out from the center in definite traveling rings of cells, by moving up the nutrient gradient created by consumption of the nutrient by the group of bacteria. The cells release an attractant aspertate when stimulated by high levels of succinate, signal other nearby cells to group with it and move in same pattern [19]. There may be some repelling action among cells, which is signaled by consuming nearby nutrients. The combined cell-to-cell interaction is given by:

$$\begin{aligned} Jcc(\theta,\theta^i(j,k,l)) = &\textstyle\sum_{i=1}^{s}\left[-d_{attract}\exp\left(-w_{attract}\sum_{m=1}^{p}\left(\theta_m-\theta_m^i\right)^2\right)\right] \\ &+\textstyle\sum_{i=1}^{s}\left[h_{repellant}\exp\left(-w_{repellant}\sum_{m=1}^{p}\left(\theta_m-\theta_m^i\right)^2\right)\right] \end{aligned} \tag{7.14}$$

where p is the dimension of search space, $J_{cc}\left(\theta,\theta^i(j,k,l)\right)$ is the cost function that is to be added to the original cost function, and $d_{attract}, w_{attract}, h_{repellant}$, $w_{repellant}$ are the coefficients which determine the depth and width of the attractant and the height and width of the repellant. These four parameters should be selected judiciously for a given problem. The total number of bacteria is S, $\theta_m^i$is the mth dimension of the position of the ith bacterium, θ^i. During the whole lifetime, each bacterium takes lots of such chemotactic steps, limited by N_c number of steps in each iteration.

Reproduction. After completing the chemotaxis step, the bacteria enter the reproduction state. Here the least healthy S_r bacteria die and are replaced by copies of S_r healthiest bacteria (those having sufficient nutrients and yielding lower values of fitness function). This means that each such healthy bacterium splits into two bacteria without mutation and are placed in the same location. Thus the total number of bacteria in a group remains same. If N_{re} number of specified reproduction steps has not been completed, then the next generation of bacteria will again undergo chemotaxis steps. Usually S_r is chosen as 50 % of the total number of bacteria.

$$S_r = \frac{S}{2} \tag{7.15}$$

Elimination and dispersal . During the lifetime of a swarm of bacteria with gradual consumption of food or nutrients, it may happen that all the bacteria die or disperse into some new environment with probability p_{ed} for some unknown reason. This dispersal destroys all the previous chemotactic processes. However, it may also have a positive impact because the bacteria may disperse into a nutrient-rich region, and the result of chemotaxis may improve in the next generation. Elimination-dispersal is part of the long-distance motile behavior in population levels. Hence this procedure may be employed when the optimization algorithm is not showing any satisfactory improvement in performance, perhaps due to a problem of stagnation.

7.4 Bacterial Foraging with Self-Adaptation

Though BFOA is applied to solve many engineering application problems, some complication arises with the increase in dimension of search space in more complex problems. The performance of BFOA largely depends on the run length unit parameter $C(i, j, k)$. Bacteria with smaller run length units have a tendency to get trapped in local optima in the search domain, whereas a comparatively larger run length unit potentially can equip the bacteria to perform global searches better [31]. Therefore, if we can adaptively change the run length unit parameter, depending on some preset condition, the performance of the bacteria can be improved [24–26]. In the proposed adaptive BFOA (termed as ABFOA), shown in Algorithm 7.2, the entire chemotaxis step can be divided into two states: exploration and exploitation. With larger run length unit parameters, bacteria will be in the exploration state and can find out prospective new search domains. Whereas, in the exploitation state, bacteria can search in nearby regions slowly, with smaller run length unit. Bacteria should always try and maintain a balance between exploration and exploitation strategies by observing two basic factors: fitness improvement, and no fitness improvement. If the bacterium registered a fitness improvement beyond a specified precision from the last chemotactic generation to the current, one can conclude that it has found a new promising region. Hence the bacterium should perform more extensive exploitation and the adapted run length unit for this bacterium should be smaller, compared to the previous chemotactic generation. Hence, following this, the bacteria will self-adapt into an exploitation state [26]. If no fitness improvement occurs for a predefined number of consecutive chemotactic generations, the augmentation of run length unit becomes a demand and this bacterium enters an exploration state to extend its search from an unproductive region to potentially relatively promising regions. Finally, as the ABFOA evolves, each bacterium alternates between two distinct search states and run length unit for the ith bacterium will be modified as required. In the ABFOA proposed in [26], the reproduction step and the elimination and dispersal step are included within each chemotactic step. However, in our proposed ABFOA, this part

Algorithm 7.1 The algorithm representation of classical BFOA [19, 32, 33]

Begin
Initialize all the free parameters of classical BFOA, $C(i)$, $i = 1, 2, \ldots, S$
Set all the counters to zero.
repeat
  for Elimination-dispersal: ell= 1 to N_{ed} **do**
    for Reproduction: k = 1 to N_{re} **do**
      for Chemotaxis: j = 1 to N_c **do**
        for each bacterium: i = 1 to S **do**
          Compute fitness function, $J\ (i, j, k, ell)$
          $J\ (i, j, k, ell) = J\ (i, j, k, ell) + J_{cc}\left(\theta, \theta^i\ (j, k, ell)\right)$
          $J_{last} = J(i, j, k, ell)$
          Tumble: Generate a random vector $\Delta(i) \in \Re^p$
          Move: $\theta^i\ (j+1, k, ell) = \theta^i\ (j, k, ell) + C\ (i)\ \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$
          Compute fitness function, $J(i, j+1, k, ell)$
          $J\ (i, j+1, k, ell) = J\ (i, j+1, k, ell) + J_{cc}\left(\theta, \theta^i\ (j+1, k, ell)\right)$
          $m = 0$
          Swim:
          while m $< N_s$ **do**
            $m = m + 1$
            if $J\ (i, j+1, k, ell) < J_{last}$ **then**
              $J_{last} = J\ (i, j+1, k, ell)$
              Move: $\theta^i\ (j+1, k, ell) = \theta^i\ (j+1, k, ell) + C\ (i)\ \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$
              Compute fitness function, $J\ (i, j+1, k, ell)$ using the nutrient concentration at the current location of each bacterium and cell-to-cell interaction
            else
              m = N_s
            end if
          end while
        end for
      end for
      for i = 1 to S **do**
        $Jhealth(i) = \sum_{j=1}^{Nc+1} J(i, j, k, ell)$
      end for
      Sort bacteria in order of cost values of J_{health} [19, 23]
      The least healthy S_r bacteria will die
      Remaining S_r healthy bacteria each split
      Each such pair resides in the original position of the parent
    end for
    for i = 1 to S **do**
      Eliminate and disperse the ith bacterium, with probability p_{ed}, in such a way that, at the end, the number of bacteria in the population remains constant
    end for
  end for
until termination criterion satisfied
End

Algorithm 7.2 The algorithm representation of the proposed adaptive BFOA

Begin
Initialize all the free parameters of classical BFOA along with $C_{initial}$, $\varepsilon_{initial}$, α, β, K_u
repeat
for Elimination-dispersal: $ell = 1$ to N_{ed} **do**
  for Reproduction: $k = 1$ to N_{re} **do**
    for Chemotaxis: $j = 1$ to N_c **do**
      for each bacterium: $i = 1$ to S **do**
        Compute fitness function, $J(i, j, k, ell)$
        $J(i, j, k, ell) = J(i, j, k, ell) + J_{cc}\left(\theta, \theta^i(j, k, ell)\right)$
        $J_{last} = J(i, j, k, ell)$
        Tumble: Generate a random vector $\Delta(i) \in \Re^p$ with each element a random number in [-1,1]
        Move: $\theta^i(j+1, k, ell) = \theta^i(j, k, ell) + C(i, j, k)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$
        Compute fitness function, $J(i, j+1, k, ell)$
        $J(i, j+1, k, ell) = J(i, j+1, k, ell) + J_{cc}\left(\theta, \theta^i(j+1, k, ell)\right)$
        $m = 0$
        Swim:
        while m $< N_s$ **do**
          $m = m + 1$
          if $J(i, j+1, k, ell) < J_{last}$ **then**
            $J_{last} = J(i, j+1, k, ell)$
            Move: $\theta^i(j+1, k, ell) = \theta^i(j+1, k, ell) + C(i, j, k)\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$
            Compute fitness function, $J(i, j+1, k, ell)$ using the nutrient concentration at the current location of each bacterium and cell-to-cell interaction
            $flag(i) = 0$
          else
            $flag(i) = flag(i) + 1$
            if $flag(i) < K_u$ **then**
              $C(i, j+1, k) = C(i, j, k)$
              $\varepsilon(i, j+1, k) = \varepsilon(i, j, k)$ [26]
            else
              $C(i, j+1, k) = C_{initial}$
              $\varepsilon(i, j+1, k) = \varepsilon_{initial}$
            end if
            $m = N_s$
          end if
        end while
        if $J(i, j+1, k, ell) < \varepsilon(i, j, k)$ **then**
          $C(i, j+1, k) = \sin\left(\frac{J(i,j+1,k,ell)}{\alpha}\right)$
          $\varepsilon(i, j+1, k) = \frac{\varepsilon(i,j,k)}{\beta}$
        else
          $C(i, j+1, k) = C(i, j, k)$
          $\varepsilon(i, j+1, k) = \varepsilon(i, j, k)$
        end if
      end for
    end for
    for $i = 1$ to S **do**
      $Jhealth(i) = \sum_{j=1}^{Nc+1} J(i, j, k, ell)$
    end for
    Sort bacteria in order of cost values of J_{health} [19, 23]
    The least healthy S_r bacteria will die
    Remaining S_r healthy bacteria each split
    Each such pair resides in the original position of the parent
  end for
  for $i = 1$ to S **do**
    Eliminate and disperse the ith bacterium, with probability p_{ed}, in such a way that, at the end, number of bacteria in the population remains constant
  end for
end for
End

follows the classical BFOA. That is, once the chemotactic steps are completed in one generation, a reproduction step is activated and, after the specified reproduction steps are completed, then one elimination-dispersal step is activated. *flag(i)* is the indicative measure of the number of generations the *i*th bacterium has not improved its own fitness, $\varepsilon\ (i,\ j,\ k)$ is the required precision of fitness improvement in the present chemotactic generation of the *i*th bacterium, α and β are user-defined constants, $C_{initial}$ is initial run length and $\varepsilon_{initial}$ is the initial precision goal, respectively. All the other identifiers in this algorithm carry the identical meaning as in the classical BFOA.

7.5 Simulation Results

Here we have considered three benchmark images, namely Lena, Pepper and Boat images, of size 512 × 512 pixels, each image having 256 gray levels. Each image is first divided into 2 × 2 blocks, resulting in 65,536 numbers of 4 × 1 dimensional vectors. These vectors are used as training image pattern vectors. The optimal codebook is generated by minimizing the average distortion measure *D*, given as:

$$D = \frac{1}{n}\sum_{k=1}^{n} \min_{1 \le j \le c} \{ \| x_k - v_j \|^2 \} \tag{7.16}$$

Once the optimal codebook is obtained, using a candidate stochastic optimization algorithm, the efficiency of the scheme is evaluated by computing the peak-signal-to-noise-ratio (PSNR), in dB. The higher the PSNR value in dB, the better the reconstructed image.

$$PSNR = 10\log_{10} \frac{255^2}{\frac{1}{N*N}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(f\ (i,\ j) - \hat{f}\ (i,\ j)\right)^2} dB \tag{7.17}$$

where $N \times N$ is the size of the original image and $f\ (i,\ j)$ and $\hat{f}\ (i,\ j)$ are the gray-level pixel values of the original image and the reconstructed image, at position (*i,j*), respectively [9, 16].

Simulation is carried out in an ordered manner. In the first stage, the suitable combination of the free parameters of the BFOA is determined. This experimentation is carried out varying one free parameter at a time, keeping all other parameters constant and hence determining the most suitable value of this parameter varied. This process is carried out for each free parameter sequentially. In these experimentations, the classical BFOA is initialized each time using a fuzzy learning vector quantization (FLVQ) algorithm. To make a robust choice of each such parameter,

Table 7.1 Impact of variation of run length unit in BFOA (Pepper and Lena: size 256 × 256 divided into 2 × 2 blocks)

Codebook size (c)	Run length unit $C(i)$	Average distortion measure (D)		PSNR (in dB)	
		Pepper	Lena	Pepper	Lena
32	1.2	319.2347	288.2058	29.1122	29.5545
	1	313.8729	**284.5976**	29.1842	**29.6091**
	0.6	311.1685	286.9871	29.223	29.5728
	0.2	**305.3132**	**286.1672**	**29.3045**	**29.5853**
	0.06	313.1841	287.7165	29.1996	29.5624
64	1.2	195.9784	200.3456	31.2295	31.1336
	1	196.3358	**195.2687**	31.2223	**31.2451**
	0.6	**194.2688**	197.9233	**31.2674**	31.1864
	0.2	197.0793	**196.2235**	31.2058	**31.2239**
	0.06	197.7111	196.8354	31.1911	31.2104
128	1.2	131.1106	143.1953	32.9754	32.5921
	1	134.0000	143.4805	32.8800	32.5835
	0.6	133.3892	141.2616	32.9000	32.6512
	0.2	**131.7628**	**135.3008**	**32.9537**	**32.8384**
	0.06	133.4900	135.5395	32.8960	32.8307

Table 7.2 Impact of variation of run length unit in BFOA (Boat and Lena: size 512 × 512 divided into 2 × 2 blocks)

Codebook size (c)	Run length unit $C(i)$	Average distortion measure (D)		PSNR (in dB)	
		Boat	Lena	Boat	Lena
32	0.6	248.8429	153.1082	30.1922	32.3014
	0.2	**238.7726**	**148.8455**	**30.3716**	**32.4240**
64	0.6	178.3276	108.5133	31.6392	33.7966
	0.2	**175.5888**	**108.4443**	**31.7064**	**33.7993**
128	0.6	124.4358	78.9621	33.2020	35.1772
	0.2	**121.4000**	**77.4772**	**33.3092**	**35.2597**

Table 7.3 Impact of variation of number of bacteria in BFOA (Pepper and Lena: 256 × 256 size divided into 2 × 2 blocks)

Codebook size (c)	No. of bacteria (S)	Average distortion measure (D)		PSNR (in dB)	
		Pepper	Lena	Pepper	Lena
32	60	316.4495	289.7696	29.1484	29.5309
	40	317.2997	287.1783	29.1380	29.5699
	20	**304.4842**	**283.7988**	**29.3160**	**29.6213**
	10	**305.3132**	**286.1672**	**29.3045**	**29.5853**
64	60	199.7119	194.5630	31.1474	31.2608
	40	200.1956	**192.4327**	31.1369	**31.3086**
	20	**190.8185**	197.9498	**31.3452**	31.1859
	10	197.0793	**193.0866**	31.2058	**31.2939**
128	60	129.8249	133.1768	33.0446	32.9088
	40	132.3453	132.4453	32.9343	32.9310
	20	136.3333	133.2083	32.8054	32.9061
	10	**129.8156**	**133.1554**	**33.0181**	**32.9078**

the experimentation is carried out for three images (Lena, Pepper and Boat) and for three different codebook sizes ($c = 32, 64, 128$). At first, we vary run length unit $C(i)$ keeping $S = 10$, $N_c = 4$ and $N_s = 2$ as constants. Tables 7.1 and 7.2 show the image compression performances in terms of average distortion measure and PSNR.

After observing the results in Tables 7.1 and 7.2, the run length unit parameter $C(i)$ is chosen as 0.2, as at this setting most of the simulation results produced the lowest average distortion measure and the highest PSNR in dB value. Then keeping $C(i) = 0.2$ as constant, size of the population of bacteria colony (S) is varied in steps of 10, 20, 40 and 60. The common notion is that as the number of bacteria is increased, the optimization algorithm should yield better results, because it is expected that at least one bacterium will move closer to the optimum point [19].

However Table 7.3 shows that with an increase in the value of S often the performance deteriorated. The best results are obtained mostly with either $S = 10$ or $S = 20$. We chose $S = 10$, as an increase in the number of bacteria results in an increase in computational complexity as well as the computation time.

Table 7.4 Impact of variation of chemotaxis steps in BFOA (Pepper and Lena: 256 × 256 size divided into 2 × 2 blocks)

Codebook size (c)	Combination of ($N_c - N_s$)	Average distortion measure (D)		PSNR (in dB)	
		Pepper	Lena	Pepper	Lena
32	16 – 8	318.2147	300.5532	29.1243	29.3726
	8 – 4	317.3322	299.5829	29.1363	29.3862
	4 – 2	**305.3132**	**286.1672**	**29.3045**	**29.5853**
64	16 – 8	**195.8200**	207.6058	**31.2329**	30.9790
	8 – 4	197.7820	202.6545	31.1895	31.0838
	4 – 2	197.0793	**193.0866**	31.2058	**31.2939**
128	16 – 8	134.5591	141.4428	32.8623	32.6457
	8 – 4	133.8687	153.0440	32.8846	32.3032
	4 – 2	**129.8156**	**133.1554**	**33.0181**	**32.9078**

Table 7.5 Free parameters for classical BFOA

Parameter	Value
S	10
N_c	4
N_s	2
$C(i)$	0.2
N_{re}	1
N_{ed}	1
p_{ed}	0.4

Table 7.6 Free parameters specifically chosen for adaptive BFOA (ABFOA)

Parameters of ABFOA	Value
α	$3*\pi$
β	2
$\varepsilon_{initial}$	2,000
$C_{initial}$	0.6
K_u	5

Once S is fixed, we varied the $N_c - N_s$ combination. Though the computational complexity increases with the increase in number of chemotactic steps, the possibility of arriving at the optimum solution also increases [27]. By observing Table 7.4 we can conclude that the $N_c = 4$ and $N_s = 2$ combination gives considerably good results for both the Lena and Pepper images.

Thus the final parameter set obtained from classical BFOA is shown in Table 7.5.

Once the suitable free parameters of the classical BFOA are determined, the next step was to determine the additional free parameters α, β, $\varepsilon_{initial}$, K_u introduced in the adaptive BFOA (ABFOA). As our proposed algorithm attempts to adapt the run length unit in each processing step, the suitable nomenclature adopted for it is $C(i, j, k)$, i.e., the run length unit for ith bacterium, in the jth chemotactic step and the kth reproduction step. We propose to utilize a sine function for the adaptive run

Table 7.7 Comparative study of PSNR for the Lena,Pepperand Boat images for different codebook sizes

Algorithms	PSNR (in dB)				
	c=8	c=16	c=32	c=64	c=128
Lena					
FABFOAVQ1	28.8580	30.7428	32.3438	33.8746	35.3879
FABFOAVQ2	28.8576	30.7213	32.3423	33.8723	35.3732
FABFOAVQ3	28.8031	30.7157	32.3366	33.8091	35.3594
FBFOAVQ1	28.8442	30.7194	32.3703	33.8078	35.2807
FBFOAVQ2	28.8405	30.7121	32.3301	33.7900	35.2730
FBFOAVQ3	28.8400	30.6938	32.3089	33.7454	35.2602
Pepper					
FABFOAVQ1	27.3026	29.0870	31.3356	32.8237	34.2747
FABFOAVQ2	27.3319	29.0668	31.3250	32.7483	34.2119
FABFOAVQ3	27.3193	29.0676	31.2721	32.7394	34.3226
FBFOAVQ1	27.3097	29.0447	31.3675	32.7319	34.1442
FBFOAVQ2	27.3195	29.0262	31.2150	32.8045	34.1926
FBFOAVQ3	27.2823	29.0664	31.2457	32.6344	34.1838
Boat					
FABFOAVQ1	26.7552	28.7165	30.3018	31.8293	33.3748
FABFOAVQ2	26.7520	28.6127	30.2853	31.8107	33.3580
FABFOAVQ3	26.7363	28.6087	30.2853	31.7036	33.2515
FBFOAVQ1	26.7377	28.6903	30.3716	31.7064	33.3092
FBFOAVQ2	26.7430	28.6422	30.2530	31.7018	33.2760
FBFOAVQ3	26.7301	28.5345	30.0124	31.6155	33.2020

Table 7.8 Comparative study of PSNR in dB for competing algorithms for the Lena image (size: 512 × 512 pixels divided into 4 × 4 image blocks)

	Codebook size (c)	
	64	128
c-Means [16, 27]	28.638	29.257
LBG [6, 16]	28.451	28.656
ELBG [16, 34]	29.412	30.234
MD [14, 16]	27.998	28.904
ISM [16, 35]	27.876	29.121
FVQ [3, 16]	29.994	31.080
FLVQ [15, 16] $m_0 = 1.5$	29.903	30.908
Improved Batch FLVQ $\theta_0 = 0.2$ [16]	29.860	31.304
FBFOAVQ (proposed)	29.678	30.734
FABFOAVQ (proposed)	29.704	30.862

length unit, such that if $J\ (i, j, k, ell)$ decreases, $C(i, j, k)$ also decreases. Hence,

$$C(i, j, k) = \sin\left(\frac{J(i, j, k, ell)}{\alpha}\right) \tag{7.18}$$

Table 7.9 Comparative study of PSNR in dB for competing algorithms for the Boat image (size: 512 × 512 pixels divided into 4 × 4 image blocks)

	Codebook size (c)	
	64	128
c-Means [16, 27]	26.504	27.456
LBG [6, 16]	26.671	27.812
ELBG [16, 34]	27.552	27.897
MD [14, 16]	27.269	26.567
ISM [16, 35]	27.122	26.452
FVQ [3, 16]	27.213	28.234
FLVQ [15, 16] $m_0 = 1.5$	27.191	28.147
Improved Batch FLVQ $\theta_0 = 0.2$ [16]	27.294	28.268
FBFOAVQ (proposed)	27.321	28.308
FABFOAVQ (proposed)	27.334	28.400

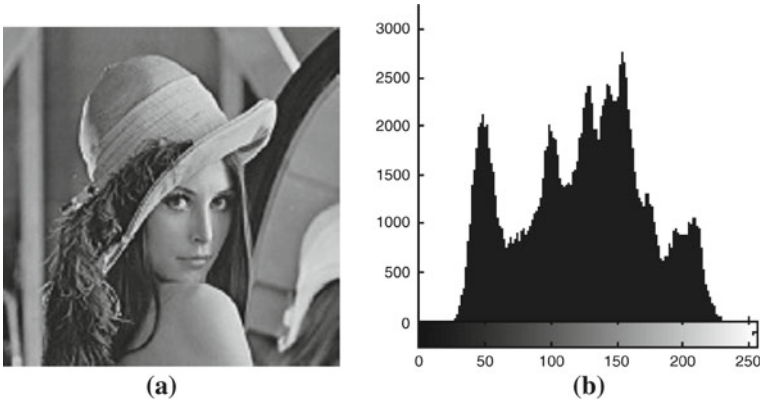


Fig. 7.3 **a** Original Lena image [36, 37]. **b** Histogram of the image in (**a**)

Precision goal $\varepsilon_{initial}$ is varied from 1,000 to 4,000 and is finally fixed at 2,000. In the literature, the suggested value of K_u is 20 [26]. However, we reduced it to 5 to ensure that the decision making is much quicker regarding whether the bacterium should be in the exploitation state or in the exploration state. Once the population of bacteria finishes chemotaxis it enters into reproduction and elimination-dispersal loop, same as classical BFOA. α and β are chosen as 3π and 2, respectively, after several trial and error-based test runs. The specific parameter settings for ABFOA are summarized in Table 7.6.

In the next stage of simulation analysis, we carried out a comparative study of the performance of classical BFOA with the performance of ABFOA. Here three benchmark images Lena, Pepper and Boat, each of size 512 × 512 pixels, were chosen. These images were first divided into small blocks of size 2 × 2 pixels, and training image vectors were accordingly generated. Then, for different codebook sizes, classical BFOA and ABFOA each underwent five independent runs. For each codebook size, the three best performances in terms of PSNR in dB are recorded

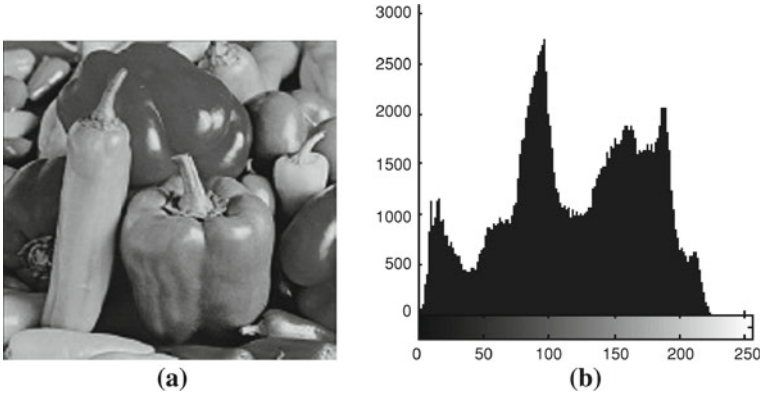


Fig. 7.4 **a** Original Pepper image [36, 37]. **b** Histogram of the image in (**a**)

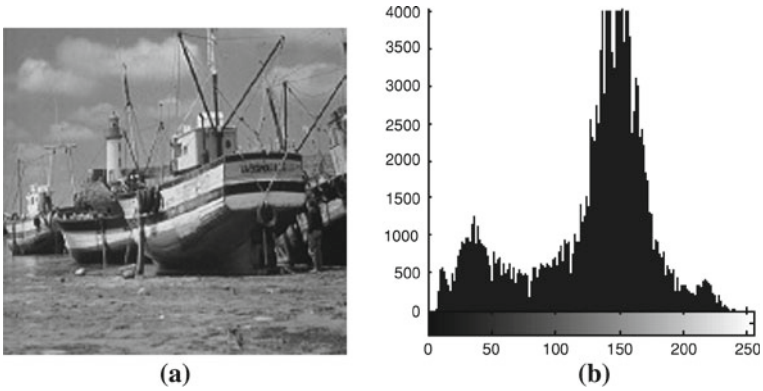


Fig. 7.5 **a** Original Boat image [36, 37]. **b** Histogram of the image in (**a**)

in Table 7.7. The three best versions of classical BFOA-based VQ are named as FBFOAVQ1, FBFOAVQ2 and FBFOAVQ3, and the three best versions of the adaptive BFOA-based VQ are named as FABFOAVQ1, FABFOAVQ2 and FABFOAVQ3, respectively. This table demonstrates that the performance of ABFOA, in most situations, dominates over the performance of classical BFOA. With bigger codebook sizes, this dominance gets more pronounced.

In the next stage of simulation, each image of size 512 × 512 pixels is divided into 4 × 4 blocks so that 16,384 numbers of 16 × 1 vectors are generated as training image pattern vectors. Then the optimal codebook is generated with the help of BFOA and ABFOA, and the compression quality is evaluated in terms of PSNR. Tables 7.8 and 7.9 present these results. One can see that FABFOAVQ has improved the performance of FBFOAVQ in each case. For the Boat image, FABFOAVQ and FBFOAVQ emerged as the two overall best performing algorithms. For the Lena

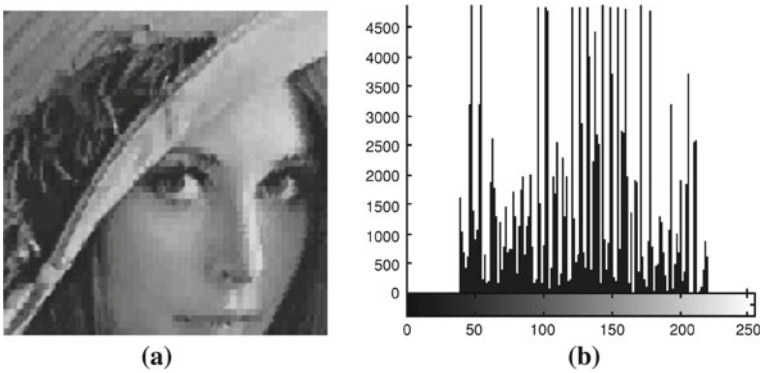


Fig. 7.6 **a** Zoomed reconstructed image of Lena with codebook size 64. **b** Histogram of the reconstructed image of Lena with codebook size 64

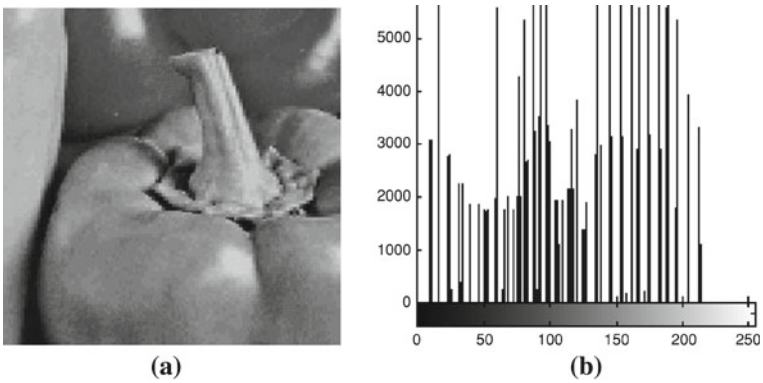


Fig. 7.7 **a** Zoomed reconstructed image of Pepper with codebook size 32. **b** Histogram of the reconstructed image of Pepper with codebook size 32

image, however, FVQ, FLVQ and improved batch FLVQ produced better results than FABFOAVQ and FBFOAVQ.

Figures 7.3, 7.4 and 7.5 show three benchmark images considered, along with the histograms constructed for pixel intensities for each of these images. Figures 7.6a, 7.7a and 7.8a show zoomed views of a subportion of each of these images, reconstructed using codebook vectors created employing FABFOAVQ scheme for image compression. These reconstructions are carried out for different sample codebook sizes or values of c. One can infer from these zoomed views that the reconstructed images are quite smooth in nature, and the compression carried out does not degrade the visual quality of the reconstructed images and, as such, no significant artifacts are visible. This is also justified by the corresponding histograms of these reconstructed images shown in Figs. 7.6b, 7.7b and 7.8b respectively. These histograms show that

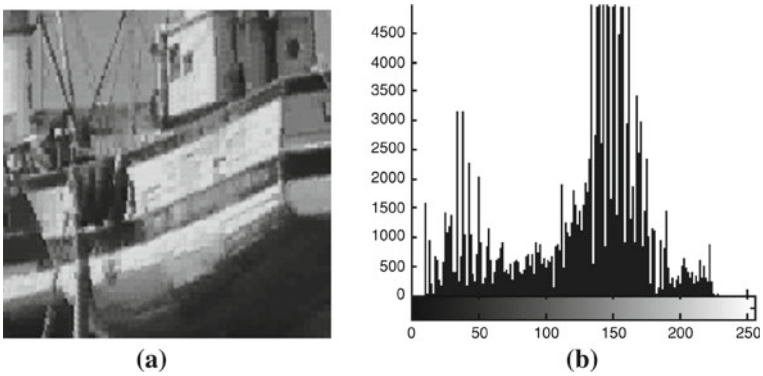


Fig. 7.8 **a** Zoomed reconstructed image of Boat with codebook size 128. **b** Histogram of the reconstructed image of Boat with codebook size 128

the distributions of pixel intensities are quite uniform throughout the entire universe of discourse of the gray levels and they are not cluttered in a small zone (or zones), even after undergoing the process of compression. Hence these figures provide a visual or qualitative justification for the efficiency of the compression algorithm proposed in this work.

7.6 Conclusion

In this work we have presented the development and evaluation of a classical bacterial foraging optimization algorithm and proposed a new adaptive bacterial foraging algorithm for VQ-based image compression. It has been shown that these BFOA and ABFOA algorithms can produce improvements in compression qualities for several images, in terms of average distortion measure and peak-signal-to-noise-ratio. Although FLVQ-based algorithms still produced better performance than classical BFOA and ABFOA-based VQ algorithms in some cases, it should be remembered that the performance of such FLVQ schemes depend largely on initial choices of free parameter(s). On the other hand, the BFO-based VQ algorithms showed relatively high robustness in their performance, and ABFOA exhibited less dependence on the initial choice of free parameters. The efficiency of the proposed algorithm is also demonstrated in terms of high fidelity of the reconstructed images.

References

1. Gonzalez, R.C., Woods, R.E.: Digital image Processing. Pearson Education India, Fifth Indian Reprint (2000)
2. Oehler, K.L., Gray, R.M.: Combining image compression and classification using vector quantization. IEEE Trans. Pattern Analy. Mach. Intell. **17**(5), 461–473 (1995)

3. Tsekouras, G.E.: A fuzzy vector quantization approach to image compression. Appl. Math. Comput. **167**, 539–560 (2005)
4. Pedreira, C.E.: Learning vector quantization with training data selections. IEEE Trans. Pattern Analy. Mach. Intell. **28**(1), 157–162 (2006)
5. Gonzalez, A.I., Grana, M., Cabello, J.R., Anjou, A.D., Albizouri, F.X.: Experimental results of an evolution Based strategy for VQ image filtering. Inf. Sci. **133**(2001), 249–266 (2001)
6. Linde, Y., Buzzo, A., Gray, R.M.: An algorithm for vector quantization design. IEEE Trans. Commun. **28**(1), 84–95 (1980)
7. Shen, F., Hasegawa, O.: An adaptive incremental LBG for vector quantization. Neural Netw. **19**, 694–704 (2006)
8. Zadeh, L.A.: Probability measures of fuzzy events. J. Math. Analy. Appl. **23**, 421–427 (1968)
9. Feng, H.M., Chen, C.Y., Ye, F.: Evolutionary fuzzy particle swarm optimization vector quantization learning scheme in image compression. Expert Syst. Appl. **32**(2007), 213–222 (2007)
10. Karayiannis, N.B.: A methodology for constructing fuzzy algorithms for learning vector quantization. IEEE Trans. Neural Netw. **8**(3), 505–518 (1997)
11. Karayiannis, N.B., Pai, P.I.: Fuzzy vector quantization algorithms and their application in image compression. IEEE Trans. Image Process. **4**(9), 1193–1201 (1995)
12. Pal, N.R., Bezdek, J.C., Hathaway, R.J.: Sequential competitive learning and the fuzzy c-means clustering algorithms. Neural Netw. **9**(5), 787–796 (1996)
13. Zeger, K., Vaisey, J., Gersho, A.: Globally optimal vector quantizer design by stochastic relaxation. IEEE Trans. Signal Process. **40**(2), 310–322 (1992)
14. Chan, C.K., Ma, C.K.: A fast method for designing better codebooks for image quantization. IEEE Trans. Commun. **42**(1994), 237–242 (1994)
15. Tsao, E.C.K., Bezdek, J.C., Pal, N.R.: Fuzzy Kohonen clustering networks. Pattern Recognit. **27**(5), 757–764 (1994)
16. Tsekouras, G.E., Mamalis, A., Anagnostopoulos, C., Gavalas, D., Economou, D.: Improved batch fuzzy learning vector quantization for image compression. Inf. Sci. **178**, 3895–3907 (2008)
17. Pasi, F.: Genetic algorithm with deterministic crossover for vector quantization. Pattern Recognit. Lett. **21**(1), 61–68 (2000)
18. Chen, C.-Y., Chen, K.-Y., Ye, F.: Evolutionary-based vector quantizer design. In: ICSS2005 International Conference on System & Signals, pp. 649–654. I-Shou University, Taiwan (2005)
19. Passino, K. M.: Biomimicry of bacterial foraging. IEEE Control Syst. Mag. **22**(3), 52–67 (2002)
20. Mishra, S.: A hybrid least square-fuzzy bacterial foraging strategy for harmonic estimation. IEEE Trans. Evol. Comput. **9**, 61–73 (2005)
21. Tripathy, M., Mishra, S., Lai, L.L., Zhang, Q.P.: Transmission loss reduction based on FACTS and bacteria foraging algorithm. In: Proceedings of the 9th International Conference on Parallel Problem Solving from Nature, pp. 222–231 (2006)
22. Mishra, S., Bhende, C.N.: Bacterial foraging technique-based optimized active power filter for load compensation. IEEE Trans. Power Deliv. **22**(1), 457–465 (2007)
23. Liu, Y., Passino, K.M.: Biomimicry of social foraging bacteria for distributed optimization: models, principles, and emergent behaviors. J. Optim. Theory Appl. **115**(3), 603–628 (2002)
24. Dasgupta, S., Biswas, A., Das, S., Abraham, A.: Adaptive computational chemotaxis in bacterial foraging optimization: an analysis. CISIS 2008, Barcelona, Spain. IEEE Computer Society Press, ISBN 0-7695-3109-1, pp. 64-71 (2008)
25. Biswas, A., Dasgupta, S., Das, S., Abraham, A.: A synergy of differential evolution and bacterial foraging algorithm for global optimization. Neural Netw. World **17**(6), 607–626 (2007)
26. Chen, H., Zhu, Y., Hu k.: Self-adaptation in bacterial foraging optimization algorithm. In: Proceedings of 2008 3rd International Conference on Intelligent System and Knowledge Engineering, pp. 1027–1031 (2008)
27. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs (1988)

28. Ishibuchi, H., Nakashima, T., Morisawa, T.: Voting in fuzzy rule-based systems for pattern classification problems. Fuzzy Sets Syst. **103**, 223–238 (1992)
29. Xu, W., Nandi, A.K., Zhang, J.: Novel fuzzy reinforced learning vector quantization algorithm and its application in image compression. IEEE Proc. Vis. Image Signal Process. **150**(5), 292–328 (2003)
30. Hirota, K., Pedrycz, W.: Data compression with fuzzy relational equations. Fuzzy Sets Syst. **126**(2002), 325–335 (2002)
31. Kim, D.H., Abraham, A., Cho, J.H.: A hybrid genetic algorithm and bacterial foraging approach for global optimization. Inf. Sci. **177**(18), 3918–3937 (2007)
32. Chatterjee, A., Fakhfakh, M., Siarry, P.: Design of second-generation current conveyors employing bacterial foraging optimization. Microelectron. J. **41622**(2010), 616–626 (2010)
33. Maitra, M., Chatterjee, A.: A novel technique for multilevel optimal magnetic resonance brain image thresholding using bacterial foraging. Elsevier Meas. **41**(2008), 1124–1134 (2008)
34. Patane, G., Russo, M.: The enhanced LBG. Neural Netw. **14**(2001), 1219–1237 (2001)
35. Kaukoranta, T., Franti, P., Nevalainen, O.: Iterative split-and-merge algorithm for vector quantization codebook generation. Opt. Eng. **10**(1998), 2726–2732 (1998)
36. http://decsai.ugr.es/cvg/CG/base.htm
37. http://sipi.usc.edu/database/database.cgi

Part III
Image Analysis Algorithms

Chapter 8
A Fuzzy Condition-Sensitive Hierarchical Algorithm for Approximate Template Matching in Dynamic Image Sequence

Rajshree Mandal, Anisha Halder, Amit Konar and Atulya K Nagar

Abstract Given a template of $m \times n$ and an image of $M \times N$ pixels, the latter being partitioned into blocks of $m \times n$ pixels with interleaving, template matching aims at determining the best matched target block in the image with respect to the template. This chapter develops a hierarchical algorithm of template matching using decision trees. Nodes in the tree, here, represent the features used for matching, while the arcs denote the conditions on the features to separate relatively better candidate solutions from the rest. The proposed hierarchical matching scheme tests the feasibility of each block by checking the satisfiability of the conditions labeled along the arcs. The block that satisfies the condition at one level is transferred to the next level, and discarded from the system otherwise. Thus blocks that traverse the largest depth are better candidate solutions. Among these solutions, the one with the smallest Euclidean distance with the template is declared as the winner. The work differs with respect to classical hierarchical template matching by two counts. First, the conditions here are induced with fuzzy measurements of the features. Fuzzy encoding eliminates small changes in imaging features due to variations in lighting conditions and head movement. Second, information gain is used to determine the order of the features to be examined by the tree for decision making. The time-

R. Mandal (✉) · A. Halder · A. Konar
Department of Electronics and Tele-Communication Engineering,
Jadavpur University, Kolkata-32, Kolkata, India
e-mail: rajshree.mandal@gmail.com

A. Halder
e-mail: halder.anisha@gmail.com

A. Konar
e-mail: konaramit@yahoo.co.in

A. K. Nagar
Department of Math and Computer Science,
Liverpool Hope University,
Liverpool, UK
e-mail: nagara@hope.ac.uk

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_8, © Springer-Verlag Berlin Heidelberg 2013

complexity of the proposed algorithm is of the order of MN/mn. The algorithm has successfully been implemented for template matching of human eyes in facial images carrying different emotions, and the classification accuracy is as high as 94 %.

8.1 Introduction

Template matching is a well-known problem in image understanding and interpretation. It has extensive applications in geographical/geological explorations, medical study, and the like. But its application in emotionally expressive facial region recognition is a novel problem. The classical template matching schemes presume static frames with distortions in the imagery. But when facial expression and in particular emotional expression is concerned, matching becomes a difficult problem. This is because of the fact that the detection of the nearest matched module/block in the image with respect to a static template sometimes gives rise to false indication, and occasionally misses the necessary target. The problem of image matching in dynamic image frames thus is a challenging problem.

There exists extensive works on image matching using correlation [1, 2], feature extraction [5], boosting process [3], distance transforms [4], subblock coding [6], moment descriptor [20] and other some other techniques [7–10, 12–15, 17]. Hierarchical image matching is also addressed in recent works [4]. However, we are afraid that there is almost no trace of research in the arena of template matching using hierarchical techniques. Studies of hierarchical template matching on emotionally excited faces are few and far between. The aim of this chapter is thus novel, and the method employed is also unique with respect to the reported works in this domain.

With the existing techniques of pattern recognition and image processing, hundreds or even more approaches to handle this problem can be addressed. But the objective here is to design a very robust algorithm, capable of matching in a finite time, where the time taken should be as small as possible. If such an algorithm can be designed, we would be able to use it for real-time matching in movie frames or even with minor modifications on real-time video. Designing a fast algorithm calls for minimum computations, without losing the target blocks. One approach to solve this problem is to consider matching template features with block features of a partitioned image in a time-staggered manner, so that important features that eliminate the possibility of matching can be used first to reduce the search space. In this chapter, we consider a hierarchical matching that explores the search space in such a time-staggered manner with multiple features, taken one at a time. One question that may be raised is how to detect the order of matching of the features. Here, we employ an entropy measure policy like the one considered in decision tree-based learning algorithms, to determine the order of features based on which the matching has to be accomplished.

In this chapter, simple statistical features like mean, standard deviation and kurtosis are considered to compare the template with the partitioned blocks in the image. Further, instead of directly matching the image attributes, the features are first mapped

on the fuzzy plane and then the comparison is carried out. The fuzzy measure reduces the scope of creeping up of noise in the matching process, and thus improves the robustness of the technique. The chapter has been divided into eight sections. Section 8.2 gives a small introduction on the principle of template matching. Section 8.3 gives the fuzzy conditions for approximate matching employed in our algorithm. The basis of hierarchical search and the decision tree learning approach is given in Sect. 8.4. The algorithm is given in Sect. 8.5. The experimental results are given in Sect. 8.6. Section 8.7 gives the performance analysis and the conclusions are listed in Sect. 8.8.

8.2 Principle of Template Matching

In template matching, we need to search a template of $m \times n$ pixels in an image of $M \times N$ pixels. A pixelwise matching of the template over the image definitely gives the best result, but is computationally very expensive and time consuming. The pixelwise matching scheme thus is prohibited for real-time applications, where time is an important factor. This calls for designing an intelligent search algorithm that alleviates the fundamental premise of pixelwise template matching. One way of formulating the template matching problem in the present context is to design a feature-based search strategy over a partitioned image of equal block size similar to that of a given template. The exploration should continue until the desired block can be identified. During the exploration phase, the template needs to be matched with the partitioned blocks with respect to the selected features. Selection of features for a general image without any knowledge of the background or context is not always easy. Here, we consider simple statistical features to be determined for each block in the image. A distance metric is defined to match the template features with those of individual image blocks. The simplest distance metric is the Euclidean distance. However, other distance metrics can also be used, depending on their suitability in applications. In this chapter, we select mean, standard deviation and kurtosis as three basic image attributes. To perform template matching in color images, here, the above features are evaluated in the r-,g- and b-planes separately.

We now formally define these parameters with respect to an α plane where $\alpha \in r, g, b$

Let

x_i^α be the intensity of the ith pixel on the α plane,

$mean_\alpha$ be the mean value of pixel intensities in a block on the α plane,

σ_α be the standard deviation of pixel intensities in a block on the α plane,

k_α be the kurtosis of pixel intensities in a block on the α plane,

n gives the total number of pixel intensities in the block.

Definition 8.1 For a particular block, *mean* gives the arithmetic average of all the pixel intensities in r-, g-, and b-planes, respectively, and is formally given by

$$mean_\alpha = \frac{1}{n}\sum_{i=1}^{n} x_i^\alpha \tag{8.1}$$

Definition 8.2 The *standard deviation* indicates a measure of deviation of the pixel intensities x_i from their mean. The standard deviation of a block of pixel intensity is given as

$$\sigma_\alpha = (\frac{1}{n}\sum_{i=1}^{n}(x_i^\alpha - mean_\alpha)^2)^{\frac{1}{2}} \tag{8.2}$$

Definition 8.3 The *kurtosis* of a block of pixels on the α plane is given by,

$$k_\alpha = \frac{E(x_i^\alpha - mean_\alpha)^4}{\sigma_\alpha^4} \tag{8.3}$$

where $E(X)$ is the expectation of the random variable X defined on a probability space.

The template matching algorithm can be realized by matching the template over equally sized partitioned blocks in the image. If the search is performed on non-overlapped regions in the image, the chances of identifying the target block becomes rare. To avoid this problem, we allow overlapped search with an interleaving of few pixels over the previously selected blocks. The more the overlap, the better the localization of the target region, at the cost of extra search time.

8.3 Fuzzy Conditions for Approximate Matching

Feature-based matching of template with partitioned blocks usually determines the distance between the measured features of the template with the respective features of a partitioned block. In many cases, it is observed that the template may not be present in the image in its exact form. This raises a fundamental problem, which is addressed here using a transformation of the measurements into fuzzy memberships. It is apparent that the logic of fuzzy sets has their inherent capability to handle approximate matching. We would explore this particular characteristic of fuzzy sets to perform approximate matching of a given block in an image with a fixed template. The definitions of fuzzy set and membership are introduced below for convenience of the readers.

Definition 4: Let X be a universe of measurements. For $x \in X$, we call A to be a fuzzy set under the universe X, where

$$A = \{x, \mu_A(x)\} \tag{8.4}$$

$\mu_A(x)$ is called the membership of x in A, where $0 \leq \mu_A(x) \leq 1$.

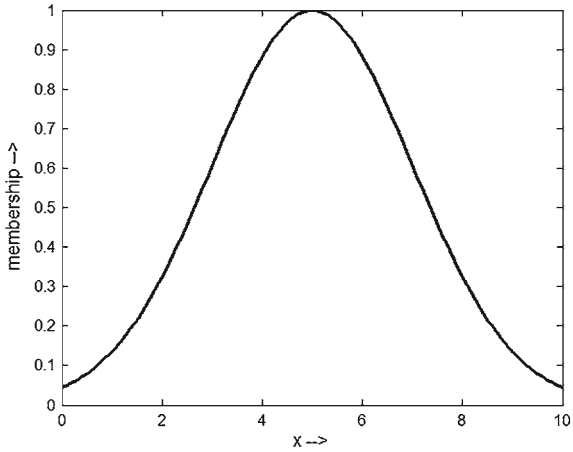


Fig. 8.1 A Gaussian membership distribution $\mu_A(x)$ versus parameter x

In this chapter, we consider Gaussian-type membership function, given by

$$\mu_A(x) = e^{-\frac{(x-m)^2}{2\sigma^2}} \tag{8.5}$$

where x is a linguistic variable in set A, and m and σ are the mean and standard deviation of x in set A. Figure 8.1 provides a Gaussian membership distribution curve. The significance of selecting Gaussian distribution is briefly outlined below.

The features of the template, here, have been modeled as fuzzy linguistic variables. Usually, for similar templates, the probability in deviation of a feature $\pm\delta(x)$ from its mean value $\bar{x}$ is presumed to be equal for large samples. This motivated us to use a Gaussian distribution as the membership distribution $\mu_A(x)$ for the feature x. Such membership distribution has a peak at the center of the span of the linguistic variable x, and can easily capture the membership of x in A, where A=EQUAL_TO_$\bar{x}$. In other words, the Gaussian membership distribution indicates the membership of x to be close to $\bar{x}$. Consequently, when x=$\bar{x}$, the membership is 1, and as x moves away from $\bar{x}$, the membership falls off.

Because of the inherent nonlinearity in the Gaussian membership functions, numerically close linguistic variables are mapped closer in the fuzzy space. Thus a search of the template on a uniformly noisy image with no background information about the noise characteristics can be performed efficiently using the proposed approach.

The feature-based fuzzy matching scheme to be proposed attempts to match the features of the template with those of a block, respectively, in the membership scale. For example, let $(\bar{x} + \delta)$ be the measurement of a feature in a given block, where $\bar{x}$ is the mean value of the feature obtained from several similar templates. Now, we say that the feature $\bar{x}$ of the template will be close enough to the feature $(\bar{x} + \delta)$ of the block, if

$$|\mu_A(\bar{x}) - \mu_A(\bar{x} + \delta)| \leq \varepsilon,$$

Table 8.1 Matching accuracy in percentage by varying threshold

	Emotion			
Threshold	Happy	Sad	Fear	Anger
0.9	100	100	80	80
0.8	80	100	60	65
0.7	80	100	60	50
0.6	70	95	50	45

Table 8.2 Position of the peak of the Gaussian distribution

Features	r-plane	g-plane	b-plane
Mean	179	121	93
Std. dev	10	8	7
Kurtosis	4	3	3

where ε is a very small preassigned positive quantity. The choice of ε is subjective to specific feature under consideration.

Usually, more than one feature is required to perform the template matching operation. Let, $f_1, f_2, \ldots, f_n$ be a set of n features used for template matching. Then, we say that the template will be close enough to a given block with respect to the above features if,

$$\begin{aligned} |\mu_{A_1}(\bar{f}_1) - \mu_{A_1}(\bar{f}_1 + \delta_1)| &\leq \varepsilon_1 \\ |\mu_{A_2}(\bar{f}_2) - \mu_{A_2}(\bar{f}_2 + \delta_2)| &\leq \varepsilon_2 \\ &\vdots \\ and\ |\mu_{A_n}(\bar{f}_n) - \mu_{A_n}(\bar{f}_n + \delta_n)| &\leq \varepsilon_n \end{aligned}$$

where

f_i is the ith feature in fuzzy set A_i,

$\bar{f}_i$ is the mean value of the ith feature in the template,

δ_i is the offset in measurement of the feature f_i in a given block,

and ε_i is the allowed tolerance level in membership matching of a feature between the template and a given block.

In our experiment, we consider three features: f_1 for mean, f_2 for standard deviation and f_3 for kurtosis of a given block and template. We experimented by varying ε_i for i=1 to 4 in [0.6, 0.9]. Table 8.1 shows the matching accuracy in percentage by varying ε_i from 0.6 to 0.9. The results given in the table are intuitively supported, as with increasing threshold the classification accuracy increases.

The above measurements of features are carried out in the r-,g- and b-planes separately, and the mean, standard deviation and kurtosis in three planes obtained from a set of thirty templates of left/right eye of a subject are evaluated. Table 8.2 gives the position of the peak value of the Gaussian curve.

One illustrative membership curve for the feature mean in the r-plane is given in Fig. 8.2 for convenience.

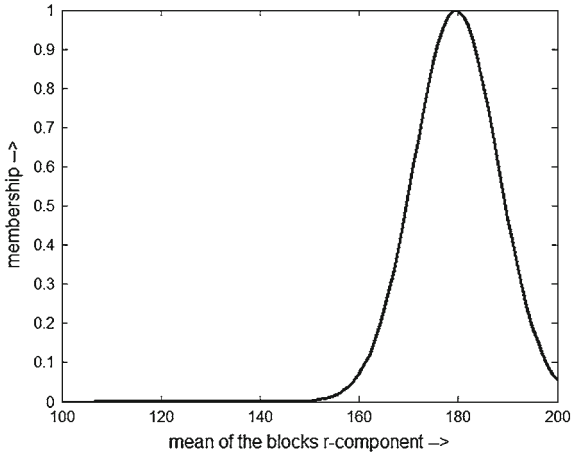


Fig. 8.2 Membership curve of mean of block r-component with $m = 179$

8.4 Template Matching by Hierarchical Search

An exhaustive feature-based matching of the template with individual partitioned blocks in the image definitely yields good results but at the cost of excessive computational complexity. This excessive complexity can, however, be reduced significantly by hierarchically matching the features of the partitioned blocks in the image with that of the template. This calls for ordering of the features in a manner so that the failures in matching can be identified earlier in the search process and final localization of the target block can be undertaken by matching other relevant features.

The hierarchical feature matching introduced here consists of both coarse and fine search. The coarse search first identifies the approximate location of the target block in a given image. The fine search is required to identify the exact location of the target block in and around the selected location of the block obtained in the coarse search.

The coarse search is accomplished using a decision-tree learning algorithm [19]. In a decision tree , the features(attributes) of a set of exemplar observation are used to classify all the data points of n features into two distinct classes. The nodes in a decision tree denote the features, and the arcs emanating from a node denote possible values of feature. For example, if 'wind velocity' is a feature to predict atmospheric storm, then the 'wind velocity' will be node, and the arcs emanating from that node are 'strong' and 'weak'. The most important issue in a decision-tree construction is to order the attributes in the tree in a hierarchical manner in order of their importance, starting at the root node and going downstream all the nodes excluding the leaf nodes. The leaf nodes in the decision tree are the boolean variables 'yes' and 'no'.

To determine the mapping of the features at different levels of the nodes, usually an entropy analysis is performed. This analysis helps in determining the sequential order of the attributes to be used at different levels of the hierarchy in the tree, starting at the root node.

The entropy calculation requires the determination of positive and negative proportions of the given instances favoring and denying the goal decision.

Let

'pos' denotes the proportion of positive instances in S

'neg' denotes the proportion of negative instances in S

We now define Information gain(S,A) where, S is number of samples of positive and negative instances for a goal/target decision, and A is a given attribute.

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \times (S_v) \tag{8.6}$$

where,

$$Entropy(S) = -pos\ log_2(pos) - neg\ log_2(neg) \tag{8.7}$$

The decision tree to be developed includes five levels of hierarchy, where the last level is the boolean variable 'yes'/'no' and the last-but-one level is the pixelwise difference in intensity between the template and the given block. We intentionally want to keep the pixel difference at the last-but-one level so that the overall search complexity of the algorithm is less. The features for which Information gain are to be determined so as to rank them for mapping in the decision tree according to their relative importance are mean, standard deviation and kurtosis at the r-,g- and b-planes together. To measure the possible values of the above three features, we use the following variables, A, B, C and P, as given below.

$$A = \mu_{EQ-TO-179}(mean) \wedge \mu_{EQ-TO-121}(mean) \wedge \mu_{EQ-TO-93}(mean) \tag{8.8}$$

$$B = \mu_{EQ-TO-10}(std.dev) \wedge \mu_{EQ-TO-8}(std.dev) \wedge \mu_{EQ-TO-7}(std.dev) \tag{8.9}$$

$$C = \mu_{EQ-TO-4}(kurtosis) \wedge \mu_{EQ-TO-121}(kurtosis) \wedge \mu_{EQ-TO-93}(kurtosis) \tag{8.10}$$

$$\begin{aligned} Z_{ij} = {} & |1 - \mu_{EQ-TO-179}(mean)| + |1 - \mu_{EQ-TO-121}(mean)| \\ & + |1 - \mu_{EQ-TO-93}(mean)| + |1 - \mu_{EQ-TO-10}(std.dev)| \\ & + |1 - \mu_{EQ-TO-8}(std.dev)| + |1 - \mu_{EQ-TO-7}(std.dev)| \\ & + |1 - \mu_{EQ-TO-4}(kurtosis)| + |1 - \mu_{EQ-TO-121}(kurtosis)| \\ & + |1 - \mu_{EQ-TO-93}(kurtosis)| \end{aligned} \tag{8.11}$$

$$P = the\ first\ n\ elements\ of\ the\ sorted\ Z_{ij}\ in\ ascending\ order. \tag{8.12}$$

Table 8.3 provides result of analysis of entropy and Information gain for mean, standard deviation and kurtosis. It is apparent from this table that the Information gain is the largest for mean, followed by that of standard deviation and that of kurtosis. Thus, mean, standard deviation and kurtosis are organized at successive levels of the decision tree, starting at the root. In Table 8.3, E stands for entropy.

Table 8.3 Entropy and Information Gain Calculation

No of blocks	E>(std. dev 0.9)		E>(mean 0.9)		E>(kurtosis 0.9)		Info Gain (mean)	Info Gain (std. dev)	Info Gain (kurtosis)
	p	n	p	n	p	n			
609	1	149	1	105	1	452	0.0041	0.0033	0.00069
							−0.0175	−0.0175	−0.0175
599	1	112	1	128	1	420	0.0036	0.0040	0.00084
							−0.0178	−0.0178	−0.0178
606	1	160	1	151	1	390	0.0032	0.0031	0.00108
							−0.0176	−0.0176	−0.0176
710	1	161	1	146	1	499	0.0031	0.0029	0.00095
							−0.0153	−0.0153	−0.0153
610	1	145	1	110	1	445	0.0045	0.0032	0.00077
							−0.0172	−0.0172	−0.0175
603	1	115	1	134	1	415	0.0038	0.0039	0.00088
							−0.018	−0.018	−0.018
702	1	158	1	135	1	376	0.0028	0.0025	0.00089
							−0.0148	−0.0148	−0.0148
714	1	165	1	148	1	502	0.0033	0.0031	0.00096
							−0.0151	−0.0151	−0.151
653	1	148	1	115	1	462	0.0053	0.0045	0.00086
							−0.0186	−0.0186	−0.0186
592	1	111	1	128	1	410	0.0035	0.0033	0.00086
							−0.0157	−0.0157	−0.0157

Besides the above, pixelwise difference is included at the lowest decision level of the tree, as given in Fig. 8.3. In this figure, we consider matching of a block when the parameters A, B, C exceed 0.9 and n is less than 10. Any time one of the conditions listed in the left-most leading edge of the tree violates the prescribed conditions, the block on which the matching was undertaken is discarded from the list. In case there exists more than one block of an image that satisfies the matching criteria in the first three steps, i.e., $A > 0.9$, $B > 0.9$ and $C > 0.9$, then we need to identify the target block by pixelwise matching between the template and the blocks so far selected by the first three steps. When number of such selected blocks is high, we then select the best ten among selected blocks based on their measure Z_{ij}. A block with smaller Z_{ij} is given preference to other blocks having relatively larger Z_{ij}. Thus pixelwise matching of the template with ten blocks reduces the computational overhead of the algorithm.

8.5 The Proposed Algorithm

The algorithm includes a coarse search followed by a fine search. The coarse search starts with evaluating A for all blocks of $m \times n$ pixels in a given image of $M \times N$ pixels with an interleaving of $m/4$ pixels for row-blocks and $n/4$ pixels for column

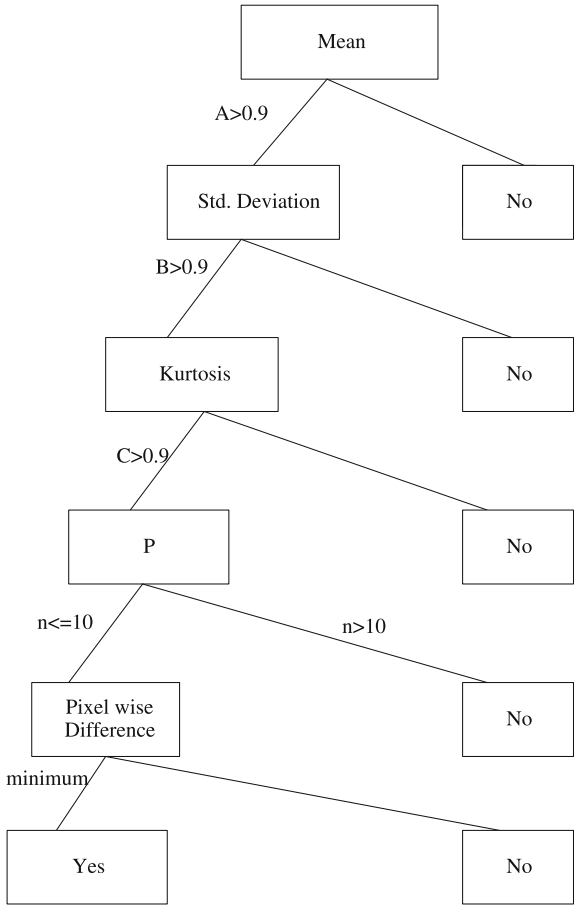


Fig. 8.3 Entropy-based decision tree for ordering the features

blocks. So, A is evaluated for $(\frac{M}{m/4} - 1)(\frac{N}{n/4} - 1) = (\frac{4M}{m} - 1)(\frac{4N}{n} - 1)$ blocks. The blocks satisfying $A > 0.9$ are passed on to the next stage, and B is evaluated for these blocks. Those blocks satisfying $B > 0.9$ are selected and passed on to the next stage. Now, C is evaluated for these blocks, and those satisfying $C > 0.9$ are considered as the nearest matched blocks with the template. Finally, the coarse search evaluates Z_{ij} for the block B_{ij} that satisfied $C > 0.9$. The $Z_{ij} \forall i, j$ are sorted in ascending order, and the indices (i, j) for the best ten blocks are saved in a set P. The fine search evaluates Euclidean distance between the template and all blocks with indices sorted in P and their neighborhood. The coarse search and the fine search algorithm are formally given next.

Input: A given template of $m \times n$ pixels to be searched on an image of $M \times N$ pixels.
Output: The block with minimum pixelwise unsigned difference with the template.

Coarse Search
Begin
Determine mean, std. deviation and kurtosis for the template;
$(S := \phi)$ // a set for holding values of Z_{ij}
For i:=1 to $(\frac{4M}{m} - 1)$ with an interleaving of $\frac{m}{4}$ pixels **do Begin**
For j:=1 to $(\frac{4N}{n} - 1)$ with an interleaving of $\frac{n}{4}$ pixels **do Begin**
find $\mu(mean)$ for the block for r-,g-,b- plane
If $A > 0.9$ **do Begin**
find $\mu(stddev)$ for the block for r-,g-,b- plane
If $B > 0.9$ **do Begin**
find $\mu(kurtosis)$ for the block for r-,g-,b- plane
If $C > 0.9$ **do Begin**
find Z_{ij}
$S := S \bigcup Z_{ij})$
End If
End If
End If
End For
End For
Sort S in ascending order of Z_{ij}
Save the indices (i,j) for the first 10 elements of S in set P

Fine Search
For $p \in P$ do Begin
For $block_p$ in the neighborhood N_p of p.
Find pixelwise Euclidean distance d_{ij} between the selected block and the template
Save the smallest distance d_{ij} in set D
End For
End For
Find the smallest element in D and print the block index(i,j)
End

Complexity: Given an image of $M \times N$ pixels, and a template of $m \times n$ pixels, the template is rolled over the image with an interleave of $m/4$ along the row and $n/4$ along the column. So, the total number of matching of the template with the image is $(\frac{M}{m/4} - 1)(\frac{N}{n/4} - 1) \approx 16(\frac{MN}{mn})$. The coarse search is performed in three steps at different levels of the tree. At the root level, the complexity is $16(\frac{MN}{mn})$. However, only a few blocks of the image are passed on to the next level, when the fuzzy

membership of the respective statistical variable (here mean), is within 10 % fall-off from the peak of the curve. Let the range of variable x for which the membership is within this 10 % fall-off from the peak be 2α. We know that the total x-span which approximately covers 99 % of the range of x is 6σ, where σ is the standard deviation of the variable x. So, the expected number of blocks to be passed on to the next level in the tree is given by $\frac{2\alpha}{6\sigma}16(\frac{MN}{mn})$ of the Gaussian curve. It can be shown that again the expected number of blocks to be passed on to the next descendent level is $(\frac{2\alpha}{6\sigma})^2 16(\frac{MN}{mn})$. So, the expected complexity of the coarse search is given by

$$\begin{aligned} T_{COARSE} &= 16(\frac{MN}{mn})[1 + (\frac{2\alpha}{6\sigma}) + (\frac{2\alpha}{6\sigma})^2] \\ &= O(\frac{MN}{mn}) \end{aligned} \tag{8.13}$$

To determine the complexity of the fine search, we first determine a zone around the selected block identified by the previous coarse search procedure. In this chapter, for a given template size of $m \times n$, the neighborhood around the centroid of the selected block is $(m + \frac{m}{2}, n + \frac{n}{2})$, and the template is searched in this region with an interleaving of $\frac{m}{16}$ along the row, and $\frac{n}{16}$ along the column. The complexity to match the template with the above interleaving is found to be $[\frac{3m/2}{m/16} - 1] \times [\frac{3n/2}{n/16} - 1] = 529$ (fixed). In our experiment, we select ten such blocks for finer matching. Thus complexity of fine search procedure is $O(5290)$.

$$T_{FINE} = O(5290) \tag{8.14}$$

The total complexity is given by $(T_{COARSE} + T_{FINE})$, which is $O(\frac{MN}{mn}) + O(5290)$. As a specific example, when $M = 640$, $N = 480$, $m = 40$, $n = 30$, the total complexity is obtained as $O(162) + O(5290)$, which means the fine search has much more complexity than the coarse counterpart. It can be verified that the overall complexity of the algorithm is approximately $21 \times (\frac{MN}{mn})$ for the given settings of M, N, m and n. Thus the expected time-complexity of the hierarchical search is $O(21\frac{MN}{mn}) \approx O(\frac{MN}{mn})$.

8.6 Experiments and Computer Simulation

The work was undertaken in the Artificial Intelligence Laboratory of Jadavpur University. The experiment was conducted with ten subjects whose facial expressions conveying different emotions such as happiness, anger, fear and sadness were captured. Before template matching was carried out, the skin region is first detected. This is done in order to localize the search space for template matching. Skin region detection is carried out on the HSV color model. Two parameters, namely x and y, are chosen using the relations:

Fig. 8.4 Eye template (**b**) is extracted from image (**a**)

Table 8.4 Detection of skin region and right eye using template in Fig. 8.4b under different emotional states (H: Happy, S: Sad, F: Fear, A: Anger)

Emotion	Input Image	Skin Region	Right Eye Identified
H			
S			
F			
A			

$$x = (0.148 \times H) - (0.291 \times S) + (0.439 \times V) + 128 \tag{8.15}$$

$$y = (0.439 \times H) - (0.368 \times S) - (0.071 \times V) + 128 \tag{8.16}$$

Fig. 8.5 Right eye template matching of a subject for different emotions

Fig. 8.6 Left eye template matching of a subject for different emotion

Table 8.5 Matching score of right eyes conveying different emotions with the eye in the relaxed state

Happy	Sad	Fear	Anger
0.7084	0.8062	0.8053	0.5675
0.656	0.7772	0.7298	0.6079
0.75	0.69	0.7129	0.612
0.702	0.6918	0.723	0.605
0.705	0.612	0.758	0.592

Table 8.6 Matching score of left eyes conveying different emotions with the eye in the relaxed state

Happy	Sad	Fear	Anger
0.81	0.759	0.586	0.554
0.712	0.728	0.608	0.563
0.725	0.796	0.593	0.607
0.695	0.726	0.612	0.593
0.673	0.805	0.72	0.584

For each and every pixel, the parameters x, y and H are determined, and if they lie in the range given below, it is considered as skin pixel.

$$140 \leq y \leq 165$$
$$140 \leq x \leq 195$$
$$0.01 \leq hue \leq 0.1$$

After the skin region is extracted, our next task is to identify the block with max^m resemblance with the given template. In our experiment, we have taken an eye-template of the subject when he/she is emotionally relaxed. With this template, the eye region of the subject conveying different emotional expressions is identified with the help of the hierarchical template-matching algorithm as mentioned in the previous section. This was repeated for ten individuals and we observed that in most of the cases the eye region matches successfully. The success rate of this experiment is found to be 94 %.

Figure 8.4a is an illustrative facial expression of a person in a relaxed state. The corresponding eye template that was manually extracted is given in Fig. 8.4b. Template matching is then performed on the facial expression of the same person, conveying different emotions. It is observed that the right eye in the third column of Table 8.4 is correctly identified for the given emotional expressions.

The right and left eyes of a subject in a relaxed state are now identified, and template matching is performed on their facial expressions conveying different emotions (Figs. 8.5 and 8.6). The matching score of the right and left eyes with respect to the eye block in the relaxed state are given in Tables 8.5 and 8.6, respectively.

The experiment is now repeated to identify the target block in group photographs of three people. In Fig. 8.7, the eye template of a subject is manually extracted from her facial expression under relaxed state. The eye template is then searched in group photographs of left column in Table 8.7 under different emotional expressions. We obtained the same emotional expression of all the three people by audio-visual stimulus developed in our previous experiment on emotional research [18]. Interestingly, in all four emotional settings, the target blocks are correctly identified. The detected blocks are shown in Fig. 8.8. Experiments reveal that for group photographs, the accuracy is 82 %.

8.7 Performance Analysis

The experiment was performed on 76 images. Out of them, the eye region was detected correctly for 72 images. The accuracy of our proposed algorithm is measured to be 94 %. Table 8.8 shows the complexity comparison of our proposed algorithm with other existing methods. It is apparent from the table that the complexity obtained from our proposed algorithm is less compared to other existing methods.

Fig. 8.7 Eye template (**b**) is extracted from image (**a**)

Table 8.7 Matching of template for experiment done on a group

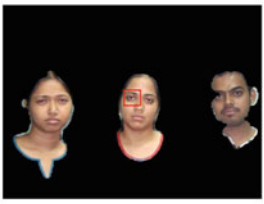
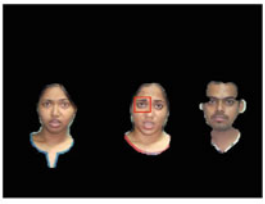
Emotion	Input Image	Skin region identified and then right eye marked
H		
S		
F		
A		

Fig. 8.8 Detected eye blocks

H: Happiness, S: Sadness, F: Fear, A: Anger.

Table 8.8 Performance analysis by complexity measure of other techniques

Technique	Imaging Parameters	Complexity
Two-dimensional template matching based on polynomial approximation [11]	Input image is $M \times N$ pixels, d is the degree of polynomial and L is the number of partial images	$O(MNd^2 + Ld^4)$
Image matching algorithm based on subblock coding [6]	Template size is $N \times N$ pixels, Image size is $M \times M$ pixels	$O(M^2)$
Sum of squared differences method	–do–	$(N - M + 1)^2$
QFT phase-only correlation template match[16]	Input image is $M \times N$ pixels template is $m \times n$ pixels	$O((log(MN))^2)$
Fast Fourier transform [16]	–do–	$O(MNlog(MN))$
Pattern matching for rotation and scaling space [3]	Template is $m \times m$ pixels and Input image $n \times n$ pixels. s and r are the dimensions of the scaling and rotation space	$O(m^2n^2sr)$
Hierarchical algorithm for approximate template matching	**Size of input image is** $M \times N$ **and size of template is** $m \times n$	$O(MN/mn)$

8.8 Conclusions

The chapter introduced a hierarchical approach to template matching. The proposed algorithm for template matching is capable of detecting both noisy and distorted partitioned image blocks similar to that of the template. Because of its hierarchical structure, the algorithm is highly time efficient, and outperforms most of the popular techniques for template matching. Experimental results confirm percentage matching accuracy is as high as around 94 % for single images and 82 % for group images.

Because of its high matching accuracy and low computational overhead, the proposed algorithm is a good choice for real-time image matching. Also, it is capable of approximate matching of a template with a given image. To support the above statement, we consider matching of emotional facial expressions with a relaxed template in our above-mentioned experiment and observed that it could correctly match the eye template with these emotional expressions. Further, due to its inherent capability of approximate matching, the algorithm can be employed for matching of salient facial attributes, such as eye or lip with those of partitioned image blocks, where the emotional content of the template need not match with that of the partitioned blocks containing the desired facial attribute.

References

1. Biswas, B., Konar, A., Mukherjee, A.: Image matching with fuzzy moment descriptors. Eng. Appl. Artif. Intell. **14**(1), 43–49 (2002)
2. Chakraborty, A., Konar, A., Chakraborty, U. K., Chatterjee, A.: Emotion Recognition From Facial Expressions and Its Control Using Fuzzy Logic. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **17**(2), 125–150 (2009)
3. Chen, W., Sun, T., Li Wang, X. Y.: Face Detection Based on Half Face Template. 9th International Conference on Electronic Measurement & Instruments, ICEMI '09, pp. 54–58 (2009)
4. Feng, Y., Li, S., Dai, M.: An Image Matching Algorithm Based on Subblock Coding. Second International Workshop on Computer Science and, Engineering (2009)
5. Gavrila, D.M.: Multi feature Hierarchical Template matching using Distance Transforms. Fourteenth International Conference on Pattern Recognition, vol. 1, pp. 439–444 (1998)
6. Gavrila, D.M.: A bayesian, exemplar-based approach to hierarchical shape matching. IEEE Trans. Pattern Anal. Mach. Intell. **29**(8), 1408–1421 (2007)
7. Jian, W., Honglian, Z.: Eye detection based on multiangle template matching. International conference on image analysis and signal processing, IASP, pp. 241–244, 2009
8. Langer, M., Kuhnert, K.-D.: A New hierarchical approach in robust real-time image feature detection and matching. Pattern recognition ICPR, University Siegen, Germany, 2008
9. Leng, J., Ni, J.: A novel fingerprint bifurcation extraction algorithm based on neural network template matching. International symposium on computer science and computational technology, Tianjin University of Technology, Tianjin, 300191, China, 2008
10. Li, W., Yan-xiang, H.: Face detection based on QFT phase-only correlation template match. ICISE (2009)
11. Lin, Z., Davis, L. S., Doermann, D., DeMenthon, D.: Hierarchical part-template matching for human detection and segmentation. IEEE 11th International conference on computer vision, ICCV 2007, pp. 1–8, 2007
12. Lin, Z.: Shape-based human detection and segmentation via hierarchical part-template matching. IEEE Trans. Pattern Anal. Mach. Intell. **32**(4), 604–618 (2010)
13. Mitchell, T.M.: Machine Learning. McGraw-Hill International, New York (1997)
14. Miyazaki, S., Takano, H., Nakamura, K.: Suitable checkpoints of feature surrounding the Eye for eye tracking using Template matching. SICE 2007, Tokoyama Prefectural University, Tokoyama, Japan, 2007
15. Omachi, M., Omachi, S.: Fast two-dimensional template matching with fixed aspect ratio based on polynomial approximation. ICSP, 2008
16. Sha, S., Jianer, C., Sanding, L.: A fast matching algorithm based on K-degree template. central south university, ICCSE, 2009
17. Suzuki, Y., Boon, C.S., Tan, T.K.: Inter frame coding with template matching averaging. ICIP, 2007
18. Uz, T., Bebis, G., Erol A., Prabhakar, S.: Minutiae-based template synthesis and matching using hierarchical delaunay triangulations. BTAS, 2007
19. Wang, J., Yang, H.: Face detection based on template matching and 2DPCA algorithm. In: Proceedings of the 2008 congress on image and signal processing CISP '08, vol. **4**, pp. 575–579, 2008
20. Wu, P., Hseieh, J.W.: Efficient image matching using concentric sampling features and boosting process. Yuan Ze University, Taiwan, ICIP, 2008

Chapter 9
Digital Watermarking Strings with Images Compressed by Fuzzy Relation Equations

Ferdinando Di Martino and Salvatore Sessa

Abstract A gray image is seen as a fuzzy relation R if its pixels are normalized with respect to the length of the used scale. This relation R is divided in submatrices defined as blocks, and each block R_B is coded to a fuzzy relation G_B, which in turn is decoded to a fuzzy relation D_B (unsigned) whose values are greater than those of R_B. Both G_B and D_B are obtained via fuzzy relation equations with continuous triangular norms (in particular, here we use the Lukasiewicz t-norm) and the involved fuzzy sets (coders) are Gaussian membership functions. Let D be the image obtained from the recomposition of the $\underline{D}'_B s$. In this work we use a watermarking method based on the well-known encrypting alphabetic text Vigenère algorithm. Indeed we embed such watermark in every G_B with the Least Significant Bit Modification (LSBM) algorithm obtaining a new matrix G_B, decompressed to a matrix $\underline{D}_B$ (signed). Both $\underline{G}_B$ and $\underline{D}_B$ are deduced with the same fuzzy relation equations used for obtaining G_B and D_B. The recomposition of the $\underline{D}'_B s$ gives the image $\underline{D}$ (signed). The quality of the reconstructed images with respect to the original images is measured from the Peak Signal-to-Noise Ratio (PSNR), and we show that $\underline{D}$ is very similar to D for low values of the compression rate. The binary watermark matrix embedded in every G_B is variable, thus this method is more secure than another our previous method, where the binary watermark matrix in every G_B is constant.

F. Di Martino (✉) · S. Sessa
Dipartimento di Costruzioni e Metodi Matematici in Architettura,
Università degli Studi di Napoli Federico II,
Via Monteoliveto 3, 80134 Napoli, Italy
e-mail: fdimarti@unina.it

S. Sessa
e-mail: sessa@unina.it

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_9, © Springer-Verlag Berlin Heidelberg 2013

9.1 Introduction

Digital watermarking (see also the optimum bibliography [13]) can be based on different technologies like fuzzy C-means [3], fuzzy relation equations [7, 15], genetic algorithms [2, 4], wavelets [1], etc. but here we improved our method [7] by using the well-known encrypting alphabetic text Vigenère algorithm.

Let $t : x, y \in [0,1]^2 \to xty = t(x,y) \in [0,1]$ be a continuous triangular norm (t-norm, for short) and "$\to_t$" be its residuum operator defined as $(x \to_t y) = \sup\{z \in [0,1] : xtz \geq y\}$ for all $x, y \in [0,1]$. The most used t-norms [10] are the classical minimum, arithmetical product and the Lukasiewicz t-norm L, given as $xLy = \max\{0, x+y-1\}$ and $(x \to_L y) = \min\{1, 1-x+y\}$ for all $x, y \in [0,1]$. These t-norms are used in the structure of fuzzy relation equations [8] for coding/decoding image processes [5, 6, 11, 12, 14].

Indeed, let $I_s = \{1, \ldots, s\}$ with s natural number. A gray image R of sizes $m \times n$ is seen as a fuzzy relation $R : (i,j) \in I_m \times I_n \to [0,1]$, $R_{ij} = R(i,j)$ being the normalized value of the pixel $P_{ij} = P(i,j)$. That is $R_{ij} = P_{ij}/Lt$ if Lt is the length of the gray scale (here, for simplicity, $Lt = 255$). R is compressed to a matrix $G : (p,q) \in I_k \times I_h \to G_{pq} = G(p,q) \in [0,1]$, of sizes $k \times h$, $k \geq m$ and $h \geq n$, by using a system of fuzzy relation equations of max $-t$ type, and successively G is decompressed to a matrix $D : (i,j) \in I_m \times I_n \to D_{ij} = D(i,j) \in [0,1]$ of sizes $m \times n$ via a system of fuzzy relation equations of min $- \to_t$ type.

The quality of D with respect to the original image R is measured by means of the Peak Signal-to-Noise Ratio (PSNR), given by:

$$\mathrm{PSNR} = \log_{10} \frac{255}{\mathrm{RMSE}} \tag{9.1}$$

where the Root Mean Square Error (RMSE) is defined by:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \left(R_{ij} - D_{ij}\right)^2}{m \times n}} \tag{9.2}$$

Our method consists in the embedding of a watermark with the encrypting alphabetic text Vigenère algorithm in the matrix G, by obtaining the matrix $\underline{G}$ of sizes $k \times h$. This matrix $\underline{G}$ is decompressed to a signed matrix $\underline{D}$ of size $m \times n$ with the same system of fuzzy relation equations of max $-t$ type used for decompressing the unsigned matrix G. In order to guarantee the solvability of all the equations, we take coders which are normal fuzzy sets. Indeed, let $A_p : i \in I_m \to A_p(i) = A_{pi} \in [0,1]$ and $B_q : j \in I_n \to B_q(j) = B_{qj} \in [0,1]$ for all $p \in I_k, i \in I_m, q \in I_h, j \in I_n$. Strictly speaking, as in [7], we consider coders with Gaussian membership function, that is $A_p : x \in [0, +\infty) \to A_p(x) \in [0,1]$ and $B_q : x \in [0, +\infty) \to B_q(x) \in [0,1]$, and we put for any $(p,q) \in I_k \times I_h$:

$$A_p(x) = \exp\left[-\alpha\left(p\frac{m}{k} - x\right)^2\right] \qquad B_q(x) = \exp\left[-\alpha\left(q\frac{n}{h} - x\right)^2\right] \tag{9.3}$$

where $\alpha \in \{0.1, 0.2, \ldots, 1.0\}$ is a parameter optimized in such a way that the RMSE is minimum. Of course A_p and B_q assume values for $x = i \in I_m$ and $x = j \in I_n$, respectively, and also are normal fuzzy sets such that $A_p(pm/k) = B_q(qn/h) = 1$. Motivated from our previous papers [5, 6], we adopt the t-norm L and various compression rates using always the PSNR between the reconstructed image D (unsigned) and $\underline{D}$ (signed) as the performance index.

9.2 Preliminary Results

With the same notation as Sect. 9.1, let R : $(i, j) \in I_m \times I_n \to [0, 1]$, be an assigned matrix with entries $R_{ij} = R(i, j)$, $A_1, \ldots, A_k : I_m \to [0, 1]$ and $B_1, \ldots, B_h : I_n \to [0, 1]$ be assigned fuzzy sets and $G : (p, q) \in I_k \times I_h \to [0, 1]$ be a matrix, with entries $G_{pq} = G(p, q)$ for all $(p, q) \in I_k \times I_h$, defined by

$$G_{pq} = \bigvee_{i=1}^{m} \bigvee_{j=1}^{n} \left[\left(A_{pi}\, t\, B_{qj}\right) t\, R_{ij}\right] \tag{9.4}$$

If the system (9.4) has solutions R, then the following result holds [8]:

Theorem 9.1 *Let A_p, B_q, G_{pq} be given for all $(p, q) \in I_k \times I_h$ and the system (9.4) be solvable in the unknown R. Then the fuzzy relation $D : (i, j) \in I_m \times I_n \to D(i, j) = D_{ij} \in [0, 1]$ defined as*

$$D_{ij} = \bigwedge_{p-1}^{k} \bigwedge_{q-1}^{h} \left[\left(A_{pi}\, t\, B_{qj}\right) \to_t G_{pq}\right] \tag{9.5}$$

is the greatest solution of (9.4); that is $D_{ij} \geq R_{ij}$ for all $(i, j) \in I_m \times I_n$ for any R.

The solvability degree ξ_{pq} [8, 9] of every fuzzy Eq. (9.5) is defined for all $(p, q) \in I_k \times I_h$ as

$$\xi_{pq} = G_{pq} \to_t \left[\bigvee_{i=1}^{m} \bigvee_{j=1}^{n} \left(A_{pi}\, t\, B_{qj}\right)\right] \tag{9.6}$$

and we know that [9]:

Theorem 9.2 *Every Eq. (9.5) has the fuzzy relation $R_{pq} : (i, j) \in I_m \times I_n \to R_{pq}(i, j) = (A_{pi}\, t\, B_{qj}) \to_t G_{pq} \in [0, 1]$ as the greatest solution iff $\xi_{pq} = 1$.*

G (resp. D) is the compression (resp. decompression) of R by means of the coders $\{A_1, \ldots, A_k\}$ and $\{B_1, \ldots, B_h\}$. As already proposed in some previous papers [5, 6, 11, 12], we divide R of sizes $m \times n$ in fuzzy submatrices R_B of sizes $m_B \times n_B$, called blocks. Every R_B is coded to a fuzzy matrix G_B of sizes $k_B \times h_B (k_B \leq m_B$ and $h_B \leq n_B)$ with the following fuzzy equation of max $-t$ type:

$$G_B(p, q) = \bigvee_{i=1}^{m_B} \bigvee_{j=1}^{n_B} \left[\left(A_{pi} t B_{qj} \right) R_B(i, j) \right] \tag{9.7}$$

for $p = 1, \ldots, k_B$ and $q = 1, \ldots, h_B$. Every G_B is decoded to a fuzzy matrix D_B of sizes $m_B \times n_B$ with the following fuzzy equation of min $-t$ type:

$$D_B(i, j) = \bigwedge_{p-1}^{k_B} \bigwedge_{q=1}^{h_B} \left[\left(A_{pi} t B_{qj} \right) \rightarrow_t G_B(p, q) \right] \tag{9.8}$$

for $i = 1, \ldots, m_B$ and $j = 1, \ldots, n_B$. We recompose the $D_B's$ for deducing the fuzzy relation D of sizes m × n. The coders of the Eqs. (9.7) and (9.8) are given from Eq. (9.3) by setting $m = m_B, n = n_B, k = k_B, h = h_B$, and α is optimized in such a way that the following RMSE

$$(\text{RMSE})_B = \sqrt{\frac{\sum_{i=1}^{m_B} \sum_{j=1}^{n_B} [R_B(i, j) - D_B(i, j)]^2}{m_B \times n_B,}} \tag{9.9}$$

assumes minimum value. Next a binary watermark matrix W_B, based on the encrypting alphabetic text Vigenère algorithm, is embedded in each unsigned G_B, and let $\underline{G}_B$ this new fuzzy relation, decoded via Eq. (9.8) with the same coders, to a fuzzy relation D_B defined, for $i = 1, \ldots, m_B$ and $j = 1, \ldots, n_B$, as

$$\underline{D}_B(i, j) = \bigwedge_{p=1}^{k_B} \bigwedge_{q=1}^{h_B} \left[\left(A_{pi} t B_{qi} \right) \rightarrow_t \underline{G}_B(p, q) \right] \tag{9.10}$$

Thus $D_B(i, j)$ is the greatest solution of the following equation:

$$\underline{G}_B(p, q) = \bigvee_{i=1}^{m_B} \bigvee_{j=1}^{n_B} \left[\left(A_{pi} t B_{qj} \right) t R_B(i, j) \right] \tag{9.11}$$

for $p = 1, \ldots, k_B$ and $q = 1, \ldots, h_B$. Since we use the normal Gaussian coders (9.3), we have that

$$\bigvee_{i=1}^{m_B} \bigvee_{j=1}^{n_B} \left[\left(A_{pi} t B_{qj} \right) \right] = 1 \tag{9.12}$$

and hence $\xi_{pq} = (G_{pq} \rightarrow_t 1) = 1$ for $p = 1, \ldots, k_B$, $q = 1, \ldots, h_B$. Thus every Eq. (9.11) has solutions by Theorem 9.2. Each D_B has sizes $m_B \times n_B$, and their recomposition gives the fuzzy relation $\underline{D}$ of sizes $m \times n$ which we compare with the unsigned image D. Without loss of generality, we consider $m_B = n_B$ and $k_B = h_B$ in all the experiments.

9.3 Usage of Watermarking Strings

In [7] the authors proposed a digital image watermarking method applied on blocks coded as above and realized on a set of images of sizes 256×256. The watermarking code applied over every compressed block is composed by a constant binary matrix with the same dimensions of the block, having values equal to 1 in the first entry and 0 otherwise. In this work we experiment with the use of a variable binary watermarking matrix applied to each unsigned compressed block; this matrix is derived starting with an initial string called decoding "key string" and used as password to obtain the unsigned image. Indeed, a binary matrix representing a watermark can be interpreted as a binary code of an ASCII character.

For example, in the ASCII code the sequence of nine bits 001010111 corresponds to the character W, which can be considered as the following 3×3 binary matrix:

$$W_B = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \tag{9.13}$$

If we have images with 256 gray levels and we normalize this image with respect to $Lt = 255$, we obtain the fuzzy relation:

$$W_{BN} = \begin{bmatrix} 0 & 0 & 0.00392156 \\ 0 & 0.00392156 & 0 \\ 0.00392156 & 0.00392156 & 0.00392156 \end{bmatrix} \tag{9.14}$$

In our tests we consider a character string and apply the well-known encrypting alphabetic text Vigenère algorithm to each block of the compressed matrix corresponding to a character string. This algorithm is a simple form of poly-alphabetic substitution. Indeed it uses a key string (password) of arbitrary length and a character string: the ith character of the text to be ciphered is compared with the ith character of the key string repeated until it covers the number of the characters of the text. The character to be ciphered is determined by using the following rule:

- If i_1 is the occurrence of the ith character in the text and i_2 is the character of the correspondent key string, the ith character assigned to the ciphered text will be corresponding to the occurrence $i_3 = i_1 + i_2 - 1$. If i_3 is greater than the

Table 9.1 Example of ciphered text Vigenère algorithm

Text char	Key char	i_1	i_2	i_3	Cipher text char
c	*c*	3	3	5	*e*
a	*i*	1	9	9	*i*
c	*a*	3	1	3	*c*
h	*c*	8	3	1	*a*
e	*i*	5	9	4	*d*

Table 9.2 Sequences shifted one position

	1	2	3	4	5	6	7	8	9
1	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*
2	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*a*
3	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*a*	*b*
4	*d*	*e*	*f*	*g*	*h*	*i*	*a*	*b*	*c*
5	*e*	*f*	*g*	*h*	*i*	*a*	*b*	*c*	*d*
6	*f*	*g*	*h*	*i*	*a*	*b*	*c*	*d*	*e*
7	*g*	*h*	*i*	*a*	*b*	*c*	*d*	*e*	*f*
8	*h*	*i*	*a*	*b*	*c*	*d*	*e*	*f*	*g*
9	*i*	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*

dimension of the alphabet, then the counting is iterated by starting from the first character of the alphabet.

We can determine the ciphered text using a table in which each row represents a sequence of characters of the alphabet, every time shifted by one position. For example, we can consider the alphabet of letters $\{a, b, c, d, e, f, g, h, i\}$ and the key string "cia". Let us consider that we wish to cipher the text char formed by the word "cache", which is compared with the key string repeated till to the length of the text obtaining the key char "ciaci". Table 9.1 shows the results of this method.

The word "cache" will be encripted in "eicad", which can be determined also using Table 9.2 in which the sequence of characters of each row is formed by performing a shifting by one position of the sequence of characters in the previous row.

The ith character in the encrypted text is identified in Table 9.2 considering the row starting with the character of the text to cipher and the column of the corresponding character in the repeated key string. For example, the first character of the encrypted word "eicad" is obtained in the cell corresponding to the row 3 (corresponding to the third symbol of the alphabet, the character c of the word "cache") and the column 3 (corresponding to the third symbol of the alphabet, the character c of the password "cia"). By using a decomposition of the image in blocks, we apply the above algorithm to a compressed block of sizes $h \times k$ by using a matrix of size $h \times k$ which represents the binary code of a watermarking key string. We set a text string and a password for obtaining the encrypted string with the Vigenère algorithm. The ith block of the compressed unmarked image will be applied to the matrix corresponding to the ith

character of the encrypted string. This character is converted into the binary form, adding, if necessary, some zeros as more significant bits with the Least Significant Bit Modification (LSBM) algorithm, so that the length of the character is equal to the size of the block. If the ith occurrence of the block is greater than the length of the encrypted string, the counting is iterated by starting from the first character of the encrypted string. The binary code of the character is performed by the choice of an alphabet of symbols. The compressed block G_B is calculated accordingly via the following formula based on the Lukasiewicz t-norm:

$$\underline{G}_B(p, q) = \min\left(1, G_B(p, q) + W_{BN}(p, q)\right) \tag{9.15}$$

where $\mathrm{W_{BN}}(p, q)$ is the binary matrix normalized corresponding to the binary code of the ith character of the watermarking string. Our process comprises the following steps:

1. We consider an alphabet of symbols, for example, an alphabet formed from the 256 ASCII characters. We set the password and a text forming the string to encrypt.
2. We apply the Vigenère algorithm to our string, thus obtaining an encrypted string. Each character of the encrypted string is converted in binary mode and is transformed in a binary matrix of the dimension of the compressed block, adding a "0" as more significant bit by the LSBM algorithm. This binary matrix is normalized by dividing the cell value per number of gray levels of the image -1 (for example, 255 if we use images with 256 gray levels).
3. The first compressed block of the image is modified using Eq. (9.15), where $\mathrm{W_{BN}}$ is the matrix corresponding to the first character of the encrypted string.
4. Return to step 3 to apply the watermark to the successive blocks of the compressed unmarked image. If the number of the blocks is greater than the length of the text string, we consider the first character of the encrypted string.
5. The process ends when all unmarked blocks in the compressed image are taken into consideration.

The following example effectively illustrates the above procedure for a generic block. Let us consider an original image which is divided in blocks of sizes 4×4, and let B (say) a block be given by

$$B = \begin{pmatrix} 155 & 142 & 142 & 161 \\ 149 & 138 & 135 & 156 \\ 129 & 135 & 142 & 166 \\ 132 & 133 & 128 & 170 \end{pmatrix} \tag{9.16}$$

Table 9.3 Values of coders

$A_p = B_q$	$i = j = 1$	$i = j = 2$	$i = j = 3$	$i = j = 4$
$A_p = B_q$	$i = j = 1$	$i = j = 2$	$i = j = 3$	$i = j = 4$
$A_1 = B_1$	0.978023	0.914947	0.573753	0.241177
$A_2 = B_2$	0.573753	0.914947	0.978023	0.700784
$A_3 = B_3$	0.165299	0.449329	0.818731	1.000000

which is normalized to the following fuzzy relation:

$$R_B = \begin{pmatrix} 0.607843 & 0.556863 & 0.556863 & 0.631373 \\ 0.584314 & 0.584314 & 0.529412 & 0.611765 \\ 0.505882 & 0.529412 & 0.556863 & 0.650980 \\ 0.527647 & 0.521569 & 0.501961 & 0.666667 \end{pmatrix} \tag{9.17}$$

The coders (9.3) with $k_B = h_B = 3$, $m_B = n_B = 4$ and $\alpha = 0.1$ assume the values given in Table 9.3.

Then we get the compressed fuzzy relation G_B of sizes 3×3 (Eq. (9.7)) given by

$$G_B = \begin{pmatrix} 0.585744 & 0.534764 & 0.620323 \\ 0.529793 & 0.534764 & 0.639931 \\ 0.506597 & 0.503795 & 0.666667 \end{pmatrix} \tag{9.18}$$

which is decompressed via Eq. (9.8), and we obtain the fuzzy relation of sizes 4×4 given by

$$D_B = \begin{pmatrix} 0.607843 & 0.588235 & 0.556863 & 0.631373 \\ 0.584314 & 0.615686 & 0.588235 & 0.662745 \\ 0.549020 & 0.584314 & 0.556863 & 0.650980 \\ 0.527647 & 0.545098 & 0.513725 & 0.666667 \end{pmatrix} \tag{9.19}$$

Denormalizing this matrix, we obtain that

$$C = \begin{pmatrix} 155 & 150 & 142 & 161 \\ 149 & 157 & 150 & 169 \\ 140 & 149 & 142 & 166 \\ 132 & 139 & 131 & 170 \end{pmatrix} \tag{9.20}$$

We mark the compressed unmarked block G_B using the process described above. For example, we assume to use the ASCII alphabet formed by the 256 symbols. We assume that the occurrence of the block (9.18) corresponds the character W, that is transformed into the matrix (9.13) and normalized into the matrix (9.14).

Fig. 9.1 Lena

We apply the operator (9.15) on the matrices (9.14) and (9.18), obtaining the following watermarked matrix:

$$\underline{G}_B = \begin{pmatrix} 0.585744 & 0.534764 & 0.624245 \\ 0.529793 & 534764 & 0.639931 \\ 0.510519 & 0.507717 & 0.670589 \end{pmatrix} \tag{9.21}$$

By utilizing the same coders of Table 9.3 and Eq. (9.10), we deduce that

$$\underline{D}_B = \begin{pmatrix} 0.607843 & 0.588235 & 0.556863 & 0.647059 \\ 0.584314 & 0.615686 & 0.588235 & 0.682353 \\ 0.549020 & 0.584314 & 0.600000 & 0.650980 \\ 0.568627 & 0.60000 & 0.572549 & 0.725490 \end{pmatrix} \tag{9.22}$$

to which corresponds the following denormalized matrix (signed image):

$$\underline{C} = \begin{pmatrix} 155 & 150 & 142 & 165 \\ 149 & 157 & 150 & 174 \\ 140 & 149 & 153 & 166 \\ 145 & 153 & 146 & 185 \end{pmatrix} \tag{9.23}$$

In Sect. 9.4 we show the results of our tests.

9.4 Results of Tests

We considered a sample of 1000 images of sizes 256×256 extracted from the well-known Corel Galery (Arizona directory) database. We take into consideration only two images, “Lena” (Fig. 9.1) and “House” (Fig. 9.2).

Fig. 9.2 House

Fig. 9.3 Lena unsigned, $\rho_B = 0.5625$

Fig. 9.4 Lena signed, $\rho_B = 0.5625$

In our tests we take the key string "Watermark", the string to be encrypted "To be or not to be" and an alphabet formed from the 256 ASCII characters. We consider several compression rates and the related PSNR for the unsigned and signed images. Initially the image R of Lena was divided in 4096 (= 64×64) blocks R_B.

We consider $m_B = n_B = 4$ and $k_B = h_B = 3$, hence each R_B (having sizes 4×4) is compressed to a block G_B of sizes 3×3 ($\rho_B = 9/1 = 0.5625$), and in turn decoded to a block D_B of sizes 4×4. The related recomposition gives the unsigned image D of Fig. 9.3. For the first unsigned compressed block we use the normalized

Fig. 9.5 Lena unsigned, $\rho_B = 0.3906$

Fig. 9.6 Lena signed, $\rho_B = 0.3906$

matrix (9.14). The processing order is $kN + 1$, with k being an integer and N the length of the key string.

Generally speaking, the matrix W is applied to the block with processing order $kN + i$. Each W is embedded in each block G_B with the LSBM algorithm, and we deduce a signed block $\underline{G}_B$ of sizes 3×3, which is decoded to a block $\underline{D}_B$ of sizes 4×4. These blocks $\underline{D}_B$'s are recomposed forming the signed image $\underline{D}$ of Fig. 9.4.

We apply the same procedure with $m_B = n_B = 8$ and $k_B = h_B = 5(\rho_B = 25/64 = 0.3906)$ by getting the unsigned (resp., signed) image of Fig. 9.5 (resp., Fig. 9.6). In Table 9.4 we give the PSNR values for other compression rates. A detailed examination of the PSNR for "Lena" shows clearly the fact that the PSNR of the unsigned images is very close to the PSNR of the signed images for low ρ_B values.

This difference is more evident if ρ_B tends to 1 as the plot of Fig. 9.7 shows clearly. For sake of completeness, we show the analogous plot of the image "Lena" from [7] in Fig. 9.8, where we use a watermarking binary constant matrix.

Hence the use of watermarking variable matrices applied to compressed blocks does not essentially change the trend of the PSNR with respect to the compression rate. Similar results are also obtained for the image "House" given in the unsigned Fig. 9.9 (resp., Fig. 9.10) and the signed Fig. 9.11 (resp., Fig. 9.12) for $\rho_B = 0.5625$ (resp. $\rho_B = 0.3906$), by always using the same technique. We obtain similar results for other images from Corel Gallery, but they are not given here because of the brevity of presentation.

Table 9.4 Values of the PSNR of "Lena" for some values of ρ_B

$m_B \times n_B$	$k_B \times h_B$	$\rho_B = k_B \times h_B / m_B \times n_B$	PSNR of unsigned image	PSNR of signed image
4×4	3×3	0.5625	29.7000	26.6317
6×6	4×4	0.4444	22.8900	20.5729
8×8	5×5	0.3906	21.7441	20.1971
5×5	3×3	0.3600	20.8900	19.6104
7×7	4×4	0.3265	20.2128	19.0113
8×8	4×4	0.2500	17.8455	16.8902

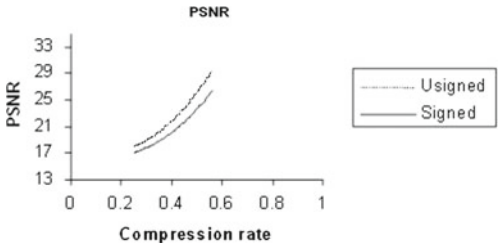


Fig. 9.7 Values of the PSNR at several values of ρ_B for "Lena"

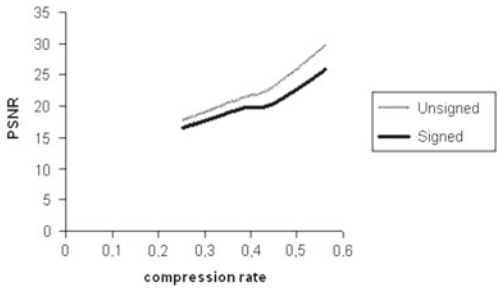


Fig. 9.8 Values of the PSNR at several values of ρ_B for "Lena" from [7]

Fig. 9.9 House unsigned, $\rho_B = 0.5625$

Fig. 9.10 House signed, $\rho_B = 0.5625$

Fig. 9.11 House unsigned, $\rho_B = 0.3906$

Fig. 9.12 House signed, $\rho_B = 0.3906$

9.5 Concluding Comments

In contrast with the work performed in [7], where the authors used a constant binary watermarking matrix for each compressed block, in this work this matrix is variable and is settled to the binary value corresponding to an ASCII character. This character is determined using the well-known encrypting alphabetic text Vigenère algorithm starting with an initial string. We showed that the signed image $\underline{D}$ is very similar to the unsigned image D for low values of the compression rate. This means that this method does not affect the quality of the signed decoded image, but it clearly gives a significant security with respect to the method of [7].

References

1. Barni, M.: Improved wavelet-based watermarking through pixelwise making. IEEE Trans. Image Process. **10**(5), 783–791 (2002)
2. Chang, C.C., Chen, Y.H., Lin, C.C.: A data embedding scheme for color images based on genetic algorithm and absolute moment block truncation coding. Soft Comput. **13**, 321–331 (2009)
3. Chen, W.C., Wang, M.S.: A fuzzy *c*-means clustering-based fragile watermarking scheme for image authentication. Expert Syst. Appl. **36**, 1300–1307 (2009)
4. Davarynejad, M., Ahn, C.W., Vrancken, J., Van den Berg, J., Coello, C.A.: Evolutionary hidden information detection by granulation based fitness approximation. Appl. Soft Comput. **10**, 719–729 (2010)
5. Di Martino, F., Loia, V., Sessa, S.: A method for coding/decoding images by using fuzzy relation equations. In: Bilgic, T., De Baets, B., Kaynak, O. (eds.) Proceedings of 10th IFSA World Congress, LNAI 2715, pp. 436–441. Springer, Heidelberg (2003)
6. Di Martino, F., Loia, V., Sessa, S.: A method in the compression/ decompression of images using fuzzy equations and fuzzy similarities. In: Proceedings of 10th IFSA World Congress, pp. 524–527. Istanbul (2003)
7. Di Martino, F., Sessa, S.: Digital watermarking in coding/decoding processes with fuzzy relation equations. Soft Comput. **10**, 238–243 (2006)
8. Di Nola, A., Pedrycz, W., Sanchez, E., Sessa, S.: Fuzzy Relation Equations and Their Applications to Knowledge Engineering. Kluwer Academic Publishers, Dordrecht (1989)
9. Gottwald, S., Pedrycz, W.: Solvability of fuzzy relational equations and manipulation of fuzzy data. Fuzzy Sets Syst. **18**(1), 45–65 (1986)
10. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Kluwer Academic Publishers, Dordrecht (2000)
11. Loia, V., Pedrycz, W., Sessa, S.: Fuzzy relation calculus in the compression and decompression of fuzzy relations. Int. J. Image Graph. **2**, 1–15 (2002)
12. Loia, V., Sessa, S.: Fuzzy relation equations for coding/decoding processes of images and videos. Inf. Sci. **171**, 145–172 (2005)
13. Mahdian, B., Saic, S.: A bibliography on blind methods for identifying image forgery. Sign. Process. Image Commun. **25**, 389–399 (2010)
14. Nobuhara, H., Pedrycz, W., Hirota, K.: Fast solving method of fuzzy relational equations and its application to lossy image compression. IEEE Trans. Fuzzy Syst. **8**(3), 325–334 (2000)
15. Nobuhara, H., Pedrycz, W., Hirota, K.: A digital watermarking algorithm using image compression method based on fuzzy relational equations. In: Proceedings of FUZZ-IEEE 2002, vol. 2, pp. 1568–1573. IEEE Press, New York (2002)

Chapter 10
Study on Human Brain Registration Process Using Mutual Information and Evolutionary Algorithms

Mahua Bhattacharya and Arpita Das

Abstract The registration of brain images is required to facilitate the study of brain mapping, treatment planning, and image-guided therapies of nervous system. In the present work a similarity measure has been implemented for affine multimodality (MR and CT) image registration of sections of the human brain. In addition, a similarity measure is built on both intensity and gradient-based images. In the present work, the region of interest (ROI), the ventricular region, is segmented using the fuzzy c-means clustering technique. The deformation or change of shape of the ventricular region captures the process of degeneracy and other abnormality in tissue regions of the human brain. The similarity metric should be maximal when two images are perfectly aligned. The genetic algorithm-based search strategy has been utilized to optimize the registration process.

10.1 Introduction

The process of registration [2–6, 8–10, 12–14, 16] has great importance for multimodal medical image processing for clinical interest. The registration is based on the transformation of one image with respect to another.Image registration should

M. Bhattacharya (✉)
Department of Information Communication Technology,
ABV-Indian Institute of Information Technology and Management,
Morena Link Road, Gwalior 474010, India
e-mail: mb@iiitm.ac.in

A. Das
Institute of Radio Physics and Electronics,
University of Calcutta, 92, A.P.C Road,
Calcutta 700009, India
e-mail: arpita.rpe@caluniv.ac.in

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_10, © Springer-Verlag Berlin Heidelberg 2013

address the differences in acquisition parameters among different sets of images. These parameters are different due to the size of the images, the position of the camera, different viewing angles, etc. The images to be registered may be of the same modality but taken at different times or of different modalities. During this process, one image is fixed (the reference image) and the other (the floating image) is transformed so that it becomes similar to the reference image [6, 12, 13]. Registration is used to describe the geometric transformations such that the generated image should be registered or aligned with some standard or reference image.

Medical imaging provides insights into the size, shape and spatial relationships among anatomical structures. In radiotherapy planning, dose calculation is based on the computed tomography (CT) data, while tumor outlining is often better performed in the corresponding magnetic resonance imaging (MRI). Functional imaging like single-photon computed tomography (SPECT) and Positron emission tomography (PET) is becoming increasingly important in medical research. PET and SPECT imaging provide information on blood flow, glucose intake and different other metabolic processes. When patients are to undergo brain surgery, both CT and MR images of the brain are used to help the surgeon. Registration of preoperative and postoperative images in surgical interventions and treatment monitoring is another developing application field.

The registration of brain images is required to facilitate the study of brain mapping, treatment planning and image guided therapies of nervous system. We have already done experiments on registration using MR (T1 and T2 weight) and CT imaging modalities of the ventricular region of the brain as the region of interest (ROI) for patients having Alzheimer's disease using a shape-theoretic approach [2]. The control points on the concavities present in the contours are chosen to reproject ROI from the respective modalities in a reference frame.

In our present work, a similarity measure has been computed for affine multimodality (MR and CT) image registration of section of human brain. In addition, the similarity measure is built on intensity and gradient-based images. In this chapter, the region of interest (ROI), which is the ventricular region, is segmented by using fuzzy c-means clustering technique. The deformation or change of shape of the ventricular region captures the process of degeneracy and other tissue abnormalities in region of the human brain. In our earlier work [2] the landmark-based registration of multimodality brain images was accomplished and which is semiautomatic in nature. In present work, the registration process proposed is automatic and is based on the measure of the similarity metric [11, 15, 16]. This should be maximal when two images are perfectly aligned. Since, similarity metric is a non convex function and contains many local optima, the choice of search strategy to optimize it is important in the registration problem. Presently we have implemented a genetic algorithm-based optimization scheme. Results are presented for registration of 2D clinical images. The robustness of the proposed method is presented by choosing a realistic, practical transformation technique and the metaheuristic search strategies.

10.2 Proposed Method for Medical Image Registration

A prior step for the process of information fusion for multimodality imaging is image registration. After the process of registration, the fusion is required in order to display the integrated images. When the region of interest (ROI) of any diseased part of a human body is captured by different imaging sensors (like CT, MR, PET, SPECT, USG) it is desirable to establish the point-to-point correspondences and finally to match the relevant multimodal images of the ROI. Our present approach for 2D registration of different modality medical images is based on similarity measure of both intensity and gradient-based information. Incorporation of edge information along with gray-level intensities improves the evaluation parameter. The correlation metric [11, 15, 16] and mutual information [1, 3, 6, 17] should be maximal when the two different images are perfectly aligned or registered. Present approach for registration exploits the natural phenomena based genetic optimization algorithm [6, 7, 17].

In the present approach we discuss intensity-based 2D/2D registration of same-modality (MR) brain images and gradient-based registration of different modalities (MR and CT) brain images. The fuzzy c-means clustering technique was used to segment the ROI (ventricular region), and, gradient is obtained using the Sobel mask. Figure 10.1 demonstrates an overview of the present work. Our present work incorporates the gradient information along with intensity distributions to improve the registration accuracy. Since margins/edges of the image carry vital information, enhancement of gradient produces significant improvement in the evaluation parameter. During the registration procedure, one image plays the role of reference image, and the second one, is called the floating image. Floating image tries to geometrically align itself with the reference image. Mutual information (MI) and correlation functions (CT) measure the similarity between reference and aligned floating images. To maximize this non convex similarity metric function, a genetic optimization algorithm is utilized. Initial population of the genetic algorithm (GA) starts with randomly chosen affine transformation of floating images. It maximizes the non convex similarity function by its selection, crossover and mutation operators. Both mutual information and a correlation-based similarity measuring function provide better matching parameters for the image registration process using gradient-based methodology.

The deformation or change of shape of the ventricular region captures the process of degeneracy on the human brain. The transformation applied to register the images can be categorized according to the degree of freedom. In present work, we define a non rigid affine transformation. It extends the degrees of freedom of rigid transformation with a scaling factor in each dimension and additionally a shearing in each dimension.

The registration function is generally not a smooth function, and contains many local maxima. The local maxima are due to the interpolation techniques required to estimate the gray values when transforming the points from one image to another. The presence of local maxima hampers the optimization process. Because of this,

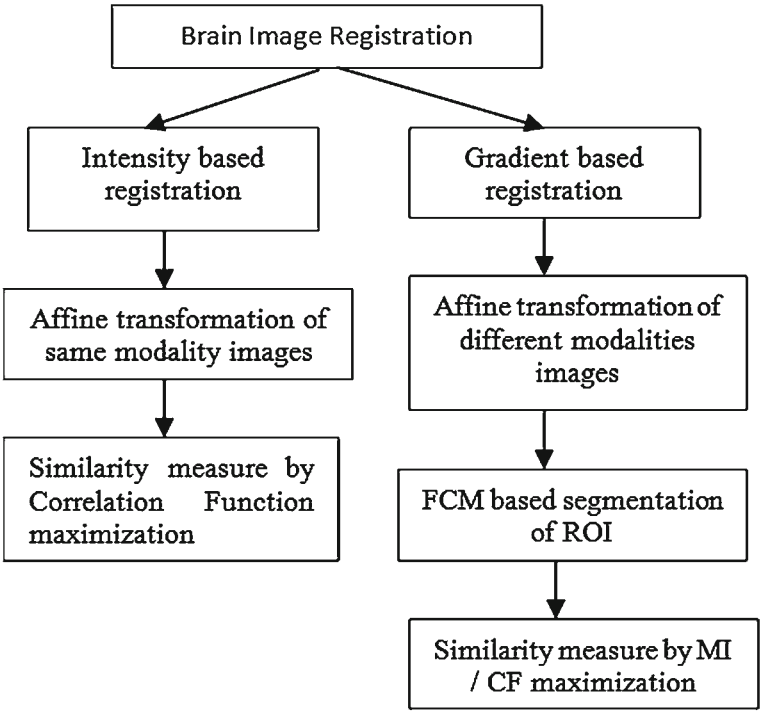


Fig. 10.1 Brief overview of the proposed work

the choice of optimization routine has a large influence on the results of the registration method, particularly on the robustness of the method with respect to the initial transformation. A popular method of optimization used in image registration is genetic algorithms (GAs), which are based on the survival-of-the-fittest principle and selecting the best of the new generation.

10.2.1 Fuzzy C-means Clustering for Segmentation of ROI

In present work we have implemented Fuzzy C-means clustering algorithm for intensity-based segmentation of regions of interest (ROI) as a prior step for registration of different modality imaging of sections of the human brain using CT and MR modalities. In the Fuzzy C-means clustering algorithm used for intensity based segmentation, the total number of fuzzy cluster centers chosen are shown in Fig. 10.2. Cluster center A represents the normal brain tissue. Second cluster B represents the shadows of the ventricular region, and C represents the actual ventricular region.

The ultimate Fuzzy partition membership functions are shown in Fig. 10.2, which shows that there is an overlap between the membership functions A, B and C.

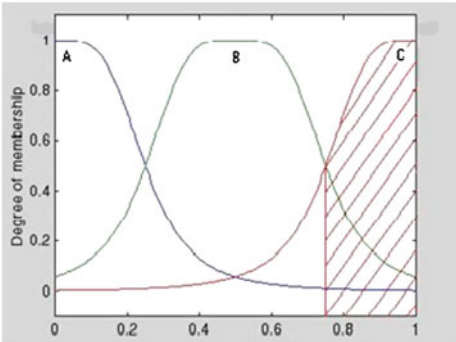


Fig. 10.2 Final fuzzy partition membership functions

In this section we have described how the decision has been made. If the possibility for a region that belongs to the ventricular region is greater than 50 % (i.e., the membership value of curve $C > 0.5$), the decision is that the region belongs to the actual ventricular region. Thus according to our decision rule, the shaded region of Fig. 10.2 is under the ventricular region or ROI in the present context.

Algorithm:

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of given data. A fuzzy c-partition of X is a family of fuzzy subsets of X, denoted by $P = \{A_1, A_2, \ldots, A_c\}$, which satisfies

$$\sum_{i=1}^{c} A_i(x_k) = 1$$

for all $k \in N_n$ and

$$0 < \sum_{k=1}^{n} A_i(x_k) < n$$

for all $i \in N_c$, where c is a positive integer.

Given a set of data $X = \{x_1, x_2, \ldots, x_n\}$, where x_k, in general, is a vector for all $k \in N_n$, the problem of fuzzy clustering is to find a fuzzy pseudopartition and the associated cluster centers by which the structure of the data is represented as well as possible. To solve the problem of fuzzy clustering, we need to formulate a performance index. Usually, the performance index is based upon cluster centers, $v_1, v_2, \ldots, v_c$, associated with the partition, calculated by the formula given below

$$v_i = \frac{\sum_{k=1}^{n} [A_i(x_k)]^m x_k}{\sum_{k=1}^{n} [A_i(x_k)]^m} \tag{10.1}$$

for all $i \in N_c$, where $m > 1$ is a real number that governs the influence of membership grades. Observe that the vector v_i calculated by Eq. 10.1 is viewed as the cluster center of the fuzzy class A_i and is actually the weighted average of data in A_i.

The performance index of a fuzzy pseudopartition P, $J_m(P)$, is defined in terms of the cluster centers by the formula

$$J_m(P) = \sum_{k=1}^{n} \sum_{i=1}^{c} [A_i(x_k)]^m \|x_k - v_i\|^2 \tag{10.2}$$

where $\|x_k - v_i\|^2$ represents the distance between x_k and v_i. Clearly, the smaller the value of $J_m(P)$, the better the fuzzy pseudopartition P. Thus, the goal of the fuzzy c-means clustering method is to find a fuzzy pseudopartition P that minimizes the performance index $J_m(P)$.

10.2.2 Affine Transformation

An affine transformation with six degrees of freedom is used to correct calibration errors in the pixel dimensions. It is the most general linear transformation on an image:

$$x' = fx + gy + h \tag{10.3}$$

$$y' = qx + ry + s \tag{10.4}$$

in transposed matrix notation:

$$[x'y'] = [xy1]T \tag{10.5}$$

where T is a $3W2$ matrix of coefficients:

$$T = [fq; gr; hs] \tag{10.6}$$

An affine transformation takes any coordinate system in a plane into another coordinate system that can be found of translation, rotation, scaling and shearing.

10.2.3 Computation of Objective Function

Mutual Information

In medical imaging the difficulty arises since images are taken from several different devices, e.g., CT, MR imaging, or ultrasound scanners. Thus, their intensities cannot be taken directly to measure the image similarity. Recent studies show that, for successful image registration in a multimodal situation the mutual information of the images will be optimum. Mutual information (MI) is a basic concept from informa-

tion theory which measures the statistical dependence between two variables or the amount of information that one variable contains about the other. It has been shown that mutual information is a robust similarity measure for multimodal registration and does not depend on the specific dynamic range or intensity scaling of the images. The mutual information of two images is a combination of the entropy values of the images, both separately and jointly. The entropy of an image can be computed by estimating the probability distribution of the image intensities, The maximum entropy value is reached for a uniform distribution of intensities of two images. In present work, we use the Shannon measure of entropy. The joint probability distribution of two images is estimated by computing a normalized joint histogram of the gray values. The definition of the mutual information of two images A and B combines the marginal entropies, $p_A(a)$ and $p_B(b)$, and joint entropy $p_{AB}(a, b)$ of the images in the following manner:

$$I(A, B) = \sum_{a,b} p_{AB}(a, b) \log \frac{p_{AB}(a, b)}{p_A(a) p_B(b)} \tag{10.7}$$

MI is related to entropy by the equation

$$I(A, B) = H(A) + H(B) - H(A, B) \tag{10.8}$$

with H(A) and H(B) being the marginal entropies of A and B, respectively, and H(A, B) their joint entropy.

$$H(A) = -\sum_{a} p_A(a) \log p_A(a) \tag{10.9}$$

$$H(A, B) = -\sum_{a,b} p_{AB}(a, b) \log p_{AB}(a, b) \tag{10.10}$$

As the optimization technique in registration procedure is used to maximize the similarity metric, MI or Correlation value plays the role of objective function, and its maximum value is achieved for correctly registered images.

Correlation Function

The proposed correlation-based registration method is described below: the correlation of two images $f_1(x, y)$, and $f_2(x, y)$, of size $M \times N$ is defined as

$$f_1(x, y), of_2(x, y) = (M-1)(N-1)\left(\frac{1}{MN}\right) \times \sum_{m=0} \sum_{n=0} f_1^*(m, n) \times f_2(x+m, y+n) \tag{10.11}$$

where f_1^* denotes the complex conjugate of f_1. Here we deal with real images, in which case $f_1^* = f_1$.

The principle use of correlation is for similarity measurement. In image registration, $f_1(x, y)$ is denoted as the reference object and $f_2(x, y)$ is the affine-transformed

version of $f_1(x, y)$. Then, if there is a match between $f(x, y)$ and $h(x, y)$, the correlation of the two images will be maximum.

10.2.4 Genetic Algorithm-Based Optimization

To maximize the non convex similarity metric function a *genetic optimization algorithm* is utilized. Initial population of *genetic algorithm* (GA) starts with the randomly chosen affine transformation of floating images. GAs are search algorithms based on the mechanics of natural selection and natural genetics. A possible solution is represented as a chromosome in a string structure with each element representing one parameter in the solution. A collection of possible chromosomes forms a population, which produces another generation through a search process. The search process adopts "the fittest survives" rule after a structured yet randomized information exchange within the existing generation to yield a new generation. GAs are not just simple random walks; they efficiently exploit the information to speculate on new search points with expected improved performance. This method is allowed to escape from local optima, and the chromosomes will approach the global optimum. To apply GA in our registration problem, we encode the transformation matrix and optimize these six parameters to achieve the best possible result.

In the registration problem let us consider that the multimodality images to be registered are *Image A* and *Image B*. Say*Image A* is the reference image, and *Image B* is the image transformed by an affine transformation technique such that it will be correctly registered with *Image A*. Now for 2D affine transformation, six parameters are required to transform an image as,

$$T = \begin{bmatrix} fgh \\ qrs \end{bmatrix}$$

These parameters are optimized by GA, so that the image formed with optimized parameters is perfectly registered with the reference image (*Image A*).

Optimization Algorithm:

1. Generate ten (between 0 to 1) random numbers for each parameter f, g, h, q, r, s. With these new sets of values of f, g, h, q, r, s each, transform the *Image B* using affine transformation and achieve ten new transformed images.
2. Calculate the similarity metric (SM) between the reference image (*Image A*) and above-obtained ten new transformed images (denoted as *Image B*(*T1*)). These ten values of SM act as the fitness/objective function for further implementation of GA. Find the maximum SM value and denote it as *max SM1*.
3. Now encode the values of f, g, h, q, r, s by simple binary numbers to form the population of chromosomes in GA.
4. Perform three genetic operations: reproduction, crossover and mutation for each encoded parameters.

Table 10.1 Similarity measure for intensity and gradient based registration of human brain images

Pair of images	Mode of registration	CF measure	MI measure
MR T2 versus CT	Intensity based registration	0.3687	1.5065
	Intensity +gradient based registration	0.3733	1.5076
CT versus MR T2	Intensity based registration	0.2675	1.3188
	Intensity +gradient based registration	0.2734	1.3231
CT versus MR T1	Intensity based registration	0.2251	1.3575
	Intensity +gradient based registration	0.2257	1.3597
MR T1 versus CT	Intensity based registration	0.5677	1.4080
	Intensity +gradient based registration	0.5680	1.4128

5. After mutation, a new generation of chromosomes is formed. These chromosomes are now decoded to obtain the new values of f, g, h, q, r, s.
6. Transform *Image B* using affine transformation with the new set of values of f, g, h, q, r, s found from the new generation of GA (let us denote these images as *Image B*(*T2*)).
7. Calculate ten new values of SM between the reference image (Image A) and the transformed image B(T2). Among these ten SM values find the maximum SM value and denote it as *max SM2*.
8. If the difference between *max SM1* and *max SM2* is less than a predefined threshold value (T), stop iteration; otherwise replace the value of *max SM1* by *max SM2* and go to step 4.

10.3 Experimental Results

In the present research, the optimal transformation parameters of matrix T are searched for by using GAs. The chromosomes are formed by concatenating six binary coded parameters of matrix T. The number of bits should be chosen as small as possible to minimize the time of convergence of the GA. In the present problem, the parameters are assigned ± 7 units each. Therefore four bits are assigned for each parameter. We start with GA population of 30 initially, and after the 3rd generation, the optimal solution is achieved.

Presently we have registered the ventricular region as the ROI of the brain images from CT and MR modalities. The present approach of registration has been implemented on same modality (MR) brain images (Fig. 10.3) and on different modalities (MR and CT) brain images (Fig. 10.4). The process evaluates correlation matching combined with the GA optimization technique and also measures the efficiency and robustness of the process.

Table 10.1 demonstrates the results of similarity measure on the intensity and intensity-gradient registration process using mutual information and correlation measure.

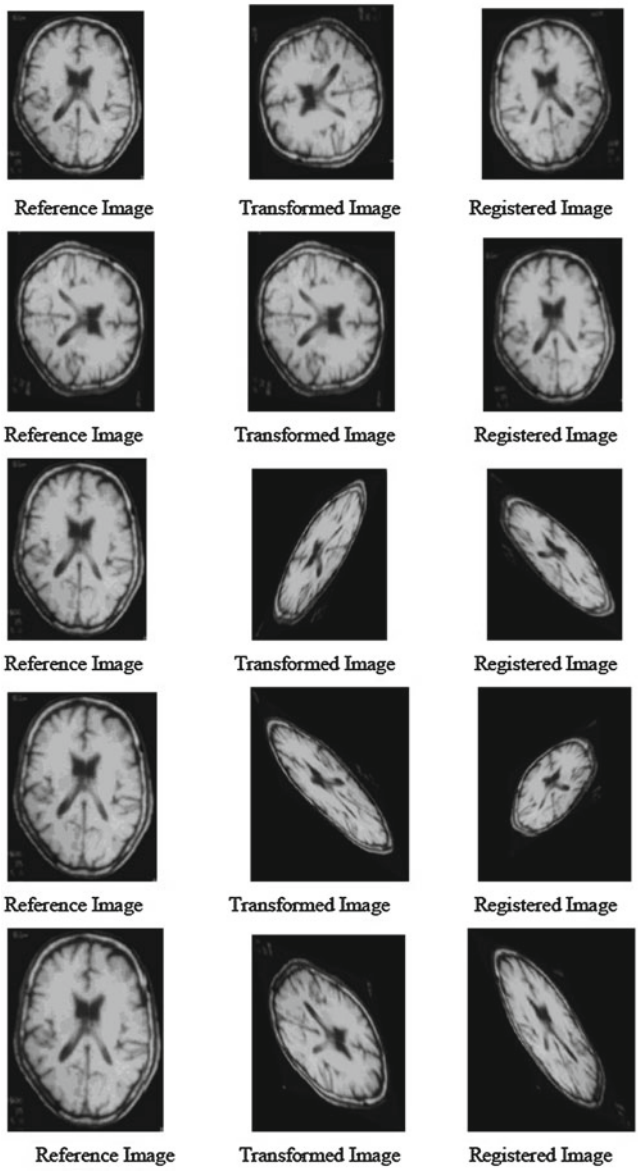


Fig. 10.3 Intensity-based registration of same-modality MR images

10.4 Conclusion

In this chapter, we presented an efficient similarity-based image registration method combining a GA search technique for non-rigid affine transformation using both

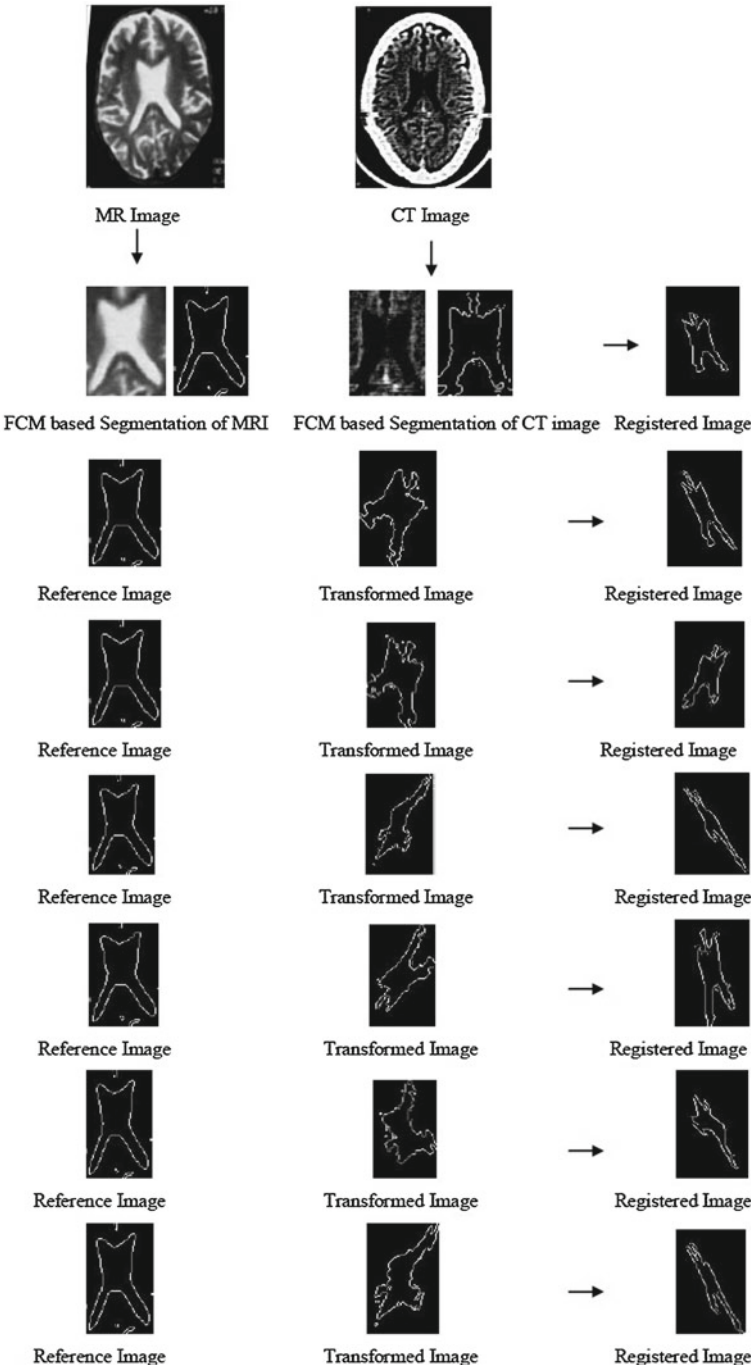


Fig. 10.4 FCM Gradient based registration of MR and CT images of section of human brain

intensity and gradient-based techniques. To overcome the influence of the existence of local maxima, we have adopted the mutation probability of 0.10, which improves the convergence ability of GAs. Experiments have shown that our algorithm can yield good results. Moreover, our experimental results have shown accuracy for both intensity-based matching and FCM gradient-based matching of ventricular regions. Future research will consider using other efficient optimization techniques. The process of GA-based optimization is time consuming and our future challenges lie in how to accelerate the process substantially without loss of accuracy.

Acknowledgments The authors would like to thank Dr. S. K. Sharma, Director, EKO Imaging and X-Ray Institute, Kolkata and All India Institute of Medical Sciences, New Delhi.

References

1. Bhattacharya, M., Dutta Majumder, D.: Multi resolution medical image registration using mutual information and shape theory. In: Proceedings of Fourth International Conference on advances in Pattern Recognition and Digital, Techniques, ICAPRDT '99 (IAPR), pp. 175–178 (1999)
2. Bhattacharya, M., Dutta Majumder, D.: Registration of multimodal images of Alzheimer's patient: a shape theoretic approach. Pattern Recognit. Lett. **21**(6–7), 531–548 (2000)
3. Butz, T., Thiran, J.-P.: Affine registration with feature space mutual information. In: Niessen, W.J., Viergever, M.A. (eds.) Medical Image Computing and Computer-Assisted Intervention. Lecture Notes in Computer Science, vol. 2208, pp. 549–556. Springer, Berlin (2001)
4. Comeau, R.M., Sadikot, A.F., Fenster, A., Peters, T.M.: Intraoperative ultrasound for guidance and tissue shift correction in image-guided surgery. Med. Phys. **27**(4), 787–800 (2000)
5. Davatzikos, C., Prince, J.L., Bryan, R.N.: Image registration based on boundary mapping. IEEE Trans. Med. Imaging **15**(1), 112–115 (1996)
6. Das, A., Bhattacharya, M.: Affine based registration of CT and MR modality images of human brain using multiresolution approaches: comparative study on genetic algorithm and particle-swarm optimization. Neural Comput. Appl. (Springer London) **20**(2), 223–237 (2011)
7. Dasgupta, D., McGregor, D.R.: Digital image registration using structured genetic algorithms. In: Proceeding of the SPIE, vol. 1766, pp. 226–234 (1992)
8. Fright, R.W., Linney, A.D.: Registration of 3-D head surfaces using multiple landmarks. IEEE Trans. Med. Imaging **12**(3), 515–520 (1993)
9. Hill, D.L.G., Hawkes, D.J., Hussain, Z., Green, S.E.M., Ruff, C.F., Robinson, G.P.: Accurate combination of CT and MR data of the head: validation and applications in surgical and therapy planning. Comput. Med. Imaging Graph. **17**(4/5), 357–363 (1993)
10. Johansson, M.: Image registration with simulated annealing and genetic algorithms. Master's Thesis in Computer Science at the School of Computer Science and Engineering, Royal Institute of Technology (2006)
11. Junck, L., Moen, J.G., Hutchins, G.D., Brown, M.B., Kuhl, D.E.: Correlation methods for the centering, rotation, and alignment of functional brain images. J. Nucl. Med. **31**, 1220–1276 (1990)
12. Maintz, J.B.A., Viergever, M.A.: A survey of medical image registration. Med. Image Anal. **2**(1), 1–36 (1998)
13. Roche, A., Pennec, X., Malandain, G., Ayache, N.: Rigid registration of 3-D ultrasound with MR images: a new approach combining intensity and gradient information. IEEE Trans. Med. Imaging **20**(10), 1038–1049 (2001)
14. Ritter, N., Owens, R., Cooper, J., Eikelboom, R.H., van Saarloos, P.P.: Registration of stereo and temporal images of the retina. IEEE Trans. Med. Imaging **18**(5), 404–418 (1999)

15. Roche, A., Malandain, G., Pennec, X., Ayache, N.: The correlation ratio as a new similarity measure for multimodal image registration. In: First International Conference on Medical Robotics, Imaging And Computer Assisted Surgery (MICCAI'98), ser. Lecture Notes in Computer Science, vol. 1496, pp. 1115–1124. Springer, Cambridge, MA (1998)
16. van den Elsen, P.A., Maintz, J.B.A., Pol, E.J.D., Viergever, M.A.: Automatic registration of CT and MR brain images using correlation of geometrical features. IEEE Trans. Med. Images **14**(2), 384–398 (1995)
17. Zhang, H., Zhou, X., Sun, J., Zhang, J.: A novel medical image registration method based on mutual information and genetic algorithm. In: IEEE Proceedings of the Computer Graphics, Imaging and Vision: CGIV, pp. 221–226 (2005)

Chapter 11
Use of Stochastic Optimization Algorithms in Image Retrieval Problems

Mattia Broilo and Francesco G. B. De Natale

Abstract In this chapter, the use of stochastic and evolutionary optimization techniques for image retrieval is addressed. We provide background and motivations of such approaches, as well as an overview of some of the most interesting ideas proposed in the recent literature in the field. The relevant methodologies refer to different applications of the optimization process, as a way of either improving the parameter setting in traditional retrieval tools, or directly classifying images within a dataset. Also, the use of stochastic optimization approaches based on social behaviors is discussed, showing how these approaches can be used to capture the semantics of a query through the interaction with the user. Strengths and weaknesses of different solutions are discussed, taking into account also implementation issues, complexity, and open directions of the research.

11.1 Introduction

Content-based image retrieval (CBIR) refers to any technology that in principle helps to organize digital picture archives by their visual content [1]. The year 1992 is considered the starting point of research and development on image retrieval by content [2]. The last two decades have witnessed great interest in research on CBIR. This has paved the way for a large number of new techniques and systems, and a growing interest in associated fields to support such systems. CBIR is a field of research which presents many facets and involves numerous unresolved issues. Despite the effort made in these years of research, there is not yet a universally accepted algorithmic way of characterizing the human perception of the relevant content of an image

M. Broilo · F. G. B. De Natale (✉)
DISI University of Trento, Via Sommarive 14, Trento, 38123 Povo, Italy
e-mail: denatale@ing.unitn.it

M. Broilo
e-mail: broilo@disi.unitn.it

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_11, © Springer-Verlag Berlin Heidelberg 2013

or, even more difficult, interpreting it. The main technical problems encompassed by CBIR are: how to mathematically describe an image (visual signature), how to assess the similarity between two descriptions (similarity metric), how to retrieve the desired content (search paradigm), how and what to learn from content or users (learning and classification). All these issues can be referred to as the *semantic gap*, that is, the gap between the subjective semantic meaning of a visual query and the numerical parameters extracted and analyzed by a computer [3]. Beyond the techniques adopted, the two key aspects of a content-based system are the purpose and the domain of the application. It is possible to simplify the content-based application types according to two main tasks: *search*, which covers retrieval by association, target, or category search; and *annotation*, which includes face and object detection and recognition, and all the different levels in concept detection (from lower to higher semantic abstraction). Understanding the nature and scope of image data plays a key role in the complexity of image search system design. Along this dimension, it is possible to classify content-based application domains into two main categories: consumer (e.g., *personal photo collections* and *social media*), and professional (e.g., content coming from specific domains such as *biomedical*, *satellite*, or *arts* image databases).

Many researchers agree that CBIR remains well behind content-based text retrieval [4–6] which is mainly due to three unresolved problems:

Understanding the semantics of media. The semantic interpretation of an image is still out of reach of current technologies. Significant efforts have been spent on using low-level image properties such as the statistics of the pixel values to detect concepts [7]. Nevertheless, from simpler methods, such as color and texture histograms, to more sophisticated ones, such as global transforms or SIFT [8], no ultimate approach has been found to reliably discover the user's perceived meaning of an image.

Scalability. The variety of visual concepts and their possible interpretations are enormous, thus calling for richer descriptions. Nevertheless, pattern classification studies demonstrate that increasing the number of features can be detrimental in a classification problem [9]. This problem is also known as the curse of dimensionality. Ideally, images in a given semantic category should be projected in nearby points in the feature space. If the number of samples is small compared to the dimension of the space, then it becomes possible to find rules to associate the feature sets of "similar" images. But when new samples or new categories are added, it is unlikely that such an association will be confirmed [10], thus bringing generalization and scalability lacks in classifiers [11].

Context and personalization. Each user is unique and his interaction with a system involves a number of human factors related to psychological affects, contextualization, personal experience, etc. Consequently, if the same photo is given to different people and they are asked to assign tags to represent that photo, there may be as many different tags as the number of people assigning them. Also, if the same photo is given to the same person at different times and in different contexts and situations, then the assigned tags could change.

The knowledge of the context and of the user profile, if available, would then be important to improve the capabilities of retrieval systems, although there is not yet a clear idea on how to represent and use such information [12].

As it can be seen, image retrieval deals with very complex problems, requiring sophisticated models that usually involve high-dimensional feature spaces and non linear objective functions (according to some similarity metrics). According to these models, the solution of a retrieval problem can be recast into an optimization problem, where the similarity of query and target can be associated to a fitness function, to be maximized within a given feature space. Such formalization makes very attractive the use of stochastic optimization algorithms, for their capability to find near-optimum solutions where linear programming and dynamic programming techniques usually fail or remain stuck in local minima [13]. Among other stochastic optimization techniques, evolutionary algorithms (EAs) represent a very interesting family of search methods based on the metaphor of the natural biological evolution and/or the social behavior of species. A comprehensive comparison of these stochastic algorithms could be found in [14]. In general, EAs share a common approach: First, the problem should be suitably modeled to fit the desired optimizer; then, the evolutionary search algorithm is applied to get a near-optimum solution. A number of *no free lunch* theorems are presented in [15], where it is established that if a given approach provides a high performance over a class of problems, it will be much less effective over another class. These theorems are associated with the fact that each optimization algorithm can be given a geometric interpretation, which typically matches a specific class of problems. When modeling a specific problem it is therefore important to identify the most suitable algorithm.

When dealing with content-based retrieval, evolutionary approaches based on social behaviors appear to be particularly interesting, as they are implicitly targeted to follow multiple or clustered sets of solutions in complex feature spaces. The general idea is, in fact, to mimic the behavior of individuals and groups, which can span over the solution space and reach different targets. Examples include how ants find the shortest route to a source of food [16], and how bees find their destination during flights. The behavior of such species is guided by learning, adaptation, and evolution [17]. To mimic the efficient behavior of these species, various researchers have developed computational systems that seek fast and robust solutions to complex optimization problems.

In the following, the use of stochastic and evolutionary optimization techniques for image retrieval problems is addressed, making special reference to such classes of methods, and in particular to particle swarm optimization (PSO) [18]. We will describe the different ways this algorithm can be used to solve typical problems in content-based image retrieval. Nowadays, the application of PSOs in CBIR can be roughly classified into three different approaches: optimization of the parameters (or weights), optimization of the classification, and optimization of the image search. After a brief introduction on the general principles of PSO, we provide a detailed description of these three approaches, and how the stochastic optimization can help in finding better solutions as compared to other techniques.

A short overview of possible alternative methodologies making use of different stochastic approaches such as Ant Colony [16] and Genetic Algorithms [19] within the same application domain is also provided in Sect. 11.6.

11.2 Particle-Swarm Optimization Principles

Particle-swarm optimization (PSO) was originally developed by Kennedy and Eberhart in 1995, and is inspired by the social behavior of swarms of bees [18]. PSO has been applied to a large number of domains for solving different optimization problems [20]. In PSO, the bees in a swarm are represented as particles. These particles are considered to be "flying" through a multidimensional feature space, looking for the optimal solution [21]. The location of a particle $\underline{p}_n$ is a point in the multidimensional space that models the problem under investigation, and represents a possible solution for that problem. PSO is an iterative algorithm that monitors at every iteration k the position of the particles in order to push the "bees" towards the optimal solution. In order to achieve this goal, the solution of each particle $\underline{p}_n$ is evaluated by a *fitness function* $\Psi\left(\underline{p}_n\right)$, which provides a quantitative value of the solution goodness. This fitness function is tuned and ad hoc designed by the engineers, according to the specific problem. The velocity $\underline{v}_n^k$ and direction of each particle moving along each dimension of the problem space are modified at each generation. The movement of particles is influenced by two factors. The first is called *global best* $\underline{g}^k$ and represents the social behavior in which the particle gets attracted towards the centroid of the group; the second is the so-called *personal best* $\underline{l}_n^k$ and represents the cognitive factor of each individual particle, i.e., the particle best solution. The stochastic nature of PSO comes from two random component r_1 and r_2, which influence the speed and the direction of the particles, respectively. A general equation for the speed of particles is the following:

$$\underline{v}_n^k = \varphi \cdot \underline{v}_n^{k-1} + C_1 r_1 \left\{\underline{l}_n^k - \underline{p}_n^{k-1}\right\} + C_2 r_2 \left\{\underline{g}^k - \underline{p}_n^{k-1}\right\} \tag{11.1}$$

where φ is the inertial factor, which defines how much the speed is influenced by the past. This parameter could be set as a constant or may decrease across iterations in order to slow down the swarm when approaching the best solution [22]. The cognitive and the social factors are influenced by two different constants, C_1 and C_2, called *acceleration coefficients*, which are two tunable weights used to modify the behavior of the particles. The higher the value of C_1, the more the particles will move towards their personal best solution; the higher the factor C_2, the more the particles tend to approach the best solution found by the whole swarm. An in-depth analysis of the relation between acceleration coefficients and optimization process could be found in [23]. At every iteration k, the position of a particle $\underline{p}_n$ is updated as follows:

$$\underline{p}_n^k = \underline{p}_n^{k-1} + \underline{v}_n^k \tag{11.2}$$

The point $\underline{p}_n$ is a new possible solution of the problem, and is again evaluated using the fitness function $\Psi\left(\underline{p}_n\right)$. The iterative process terminates when a predefined number of iterations is reached or the best possible solution $\underline{p}_n$ is found.

Even though PSO has been used for more than ten years, there is not a unique or standardized setting of the parameters able to guarantee control efficiency, stability, and, more in general, the performance of the algorithm. The number of particles, the inertial weight, and the cognitive and social factors are parameters strictly connected to each other, which have to be carefully tuned according to the considered optimization problem. A useful insight about the behavior of the algorithm with different parameter settings can be found in [24].

11.3 PSO for Optimal Tuning of Parameters

The first attempt to introduce PSO in content-based image retrieval concerned the adaptive setting of parameters into CBIR tools.

In [25], the optimization is applied to a query-by-example tool based on a similarity measure calculated on the sublevels of the wavelet transform. In this work, the authors calculate the similarity between two images D as a weighted sum of the histograms coming from the wavelet transform of the image entropy, namely:

$$D = \sum_{i=0}^{r-1} \sum_{j=0}^{i-1} w_{ij} \cdot \delta \left[\Psi_j^i(s_j^i), \Psi_j^i(s_j^{'i}) \right] \tag{11.3}$$

where Ψ is the Haar wavelet transform of an entropy subhistogram s_j^i of the image. In this equation, i represents the number of bit-planes of the image. The PSO is used to find the best parameter w_{ij} (weight of the subhistogram j at level i) and the number of decomposition levels i, in order to obtain the best similarity. r is fixed to 16 so that the multidimensional feature space has a maximum of 128 dimensions.

Zhiwei Ye et al. used the PSO in a similar way [26]. In this case, the authors wanted to find the best set of weights in order to stress the spatial importance of the color histogram. The main difference from [25] lies in the fact here that the weights are discrete binary values, instead of continuous values in the range [-2, 2].

Kameyama et al. used the PSO to optimize the similarity parameters [27–29]. In this case, a relaxation matching [30] is adopted to establish the similarity between images. This method requires the setting of eight independent parameters, and PSO is proved to achieve an effective solution to this purpose. In all these works the PSO fitness function involves the precision and the recall calculated on a small set of test images. This means on one hand that there is an assessable optimal setting, but on the other hand that some over-fitting problem can happen. For this reason, the number of images used in the optimization process is of key importance.

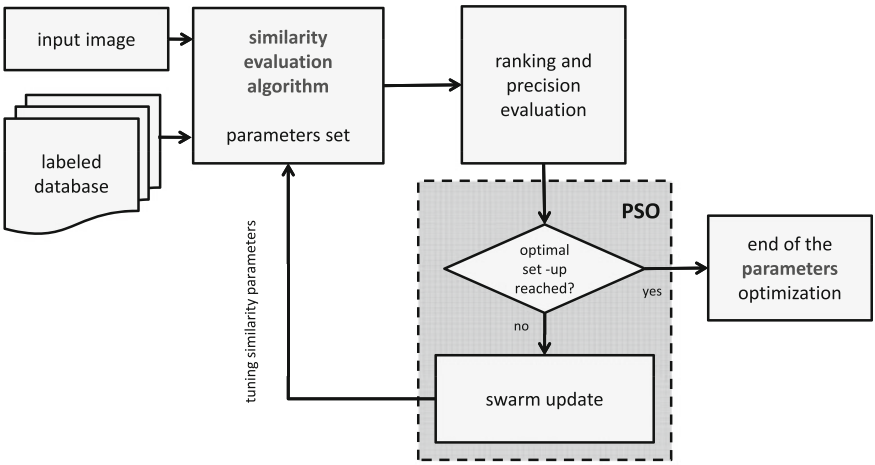


Fig. 11.1 General schema of parameters optimization related to visual image similarity. Each particle $\underline{p}_n$ represents a possible combination of parameters. The fitness function represents the ability of this combination to improve the image ranking according to the adopted similarity metric

A general schema depicting how the PSO algorithm works in this context is presented in Fig. 11.1.

11.4 PSO in Image Classification

A second possible way of introducing an optimization tool into an image retrieval system consists in stating the classification problem as an optimization process. Again, the optimization process allows finding an optimal set of parameters, but in this case such parameters describe the hyper-planes or the rules needed to segment the multi-dimensional feature space into a set of partitions, corresponding to a set of image classes. In [32–34] the authors propose to apply a PSO to find the optimal set-up for a self organizing map (SOM) classifier [31]. The idea is to associate to each image a neuron of the SOM, with dimension equal to the feature vector. The weights of the neurons are then iteratively updated by the PSO algorithm, until a predefined number of iterations is reached or the distance between input feature vector and global best reaches a fixed threshold. At the end of the optimization, the resulting set of weights minimizes the classification error of the training set.

PSO has been also used to optimize K-means classifiers [35, 36]. Although K-means remains one of the most widely used clustering algorithms for unsupervised classification problems, it gets typically trapped into local minima. In order to overcome this problem, the authors propose to use a PSO algorithm to search for the best K-means solution. To achieve this goal, the authors propose to initialize m

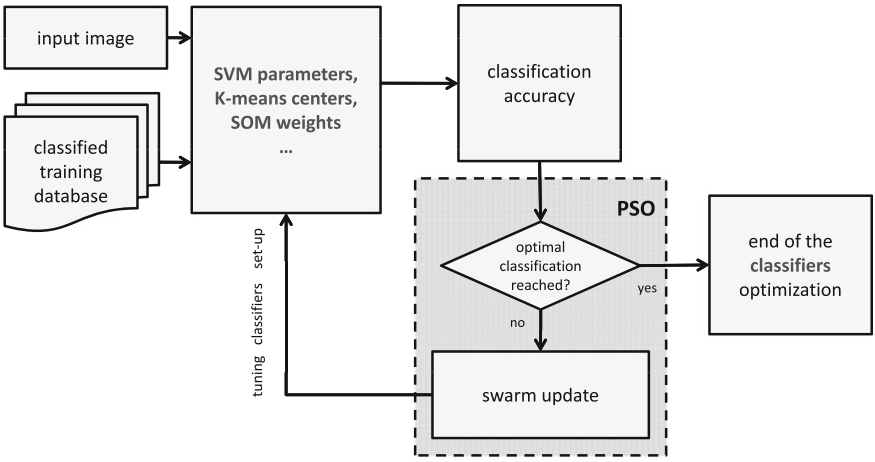


Fig. 11.2 General schema of classifier optimization. Each particle $\underline{p}_n$ represents a possible set-up of the classifier. At every iteration the classification accuracy on a training set of images is evaluated as a fitness function. The process terminates when either the target accuracy is obtained or a predefined number of iterations is reached. The set-up corresponding to the best accuracy is kept as the optimal one

sets of K random cluster centers to be used by m particles. Each particle is then a point in a multidimensional space and is used as a seed for the clustering. At the end of the optimization process, the particles highlight the best cluster centers, with the goal of maximizing the classification accuracy. This kind of approach turns out to be very useful in image segmentation tasks, where a predefined number k of selections is given.

A classification scheme largely used in image retrieval as well as other applications is the support vector machine (SVM)[37]. SVMs employ the structural risk minimization principle, which provides them with good generalization properties. In order to solve the problem of parameter selection in support vector machines, the use of PSO has been proposed [38].

Finally, it has to be mentioned that the use of PSO for data classification not only refers to natural images, but also to other data sources, such as ultrasound images [39], hyperspectral remote sensing images [40, 41], SAR images [42], or medical images [43]. In all those cases, PSO provide significant gains in terms of efficiency against empirical parameter tuning.

A general schema depicting how PSO algorithm works in this type of problems is presented in Fig. 11.2.

11.5 PSO and Relevance Feedback

A completely different use of PSO refers to the solution of the retrieval problem when the user is involved in the search process. This is the typical case of relevance feedback (RF) approaches, where the user interacts with the system to improve the retrieval performance.

The retrieval process usually relies on presenting a visual query (natural or synthetic) to the system, which extracts the most similar images from a database. Such mechanism, referred to as query-by-example, is typically deterministic, thus producing always the same result according to the query. The choice of the query is then fundamental for the quality of the final result. RF introduces several mechanisms to improve this process, making it possible to change the retrieval result by iteratively modifying either the query or the feature space according to the user feedback. Nevertheless, the process is again deterministic (provided that the user feedback is consistent) and the initial query has still a great impact on the final result.

By introducing some degree of randomness in the search, it is possible to explore in a completely different way the solution space, thus allowing the convergence to a better solution and/or a lower dependency on the solution from the starting point. Stochastic optimization methods provide a viable approach to achieve this goal.

In [44], PSO is, for the first time, directly used to find the requested image cluster inside a database, i.e., to find the images that minimize a given cost function. The swarm particles fly through the database, and their location in the multidimensional feature space represents a possible solution. The underlying idea is to steer the swarm particles through the solution space (the multidimensional feature space and the image database projected thereby) and to run the swarm migration process in order to identify the set of images that best fit the user's request. In this way, the image retrieval problem can be seen as an optimization problem of a parametric fitness function that expresses the quality of the retrieved solution: the lower the fitness value, the better the solution. Using the swarm intelligence it is possible to substitute a generic query shifting with a completely different process, where each particle of the swarm can be seen as a single query, moving towards relevant samples and far from irrelevant ones (thanks to the personal best bias), but also taking into account the social behavior (according to the global best term). A further added value of the proposed algorithm with respect to other proposed CBIR methods is the fact that it introduces a random component to the process, thus allowing to explore the solution space in different ways, converging to a good solution independent of the starting point and of the path followed. A general schema depicting how PSO algorithm works jointly with RF is presented in Fig. 11.3.

An important improvement of the idea presented in [44] consists in understanding that also the diversity could be exploited in order to better satisfy the user. Sometimes it could happen that images that are relevant to the user or belong to the same semantic concept, could be very diverse in terms of visual appearance. Because of this, also stochastic methods could suffer from stagnation problems, being unable to follow manifold clusters. For this reason, the concept of swarm evolution is introduced in

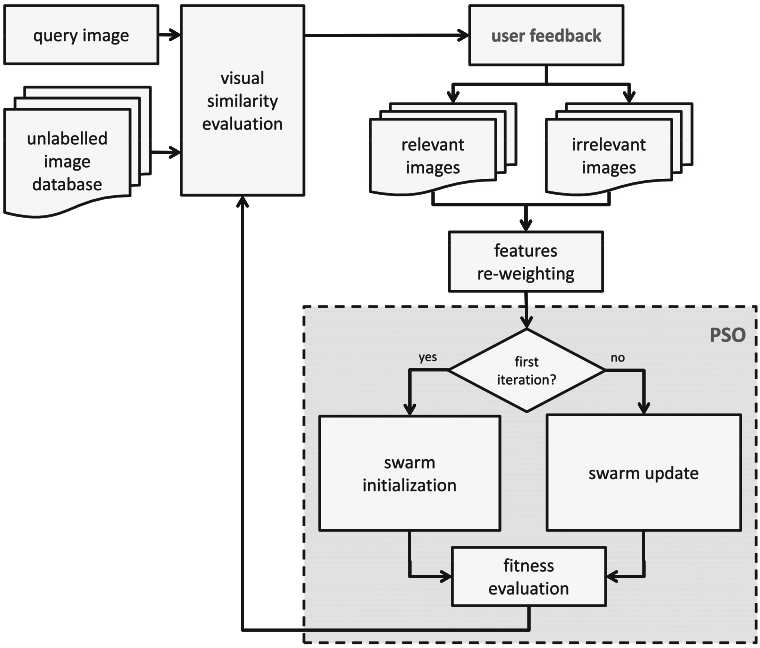


Fig. 11.3 General schema of relevance feedback and particle-swarm optimization retrieval system. Each particle $\underline{p}_n$ represents a visual feature set. At every iteration the distance of each particle from the relevant examples tagged by the user is evaluated. The particles that minimize the distance from relevant points and maximize the distance from irrelevant ones are presented to the user as possible new relevant images. The process terminates when the user is satisfied with the retrieved images

the stochastic retrieval optimization process [45]. The key idea is to split the swarm according to the number of the tagged relevant images. Accordingly, each subswarm performs an independent optimization process, searching in the surrounding solution space. The evolutionary splitting consists in forcing the personal best of all the subswarm particles to the same relevant point. This process does not provide major advantages when the relevant images are consistently clustered in the solution space, while significantly improving the performance when relevant pictures are scattered in the feature space [46] (see Fig. 11.4).

Another degree of freedom when using PSO in CBIR lies in the definition of the fitness function. When the optimization is used to find a set of parameters, the fitness is typically calculated based on the classification accuracy over a well-defined set of data. Consequently, if a minimum is present, the slope of the function for all the particles should be monotonically decreasing. This happens because the bounding conditions of the problem remain constant during the optimization process. This is not the case when PSO is adopted with the user feedback. In this case the fitness is evaluated in terms of distances among relevant and irrelevant examples [47, 48]. These pictures change at every iteration and, in many cases, the feature space changes

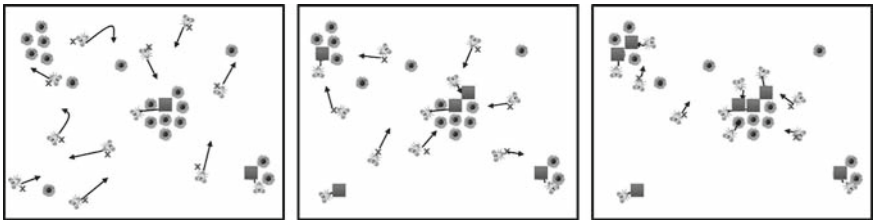


Fig. 11.4 Example of evolutionary PSO. At every iteration the personal best $\underline{l}_n^k$ is set according to the nearest relevant image, in order to search the surroundings of each relevant point, thus avoiding early stagnation of the swarm

as well due to reweighting [44]. In this context, the PSO algorithm loses the meaning of *optimization*, since there is no unique optimal solution, but just a best guess of the new images to be presented to the user. In [46] the fitness function is defined in such a way to represent the effectiveness of the solution reached by the swarm particles. The authors take into account the relevant and irrelevant image sets, and define Eq. 11.4 as a weighed cost function $\Psi^k\left(\underline{p}_n\right)$ that expresses the fitness associated to the solution found by the generic particle $\underline{p}_n$:

$$\Psi^k\left(\underline{p}_n\right) = \frac{1}{N_{rel}^k}\sum_{r=1}^{N_{rel}^k} D\left(\underline{p}_n^k; \underline{x}_r^k\right) + \left(\frac{1}{N_{irr}^k}\sum_{i=1}^{N_{irr}^k} D\left(\underline{p}_n^k; \underline{x}_i^k\right)\right)^{-1} \tag{11.4}$$

In this equation, $\underline{x}_r^k; r = 1, \ldots, N_{rel}^k$ and $\underline{x}_i^k; i = 1, \ldots, N_{irr}^k$ are the images in the relevant and irrelevant subsets, respectively, and D represents a generic similarity metric between the visual signatures of the selected images. It is to be observed that the function $\Psi^k\left(\underline{p}_n\right)$ produces lower values when the particle is close to the relevant set and far from the irrelevant one. Therefore, the lower the fitness, the better the position of the particle. According to the fitness value, it is possible to reorder the particle-swarm obtaining a new ranking. It is also worth noting that both the weights and the fitness function change across iterations, because of the dynamic nature of χ_{REL}^k and χ_{IRR}^k subsets. Accordingly, features that were relevant to discriminate some images can lose importance during the process, and particles that were considered very close to the global best can become far from the relevant zone of the solution space. In most cases, the number of irrelevant images collected across iterations is greater than the number of relevant ones. This aspect has been taken into consideration during the formulation of the fitness function making it dependent on the inverse of the distance from irrelevant images. In this way, the more the average distance of the particle from irrelevant images grows, the more the fitness depends only from relevant images.

The idea presented in [44] has been adopted by other authors in different works; X. Xu et al. first reimplemented the same retrieval system in [49], then adopted PSO as an online optimization of the weights in the evaluation of the similarity between features vectors [50]. The translation of the retrieval process into an optimization one is also adopted by K. Wei et al. in [51] and in [52], where there is no user feedback, but the PSO iteratively presents to the user a set of images related to an initial query, till reaching satisfaction.

Recently a CBIR application involving RF and PSO has been presented in [53]. Based on Fourier descriptors, the method decomposes the contour of an object into a series of concentric ellipses. An ellipse can be represented with only three parameters: semimajor axis, semiminor axis, and orientation angle. Since the difference among such ellipses is remarkable, the weights associated to the relevant parameters in similarity computation are fine tuned by a PSO. Finally, relevance feedback has been used also by Chandramouli and Izquierdo, who modified the fixed off-line training of the SOM [32] to generate an on-line training system, by introducing a user input [54–56].

11.6 Ant Colony and Genetic Algorithms

The optimization issues presented in the above sections could also be addressed using different optimization approaches. In the field of stochastic algorithms, ant colony [16] and genetic algorithms [57] deserve a special mention. Ants present a very good natural metaphor for evolutionary computation. With their small size and small number of neurons, they are not capable of dealing with complex tasks individually. An important and interesting behavior of ant colonies is, in particular, how ants can find the shortest paths between food sources and their nest. For this reason, this algorithm has been widely exploited in network-related problems [58], although some proposals have been presented also in the field of image retrieval [59] and in general for classification purposes [60].

Also genetic algorithms (GA) are considered effective stochastic optimization tools and are applied in a number of research fields, including image retrieval problems. One of the first applications can be found in [61]. Here, the optimization starts with an initial population formed of randomly generated individuals. In the beginning, the fitness value of each individual is evaluated to determine how appropriate it is for the given problem. Two individuals of relatively high fitness are then selected from the population, and they are regarded as the *parents*. New individuals called *children* are created by recombining the chromosomes of parents. Here, crossover and mutation operators are used to induce variations in the population.

GAs have been exploited both to optimize the parameter selection [62] and to exploit user interaction [63]. While in the first case it is easy to compare the performance of different optimizers, since there is a measurable objective, this does not apply to the second case. In fact, when users are involved in the loop, it is possible to observe that each user has his own idea of *relevance*, and the images selected

by a user could be very dissimilar in terms of visual features [46]. For this reason, the principle of evolution of a GA looses effectiveness. In order to avoid this problem, multiobjective GAs have been proposed, which multiply the number of parents according to the relevant images found [64], mimicking what is done in swarm-based approaches.

11.7 Conclusions

In this chapter, the use of stochastic and evolutionary optimization techniques for image retrieval problems has been presented, making special reference to social approaches, and in particular to particle swarm optimizers (PSO). The different ways how these tools can be usefully exploited were analyzed in detail, in order to explain how statistical optimization can improve the performance of content-based image retrieval techniques. It was shown that state-of-the-art stochastic optimization approaches make it possible to achieve nearly optimal parameter setting in multidimensional continuous or discrete spaces, while providing time- and computation-effective solutions. It was also illustrated how the social behavior and the active learning capability of such approaches can be exploited directly in the retrieval process, letting the user interact with the optimization process. PSO is characterized by a set of good properties in terms of generalization, scalability and diversity, which are all important features in solving a retrieval problem. The translation of the retrieval process into an optimization problem has opened different research directions in the human-machine interaction field.

References

1. Datta, R., Joshi, D., Li, J., Wang, J.Z.: Image retrieval: Ideas, influences, and trends of the new age. ACM Compute **40**(2), 1–60 (2008)
2. Hirata, K., Kato, T.: Query by Visual Example - Content based Image Retrieval. In: Proceedings of the 3rd International Conference on Extending Database Technology: Advances in Database Technology, pp. 56–71. Springer, UK (1992)
3. Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of the early years. IEEE Trans. Pattern Anal. Mach. Intell. **22**(12), 1349–1380 (2000)
4. Hanjalic, A., Lienhart, R., Ma, W.-Y., Smith, J.R.: The Holy Grail of Multimedia Information Retrieval: So Close or Yet So Far Away? JPROC **96**(4), 541–547 (2008)
5. Yanai, K., Shirahatti, N.V., Gabbur, P., Barnard, K.: Evaluation strategies for image understanding and retrieval. In: Proceedings of the 7th ACM SIGMM international workshop on Multimedia information retrieval (MIR '05), pp. 217–226. ACM, New York (2005)
6. Deserno, T.M., Antani, S., Long, R.: Ontology of Gaps in Content-Based Image Retrieval. J. Digit. Imaging. (Springer, New York) **22**(2), 202–215 (2008)
7. Deselaers, T., Keysers, D., Ney, H.: Features for image retrieval: an experimental comparison. Inf. Retrieval (Springer, Netherlands) **11**(2), 77–107 (2008)
8. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. Int. J. Comput. Vision, (Springer, Netherlands) **60**(2), 91–110 (2004)

9. Duda, R.O.: Hart . Pattern Classification and Scene Analysis. Wiley, New York (1973)
10. Deng, J., Berg, A., Li, K., Fei-Fei, L.: What does classifying more than 10,000 image categories tell us?. In: Proceedings of European Conference of Computer Vision (ECCV) (2010)
11. Russakovsky, O., Fei-Fei, L.: Attribute learning in large-scale datasets. In: Proceedings of European Conference of Computer Vision (ECCV), International Workshop on Parts and Attributes (2010)
12. Jain, R., Sinha, P.: Content without context is meaningless. In: Proceedings of the international conference on Multimedia (MM '10), pp. 1259–1268. ACM, New York (2010)
13. Pereira, M.V.F., Pinto, L.M.V.G.: Multi-stage stochastic optimization applied to energy planning. Math. Program. (Springer, Berlin), **52**(1), 359–375 (1991)
14. Elbeltagi, E., Hegazy, T., Grierson, D.: Comparison among five evolutionary-based optimization algorithms. Adv. Eng. Inform. **19**(1), 43–53 (2005). http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=05337717
15. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**(1), 67–82 (1997)
16. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Trans. Syst. Man Cybern. Part B Cybern. **26**(1), 29–41 (1996)
17. Masahiro, I., Jaroslav, R.: Possibilistic linear programming: a brief review of fuzzy mathematical programming and a comparison with stochastic programming in portfolio selection problem. Fuzzy Sets Syst. **111**(1), 3–28 (2000)
18. Kennedy, J., Eberhart, R.: Particle-swarm optimization. Proc. Fourth IEEE Int. Conf. Neural Netw. **4**, 1942–1948 (1995)
19. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
20. Eberhart, R., Shi, Y.: Particle-swarm optimization: developments, applications and resources. Proc. Congress Evol. Comput. **1**, 81–86 (2001)
21. Clerc, M., Kennedy, J.: The particle-swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6**(1), 58–73 (2002)
22. Hinchey, M.G., Sterritt, R., Rouff, C.: Swarms and swarm intelligence. Computer **40**(4), 111–113 (2007)
23. Robinson, J., Rahmat-Samii, Y.: Particle-swarm optimization in electromagnetics. IEEE Trans. Antennas Propag. **52**(2), 397–407 (2004)
24. Zheng, Y.L., Ma, L.H., Zhang, L.Y., Qian, J.X.: On the convergence analysis and parameter selection in particle-swarm optimization. Int. Conf. Mach. Learn. Cybern. **3**, 1802–1807 (2003)
25. Chao, X., Chengjian, W., Jun, X.: Evolutionary wavelet-based similarity search in image databases. In: Proceedings of IEEE International Workshop on VLSI Design and Video Technology, pp. 385–388 (2005)
26. Ye, Z., Xia, B., Wang, D., Zhou, X.: Weight optimization of image retrieval based on particle-swarm optimization algorithm. In: International Symposium on Computer Network and Multimedia Technology, pp. 1–3 (2009)
27. Kameyama, K., Oka, N., Toraichi, K.: Optimal parameter selection in image similarity evaluation algorithms using particle-swarm optimization. In: IEEE Congress on, Evolutionary Computation, pp. 1079–1086 (2006)
28. Okayama, M., Oka, N., Kameyama, K.: Relevance optimization in image database using feature space preference mapping and particle-swarm optimization. Lect. Notes Comput. Sci. Neural Inf. Process. (Springer, Berlin), 4985, 608–617 (2008)
29. Oka, N., Kameyama, K.: Relevance tuning in content-based retrieval of structurally-modeled images using Particle-Swarm Optimization. In: IEEE Symposium on Computational Intelligence for Multimedia Signal and Vision Processing, pp. 75–82 (2009)
30. Rosenfeld, A., Hummel, R.A., Zucker, S.W.: Scene labeling by relaxation operations. IEEE Trans. Syst. Man Cybern. **6**(6), 420–433 (1976)
31. Kohonen, T.: The self-organizing map. Proc. IEEE **78**(9), 1464–1480 (1990)
32. Chandramouli, K., Izquierdo, E.: Image classification using chaotic particle-swarm optimization. In: IEEE International Conference on Image Processing, pp. 3001–3004 (2006)

33. Piatrik, T., Chandramouli, K., Izquierdo, E.: Image classification using biologically inspired systems. In: Proceedings of the 2nd International Conference on Mobile Multimedia Communications (MobiMedia '06). ACM, New York (2006)
34. Chandramouli, K.: Particle-swarm optimisation and self organising maps based image classifier. In: Second International Workshop on Semantic Media Adaptation and Personalization, pp. 225–228 (2007)
35. Su, S.: Image classification based on particle-swarm optimization combined with K-means. In: International Conference on Test and Measurement, ICTM '09, **2**, pp. 367–370 (2009)
36. Hung, C.C., Wan, L.: Hybridization of particle-swarm optimization with the K-Means algorithm for image classification. In: IEEE Symposium on Computational Intelligence for Image Processing, CIIP '09, pp. 60–64 (2009)
37. Chapelle, O., Haffner, P., Vapnik, V.N.: Support vector machines for histogram-based image classification. IEEE Trans. Neural Netw. **10**(5), 1055–1064 (1999)
38. Zhang, Y., Xie, X., Cheng, T.: Application of PSO and SVM in image classification. 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), **6**, pp. 629–631 (2010)
39. Chang, C.Y., Lai, C.T., Chen, S.J.: Applying the particle-swarm optimization and boltzmann function for feature selection and classification of lymph node in ultrasound images. ISDA '08. In: Eighth International Conference on Intelligent Systems Design and Applications **1**, pp. 55–60 (2008)
40. Ding, S., Chen, L.: Classification of hyperspectral remote sensing images with support vector machines and particle-swarm optimization. In: ICIECS International Conference on Information Engineering and Computer Science, pp. 1–5 (2009)
41. Linyi, L., Deren, L.: Fuzzy classification of remote sensing images based on particle-swarm optimization. In: International Conference on Electrical and Control Engineering (ICECE), pp. 1039–1042 (2010)
42. Xu, X., Zhang, A.: An unsupervised particle-swarm optimization classifier for SAR image. Int. Conf. Computational Intell. Secur. **2**, 1630–1634 (2006)
43. Zhang, Q., Gao, L.: Medical image retrieval algorithm using setting up weight automatically. In: 3rd International Conference on Intelligent Networks and Intelligent Systems (ICINIS), pp. 60–63 (2010)
44. Broilo, M., Rocca, P., De Natale, F.G.B.: Content-based image retrieval by a semi-supervised Particle-Swarm Optimization. In: IEEE 10th Workshop on Multimedia, Signal Processing, pp. 666–671 (2008)
45. Broilo, M., De Natale, F.G.B.: Evolutionary image retrieval. In: 16th IEEE International Conference on Image Processing (ICIP), pp. 1845–1848 (2009)
46. Broilo, M., De Natale, F.G.B.: A stochastic approach to image retrieval using relevance feedback and particle-swarm optimization. IEEE Trans. Multimedia **12**(4), 267–277 (2010)
47. Luo, T., Yuan, B., Tan, L.: Blocking wavelet-histogram image retrieval by adaptive particle-swarm optimization. In: 1st International Conference on Information Science and Engineering (ICISE), pp. 3985–3988 (2009)
48. Luo, T., He, J.: Fast similarity search with blocking wavelet-histogram and adaptive particle-swarm optimization. In: Third International Conference on Knowledge Discovery and Data Mining, WKDD '10, pp. 334–337 (2010)
49. Xu, X., Liu, X., Yu, Z., Zhou, C., Zhang, L.: Re-weighting relevance feedback image retrieval algorithm based on particle-swarm optimization. Sixth Int. Conf. Nat. Comput. (ICNC) **7**, 3609–3613 (2010)
50. Xu, X., Zhang, L., Yu, Z., Zhou, C.: The application of particle-swarm optimization in relevance feedback. In: FBIE International Conference on Future BioMedical Information, Engineering pp. 156–159 (2009)
51. Wei, K., Lu, T., Zhang, Q., Bi, W.: Research of image retrieval algorithm based on PSO and a new sub-block idea. In: 2nd International Conference on Advanced Computer Control (ICACC), **1**, pp. 431–435 (2010)

52. Wei, K., Lu, T., Bi, W., Sheng, H.: A kind of feedback image retrieval algorithm based on PSO, Wavelet and subblock sorting thought. In: 2nd International Conference on Future Computer and Communication (ICFCC), **1**, pp. V1–796-V1-801 (2010)
53. Wu, F., Li, Y. X., Xu, P., Liang, X.: Image Retrieval Using Ellipse Shape Feature with Particle-Swarm Optimization. In: International Conference on Multimedia Technology (ICMT), pp. 1–4 (2010)
54. Chandramouli, K., Kliegr, T., Nemrava, J., Svatek, V., Izquierdo, E.: Query refinement and user relevance feedback for contextualized image retrieval. In: VIE 5th International Conference on Visual Information, Engineering, pp. 453–458 (2008)
55. Chandramouli, K., Izquierdo, E.: Multi-class relevance feedback for collaborative image retrieval. In: 10th Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS '09, pp. 214–217 (2009)
56. Ibrahim, S.N.A., Selamat, A., Selamat, M.H.: Query optimization in relevance feedback using hybrid GA-PSO for effective web information retrieval. In: Third Asia International Conference on Modelling and Simulation, AMS '09, pp. 91–96 (2009)
57. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Boston, MA (1989)
58. Picard, D., Cord, M., Revel, A.: Image retrieval over networks: active learning using ant algorithm. IEEE Trans. Multimedia **10**(7), 1356–1365 (2008)
59. Piatrik, T., Izquierdo, E.: Subspace clustering of images using Ant colony Optimisation. In: 16th IEEE International Conference on Image Processing (ICIP), pp. 229–232 (2009)
60. Liu, X., Li, X., Liu, L., He, J., Ai, B.: An innovative method to classify remote-sensing images using ant colony optimization. IEEE Trans. Geosci. Remote Sens. **46**(12), 4198–4208 (2008)
61. Kato, S., Iisaku, S.I.: An image retrieval method based on a genetic algorithm. In: Proceedings of Twelfth International Conference on Information Networking (ICOIN-12), pp. 333–336 (1998)
62. Papadias, D., Mantzourogiannis, M., Ahmad, I.: Fast retrieval of similar configurations. IEEE Trans. Multimedia **5**(2), 210–222 (2003)
63. Cho, S.B., Lee, J.Y.: A human-oriented image retrieval system using interactive genetic algorithm. Syst. Man Cybernetics Part A IEEE Trans. Syst. Hum. **32**(3), 452–458 (2002)
64. Tran, K. D.: Content-based retrieval using a multi-objective genetic algorithm. In: IEEE Proceedings of the SoutheastCon, pp. 561–569 (2005)

Chapter 12
A Cluster-Based Boosting Strategy for Red Eye Removal

Sebastiano Battiato, Giovanni Maria Farinella, Daniele Ravì, Mirko Guarnera and Giuseppe Messina

Abstract Red eye artifact is caused by the flash light reflected off a person's retina. This effect often occurs when the flash light is very close to the camera lens, as in most compact imaging devices. To reduce these artifacts, most cameras have a red eye flash mode which fires a series of preflashes prior to picture capture. The major disadvantage of the preflash approach is power consumption (e.g., flash is the most power-consuming device on the camera). Alternatively, red eyes can be detected after photo acquisition. Some photo-editing softwares make use of red eye removal tools that require considerable user interaction. To overcome this problem, different techniques have been proposed in literature. Due to the growing interest of industry, many automatic algorithms, embedded on commercial software, have been patented in the last decade. The huge variety of approaches has permitted research to explore different aspects and problems of red eyes identification and correction. The big challenge now is to obtain the best results with the minimal number of visual errors. This chapter critically reviews some of the state-of-the-art approaches for red eye removal. We also discuss a recent technique whose strength is due to

S. Battiato (✉) · G. M. Farinella · D. Ravì
Image Processing Laboratory (IPLAB), Dipartimento di Matematica e Informatica, Università di Catania, Viale A. Doria 6, 95125 Catania, Italy
e-mail: battiato@dmi.unict.it

G. M. Farinella
e-mail: gfarinella@dmi.unict.it

D. Ravì
e-mail: ravi@dmi.unict.it

M. Guarnera · G. Messina
Advanced System Technology, STMicroelectronics, Stradale Primosole 50, 95125 Catania, Italy
e-mail: mirko.guarnera@st.com

G. Messina
e-mail: giuseppe.messina@st.com

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_12, © Springer-Verlag Berlin Heidelberg 2013

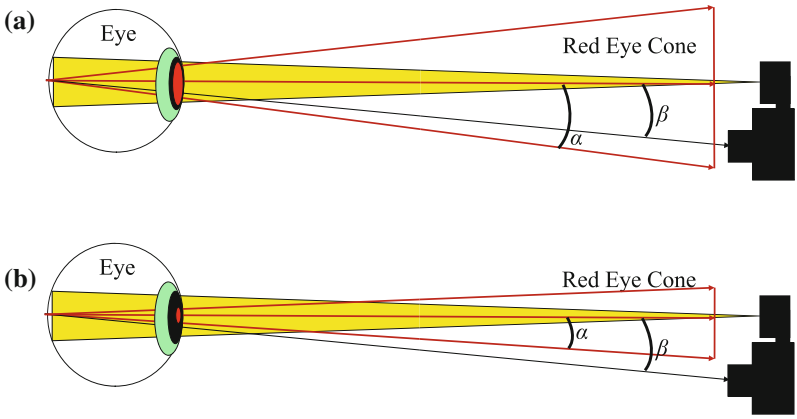


Fig. 12.1 Red eye is caused by the reflection of the flash off the blood vessels of the retina. The camera records this red hue if the angle β is not greater than α (**a**), otherwise the red eye is not recorded (**b**)

a multimodal classifier which is obtained by combining clustering and boosting in order to recognize red eyes represented in the gray codes feature space.

12.1 Introduction

Red eye artifacts are a well-known problem in digital photography. They are caused by direct reflection of light from the blood vessels of the retina through the pupil to the camera objective. When taking flash-lighted pictures of people, light reflected from the retina forms a cone, whose angle α depends on the opening of the pupil. Let β be the angle between the flash and the camera lens (centered on the retina). The red eye artifact is formed if the red light cone hits the lens, that is, if α is greater than or equal to β (Fig. 12.1). Small compact devices and point-and-click usage, typical of non-professional photography, greatly increase the likelihood for red eyes to appear in acquired images.

High-end cameras can be equipped with separate flashes with an extensible and steerable bracket, which allows for more distance between the flash and the lens, thus reducing the probability for red eyes to appear. One preventive measure suitable to both high-end and low-end devices is to make additional flashes before actually taking the photograph (preflash). This method, first proposed by Kodak [1], gives pupils time to shrink in order to reduce the reflectance surface, thus making red eyes less likely. It is important that enough time elapses between flashes to account for the response time of the pupils (Fig. 12.2). This approach is effective, but it has the disadvantage of greatly increasing power consumption, which may be problematic for

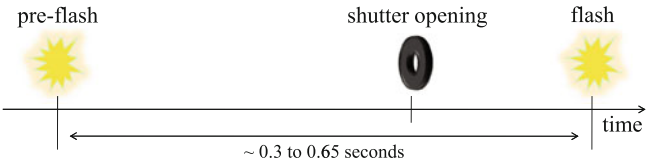


Fig. 12.2 Timeline explaining the preflash approach. Before the actual acquisition, a flash is fired. After a short time, the shutter opens and light enters the sensor. At the end of the exposure time the "true" flash is fired. Time between flashes is such that the pupils have time to react and shrink

Fig. 12.3 Examples of the variability of the red eye phenomenon. Golden eyes are also visible

power-constrained mobile devices [2]. Also, the additional flashing may sometimes be uncomfortable for people.

Red eye prevention methods reduce the probability of the phenomenon but do not remove it entirely. Most of the time, then, the picture must be corrected during postprocessing. Red eye removal is a very challenging task: Red eyes may vary in shape and color, and may also differ in position and size relative to the whole eye. Sometimes light is reflected on a part of the retina not covered with blood vessels, yielding a yellow or white reflection (golden eyes). Some examples of the phenomenon are showed in Fig. 12.3. Designing a system which can effectively address all the possible cases is very difficult.

For red eyes to be removed, they must be first reliably detected and then properly corrected. Detection methods are divided into semiautomatic methods, which ask the user to manually localize and point to the red eyes, and automatic methods, which detect the red eyes themselves. In the first case the eyes are manually selected using

a visual interface (e.g., Adobe Photoshop [3], Corel Paint Shop Pro [4], ACDSee [5], etc.). This is feasible because eyes are easy to localize for humans, but requiring manual intervention for every picture is unsuitable expecially for non-professional usage. Moreover, it may be difficult to have and use such an interface on a mobile device. Automatic methods attempt to find red eyes on their own. Since they do not require user intervention, they are easier to use and more appealing, and thus are suitable for embedded devices. However, automatic detection of red eyes is a very challenging task, due to the variability of the phenomenon and the general difficulty in discriminating eyes from other details.

Red eye correction techniques, on the other hand, can be more or less invasive. Generally speaking, "easier" cases may be addressed with a softer correction, while sometimes a stronger intervention is needed. Since the aim is to provide a corrected image which looks as natural as possible, a less invasive correction is preferred when the natural aspect of the eye is reconstructible from the acquired image.

This chapter aims to provide an overview of well-known automatic red eye detection and correction techniques, pointing out working principles, strengths and weaknesses of the various solutions. For further information about red eye removal, see recent surveys on academic papers [6, 7] and patents [8]. Moreover, the chapter discusses a recent technique whose strength is due to a multimodal classifier which is obtained by combining clustering and boosting in order to recognize red eyes represented in the gray codes feature space [9–11].

The chapter is organized as follows. Section 12.2 explores red eye detection. Section 12.3 describes methods for red eye correction, whereas Sect. 12.4 gives an insight into the problem of unwanted and improper corrections, showing their side effects. Lastly, Sect. 12.5 provides criteria to evaluate the quality of the results. Finally, Sect. 12.6 introduces an advanced technique based on boosting and gray codes representation. Conclusion are given in Sect. 12.7.

12.2 Eye Detection

The main difficulty in detection of red eyes is their great degree of variability. In the easier cases, the pupil has a "normal" shape and size and differs from a regular one only by its color. However, it is not uncommon for the red reflection to spread over the iris generating an unnatural luminance distribution. Usually a small white glint is also present, representing the direct reflection of the flash on the surface of the eye and giving the eye a much more natural appearance.

Typical red eye detection approaches involve extraction of red zones combined with skin extraction, shape template matching, and/or face detection. Some approaches also make use of ad hoc classifiers to further refine their results.

(a) **(b)**

Fig. 12.4 Picture **a** shows two very different red eyes; picture **b** shows one red eye along with a regular one

12.2.1 Color-Based Approaches

Color-based approaches are based on detecting red zones, which may correspond to red eye artifacts. As a typical constraint for the position of the red eyes, they also detect the human skin, considering also some criteria about the relative position of the red eyes and the skin (usually, the eyes must be almost completely surrounded by nearby skin). Some color-based approaches also detect the sclera (the white part of the eye), distinguishing it from the skin. Possible constraints may be imposed about the geometry of the red zones, such as discarding candidates that are too much elongated to represent a red pupil. This kind of approaches is quite simple, but does not take into account more complex features like, e.g., the presence of the various parts of the eye or the detection of the face.

One of the biggest problems of color-based techniques is the characterization of the colors to look for. Usually, interesting portions of the color space (corresponding to red, skin color, etc.) are delimited by hard thresholds, but they may also be delimited by soft margins, yielding a fuzzy probability for the color to belong to the region. However, finding proper boundaries for the regions is a challenging task. The color of red eyes is heavily influenced by the type of flash used, the sensor and the processing pipeline. While this is not a big issue, since the thresholds may be fine-tuned to adapt to the acquisition system, there are external factors which may influence the color of the eyes, including (but not limited to) the age of the person, the opening of the pupils, the distance from the camera, and the angle between the eyes and the flash. The variability is so high that even the same subject in one picture may have two different colored red eye artifacts, or a red eye and a regular one (see Fig. 12.4). Moreover, if the flash is not very strong (as is often the case with mobile devices), the external illuminant may produce a noticeable color cast on the picture, which adds another degree of variability to the colors. Similar considerations apply to the color of the skin and of the sclera.

The red color region may be defined in different color spaces. In the RGB space, a possible definition is [12]:

$$\begin{cases} R > 50 \\ R/(R+G+B) > 0.40 \\ G/(R+G+B) < 0.31 \\ B/(R+G+B) < 0.36 \end{cases} \tag{12.1}$$

Often, instead of hard thresholds, a *Redness* function is provided. This function is an estimate of how well the color of each pixel resembles a red eye artifact, and is used as a way to define soft margins for the red color region. Some possible redness functions are [13–16]:

$$Redness = (R - \min\{G, B\}) \tag{12.2}$$

$$Redness = \frac{R^2}{\left(G^2 + B^2 + 14\right)} \tag{12.3}$$

$$Redness = \frac{\max\{0, (R - \max\{G, B\})\}^2}{R} \tag{12.4}$$

$$Redness = \max\left\{0, \frac{2R - (G+B)}{R}\right\}^2 \tag{12.5}$$

As an alternative to select an interesting portion of the color space, it is possible to compare a redness function with a luminance function, discarding pixels whose luminance is more noticeable than the redness [17]:

$$Redness = R - (G+B)/2 \tag{12.6}$$

$$Luminance = 0.25R + 0.6G + 0.15B \tag{12.7}$$

$$RedLum = \max\{0, 2 \times Redness - Luminance\} \tag{12.8}$$

Search for red regions may also be performed in color spaces different from RGB, such as YCC [18] or HSL [9, 19].

Given a particular choice for the red color region, it is possible to convert each image to a representation which shows whether each pixel belongs to the region. Such representations are called redness maps. According to the employed definition for the red color region (hard-thresholded or soft-delimited), the redness map is a black-and-white or full-grayscale image (in the latter case, the redness function is adjusted to the possible maximums and minimums of the redness function, or to the maximums and minimums over each particular image). Figures 12.5 and 12.6 show

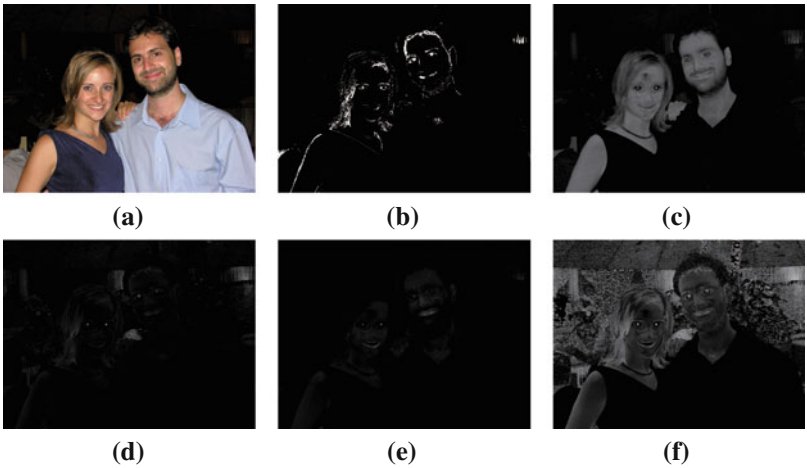


Fig. 12.5 Examples of redness maps: **a** original image; **b–f** redness maps obtained with Eqs. (12.1)–(12.5), respectively

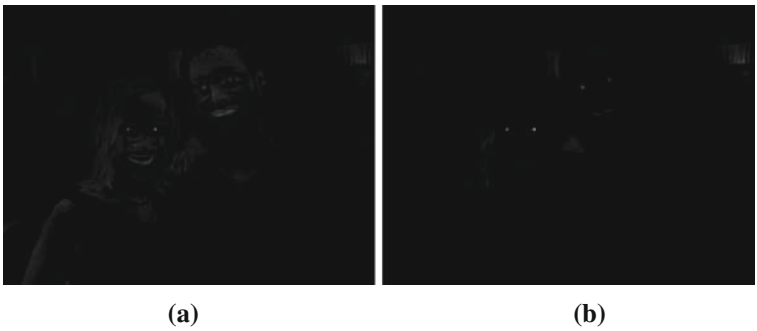


Fig. 12.6 **a** Redness map obtained from Eq. (12.6); **b** redness versus luminance map computed according to (12.8)

redness maps computed using the above formulae. Skin extraction may be performed in a similar way as red color extraction.

Other color-based information useful to detect red eyes may be gained by searching for the sclera [20] and selecting the zones where the flash has noticeably affected the image (discarding, e.g., a distant background) [21]. Using thresholding and morphological operators to combine different masks, it is possible to effectively extract red pupils.

12.2.2 Shape-Based Approaches

Shape-based approaches attempt to find eyes by exploiting simple information about their shape. They typically use templates which are matched at different positions and resolutions, in order to search the image for shapes which may correspond to eye features. The region of interest is then restricted to zones where the response of the templates is stronger. Using simple circular or square templates it is possible to recognize, e.g., the difference in intensity between the inner pupil and the outer skin and sclera. Slightly more complex templates may be useful in locating the other parts of the eye, which helps to effectively assess the presence or the absence of a red eye [22].

Edge detection filters may also be useful to extract information about shape. It is possible to use them in conjunction with color tables to make advantage of both spatial and chromatic information [23].

12.2.3 Pairing Verification

One of the constraints which can be used to filter out false detections is eye pairing verification [24]. It is based on the assumption that every eye found must be paired with the other one on the same subject's face. The two eyes must have similar size, and they must be in a certain range of distances (possibly proportional to the size, in order to account for the distance of the subject from the camera) from each other, in a horizontal or almost horizontal direction. If an eye cannot be paired because it has no suitable match, it is discarded, since it is most probably a false detection.

This approach is effective, since it is very unlikely for two false positives to satisfy the pairing criteria, but it presents a major drawback: If a face is partially occluded, so that only one eye is visible in the picture, and that eye is red, it will not be corrected, since it cannot be matched to the other one. The same problem will occur when both eyes are visible but only one is red, or when both are red but only one is detected, possibly due to a difference in color (Fig. 12.7).

12.2.4 Face Detection

Restricting the search region to the zones where faces are detected, it is possible to discard a great number of false positives [14]. In detail, a face detection system determines the locations and sizes of human faces in arbitrary digital images by making use, in some cases, of ad hoc facial features. The localization is done by considering a bounding box that encloses the region of interest. The detection problem is often achieved as a binary pattern classification task. The content of a given part of an image is transformed into features used to train a classifier on example faces so that it able to decide whether that particular region of the image is a face, or not. For practical situations it is very common to employ a sliding-window technique

(a) **(b)** **(c)**

Fig. 12.7 In picture **a**, only one of the eyes is visible; the red eyes in picture **b** are very different, and in most cases only one of them will be properly detected; in picture **c** only one of the eyes is affected by the red eye phenomenon. In all these cases, the pairing verification will fail

just using the classifier on small portions of an image (usually squared or rectangular), at all locations and scales, as either faces or non-faces. In the more general case, face localization is achieved regardless of position, scale, in-plane rotation and orientation, pose (out-of-plane rotation), and illumination. Further sources of problems are the presence or absence of structural components (e.g., beards, mustaches and glasses), facial expressions that have a great impact over the face appearance and the occlusions that occur when faces may be partially occluded by other objects.

To implement a robust face detector it is fundamental to fix some points relative to the specific application. In particular it is important to decide the facial representation, the involved preprocessing, the particular "cues" (e.g., colors, shape, etc.) and the classifier design.

In literature a lot of approaches have been published with different capabilities, advantages and limitations. Of course, the implementation of a face detector system inside an embedded device requires ad hoc peculiarities due the limited available resources. The constrained domain forces us to consider methods that are able to guarantee a reasonable trade-off between robustness and computational issues. It is out of the scope of this chapter to provide a detailed review of all related technologies. See [25–29] for more specific details.

In this case the quality of red eye detection greatly depends on the quality of the face detector. Sometimes it is limited to frontal upright faces, while red eye artifacts may be located in profile or three-quarter views of subjects (especially when taking snapshots). Therefore, face detectors with such limitations are not suitable for red eye detection. Another important degree of variability is the age of the subject: Children are difficult to detect, since their faces have different shapes and different features than those of adults. Nonetheless, they have a higher chance to present red eye artifacts, since their pupils are usually more open. Thus, it is important for face detection to be robust both to the angle of view and to the age of the subject.

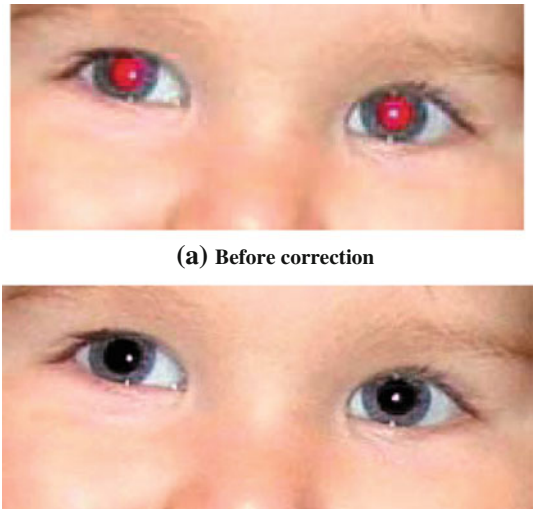
Another important issue related to face detection is that it does not help discard false detections of the face, which are usually critical. An additional constraint which may be imposed is to only accept eyes located in the upper half of the detected face. This helps filter out some false detections (e.g., lips or tongue) but it keeps the ones near the eyes (e.g., details of glasses or pimples on the forehead).

12.2.5 Combining Preprocessing and Classification

Different algorithms based on a two-stage approach have been proposed in literature. In the first stage red eye candidates are detected through preprocessing (e.g., through color-based methods). In the second step the detection is refined making use of a classifier. Zhang et al. presented a two-stage algorithm in [12]. At the first stage, red pixels are grouped and a cascade of heuristic algorithms to deal with color, size and highlight are used to decide whether the grouped region is red eye or not. At the second stage, candidate red eye regions are checked by using adaptive boosting (Adaboost) classifier. Luo and Tretter [22] proposed an algorithm that first uses square concentric templates to assess the candidate red eye regions, and then employs an Adaboost classifier coupled with a set of ad hoc selected Haar-like features for final detection. Multiscale templates are used to deal with the scale of red eye patches. For each scale, a thresholding process has been used to determine which pixels are likely to be red eye pixels. A wide class of techniques makes use of geometric constraints to restrict possible red eye regions in combination with reliable supervised classifiers for decision making. Corcoran et al. [30] proposed an algorithm for real-time detection of flash eye defects in the firmware of a digital camera. The detection algorithm comprises different substeps on *CIELAB* color space to segment artifact regions that are finally analyzed with geometric constraints. Safonov et al. [31] suggested a supervised approach taking into account color information via 3D tables and edge information via directional edge detection filters. In the classification stage a cascade of supervised classifiers including Gentleboost has been used. In [9] an effective two-stage algorithm for red eyes detection was introduced. In the first stage candidate red eyes are extracted from the input image through an image-filtering pipeline. This process is mainly based on red color segmentation in the *HSL* color space coupled with geometric constraints related to the size and the roundness of the red eye regions. In the second stage a multimodally classifier, obtained by using clustering and linear discriminant analysis (LDA), is used to distinguish between true *red eyes* patches versus *other* patches.One of the main contributions of the approach proposed in [9] is to demonstrate that better results are achieved if the multimodally nature of candidate red eyes is taken into account during classification task.

12.3 Red Eye Correction

The goal of red eye correction is to modify the image in such a way that it looks as natural as possible, given the assumption that there are red eye artifacts in the detected zones (according to the eye detector). The correction algorithm may need to adjust the hue, brightness, luminance distribution, and/or even the shape and size of the pupil. Since naturalness of the image is the goal, it is best to use a minimally invasive technique to correct each case. This also means that a way to evaluate the degree of correction of each artifact (either in the detection phase or at the very



(a) Before correction

(b) After correction

Fig. 12.8 In the simplest cases, pupil desaturation produces good results

beginning of the correction phase) is to be preferred, in order to be able to adapt the correction method by taking into account the case under consideration [32].

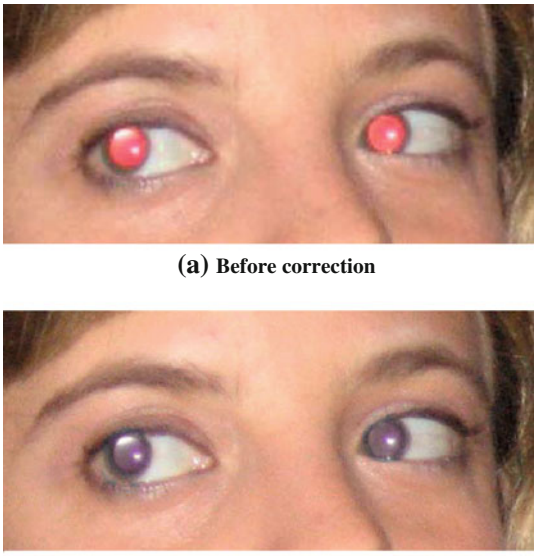
12.3.1 Desaturation

In the simplest cases, the eye has a regular shape, and the artifact only consists in the wrong color of the pupil. In these cases, the solution is equally simple: The red eye is desaturated; its chrominance is (totally or partially) suppressed, while its luminance is left intact or only slightly lowered (Fig. 12.8).

One simple way of desaturating red pupils is to replace each pixel with a gray shade at 80 % of original pixel luminance [18]. An adaptive desaturation may be performed in the CIELAB color space by stretching the lightness values of the pupil so that its darkest point becomes black [33]:

$$\begin{aligned} L^*_{corrected} &= \frac{\max L^*}{(\max L^* - \min L^*)} (L^* - \min L^*) \\ a^*_{corrected} &= 0 \\ b^*_{corrected} &= 0 \end{aligned} \tag{12.9}$$

Desaturation may suffer from a boundary effect: The transition between the corrected and uncorrected area may be noticeable and unpleasant. Moreover, some pixels outside the pupil may be incorrectly considered to be part of the red eye artifact and desaturated. To overcome these problems, a smoothing (usually Gaussian) mask may be used to modulate the strength of the correction. For each pixel (i, j) in the red eye artifact area, let be $c_{original}(i, j)$ its color in the uncorrected image, $c_{target}(i, j)$



(a) **Before correction**

(b) **After correction**

Fig. 12.9 When reflected light spreads over the iris, simple desaturation gives unnatural results

the target color of the correction and $m(i, j)$ the value of the smoothing mask. The final corrected color $c_{corrected}(i, j)$ is then:

$$c_{corrected}(i, j) = c_{target}(i, j) \times m(i, j) + c_{original}(i, j) \times (1 - m(i, j)) \quad (12.10)$$

12.3.2 Inpainting

In the hardest cases, a more invasive correction is needed. Often, the distribution of reflected light is influenced by the direction of the flash with respect to the face. Sometimes eyes present a "washed out" effect, where the reflected light spreads off the pupil onto the iris. In these cases a simple desaturation may yield incorrect and unnatural results (Fig. 12.9).

It is then necessary to use a more complex method to reconstruct a realistic image of the eye. Inpainting may vary from an adaptive recoloring of red pixels to a complete redrawing of iris and pupil [34]. The results, however, tend to be unrealistic, up to the point that they sometimes resemble glass eyes (Fig. 12.10[1]).

[1] Corel Paint Shop Pro red eye removal tool.

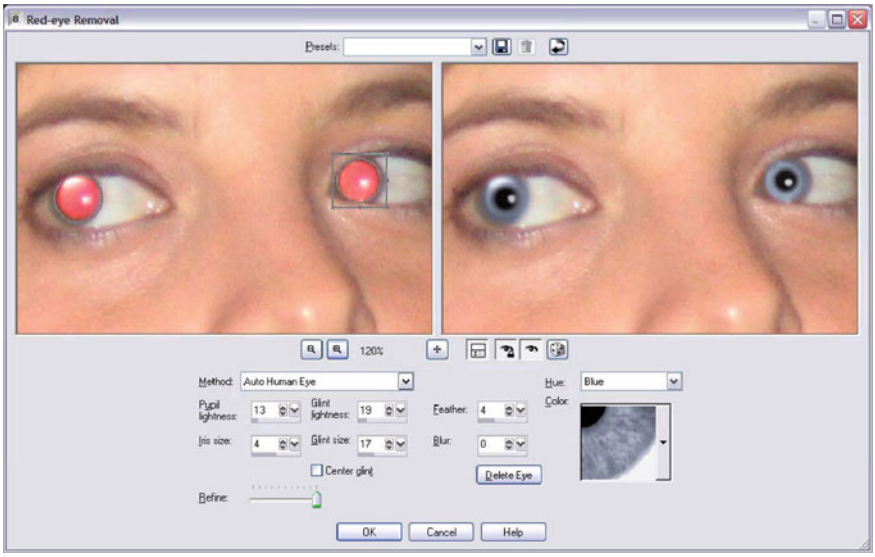


Fig. 12.10 Correction of washed-out red eyes with an inpainting technique

12.3.3 Flash/No Flash

Another way of obtaining simultaneous detection and correction of red eye artifacts is the "flash/no flash" technique [35, 36], which aims to combine the advantages of taking a nonflashed picture and a flashed one. The main idea is to take a high-quality flashed picture and a low-quality nonflashed one, which is used to detect the red eyes and recover the natural colors of the affected zones (Fig. 12.11[2]).

The method works as follows: Two pictures are taken in quick succession. The first one is shot without flash with high sensitivity, large lens aperture and with a short (for a nonflashed picture in low light conditions) exposure time. This yields a dark and noisy picture with small depth of focus, but still suitable to help recover the unaltered colors of the eyes. The second one is a regular flashed picture, which represents the "real" picture to correct. It is important that the two pictures are taken with the same focal length and that very little time elapses in between, in order to prevent misalignment. Search for red eye artifacts is performed in a luminance-chrominance color space, usually CIELAB. The a^* channel is used as a measure of redness. Pixels whose a^* component exceeds a certain threshold are considered red. Among such pixels, those whose difference between the a^* channel in the flashed image and the same channel in the nonflashed image is larger than another threshold are marked as possible red eye pixels. Morphological operators are used to cluster them into blobs, discarding isolated pixels or very narrow regions as noisy results.

[2] Picture taken from Petschnigg et al. [36].

(a) **(b)**

Fig. 12.11 Flash/no flash technique. **a** *Dark* nonflashed picture used to recover the correct color of the eyes; **b** high-quality picture affected by red eye artifacts; **c** corrected picture

To correct red eyes using information from the nonflashed picture, it is important to first compensate differences in color cast between the two images. To this end, for each of the chroma channels a^* and b^*, the difference between the two images is averaged over all non-red eye pixels, thus obtaining a color compensation term. Correction of red eye artifacts is then performed by substituting the chrominance of affected pixels in the flashed image with the chrominance of the corresponding pixels in the nonflashed image, then adding the color compensation term.

The approach is quite simple and theoretically effective, but it presents a number of drawbacks. First of all, the memory and processing requirements double, since there are two pictures being taken in place of one. Moreover, the images may suffer from registration problems, or they may simply be misaligned due to movement of the hand or of the subjects. This makes this method especially unsuitable for snapshots, where people may be caught while moving. Another important issue of this approach is uneven illumination, which is recorded by the nonflashed image but not by the flashed one: A dark shadow on a red detail (such as the shadow of the nose projected on the lips) may trigger a false detection, which in turn causes image degradation (especially if the chrominance of the shaded part is not correctly perceived due to insufficient illumination).

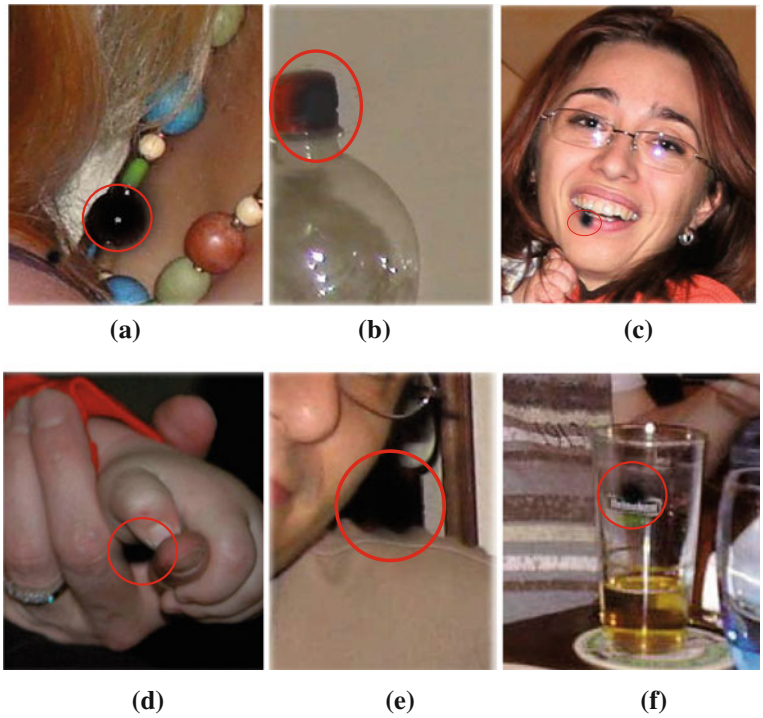


Fig. 12.12 Examples of corrections of false positives. Some are barely noticeable, while others are totally unacceptable

12.4 Correction Side Effects

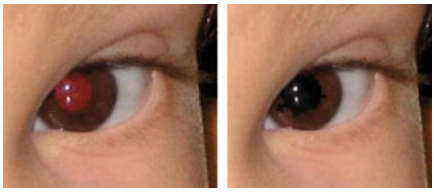
12.4.1 False Positive

One of the biggest issues in red eye removal is false positives in the detection phase. Correcting a red detail falsely detected as a red eye artifact may have a much more displeasing effect than leaving an artifact uncorrected. For this reason, getting as few false positives as possible is more important than catching as many red eyes as possible. Examples of image degradation resulting as correction of false positives are shown in Fig. 12.12.

False positives can be classified according to the severity of the associated degradation risk, as discussed in Sect. 12.5.

(a) Before correction (b) After correction

Fig. 12.13 Partial red eye correction, where the brighter area was not considered to belong to the red pupil



(a) Before correction (b) After correction

Fig. 12.14 Correction of the red pupil extends over the iris, due to red pixels caused by image noise

12.4.2 Partial Detection/Correction

Sometimes eyes are properly detected, but wrongly corrected. In such cases unnatural corrections appear in the final image. Unnatural corrections, like false positives, are very undesirable, since they are often more evident and displeasing than untouched red eyes. One type of unnatural correction is partial correction, caused by an incorrect segmentation of the red eye zone (possibly due to a difference in hue or luminance between the detected and the undetected parts) as shown in Fig. 12.13.

12.4.2.1 Noisy Correction

Noisy correction is another kind of unnatural correction. Noisy corrections appear when, in presence of heavy image noise, red pixels are present around the pupil. In this case, the detector may assume that such pixels belong to the red eye, and correction may spread over the iris, giving a strange and unnatural look to the corrected eye as shown in Fig. 12.14.

It is worthwhile to note that a strong lossy image compression (e.g., low-bitrate JPEG) may cause the same phenomenon; however, in the context of automatic algorithms which act just after the picture is taken, it is reasonable to assume that red eye removal is performed before image compression (to improve red eye detection and to avoid compressing twice).

(a) Before correction **(b)** After correction **(c)** Before correction **(d)** After correction

Fig. 12.15 In some cases, an unnatural luminance distribution is visible in the corrected image (**b**). Sometimes, instead, the eye has a "dead" look, due to the absence of the glint (**d**)

12.4.2.2 Dead Eye

Sometimes red eyes are properly detected, but the corrected image just does not look natural. This may happen when a wrong luminance distribution, caused by reflected light, is kept through the correction and is evident in the resulting image. This may also happen when the color of the corrected pupils is not quite natural, possibly because the correction is not strong enough. Finally, the absence of the glint, which may be due to inpainting or excessive correction, may cause the eye to look "dead", as shown in Fig. 12.15.

12.5 Quality Criteria

The formulation of a quality metric allows us to choose the best solution and to adjust parameters of the algorithm in the best way. To achieve quality control on a red eye removal algorithm is a challenging issue. Usually the quality of the algorithm is estimated considering the ratio between corrected eyes and false positives. Obviously this is strictly related to the nature of the database and the quantity of images. Safonov [23] introduced an interesting quality metric that allows users to remove correlation between quantity and quality.

First of all, the author enumerated all possible cases; further he prioritized them using an analytic hierarchy process (AHP) [37]. Obviously a representative set of photos affected by red eye defects should be used for calculation of these unwanted cases. Furthermore good solutions must have low false negatives (FN) and false positives (FP); ideally FN and FP are equal to zero. However the severity of the FPs differs significantly. Almost indistinguishable small FP on foreground is undesirable but sometimes allowable. Visible FP on the foreground, especially on human faces and bodies, is absolutely not allowable; such FP artifacts damage a photo more than red eyes. Therefore he divided FPs in two classes: FP_c is the number of critical FP and FP_n is the number non-critical FP.

A similar situation is described for the FNs. Several red eye regions are relatively large and well distinguishable; other regions are small and have low local contrast.

Table 12.1 Analytic hierarchy process table, where the coefficients a_i, that refer to the assigned importance values in each row, are used to estimate the Geometric Mean

Req. quality	FN_m	FN_d	FP_c	FP_n	N_p	C_i	C_n	$\sqrt[7]{\prod_{i=1}^{7} a_i}$	Weight %
FN_m	1.00	5.00	0.20	5.00	1.00	0.20	5.00	1.26	13.13
FN_d	0.20	1.00	0.33	5.00	0.20	0.20	5.00	0.68	7.08
FP_c	5.00	3.00	1.00	3.00	5.00	5.00	5.00	3.43	35.73
FP_n	0.20	0.20	0.33	1.00	0.20	0.20	1.00	0.34	3.54
N_p	1.00	5.00	0.20	5.00	1.00	1.00	5.00	1.58	16.46
C_i	5.00	5.00	0.20	5.00	1.00	1.00	5.00	1.99	20.72
C_n	0.20	0.20	0.20	1.00	0.20	0.20	1.00	0.32	3.33

The weight is estimated in percentage from the sum of the Geometric Means (=9.60)

Detection of the first red eyes is defined as mandatory by Safonov, whereas detection of the second regions is desirable. In accordance with such hypotheses he divided all FN in two groups: FN_m is defined as the number of regions which are mandatory for detection; FN_d is the number of regions which are desirable for detection.

One more unwanted situation is the correction of only one eye from a pair. For semiautomatic approaches it is not so crucial because users have the possibility to correct the second eye manually, but for embedded implementations it is quite unpleasant. N_P is then defined as the number of faces with one corrected eye from pair of red eyes.

The retouching quality is important too. Regarding correction, Safonov distinguished two cases: If the corrected eye looks worse than the original red eye, for example, only part of the red region is corrected, it is an irritating case; it is noticeable that eye has been corrected but it does not irritate strongly. Accordingly C_I is the number of irritating cases, and C_n is the number of situations when retouching is noticeable.

As described above, Safonov used prioritization of the factors through AHP table (Table 12.1) according to observer's opinions. The simplest way of filling the table is: If left item is more important than top, then cell is assigned to 5; if the severities of the two items are the same then cell is set to 1; if top item is more important than left then cell is set to 1/5. Taking into account weights from AHP table, and taking into consideration a global weight of 10 for all the features, Safonov proposed the following quality criterion:

$$\begin{aligned} Q_c = 1 &- \frac{1}{N_t}\left(1.3 \times FN_m + 0.7 \times FN_d\right) \\ &- \frac{1}{N_t}\left(3.6 \times FP_c + 0.4 \times FP_n + 1.6 \times N_p\right) \\ &- \frac{1}{N_t}\left(2.1 \times C_i + 0.3 \times C_n\right) \end{aligned} \tag{12.11}$$

where N_t is total number of red eyes.

12.6 Red Eye Detection and Correction Through Cluster-Based Boosting on Gray Codes

The red eyes removal pipeline detailed in the following subsection uses three main steps to identify and remove red eyes artifacts. First, candidate red eyes patches are extracted, then classified to distinguish between eyes and non-eye patches. Finally, correction is performed on detected red eyes.

12.6.1 Red Patch Extraction

To extract the red eye candidates, we first built a color model from the training set to detect pixels belonging to possible red eye artifacts. We constructed red eye pixel and non red eye pixel histogram models using a set of pixels of the training images. Specifically, for each image of the training set, the pixels belonging to red eyes artifacts have been labeled as red-eye-pixels (*REP*), whereas the surrounding pixels within a window of fixed size have been labeled as non-red-eye-pixels (*NREP*). The labeled pixels (in both RGB and HSV spaces) have been mapped in a three-dimensional space $C_1 \times C_2 \times C_3$ obtained taking into account the first three principal components of the projection through principal component analysis [38]. By using the principal component analysis, the original six-dimensional space of each pixel considered in both RGB and HSV color domains, is transformed into a reduced three-dimensional space maintaining as much of the variability in the data as possible. This is useful to reduce the computational complexity related to the space dimensionality. We used a 3D histogram with $64 \times 64 \times 64$ bins in the $C_1 \times C_2 \times C_3$ space. Since most of the sample pixels of the training set lie within three standard deviations of the mean, each component C_i has been uniformly quantized in 64 values taking into account the range $[-3\lambda_i, +3\lambda_i]$, where λ_i is standard deviation of the ith principal component (i.e., the ith eigenvalue). The probability that a given pixel belongs to the classes *REP* and *NREP* is computed as follows:

$$P(C_1, C_2, C_3|REP) = \frac{h_{REP}[C_1, C_2, C_3]}{T_{REP}} \tag{12.12}$$

$$P(C_1, C_2, C_3|NREP) = \frac{h_{NREP}[C_1, C_2, C_3]}{T_{NREP}} \tag{12.13}$$

where $h_{REP}[C_1, C_2, C_3]$ is the REP count contained in bin $C_1 \times C_2 \times C_3$ of the 3D histogram, $h_{NREP}[C_1, C_2, C_3]$ is the equivalent count for *NREP*, T_{REP} and T_{NREP} are the total counts of red eye pixels and non red eye pixels respectively. We derive a *REP* classifier through the standard likelihood ratio approach. A pixel is labeled *REP* if

$$P(C_1, C_2, C_3|REP) > \alpha P(C_1, C_2, C_3|NREP) \tag{12.14}$$

where α is a threshold which is adjusted to maximize correct detection and minimize false positives. Note that a pixel is assigned to the *NREP* class when both probabilities are equal to zero.

Employing such filtering, a binary map with the red zones is derived. To remove isolated red pixels, a morphology operation of closing is applied to this map. In our approach we have used the following 3×3 structuring element:

$$m = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{12.15}$$

Once the closing operation has been accomplished, a search of the connected components is achieved using a simple scanline approach. Each group of connected pixels is analyzed making use of simple geometric constraints. As in [20], the detected regions of connected pixels are classified as possible red eye candidates if the geometrical constraints of *size* and *roundness* are satisfied. Specifically, a region of connected red pixels is classified as a possible red eye candidate if the following constraints are satisfied:

- The size S_i of the connected region i is within the range $[Min_s, Max_s]$, which defines the allowable size for eyes.
- The binary roundness constraint R_i, of the connected region i is verified:

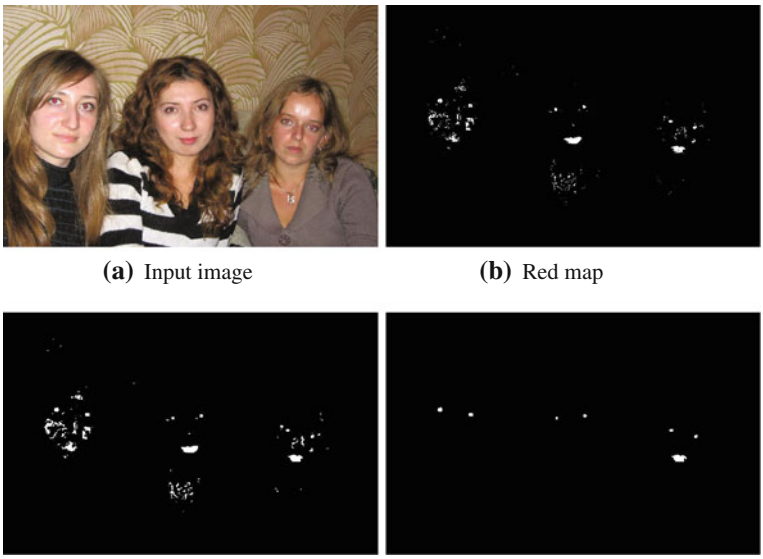
$$R_i = \begin{cases} True \quad \rho_i \in [Min_\rho, Max_\rho];\ \eta_i \leq Max_\eta;\ \xi_i \gg 0 \\ \\ False\ otherwise \end{cases} \tag{12.16}$$

  where

  - $\rho_i = \frac{4\pi \times A_i}{{P_i}^2}$ is the ratio between the estimated area A_i and the perimeter P_i of the connected region; the closer this value to 1, the more the shape will be similar to a circle.
  - $\eta_i = \max\left(\frac{\Delta_{x_i}}{\Delta_{y_i}}, \frac{\Delta_{y_i}}{\Delta_{x_i}}\right)$ is the distortion of the connected region along the axes.
  - $\xi_i = \frac{A_i}{\Delta_{x_i}\Delta_{y_i}}$ is the filling factor; the closer this parameter to 1, the more the area is filled.

The parameters involved in the aforementioned filtering pipeline have been set through a learning procedure as discussed in Sect. 12.6.5.

In Fig. 12.16 all the involved steps in filtering pipeline are shown. The regions of connected pixels which satisfy the geometrical constraints are used to extract the red eye patches candidates from the original input image (Fig. 12.17). The derived patches are reasembled to a fixed size (i.e., 30×30 pixels) and converted into gray code [39] for further classification purpose (Fig. 12.18). Gray code representation



(a) Input image **(b)** Red map

(c) Closing operation **(d)** Final candidates

Fig. 12.16 Filtering pipeline on an input image. **a** Input image, **b** red map, **c** closing operation and **d** final candidates

Fig. 12.17 Examples of possible candidates after red patches extraction

allows us to have a natural way (e.g., no strong transaction between adjacent values) to pickup the underlying spatial structures of a typical eye.

The gray levels of an m-bit gray-scale image (i.e., a color channel in our case) is represented in the form of the base 2-polynomial:

$$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \cdots + a_1 2^1 + a_0 2^0 \tag{12.17}$$

Based on this property, a simple method of decomposing the image into a collection of binary images is to separate the m coefficients of the polynomial into m bit planes. The m-bit Gray Code $(g_{m-1} \ldots g_2, g_1, g_0)$ related to the polynomial in Eq. (12.17) can be computed as follows:

$$g_i = \begin{cases} a_i \bigoplus a_{i+1} & 0 \leq i \leq m-2 \\ a_{m-1} & i = m-1 \end{cases} \tag{12.18}$$

Fig. 12.18 Example of gray code planes on the three RGB channels of a red eye patch

where $\oplus$ denotes the exclusive OR operation. This code has the unique property that successive code words differ only by one bit position. Thus, small changes in gray level are less likely to affect all m bit planes.

12.6.2 Red Patch Classification

The main aim of the classification stage is the elimination of false positive red eyes in the set of patches obtained performing the filtering pipeline described in previous section. At this stage we deal with a binary classification problem. Specifically, we want to discriminate between eye and non-eye patches. To this aim we employ an automatic learning technique to make accurate predictions based on past observations. The approach we use can be summarized as follows: Start by gathering as many examples as possible of both eye and non-eye patches. Next, feed these examples, together with labels indicating if they are eyes or not, to a machine-learning algorithm which will automatically produce a classification rule. Given a new unlabeled patch, such a rule attempts to predict if it is eye or not.

Building a rule that makes highly accurate predictions on new test examples is a challenging task. However, it is not hard to come up with rough weak classifiers that are only moderately accurate. An example of such a rule for the problem under consideration is something like the following: "If the pixel p located in the sclera region of the patch under consideration is not white, then predict it is non-eye". In this case such a rule is related to the knowledge that the white region corresponding to the sclera should be present in an eye patch. On the other hand, such a rule will cover all possible non-eye cases; for instance, it is correct to say nothing about what to predict if the pixel p is white. Of course, this rule will make predictions that are significantly better than random guessing. The key idea is to find many weak classifiers and combine them in a proper way to derive a single strong classifier.

Among others, Boosting [40–42] is one of the most popular procedure for combining the performance of weak classifiers in order to achieve a better classifier. We use a boosting procedure on patches represented as gray codes to build a strong classifier that is useful to distinguish between eye and non-eye patches.

Specifically, boosting is used to select the positions $\{p_1, \ldots, p_n\}$ corresponding to n gray code bits that best discriminate between the classes eye versus non-eye, together with n associated weak classifiers of the form:

$$h_i(\mathbf{g}) = \begin{cases} a_i & g_{p_i} = 1 \\ b_i & g_{p_i} = 0 \end{cases} \tag{12.19}$$

where $\mathbf{g} = [g_1, g_2, \ldots, g_D]$ is the gray code vector ($g_i \in \{0, 1\}$) of size $D = 30 \times 30 \times 3 \times 8$ corresponding to a 30×30 patch extracted as described in previous section. The parameters a_i and b_i are automatically learned by Gentleboost procedure [40], as explained in Sect. 12.6.3. The classification is obtained considering the sign of the learned additive model:

$$H(\mathbf{g}) = \sum_{i=1}^{n} h_i(\mathbf{g}) \tag{12.20}$$

where $n \ll D$ indicates the number of weak classifiers involved in the strong classifiers H.

The rationale behind the use of gray code representation is the following. In the gray code space just a subset of all possible bit combinations are related to the eyes patches. We wish to select those bits that usually differ in terms of binary value between eye and non-eye patches. Moreover, by using gray code representation rather than classic bit planes decomposition we reduce the impact of small changes in intensity of patches that could produce significant variations in the corresponding binary code [39].

In Fig. 12.19 an example of $n = 1000$ gray code bits selected with a Gentleboost procedure is reported. Selected bits are shown as black or white points on the different gray code planes. This map indicates that a red eye patch should have 1 in the positions coloured in white and 0 in the positions coloured in black. Once gray code bits and the corresponding weak classifiers parameters are learned, a new patch can be classified by using the sign of Eq. (12.20).

The approach described above does not take into account spatial relationship between selected gray code bits. Spatial information is useful to strengthen the classification task (e.g., pupil is surrounded by sclera). To overcome this problem we coupled the gray code bits selected at the first learning stage to obtain a new set of binary features.

Due to the multi-modal nature of the patches involved in our problem (i.e., colours, orientation, shape, etc.), a single discriminative classifier could fail during classification task. To get through this weakness we propose to perform first a clustering of the input space and then to apply the two-stage boosting approach described above on each cluster. More specifically, during the learning phase, the patches are clustered by using K-means [38] in their original color space producing the subsets of the input patches with the relative prototypes; hence the two-stage of boosting described above are performed on each cluster. During the classification stage, a new patch

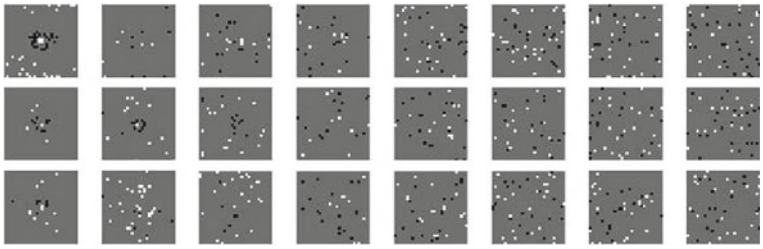


Fig. 12.19 Selected *gray* code bits

is first assigned to a cluster according to the closest prototype and then classified taking into account the two additive models properly learned for the cluster under consideration. Experimental results reported in Sect. 12.6.5 confirm the effectiveness of the proposed strategy.

12.6.3 Boosting for Binary Classification Exploiting Gray Codes

Boosting provides a way to sequentially fit additive models of the form in Eq. (12.20) optimizing the following cost function [40]:

$$J = E[e^{-yH(\mathbf{g})}], \tag{12.21}$$

where $y \in \{-1, 1\}$ is the class label associated to the feature vector $\mathbf{g}$. In this work $y = 1$ is associated to the eye class, whereas $y = -1$ is the label associated to the non-eye class. The cost function in the Eq. (12.21) can be thought as a differentiable upper bound of the misclassification rate [41].

There are many ways to optimize this function. A simple and numerically robust way to optimize this function is called Gentleboost [40]. This version of a boosting procedure outperforms other boosting variants for computer vision tasks (e.g., face detection) [43]. In Gentleboost the optimization of Eq. (12.21) is performed minimizing a weighted squared error at each iteration [44]. Specifically, at each iteration i the strong classifier H is updated as $H(\mathbf{g}) := H(\mathbf{g}) + h_{best}(\mathbf{g})$, where the weak classifier h_{best} is selected in order to minimize the second-order approximation of the cost function in the Eq. (12.21):

$$h_{best} = \arg\min_{h_d} J(H(\mathbf{g}) + h_d(\mathbf{g})) \simeq \arg\min_{h_d} E[e^{-yH(\mathbf{g})}(y - h_d(\mathbf{g}))^2] \tag{12.22}$$

Defining as $w_j = e^{-y_j H(\mathbf{g}_j)}$ the weight for the training sample j and replacing the expectation with an empirical average over the training data, the optimization reduces to minimizing the weighted squared error:

$$J_{wse}(h_d) = \sum_{j=1}^{M} w_j (y_j - h_d(\mathbf{g}_j))^2, \tag{12.23}$$

where M is the number of samples in the training set.

The minimization of J_{wse} depends on the specific form of the weak classifiers h_d. Taking into account the binary representation of samples (i.e., the gray code of each patch), in the present proposal we define the weak classifiers as follows:

$$h_d(\mathbf{g}) = \begin{cases} a_d & if\ g_d = 1 \\ b_d & if\ g_d = 0 \end{cases} \tag{12.24}$$

In each iteration the optimal a_d and b_d for each possible h_d can be obtained through weighted least squares as follows:

$$a_d = \frac{\sum_{j=1}^{M} w_j y_j \delta(g_d = 1)}{\sum_{j=1}^{M} w_j \delta(g_d = 1)} \tag{12.25}$$

$$b_d = \frac{\sum_{j=1}^{M} w_j y_j \delta(g_d = 0)}{\sum_{j=1}^{M} w_j \delta(g_d = 0)} \tag{12.26}$$

The best weak classifier h_{best} is hence selected in each iteration of the boosting procedure such that the cost of Eq. (12.23) is the lowest:

$$h_{best} = \arg\min_{h_d} J_{wse}(h_d) \tag{12.27}$$

Finally, before a new iteration, the boosting procedure makes the following multiplicative update to the weights corresponding to each training sample:

$$w_j := w_j e^{-y_j h_{best}(\mathbf{g}_j)} \tag{12.28}$$

This update increases the weight of samples which are misclassified (i.e., for which $y_j H(\mathbf{g}_j) < 0$), and decreases the weight of samples which are correctly classified.

The procedures employed for learning and classification in the proposed representation are summarized in Algorithms 12.1 and 12.2. In the learning stage we initialize the weights corresponding to the elements of the training set such that the number of the samples within each class is taken into account. This is done to overcome the problems that can occur due to the unbalanced number of training samples within the considered classes.

Algorithm 12.1 Learning

Input: A set of gray code vectors $\mathbf{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_M\}$, and corresponding labels $\mathbf{Y} = \{y_1, \ldots, y_M\}$
Input: A strong classifier $H(\mathbf{g}) = \sum_{i=1}^{n} h_i(\mathbf{g})$
Begin
$C^+ := \{j | y_j = 1\}$
$C^- := \{j | y_j = -1\}$
$w_{j \in C^+} := \frac{1}{2|C^+|}$
$w_{j \in C^-} := \frac{1}{2|C^-|}$
for $i = 1, 2, \ldots, n$ **do**
  for $d = 1, 2, \ldots, D$ **do**
    $a_d^* := \frac{\sum_{j=1}^{M} w_j y_j \delta(g_d = 1)}{\sum_{j=1}^{M} w_j \delta(g_d = 1)}$
    $b_d^* := \frac{\sum_{j=1}^{M} w_j y_j \delta(g_d = 0)}{\sum_{j=1}^{M} w_j \delta(g_d = 0)}$
    $h_d^*(\mathbf{g}) := \begin{cases} a_d^* \; if \; g_d = 1 \\ b_d^* \; if \; g_d = 0 \end{cases}$
    $J_{wse}(h_d^*) := \sum_{j=1}^{M} w_j (y_j - h_d^*(\mathbf{g}_j))^2$
  end for
  $p_i := \arg\min_d J_{wse}(h_d^*)$
  $a_i := a_{p_i}^*$
  $b_i := b_{p_i}^*$
  $h_i(\mathbf{g}) := \begin{cases} a_i \; if \; g_{p_i} = 1 \\ b_i \; if \; g_{p_i} = 0 \end{cases}$
  $w_j := w_j e^{-y_j h_i(\mathbf{g}_j)}$
end for
$H(\mathbf{g}) := \sum_{i=1}^{n} h_i(\mathbf{g})$

Algorithm 12.2 Classification

Input: The strong classifier H, and a new gray code sample $\mathbf{g}$ to be classified
Input: The inferred class $y \in \{-1, 1\}$
Begin
$y := sign(H(\mathbf{g}))$

12.6.4 Correction of Detected Red Eyes

Once the red eyes have been detected the correction step is performed. Usually the red eye artifact consists of a red pupil with a white glint. This area absorbs light and thus should be dark. To transform the red pupil to a dark region, a desaturation and a brightness reduction is accomplished [6, 7]. The region of connected red pixels is used to fix the area that must be desaturated. To prevent an unpleasant transition from the iris to the pupil, the red eye artifact is replaced by a mask with equal dimensions, where each value is used as weighted brightness/desaturation reduction factor. The correction mask M is based on a 32×32 fixed point LUT with Gaussian shape (Fig. 12.20). The mask is resized through a bilinear resampling to fit the dimension of the region of connected red pixels under consideration.

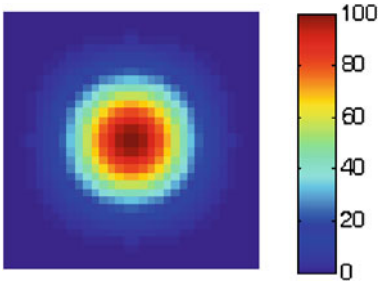


Fig. 12.20 Brightness saturation mask

Let I_c^r be the channel $c \in \{R, G, B\}$ of a region of interest r within the image I. For each channel $c \in \{R, G, B\}$ the pixels (x, y) belonging to the region I^r are corrected as follows:

$$I_c^r(x, y) = \begin{cases} I_c^r(x, y) & [I_R^r(x, y), I_G^r(x, y), I_B^r(x, y)] \in W \\ \frac{I_G^r(x,y)}{M(x,y)} & otherwise \end{cases} \tag{12.29}$$

where W is a surrounding of the "white" color, which can slightly vary in terms of lightness, hue and saturation. This means that to prevent the glint from disappearing, only red pixels are desaturated (the whitish pixels are excluded from the brightness processing).

12.6.5 Experimental Settings and Results

The proposed red eye removal pipeline was tested on a dataset of 390 images in which 1,049 red eyes were manually labeled. The dataset was collected from various sources, including digital single-lens reflex (DSLR) cameras, compact cameras, personal collections and Internet photos. Single red eyes, as well as high variability of red eye colors, poses and shapes have been considered in building the dataset. In order to accurately assess the proposed approach, the size of the eyes to be detected in the collected images must be small enough to ensure that even the smallest red eyes can be detected and corrected. The basic requirement considered in our experimental phase is that the red eyes must be accurately detected and corrected up to three meters distance from the camera. Table 12.2 presents the estimated eye sizes, in pixels, for XGA image size (1024 × 768), with the assumption that the average eye is directed to the camera. In this paper the collected images were considered with a XGA image resolution, and the minimum and maximum estimated pupil diameter (Table 12.2) were taken into account in building the dataset for testing purposes.

Fig. 12.21 Example of clusters prototypes obtained in a LOOCV run

Table 12.2 Estimated eye sizes taking into account the distance from the camera

Distance (m)	0.20	0.40	0.60	0.80	1.00	1.20	1.40	1.60	1.80	2.00	2.20	2.40	2.60	2.80	3.00
Diameter (pixels)	52	26	17	13	10	9	7	7	6	5	5	4	4	4	3

For each image of the dataset, the pixels belonging to red eye artifacts have been manually labeled as REPs. The parameters Min_h, Max_h, t_s, Min_s, Max_s, Min_ρ, Max_ρ, and Max_η involved in the first stage of the proposed approach (see Sect. 12.6.1) have been learned, taking into account the true and false red eyes pixels within the labeled dataset. To this aim, a full search procedure on a grid of equispaced points in the eight-dimensional parameters' space was employed. For each point of the grid, the correct detection and false positives rates of the true REPs within the dataset were obtained. The tuple of parameters with the best trade-off between correct detection and the false positives was used to perform the final filtering pipeline. A similar procedure was employed to determine the subspace W of the RGB space involved in the correction step to identify pixels belonging to the glint area.

In order to evaluate the classification performance of the proposed method, the leave-one-out cross-validation procedure (LOOCV) was employed. Each run of LOOCV involved a single image as test, and the remaining images as training data. This is repeated to guarantee that each input image is used once as test image. At each run of LOOCV the parameters of the filtering pipeline were set to maximize correct detection and minimize false positives. At each run of LOOCV the training images were clustered and then the two-stage boosting approach described in Sect. 12.6.2 was performed on each cluster. Seven clusters (shown in Fig. 12.21) and 800 binary features for the additive classifiers corresponding to the clusters were used on each LOOCV run. The maximum number of iterations used by the boosting procedure to obtain the 800 binary features was 1400. The final results have been obtained by averaging the results of the overall LOOCV runs.

Taking into account both the filtering and the classification stages, the hit-rate of the proposed red eye detector is 83.41 %. This means that 875 red eyes were correctly detected with respect to the 1049 red eyes of the 390 input images, whereas only 34 false positives were introduced. In Fig. 12.22 the training ability increasing the number of bits is shown in terms of hit rates (Fig. 12.22a) and false positives (Fig. 12.22b).

In Fig.12.23 two examples of misclassified patches are reported. In Fig. 12.23a a "golden" eye is depicted (another possible artifact due to similar acquisition problems). The underlying structure in Fig. 12.23b is probably the main reason of mis-

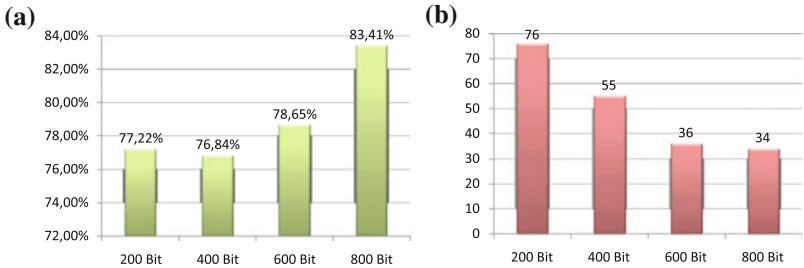
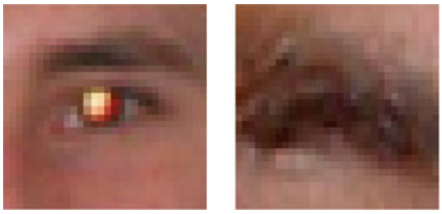


Fig. 12.22 Performances with increasing number of bits

Fig. 12.23 Examples of misclassified patches



(a) False negative **(b)** False positive

Table 12.3 Comparison of different configurations

Configuration	Hit rate (%)	False positives
Gray codes	75.98	47
Gray codes + clustering	77.51	44
Gray codes + spatial relationship	79.31	36
Gray codes + clustering + spatial relationship	**83.41**	**34**

classification. Results reported in Table 12.3 confirm the effectiveness of the rationale behind the proposed method.

To properly evaluate the overall red eyes removal pipeline, the qualitative criterion (see Sect. 12.5) was adopted to compare the proposed solution with respect to existing automatic solutions. The proposed pipeline has been compared with respect to the following automatic (mainly commercial) solutions: Volken [20], NikonView V6.2.7, KodakEasyShare V6.4.0, StopRedEye! V1.0, HP RedBot, Arcsoft PhotoPrinter V5, Cyberlink MediaShow. Experiments were done using effective commercial software and the implementation of [20] provided by the authors. The NikonView approach is mainly based on [30].

As reported in Table 12.4, the proposed approach has obtained the best performances in terms of both hit rate and quality criterion. Moreover, the proposed approach outperforms the method we presented in [9] also in terms of computational complexity.

Table 12.4 Quality score of different red eye removal approaches

Method	FN_m	FN_d	FP_c	FP_n	N_p	C_i	C_n	Q_c	Hit rate (%)
Cyberlink MediaShow	270	86	40	19	39	122	**61**	0.1423	66.06
Volken et al. [20]	179	117	150	1540	83	**17**	79	−0.5851	71.78
KodakEasyShare V6.4.0	194	99	5	20	**5**	104	100	0.4243	72.07
HP RedBot	174	109	26	45	85	99	150	0.2345	73.02
NikonView V6.2.7	143	116	6	29	88	124	129	0.2944	75.31
StopRedEye! V1.0	124	125	8	12	83	81	91	0.4161	76.26
Arcsoft PhotoPrinter V5	132	103	10	78	80	89	82	0.3800	77.60
Battiato et al. [9]	122	85	**2**	**2**	60	20	64	**0.6346**	80.26
Proposed Pipeline	**114**	**60**	9	25	46	34	79	0.6174	**83.41**

12.6.5.1 Computational Complexity

To evaluate the complexity, a deep analysis has been performed by running the proposed pipeline on an ARM926EJ-S processor instruction set simulator. We have chosen this specific processor because it is widely used in embedded mobile platforms. The CPU is run at 300 MHz, and both data and instruction caches have been fixed to 32 KB. The bus clock has been set to 150 MHz, and the memory read/write access time is 9 ns. The algorithm has been implemented using bitwise operators to work on colour maps and fixed-point operations. Due to the dependence of the operations on the number of red clusters found in the image, we have analyzed a mid case, that is, an image containing around 40 potential red eye zones, but only two of them are real eyes to be corrected.

Table 12.5 contains a report of the performance of the main steps of the proposed pipeline, assuming to work on a XGA version (scaled) of the image: the redness detection (Color Map), the processing on the generated maps (Morphological Operations), the candidate extraction, the classification step and finally the correction of the identified eyes. The performance information reported in Table 12.5 is related to the following computational resources:

Instructions: Counts the executed ARM instructions;
Core cycles: Core clock ticks needed to make the Instructions;
Data (D$): Read/write Hits and Misses, cache memory hits and misses;
Seq and Non Seq: Sequential and nonsequential memory accesses;
Idle: Represents bus cycles when the instruction bus and the data bus are idle, that is, when the processor is running;
Busy: Counts busy bus cycles, that is when the data are transferred from the memory into the cache;
Wait States: The number of bus cycles introduced when waiting to access the RAM (is an indicator of the impact of memory latencies);
Total: Is the total number of cycles required by the specific function, expressed in terms of bus cycles;

Table 12.5 Performances of the main steps of the proposed pipeline

	Color map	Morph. oper.	Candidate extr.	Classification	Correction
Instructions	19.845.568	22.990.051	9.418.650	4.446.349	1.698.946
Core cycles	28.753.276	30.489.180	16.407.293	5.668.496	2.390.279
D$ R hits	4.722.760	2.903.178	2.504.092	945.959	205.188
D$ W hits	97.636	261.213	428.924	135.634	94.727
D$ R misses	75.495	6.293	5.666	3.450	244
D$ W misses	2	193.891	3.290	24.069	1.133
SEQ	538.136	17.486.089	48.539	40.177	4.100
NON-SEQ	77.321	122.234	9.841	22.366	1.533
IDLE	16.282.401	7.325.256	10.345.379	3.203.188	1.372.407
Wait states	615.457	253.103	58.380	62.543	5.633
Total	17.513.316	16.208.789	10.462.139	3.328.274	1.383.673
Milliseconds	**117**	**108**	**70**	**22**	**9**

Milliseconds: Time required by the specific function expressed in milliseconds.

The overall time achieved on this mid-case is 326 ms. The table highlights the efficiency of the classifier, because it is mainly based on bit comparisons. Considering patches scaled at 32×32 before the classification stage, the classifier is essentially a comparison of 32×32 bit words for each channel with complexity in the range of one operation per pixel. For this reason it is very fast and light. Also the correction is very light because, as explained in Sect. 12.6.4, it is based on the resampling of a precomputed Gaussian function. The impact on memory is valuable only on the map processing, where data are processed several times, whereas in the remaining steps of the pipeline the weight of the instructions determines the main part of process timing.

We cannot compare the performances and complexity of our methodology with other methods because the other proposed methods are commercial ones, hence the related codes are not available for the analysis.

12.7 Summary and Conclusion

Since the extensive introduction of mobile devices with embedded cameras and flashgun, automatic detection and correction of red eyes have become important tasks. In this chapter we have reviewed different techniques for red eyes detection and correction. Moreover, an advanced pipeline which makes use of a two-stage approach has been discussed. Future work in this field should be devoted to deal with detection and correction of other artifacts (eg., “golden eye” or ‘silver eye”).

References

1. Mir, J. M.: Apparatus & method for minimizing red-eye in flash photography. U.S. Patent, no. US4285588 (1981)
2. Battiato, S., Bruna, A. R., Messina, G., Puglisi, G.: Image Processing for Embedded Devices. Bentham Science Publisher, Karachi (2010)
3. Adobe Photoshop, www.adobe.com/products/photoshop
4. Corel Paint Shop Pro, www.jasc.com
5. ACDSee, www.acdsee.com
6. Gasparini, F., Schettini, R.: Automatic red-eye removal for digital photography. In: Lukac, R. (ed.) Single-Sensor Imaging: Methods and Applications For Digital Cameras, pp. 429–457. CRC Press, Boston (2008)
7. Messina, G., Meccio, T.: Red eye removal. In: Battiato, S., Bruna, A.R., Messina, G., Puglisi, G. (eds.) Image Processing for Embedded Devices. Applied Digital Imaging Ebook Series. Bentham Science, Karachi (2010)
8. Gasparini, F., Schettini, R.: A review of redeye detection and removal in digital images through patents. Recent Pat. Electr. Eng. **2**(1), 45–53 (2009)
9. Battiato, S., Farinella, G.M., Guarnera, M., Messina, G., Ravì, D.: Red-eyes removal through cluster based linear discriminant analysis. In: International Conference on Image Processing (ICIP 2010), Hong Kong (2010)
10. Battiato, S., Farinella, G.M., Guarnera, M., Messina, G., Ravì, D.: Boosting gray codes for red eyes removal. In: International Conference on Pattern Recognition (ICPR 2010), Instanbul (TK) (2010)
11. Battiato, S., Farinella, G.M., Guarnera, M., Messina, G., Ravì, D.: Red-eyes removal through cluster based boosting on gray codes. EURASIP Journal on Image and Video Processing, Special Issue on Emerging Methods for Color Image and Video Quality Enhancement, 2010, pp. 1–11 (2010)
12. Zhang, L., Sun, Y., Li, M., Zhang, H.: Automated red-eye detection and correction in digital photographs. In: International Conference on Image Processing (2004)
13. Held, A.: Model-based correction of red-eye defects. In: IS&T Color Imaging Conference (CIC-02), pp. 223–228 (2002)
14. Gaubatz, M., Ulichney, R.: Automatic red-eye detection and correction. In: International Conference on Image Processing (2002)
15. Smolka, B., Czubin, K., Hardeberg, J.Y., Plataniotis, K.N., Szczepanski, M., Wojciechowski, K.W.: Towards automatic redeye effect removal. Pattern Recognit. Lett. **24**(11), 1767–1785 (2003)
16. Gasparini, F., Schettini, R.: Automatic redeye removal for smart enhancement of photos of unknown origin. In: Visual Information and Information Systems (VISUAL-2005). Lecture Notes in Computer Science, vol. 3736, pp. 226–233 (2005)
17. Willamowski, J., Csurka, G.: Probabilistic automatic red eye detection and correction. In: IEEE International Conference on Pattern Recognition (ICPR-06), pp. 762–765 (2006)
18. Patti, A. J., Konstantinides, K., Tretter, D., Lin, Q.: Automatic digital redeye reduction. In: International Conference on Image Processing, Chicago (1998)
19. Benati, P., Gray, R., Cosgrove, P.: Automated detection and correction of eye color defects due to flash illumination. U.S. Patent, no. US5748764 (1998)
20. Volken, F., Terrier, J., Vandewalle, P.: Automatic red-eye removal based on sclera and skin tone detection. In: European Conference on Color in Graphics, Imaging and Vision, pp. 359–364 (2006)
21. Ferman, A. M.: Automatic detection of red-eye artifacts in digital color photos. In: International Conference on Image Processing (2008)
22. Luo, H., Yen, J., Tretter, D.: An efficient automatic redeye detection and correction algorithm. In: International Conference on Pattern Recognition (2004)
23. Safonov, I.V.: Automatic red-eye detection. In: International conference on the Computer Graphics and Vision (2007)

24. Schildkraut, J. S., Gray, R. T.: A fully automatic redeye detection and correction algorithm. In: International Conference on Image Processing (2002)
25. Yang, M.-H., Kriegman, D.J., Ahuja, N.: Detecting faces in images: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **24**(1), 34–58 (2002)
26. Hsu, R.L., Abdel-Mottaleb, M., Jain, A.K.: Face detection in color images. IEEE Trans. Pattern Anal. Mach. Intell. **24**(5), 696–706 (2002)
27. Viola, P., Jones, M.: Robust real-time face detection. Int. J. Comput. Vis. **57**(2), 137–154 (2004)
28. Hongliang, L., Ngan, K.N., Qiang, L.: Faceseg: automatic face segmentation for real-time video. IEEE Trans. Multimed. **11**(1), 77–88 (2009)
29. Phung, S.L., Bouzerdoum, A., Chai, D.: Skin segmentation using color pixel classification: analysis and comparison. IEEE Trans. Pattern Anal. Mach. Intell. **27**(1), 148–154 (2005)
30. Corcoran, P., Bigioi, P., Steinberg, E., Pososin, A.: Automated in-camera detection of flash eye-defects. In: International Conference on Consumer Electronics (2005)
31. Safonov, I.V., Rychagova, M.N., Kang, K., Kim, S.H.: Automatic red eye correction and its quality metric. In: SPIE Electronic Imaging (2008)
32. Marchesotti, L., Bressan, M., Csurka, G.: Safe red-eye correction plug-in using adaptive methods. In: International Conference on Image Analysis and Processing—Workshops (ICIAPW-07), pp. 192–165 (2007)
33. Hardeberg, J.Y.: Red eye removal using digital color image processing. Image Processing, Image Quality, Image Capture, System Conference, Montreal, Canada, pp. 283–287 (2001)
34. Yoo, S., Park, R.-H.: Red-eye detection and correction using inpainting in digital photographs. IEEE Trans. Consum. Electr. **55**(3), 1006–1014 (2009)
35. Miao, X.-P., Sim, T.: Automatic red-eye detection and removal. In: International Conference on Multimedia and Expo (2004)
36. Petschnigg, G., Szeliski, R., Agrawala, M., Cohen, M. F., Hoppe, H., Toyama, K.: Digital photography with flash and no-flash image pairs. ACM Trans. Graph. **21**(3), 673–678 (2004)
37. Saaty, T.L.: Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World, vol. 2. Analytic Hierarchy Process Series, New Edition (2001)
38. Duda, R. O., Hart, P. E., Stork, D. G.: Pattern Classification, 2nd edn. Wiley-Interscience, Hoboken (2000)
39. Gonzalez, R. C., Woods, R. E.: Digital Image Processing, 3rd edn. Prentice Hall, Upper Saddle River (2008)
40. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Ann. Stat. **32**, 102–107 (2000)
41. Schapire, R.E.: The boosting approach to machine learning: an overview. In: MSRI Workshop on Nonlinear Estimation and Classification (2001)
42. Schapire, R. E.: The strength of weak learnability. In: Machine Learning, pp. 197–227 (1990)
43. Lienhart, R., Kuranov, E., Pisarevsky, V.: Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In: DAGM 25th Pattern Recognition Symposium, pp. 297–304 (2003)
44. Torralba, A., Murphy, K.P.: Sharing visual features for multiclass and multiview object detection. IEEE Trans. Pattern Anal. Mach. Intell. **29**(5), 854–869 (2007)

Part IV
Image Inferencing Algorithms

Chapter 13
Classifying Pathological Prostate Images by Fractal Analysis

Po-Whei Huang, Cheng-Hsiung Lee and Phen-Lan Lin

Abstract This study presents an automated system for grading pathological prostate images based on texture features of multicategories including multiwavelets, Gabor-filters, gray-level co-occurrence matrix (GLCM) and fractal dimensions. Images are classified into appropriate grades by using k-nearest neighbor (k-NN) and support vector machine (SVM) classifiers. Experimental results show that a correct classification rate (CCR) of 93.7 % (or 92.7 %) can be achieved by fractal dimension (FD) feature set by using k-NN (or SVM) classifier without feature selection. If the FD feature set is optimized, the CCR of 94.2 % (or 94.1 %) can be achieved by using k-NN (or SVM) classifier. The CCR is promoted to 94.6 % (or 95.6 %) by k-NN (or SVM) classifier if features of multicategories are applied. On the other hand, the CCR drops if the FD-based features are removed from the combined feature set of multicategories. Such a result suggests that features of FD category are not negligible and should be included for consideration for classifying pathological prostate images.

P.-W. Huang (✉) · C.-H. Lee
Department of Computer Science and Engineering,
National Chung Hsing University,
250 Kuo Kuang Road,
Taichung 40227, Taiwan, Republic of China
e-mail: powhei.huang@msa.hinet.net

P.-L. Lin
Department of Computer Science and Information Management,
Providence University, Shalu,
Taichung 44301, Taiwan, Republic of China
e-mail: lan@pu.edu.tw

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_13, © Springer-Verlag Berlin Heidelberg 2013

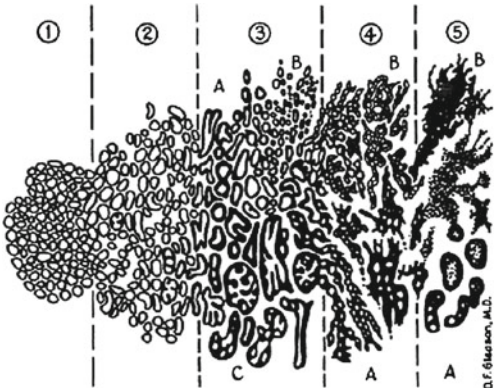


Fig. 13.1 The Gleason grading diagram

13.1 Introduction

Prostate cancer is the most frequently diagnosed cancer and ranks second among cancer deaths in the US [1]. Biopsy of the prostate, usually stained by hematoxylin and eosin (H&E) technique, is a key step for confirming the diagnosis of malignancy and guiding treatment [2]. By viewing the microscopic images of biopsy specimens, pathologists can determine the histological cancer grades according to the Gleason grading system. The Gleason grading system [3] is the most widespread method for histological grading of prostate carcinoma.

Although pathologists can determine the histological grades by viewing the microscopic images of biopsy specimens, the process of human visual grading is time-consuming and very subjective due to inter and intraobserver variations. Therefore, how to develop a more objective computer-aided technique for automatically and correctly grading prostatic carcinoma is the goal of this research study.

A classic Gleason grading diagram containing the five basic tissue patterns associated with the five tumor grades is shown in Fig. 13.1. As reported in [4], the use of texture analysis for prostatic lesions is very essential to the identification of tissue composition in prostatic neoplasia. Figure 13.2 shows four pathological images of prostatic carcinoma from well differentiated (grade 2) to very poorly differentiated (grade 5) in our image set. From Figs. 13.1 and 13.2, we can also see that the texture of prostate tissue plays an important role in Gleason grading for prostate cancer.

There are several well-known techniques for texture analysis such as extracting texture features from gray-level co-occurrence matrix (GLCM), Gabor filters, and multiwavelet transforms. The concept of fractal dimension (FD) is applied in this study for analyzing the texture of prostate tissue. The growth of cancer shows the features of fractal in physical phenomena. The fractal theory can provide clinically useful information for discriminating pathological tissue from healthy tissue [5].

The fractal dimension (FD) based features can be extracted through the differential box-counting (DBC) method [6] and entropy-based fractal dimension estimation (EBFDE) method [13] to analyze pathological images of prostatic carcinoma. We

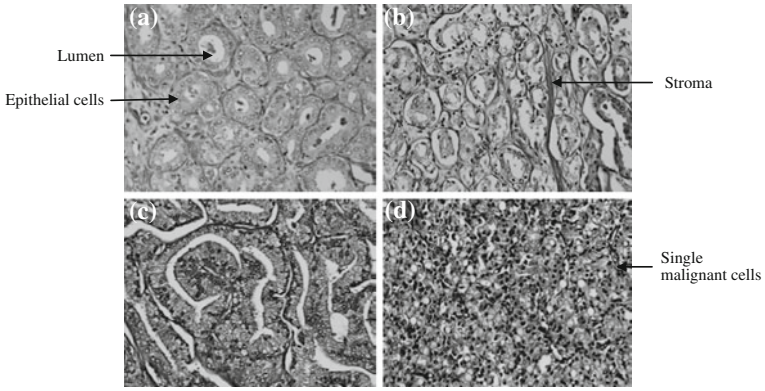


Fig. 13.2 Prostate images of different cancer grades: **a** Gleason grade 2. **b** Gleason grade 3. **c** Gleason grade 4. **d** Gleason grade 5

combine the FD-based features with those extracted from multiwavelets, Gabor filters, and GLCM to form a set of 144 texture features of multicategories for classification. To evaluate the effectiveness of classification results based on these features, we used a k-fold cross-validation procedure [7] (with $k = 5$) to a set of 205 pathological prostate images and tested against these samples using k-nearest neighbor (k-NN) and support vector machine (SVM) classifiers to estimate the correct classification rates (CCR), respectively. For selecting an optimal set of features, this study applies the sequential floating forward selection (SFFS) feature selection method [8].

Our system performs very well on classifying pathological prostate images in terms of CCR. Experimental results show that the FD feature set can achieve 93.7 % for $k = 1$ and 92.7 % of CCR without feature selection, and 94.2 % for $k = 1$ and 94.1 % of CCR with feature selection by k-NN and SVM classifiers, respectively. If features of multicategories are considered and optimized, the CCR can be improved to 94.6 and 95.6 % by k-NN and SVM classifiers, respectively. In addition, the CCRs will decrease if FD-based features are removed from the texture feature set, no matter which classifier is used. Such a result suggests that features of the FD category are not negligible and should be included for consideration if features are selected from multicategories.

13.2 Feature Extraction

In this section, we present various well-known texture feature sets derived from multiwavelets, Gabor filters, GLCM, and FD-based methods for classifying histological prostate images.

Jafari-Khouzani et al. [4] proposed a method for grading the pathological images of prostate biopsy samples by using energy and entropy features calculated from

multiwavelet coefficients of an image. Ten sets of multiwavelet features were captured from different multiwavelet methods. Among them, multiwavelet-SA4 method has the best performance. The details of defining multiwavelet-SA4 can be found in [9]. In the multiwavelet method, a two-level multiwavelet transform of an image is performed to generate 28 sub-band images. Since two features, energy and entropy, are extracted from each sub-band image for classification, there will be a total number of 56 features in a feature set extracted from multiwavelet-SA4 method. We call this type of features the Multiwavelet category.

A Gabor filter can be viewed as a sinusoidal plane of particular frequency and orientation modulated by a Gaussian envelope. It is a promising method for texture feature extraction in existing multichannel filtering approaches [10, 11]. An image is filtered with a set of Gabor filters of different preferred orientations and spatial frequencies to generate filtered images from which texture features can be extracted. In our experimental system, we implemented a bank of Gabor filters using five radial frequencies $\sqrt{2}/2^6$, $\sqrt{2}/2^5$, $\sqrt{2}/2^4$, $\sqrt{2}/2^3$, $\sqrt{2}/2^2$, and four orientations: 0°, 45°, 90°, and 135°. How to choose appropriate radial frequencies for a bank of Gabor filters can be found in [10]. In our case, a set of 20 filtered images will be generated by the Gabor filter method. We extract three features, energy, entropy, and magnitude, from each of the 20 filtered images. As a consequence, three sets of features called Gabor energy, Gabor entropy, and Gabor magnitude are formed with each containing 20 features. We can combine these three sets of features to form a feature set of dimension 60 and call this type of features the Gabor category.

Five statistical texture feature sets (energy, entropy, contrast, correlation, and homogeneity) are extracted from co-occurrence matrices based on a particular scalar distance and four orientations, 0°, 45°, 90°, and 135°. To determine an appropriate scalar distance to better capture a specific feature, we estimate the CCR of that feature using ten distances (from 1 to 10 pixels) and choose the distance which generates the highest CCR. In our experiment, the best distance is 3 pixels to capture energy feature, 4 pixels to capture the entropy and contrast features, 1 pixel to capture the correlation feature, and 8 pixels to capture the homogeneity feature. Once we obtain the best distance which achieves the highest CCR for a specific feature, we use that distance to generate four co-occurrence matrices with each matrix corresponding to an orientation. Therefore, five feature sets are generated with each one containing four features. Like the Gabor filter method, we combine the above five feature sets together to form a feature set of dimension 20 and call this type of features the GLCM category.

The concept of self-similarity can be used to estimate the fractal dimension as follows. Given a bounded set S in Euclidean n-space, S is self-similar if it is the union of N_r distinct (nonoverlapping) copies of itself scaled down by a ratio r. The fractal dimension FD of S is given by the relation $1 = N_r r^D$ and is calculated by the following equation [12]:

$$FD = \frac{\log(N_r)}{\log(1/r)}. \tag{13.1}$$

The fractal dimension texture features can be generated from a pathological image using differential box-counting (DBC) and entropy-based fractal dimension estimation (EBFDE) methods [13]. They can provide very useful information for classifying pathological prostate images into four classes in the Gleason grading system. In our system, fractal dimension is estimated according to different ranges of scales for capturing the various self-similarity properties in a prostatic carcinoma image. Before applying the DBC method, color pathological images of prostatic tissues are transformed to gray-level images by getting the R channel from the RGB color space to enhance the contrast between malignant cells and background tissues. In the above preprocessing step, the malignant cells will become darker because they are stained blue. Other pathological objects such as stroma and lumens are stained red or do not get stained in H&E-stained pathological images.

The DBC method is briefly described as follows. Consider an image of size $M \times M$ pixels that has been scaled down to a size $s \times s$, where $1 < s \leq M/2$ and s is an integer. Then, we can get the scale ratio $r = s/M$. Consider the image as a 3D space such that (x, y) represents a 2D position, and the third coordinate (z) represents the gray level of an image at position (x, y). The (x, y) space is divided into grids of size $s \times s$. Thus, there will be a column of boxes of size $s \times s \times h$ on each grid, where $\lfloor \frac{G}{h} \rfloor = \lfloor \frac{M}{s} \rfloor$ and G is the total number of gray levels in an image. Let the maximum and minimum gray levels of an image in the (i, j)th grid fall in boxes k and l, respectively. The contribution of N_r in the (i, j)th grid is expressed as follows:

$$n_r(i, j) = k - l + 1. \tag{13.2}$$

The contribution from all grids is

$$N_r = \sum_{i,j} n_r(i, j). \tag{13.3}$$

N_r is counted for different scale ratio r. Then, the fractal dimension D can be estimated from the slope of line approximated by least-squares linear fitting for $\log(N_r)$ versus $\log(1/r)$ in Eq. (13.1).

The DBC method only captures the information about intensity difference which is necessary but not sufficient enough to differentiate all patterns of different Gleason grades. The entropy-based fractal dimension estimation (EBFDE) method can further capture the information about randomness of pixels. The EBFDE method is described as follows. First, a 2D image is partitioned into several grids of size $s \times s$. Then, we compute the entropy for the (i, j)th grid using the following equation:

$$e_r(i, j) = -\sum_{k=0}^{G-1} p_k \log_2(p_k), \tag{13.4}$$

where the index k is taken over all gray scales in the (i, j)th grid of an image, p_k is the probability of gray-level k occurring in the (i, j)th grid of an image, and G is the

total number of gray levels. The contribution from the (i, j)th grid is $e_r(i, j)^2$. So the total contribution from all grids is

$$E_r(i, j) = \sum_{i,j} e_r(i, j)^2. \tag{13.5}$$

Again, by applying Eq. (13.1), the fractal dimension D of an image can be estimated using least-squares linear fitting for $\log(E_r)$ versus $\log(1/r)$.

In this study, we assume that various self-similarity properties in a prostatic carcinoma (PCa) image may be reflected in different individual ranges of scales. Remember that the scaled down ratio is $r = s/M$, where s^2 is the grid size and M^2 is the image size. Since $M = 384$ for prostate images, we choose $s = 2, 4, 8, 16, 32, 64$, and 128 to include all feasible grid sizes, from 2×2 (the smallest one) to 128×128 (one-ninth of the whole image). Therefore, the range of scales (r) is {1/192, 1/96, 1/48, 1/24, 1/12, 1/6, 1/3}, which is subsequently divided into three subranges: the subrange of small scales {1/192, 1/96, 1/48}, the subrange of medium scales {1/48, 1/24, 1/12}, and the subrange of large scales {1/12, 1/6, 1/3}. Here, we allow a small portion of overlapping between two neighboring subranges because there is no clear distinction between two subranges reflecting different self-similarity properties. We choose three scales in each subrange because this is the minimum requirement for using the technique of least-square linear fit. Since we do not exclude the possibility that the same self-similarity property is reflected in all scales, we also use all of the seven scales to estimate the fractal dimension of an image. As a result, four fractal dimension texture features can be obtained by DBC method and another four fractal dimension texture features can be obtained by our EBFDE method. Then, we combine these eight features to obtain a feature set $\{f_{D1}, f_{D2}, f_{D3}, f_{D4}, f_{E1}, f_{E2}, f_{E3}, f_{E4}\}$ as follows:

- f_{D1} is the FD calculated from grids of size $s^2 (s = 2, 4, 8)$ using the DBC method.
- f_{D2} is the FD calculated from grids of size $s^2 (s = 8, 16, 32)$ using DBC method.
- f_{D3} is the FD calculated from grids of size $s^2 (s = 32, 64, 128)$ using the DBC method.
- f_{D4} is the FD calculated from grids of size $s^2 (s = 2, 4, 8, 16, 32, 64, 128)$ using the DBC method.
- f_{E1} is the FD calculated from grids of size $s^2 (s = 2, 4, 8)$ using the EBFDE method.
- f_{E2} is the FD calculated from grids of size $s^2 (s = 8, 16, 32)$ using the EBFDE method.
- f_{E3} is the FD calculated from grids of size $s^2 (s = 32, 64, 128)$ using the EBFDE method.
- f_{E4} is the FD calculated from grids of size $s^2 (s = 2, 4, 8, 16, 32, 64, 128)$ using the EBFDE method.

Then, we can combine these eight features to form a feature set denoted by $\mathbf{f}_D + \mathbf{f}_E$. We call this type of features the FD category. Details of extracting FD-based features from prostate images can be found in [13]. Finally, we combine the FD-based features

with those extracted from multiwavelets, Gabor filters, and GLCM to form a set containing 144 texture features of multicategories for classification.

13.3 Classification and Feature Selection

We used two different classifiers, the k-NN and SVM, to cooperate with texture features of multicategories as described in the previous section. This study normalizes each feature to have a mean of zero and a standard deviation of one for the entire data set avoid the system performance influencing by the characteristic of a single feature. If one of the features has a very wide range of possible values compared with the other features, it will have a large effect on the dissimilarity, and the decisions will be based primarily upon this single feature [4].

The k-nearest-neighbor decision rule classifies an observation by assigning it the label which is most frequently represented among the nearest neighbors. A decision is made by examining all the labels on the nearest neighbors and taking a vote. The operation of a k-NN classifier can be summarized by the following basic steps [14]:

1. Compute the distances between the new sample and all previous samples already classified into clusters.
2. Sort the distances in increasing order and select samples with the smallest distance values.
3. Apply the voting principle: a new sample is added (classified) to the largest cluster out of selected samples.

Another classification technique used in this study for classifying carcinoma prostate images is the SVM method. Compared with traditional classification methods which minimize the empirical training error, the goal of SVM is to minimize the upper bound of the generalization error by finding the largest margin between the separating hyperplane and the data. In this study, we use the one-against-one multiclass classification method based on a library for support vector machines (LIBSVM) [15, 16] with the radial basis function (RBF) kernel by combining all pair-wise comparisons of binary SVM classifiers.

In order to estimate classification performance for different classifiers, the correct classification rate (CCR) [17] is defined as

$$CCR = \sum_{i=1}^{C} P(c_i) \frac{n_i}{N_i}, \tag{13.6}$$

where n_i is the number of samples correctly classified to the ith class via the k-NN or SVM classifiers, C is the total number of classes, N_i is the total number of samples in the ith class, and $P(c_i)$ is the prior probability that an observed data falls in class c_i.

In the k-fold cross-validation method, the entire sample set is randomly partitioned into k disjoint subsets of equal size, where n is the total number of samples in the entire set. Then, $k-1$ subsets are used to train the classifier and the remaining subset is used to test for accuracy estimation. This process is repeated for all distinct choices of k subsets, and the average of correct classification rates is calculated.

To obtain an optimal set of texture features of multicategories, we apply the sequential floating forward selection (SFFS) feature selection method. The SFFS feature selection method is very effective in selecting an optimal subset of features [8]. In this study, we first use the five-fold cross-validation procedure to estimate the CCR for the candidate feature subsets selected by the SFFS method at each stage for each of the k-NN and SVM classifiers. Then, we apply the five-fold cross-validation procedure to evaluate the performance of the selected feature set using each of the above two classifiers. Notice that, in applying the 5-fold cross-validation procedure, the five groups of data used in feature selection are different from the five groups of data used in training and testing by random reassignment.

13.4 Experimental Results

We used 205 pathological images with 512×384 pixels of resolution for our experiment. To avoid inter and intraobserver variations that may cause possible ambiguities in classification: (1) images were commonly analyzed by a group of experienced pathologists in Taichung Veterans General Hospital of Taiwan and classified into four classes in advance as "gold standard" for later comparison; (2) the pattern of the cancer observed in each sample must be greater than 60 % of the total pattern seen in order to assign a primary Gleason grade to that sample. Since Grade1 patterns are very rare, Grade1 and Grade 2 patterns are regarded as the same class. As a result, our image set was divided into four classes: 50 images in class1 (Grade1 and Grade 2), 72 images in class 2 (Grade 3), 31 images in class 3 (Grade 4), and 52 images in class 4 (Grade 5).

In our experiment, the classification results in terms of CCR of three feature sets multicategory1, multicategory2, and $\mathbf{f}_D + \mathbf{f}_E$ feature set are compared and analyzed. Multicategory1 denotes the set of 144 features whose extraction methods was described in Sect. 13.2. Multicategory2 is a set of 136 features formed by removing the 8 FD-based features from multicategory1. The $\mathbf{f}_D + \mathbf{f}_E$ feature set contains 8 FD-based features purely extracted from DBC and EBFDE methods [13].

Table 13.1 shows the classification results of multicategory feature sets using k-NN and SVM classifiers, respectively, without SFFS feature selection. In k-NN classifier, the CCR of multicategory1 for $k = 3$ is 92.7 %, CCR of multicategory2 for $k = 3$ is 93.2 %, and CCR of $\mathbf{f}_D + \mathbf{f}_E$ for $k = 1$ is 93.7 %. Furthermore, the CCRs of multicategory1 and multicategory2 are both 92.2 % while the CCR of $\mathbf{f}_D + \mathbf{f}_E$ is 92.7 % in SVM classifier. It seems that feature set $\mathbf{f}_D + \mathbf{f}_E$ has the same discriminating capability that multicategory1 and multicategory2 have. However, the dimensional-

Table 13.1 Comparisons of three feature sets in terms of CCR using k-NN and SVM classifiers evaluated using five-fold cross-validation procedure without SFFS feature selection

Feature sets	k-NN($k = 1$) (%)	k-NN($k = 3$) (%)	k-NN($k = 5$) (%)	SVM
Multicategory1(*144*)	91.7	92.7	90.7	92.2
Multicategory2(*136*)	91.7	93.2	90.7	92.2
$\mathbf{f}_D + \mathbf{f}_E$ (*8*)	93.7	92.7	93.2	92.7

ities of the multicategory1 and multicategory2 are 144 and 136, respectively, while the dimensionality of $\mathbf{f}_D + \mathbf{f}_E$ is only 8.

Table 13.2 shows the classification results of multicategory feature sets using k-NN and SVM classifiers, respectively, with SFFS feature selection. When the SFFS feature selection method is applied to optimize the above three feature sets, the dimension of optimized multicategory1 is reduced to 13, the dimension of optimized multicategory2 is reduced to 10, and the dimension of optimized $\mathbf{f}_D + \mathbf{f}_E$ is reduced to 3 in k-NN classifier. The dimension of optimized multicategory1 is reduced to 10, the dimension of optimized multicategory2 is reduced to 8, and the dimension of optimized $\mathbf{f}_D + \mathbf{f}_E$ is reduced to 5 in SVM classifier.

More specifically, in k-NN classifier, optimized multicategory1 contains one FD-based feature, one Gabor entropy feature, three Gabor energy features, one Gabor magnitude feature, one GLCM correlation feature, and six Multiwavelet-SA4 features. Optimized multicategory2 contains one Gabor entropy feature, one GLCM contrast feature, and eight Multiwavelet-SA4 features. The optimized $\mathbf{f}_D + \mathbf{f}_E$ feature set contains two FD-based features extracted by DBC method and another one FD-based feature extracted by EBFDE method. In SVM classifier, optimized multicategory1 contains one FD-based feature, three Gabor entropy features, three Gabor energy features, and three Multiwavelet-SA4 features. Optimized multicategory2 contains two Gabor entropy features, one Gabor magnitude feature, and five Multiwavelet-SA4 features. The optimized $\mathbf{f}_D + \mathbf{f}_E$ feature set contains three FD-based features extracted by DBC method and another two FD-based features extracted by EBFDE method.

As we can see from Table 13.2, the CCRs of multicategory1, multicategory2, and $\mathbf{f}_D + \mathbf{f}_E$ using k-NN classifier are 94.6, 92.2, and 94.2 % for $k = 1$, respectively; the CCRs of multicategory1, multicategory2, and $\mathbf{f}_D + \mathbf{f}_E$ using SVM classifier are 95.6, 92.7, and 94.1 %, respectively. Notice that the CCR of optimized $\mathbf{f}_D + \mathbf{f}_E$ feature set is pretty close to the performance of optimized multicategory1 and outperforms the CCR of optimized multicategory2, no matter which classifier is used. This implies that the FD-based features are important in classifying pathological prostate images. If texture features from multicategories are considered, FD-based features must be included. Otherwise, the classification performance will be degraded.

Table 13.2 Comparisons of three feature sets in terms of CCR using k-NN and SVM classifiers evaluated using five-fold cross-validation procedure with SFFS feature selection

Feature sets	k-NN ($k = 1$) (%)	k-NN ($k = 3$) (%)	k-NN ($k = 5$) (%)	No. of features	SVM (%)	No. of features
Multicategory1	94.6	94.6	92.7	(*13*)	95.6	(*10*)
Multicategory2	92.2	91.7	89.8	(*10*)	92.7	(*8*)
$\mathbf{f}_D + \mathbf{f}_E$	94.2	93.7	93.2	(*3*)	94.1	(*5*)

13.5 Conclusions

This study presents an automated system for grading pathological images of prostatic carcinoma based on a set of texture features extracted from multicategories of methods, including multiwavelets, Gabor filters, GLCM, and fractal dimension. Our system has shown very good performance in classifying pathological prostate images in terms of correct classification rate. Experimental results show that the FD-based feature set can provide very useful information for classifying pathological prostate images. Without feature selection, the CCRs of 93.7 % for $k = 1$ and 92.7 % can be achieved by a set of 8 FD-based texture features using k-NN and SVM classifiers, respectively. When SFFS method was applied for feature selection and optimization, the CCRs can be improved to 94.2 % for $k = 1$ and 94.1 % by optimized sets of 3 FD-based and 5 FD-based features using k-NN and SVM classifiers, respectively. If multicategories of features are considered and optimized, the CCRs can be promoted to 94.6 % for $k = 1$ and 95.6 % by k-NN and SVM classifiers, respectively. Notice that the CCRs decreased no matter which classifier is used if FD-based features are not included in the feature set. Thus, FD-based features play an important role in prostate image classification.

References

1. American Cancer Society: Cancer Facts & Figures 2007. American Cancer Society, Atlanta, GA (2007)
2. Zhu, Y., Williams, S., Zwiggelaar, R.: Computer technology in detection and staging of prostate carcinoma: a review. Med. Image Anal. **10**, 178–199 (2006)
3. Gleason, D.F.: The veteran's administration cooperative urologic research group: histologic grading and clinical staging of prostatic carcinoma. In: Tannenbaum, M. (ed.) Urologic Pathology: The Prostate, pp. 171–198. Lea and Febiger, Philadephia, PA (1977)
4. Jafari-Khouzani, K., Soltanian-Zadeh, H.: Multiwavelet grading of pathological images of prostate. IEEE Trans. Biomed. Eng. **50**, 607–704 (2003)
5. Baish, J.W., Jain, R.K.: Fractals and cancer. Cancer Res. **60**, 3683–3688 (2000)
6. Sarkar, N., Chaudhuri, B.B.: An efficient differential box-counting approach to compute fractal dimension of image. IEEE Transa. Syst. Man Cybern. **24**, 115–120 (1994)
7. Fukunaga, K.: Introduction to Statistical Pattern Recognition, 2nd edn. Academic Press, New York (1990)

8. Pudil, P., Novovicova, J., Kittler, J.: Floating search methods in feature selection. Pattern Recognit. Lett. **15**, 1119–1125 (1994)
9. Shen, L.X., Tan, H.H., Tham, J.Y.: Symmetric-antisymmetric orthonormal multiwavelets and related scalar wavelets. Appl. Comput. Harmon. Anal. (ACHA) **8**, 258–279 (2000)
10. Jain, A.K., Farrokhnia, F.: Unsupervised texture segmentation using Gabor filters. Pattern Recognit. **24**, 1167–1186 (1991)
11. Pichler, O., Teuner, A., Hosticha, B.J.: A comparison of texture feature extraction using adaptive Gabor filtering, pyramidal and tree structured wavelet transforms. Pattern Recognit. **29**, 733–742 (1996)
12. Chaudhuri, B.B., Sarkar, N.: Texture segmentation using fractal dimension. IEEE Trans. Pattern Anal. Mach. Intell. **17**, 72–77 (1995)
13. Huang, P.W., Lee, C.H.: Automatic classification for pathological prostate images based on fractal analysis. IEEE Trans. Med. Imaging **28**, 1037–1050 (2009)
14. Kantardzic, M.: Data Mining: Concepts, Models, Methods, and Algorithms. Wiley, New Jersey (2002)
15. Wu, T.F., Lin, C.J., Weng, R.C.: Probability estimates for multi-class classification by pairwise coupling. J. Mach. Learn. Res. **5**, 975–1005 (2004)
16. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines. Software http://www.csie.ntu.edu.tw/cjlin/libsvm (2001)
17. Lee, W.L., Chen, Y.C., Hsieh, K.S.: Ultrasonic liver tissues classification by fractal feature vector based on M-band wavelet transform. IEEE Trans. Med. Imaging **22**, 382–392 (2003)

Chapter 14
Multiobjective PSO for Hyperspectral Image Clustering

Farid Melgani and Edoardo Pasolli

Abstract In this chapter, a multiobjective particle-swarm optimization approach is presented as an answer to the problem of hyperspectral remote sensing image clustering. It aims at simultaneously solving the following three different issues: (1) clustering the hyperspectral cube under analysis; (2) detecting the most discriminative bands of the hypercube; (3) avoiding the user to set a priori the number of data classes. The search process is guided by three different statistical criteria, which are the log-likelihood function, the Bhattacharyya distance, and the minimum description length. Experimental results clearly underline the effectiveness of particle-swarm optimizers for a completely automatic and unsupervised analysis of hyperspectral remote sensing images.

14.1 Introduction

In the field of stochastic optimization methods, particle-swarm optimization (PSO) represents an interesting approach to solve complex optimization problems [8]. Introduced recently by Kennedy and Eberhart [12], it is inspired by social behavior of bird flocking and fish schooling. Similar to other evolutionary computation algorithms such as genetic algorithms, PSO is a population-based search method that exploits the concept of social sharing of information. This means that each individual (called *particle*) of a given population (called *swarm*) can profit from the previous experiences of all other individuals from the same population. During the search process in the solution space, each particle (i.e., candidate solution) will adjust its flying

F. Melgani (✉) · E. Pasolli
Department of Information Engineering and Computer Science,
University of Trento, Via Sommarive 14, 38123 Trento, Italy
e-mail: melgani@disi.unitn.it

E. Pasolli
e-mail: pasolli@disi.unitn.it

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_14, © Springer-Verlag Berlin Heidelberg 2013

velocity and position according to its own flying experience as well as the experiences of the other companion particles of the swarm. PSO has been shown to be promising for solving problems in different fields, such as automatic control [14], antenna design [21], inverse problems [29], multimedia [7], biomedical signal classification [24], and remote sensing [3].

In the literature, an important research field is represented by classification methods. From a methodological viewpoint, a classification process consists of associating a pattern (sample) to a class label opportunely chosen from a predefined set of class labels. Two main approaches to the classification problem have been proposed: (1) the supervised approach and (2) the unsupervised approach. Supervised techniques require the availability of a training set for learning the classifier. Unsupervised methods, known also as clustering methods, perform classification just by exploiting information conveyed by the data, without requiring any training sample set. The supervised methods offer a higher classification accuracy compared to the unsupervised ones, but in some applications, it is necessary to resort to unsupervised techniques because training information is not available.

Depending on the application, clustering methods may or may not be useful. In this chapter, we focus on works specifically developed for remote sensing applications. Remote sensing is a vital technology for monitoring man-made and natural resources at large scale with low costs. Regarding clustering methods in remote sensing, an early method is the one presented in [33], which exploits the notions of scale and cluster independence to classify multispectral and polarimetric synthetic aperture radar (SAR) images. In [25], the authors introduced the concept of global classification of remote sensing images in large archives, e.g., covering the whole globe. The classification is realized through a two-step procedure: (1) unsupervised clustering and (2) supervised hierarchical classification. Features, derived from different and noncommensurable models, are combined using an extended k-means clustering algorithm and supervised hierarchical Bayesian networks incorporating any available prior information. In [2], a fuzzy clustering method for multispectral images was presented. It groups data samples, even when the number of clusters is not known or when noise is present, by replacing the probabilistic constraint that memberships across clusters must sum to one with a composite constraint. In [15], the hybrid supervised-unsupervised approach to image classification was improved by introducing the concept of cluster-space classification for hyperspectral data. The cluster-space representation is used for associating spectral clusters with corresponding information classes automatically, thus overcoming the manual assignment of clusters and classes carried out in the hybrid approach. This method is further enhanced for efficient data transmission and classification in [16]. In [28], a method of hyperspectral band reduction based on rough sets and fuzzy C-means clustering was proposed. It consists of two steps. First, the fuzzy C-means clustering algorithm is used to classify the original bands into equivalent band groups. Then, data dimensionality is reduced by selecting only the band with maximum grade of fuzzy membership from each of the groups. In [13], limitations of k-means algorithm implementation were discussed. In order to accelerate the k-means clustering, a hardware implementation was proposed. In [18], the authors presented a two-stage hierarchical

clustering technique for classifying hyperspectral data. First, a "local" segmentator performs region-growing segmentation by merging spatially adjacent clusters. Then, a "global" segmentator clusters the segments resulting from the previous stage using an agglomerative hierarchical clustering scheme based on a context-free similarity measure. In [17], five clustering techniques were compared for classifying polarimetric SAR images. Two techniques are fuzzy clustering algorithms based on the standard l_1 and l_2 metrics. Two others combine a robust fuzzy C-means clustering technique with a distance measure based on the Wishart distribution. The fifth technique is an application of the expectation-maximization (EM) algorithm, assuming that data follow a Wishart distribution. In [22], the authors proposed an agglomerative hierarchical clustering method for multispectral images, which uses both spectral and spatial information for the aggregation decision. In [30], Markov-random field (MRF) clustering, exploiting both spectral and spatial interpixel dependency information, for polarimetric SAR images was presented. Because of its strong sensitivity to initial conditions, an initialization scheme was suggested. It aims at deriving initial cluster parameters from a set of homogenous regions, and estimating the number of clusters with the pseudo-likelihood information criterion. In [35], a two-step unsupervised artificial immune classifier for multi/hyperspectral images was proposed. In [1], unsupervised land cover classification is performed by clustering pixels in the spectral domain into several fuzzy partitions. A multiobjective (MO) optimization algorithm is utilized to tackle the fuzzy partitioning problem by means of a simultaneous optimization of different fuzzy cluster validity indexes. The resulting near-Pareto-optimal front contains a set of nondominated solutions, from which the user can pick the most promising one according to the problem requirements. In [20], a weighted fuzzy C-means clustering algorithm was proposed to carry out the fuzzy or the hard classification of multispectral images. In [34], the authors presented a rapid clustering method for SAR images by embedding an MRF model in the clustering space and using graph cuts to search for data clusters optimal in the sense of the maximum a posteriori (MAP) criterion. In [31], a multistage unsupervised classification technique for multispectral images was presented. It is composed of a context-sensitive initialization and an iterative procedure aiming at estimating the statistical parameters of classes to be used in a Bayesian decision rule. The initial steps exploit a graph cut segmentation algorithm followed by a fuzzy C-means clustering, while the iterative procedure is based on the EM algorithm. In [6], unsupervised classification of hyperspectral images was performed by applying fuzzy C-means clustering as well as its extended version, i.e., Gustafson-Kessel clustering, which is based on an adaptive distance norm. An opportune phase-correlation-based similarity measure was used to improve the fuzzy clustering by taking spatial relations into account for pixels with similar spectral characteristics.

In this chapter, we focus on hyperspectral image clustering. Compared with conventional multispectral data, hyperspectral data are characterized by a higher spectral resolution, thus giving the opportunity to further enhance the information extraction capability. However, hyperspectral imagery involves a greater quantity of data to memorize and to process. Moreover, given a specific classification problem, hyperspectral data often exhibit redundant information, thus calling for opportune band

(feature) selection algorithms. While feature selection has been widely studied in the supervised classification context, little has been done in the image clustering context due to the lack of training samples. Another intrinsic problem in image clustering in general and in hyperspectral image clustering in particular is how to set a priori the number of data classes because of the absence of prior information.

In this chapter, we present a methodology for hyperspectral images capable of solving simultaneously the above problems, i.e., clustering, feature detection (i.e., selection of the features without requiring the desired number of most discriminative features a priori from the user), and class number estimation. Clustering and feature detection are dealt with within an MO optimization process based on PSO to estimate the cluster statistical parameters and to detect the most discriminative features. Class number estimation is performed using a strategy based on the minimum description length (MDL) criterion. To illustrate the performance of the presented methodology, we conducted an experimental study based on a real hyperspectral remote sensing image acquired by the Reflective Optics System Imaging Spectrometer (ROSIS) sensor. In general, the obtained experimental results show that interesting performances can be achieved though the processing context is completely unsupervised. The remaining part of this work is organized as follows. The problem formulation is described in Sect. 14.2. The presented clustering methodology is described in Sect. 14.3. The experimental results are reported in Sect. 14.4. Finally, conclusions are drawn in Sect. 14.5.

14.2 Hyperspectral Clustering Problem Formulation

Let us consider a hyperspectral image composed of d bands and n pixels. Each pixel is represented by a vector $\mathbf{x}_i \in \Re^d = \left[x_{i,1}, x_{i,2}, \ldots, x_{i,d}\right], i = 1, 2, \ldots, n$. Let us assume that no prior knowledge is available for this image in terms of training samples. Moreover, let us suppose that the number C of data classes present in the image is not known. In optical imagery, the assumption that the distribution of images can be approximated as a mixture of normally distributed samples is generally well accepted [4, 11]. Accordingly, the probability distribution function (pdf) of the image can be written as

$$p(\mathbf{x}) = \sum_{j=1}^{C} P\left(\omega_j\right) \cdot p\left(\mathbf{x}|\omega_j\right) \tag{14.1}$$

where $P\left(\omega_j\right)$ and $p\left(\mathbf{x}|\omega_j\right) = N\left(\mu_j, \Sigma_j\right)$ are the prior probability and the conditional pdf associated with the jth data class (Gaussian mode) of the image, respectively. μ_j and Σ_j stand for the mean vector and the covariance matrix of the jth data class. Let us suppose that the features (bands) are independent and, hence, Σ_j is a diagonal matrix. Although the assumption of independence between adjacent bands is typically not satisfied, it is important to render our clustering problem computationally tractable.

The objective is to classify the image in an unsupervised way. Given its hyperspectral nature, it is preferable to perform beforehand a feature detection operation. Indeed, this last step is useful to reduce data redundancy and to remove the bands that are characterized by a strong noise component. Note that by feature detection, we intend that the selection of the features be carried out without requiring the desired number of most discriminative features a priori from the user. A further goal is to estimate automatically the number of data classes characterizing the image. We desire to meet all these requirements simultaneously, without any prior knowledge about the investigated study area.

In this chapter, we formulate this complex problem within a multiobjective particle-swarm optimization (MOPSO) framework so that to simultaneously estimate the cluster statistical parameters, detect the most discriminative features, and estimate the class number. The MO approach to the problem is motivated by the different nature of the desired tasks. In particular, the first PSO fitness function will have the purpose of estimating the cluster parameters, while the second one will guide the detection of the best features and, thus, the removal of redundant and/or noisy bands. The class number estimation will be carried out by repeating the PSO process over a predefined range of values of class number for optimizing the MDL criterion.

14.3 MOPSO Clustering Approach

14.3.1 PSO Setup

In an optimization problem that is formulated within a PSO framework, the solution space is explored by means of a swarm of particles whose positions point to candidate solutions. The first task to perform consists of defining the ingredients of the PSO algorithm, namely, the particle position $\mathbf{p}$ and the fitness functions $f(\mathbf{p})$.

Since our clustering problem consists of finding the best estimate of the cluster statistical parameters and the best discriminative features, the position $\mathbf{p}$ of each particle will simply be a vector that encodes all these variables. A representation of the particle position is provided in Fig. 14.1. Given C data classes, the cluster parameters are defined by a number of real variables equal to $2Cd$ since for each class ω_j $(j = 1, 2, \ldots, C)$, the mean vector $\boldsymbol{\mu}_j \in \Re^d = \left[\mu_{1j}, \mu_{2j}, \ldots, \mu_{dj}\right]$ and the variance vector $\boldsymbol{\sigma}_j^2 \in \Re^d = \left[\sigma_{1j}^2, \sigma_{2j}^2, \ldots, \sigma_{dj}^2\right]$ have to be estimated. Moreover, the feature detection task requires the setting of d coordinates expressed in terms of Boolean values. The feature variable $f_i, i = 1, 2, \ldots, d$ is equal to one if the correspondent feature is selected; otherwise, it is equal to zero.

Another important aspect in the setup of a PSO process is the choice of the fitness functions $f(\mathbf{p})$, which will be used to evaluate the performance of each position $\mathbf{p}$.

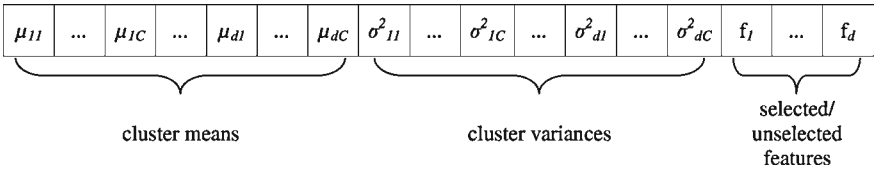


Fig. 14.1 PSO particle structure

For this purpose, we optimize jointly two different criteria to deal with both the class statistical parameter estimation and the feature detection issues.

The first fitness function is the log-likelihood function. It aims at determining the distribution parameter values that best approximate the data distribution. Supposing that samples are independent and identically distributed, the log-likelihood function is given by

$$L\left(\mathbf{x}|\mathbf{p},\mathbf{P}_C\right)=\sum_{i=1}^{n}\ln p\left(\mathbf{x}_i|\mathbf{p},\mathbf{P}_C\right). \tag{14.2}$$

where $\mathbf{P}_C=\left[P\left(\omega_1\right),P\left(\omega_2\right),\ldots,P\left(\omega_C\right)\right]$ is the prior probability vector.

Since the samples originate from a multivariate Gaussian distribution, the log-likelihood function can be rewritten as

$$L\left(\mathbf{x}|\mathbf{p},\mathbf{P}_C\right)=\sum_{i=1}^{n}\ln\sum_{j=1}^{C}\frac{P\left(\omega_j\right)}{\left(2\pi\right)^{\tilde{d}(\mathbf{p})/2}\cdot\left|\tilde{\Sigma}_j\left(\mathbf{p}\right)\right|^{1/2}}$$
$$\times\exp\left\{-\frac{1}{2}\left(\tilde{\mathbf{x}}_i\left(\mathbf{p}\right)-\tilde{\boldsymbol{\mu}}_j\left(\mathbf{p}\right)\right)^T\tilde{\Sigma}_j\left(\mathbf{p}\right)^{-1}\left(\tilde{\mathbf{x}}_i\left(\mathbf{p}\right)-\tilde{\boldsymbol{\mu}}_j\left(\mathbf{p}\right)\right)\right\} \tag{14.3}$$

where $\tilde{d}\left(\mathbf{p}\right)$ is the number of features detected by $\mathbf{p}$, $\tilde{\mathbf{x}}_i\left(\mathbf{p}\right)$ is the ith sample defined in the subspace formed by the detected features, and $\tilde{\boldsymbol{\mu}}_j\left(\mathbf{p}\right)$ and $\tilde{\Sigma}_j\left(\mathbf{p}\right)$ are the mean vector and the diagonal covariance matrix associated with the jth class in that subspace respectively.

The second fitness function has the purpose of evaluating the statistical distance between classes in the subspace of detected features defined by $\mathbf{p}$. For such purpose, we adopt the Bhattacharyya distance [5], which, for a couple of classes that are normally distributed (e.g., the ith and jth classes), is expressed as follows:

$$B_{i,j}\left(\mathbf{p}\right)=\frac{1}{8}\left(\tilde{\boldsymbol{\mu}}_i\left(\mathbf{p}\right)-\tilde{\boldsymbol{\mu}}_j\left(\mathbf{p}\right)\right)^T\left\{\frac{\tilde{\Sigma}_i\left(\mathbf{p}\right)+\tilde{\Sigma}_j\left(\mathbf{p}\right)}{2}\right\}^{-1}$$
$$\times\left(\tilde{\boldsymbol{\mu}}_i\left(\mathbf{p}\right)-\tilde{\boldsymbol{\mu}}_j\left(\mathbf{p}\right)\right)+\frac{1}{2}\ln\left\{\frac{\left|\frac{\tilde{\Sigma}_i(\mathbf{p})+\tilde{\Sigma}_j(\mathbf{p})}{2}\right|}{\left|\tilde{\Sigma}_i\left(\mathbf{p}\right)\right|^{\frac{1}{2}}\left|\tilde{\Sigma}_j\left(\mathbf{p}\right)\right|^{\frac{1}{2}}}\right\}. \tag{14.4}$$

At this point, the multiclass distance can be determined using different strategies. We calculate it according to the following simple rule:

$$B(\mathbf{p}) = \min_{i=\{1,\ldots,C\}, j=\{1,\ldots,C\}, i\neq j} \left\{ B_{i,j}(\mathbf{p}) \right\}. \tag{14.5}$$

Since the particles of the swarm can define feature subspaces of different dimensionalities, in order to remove (reduce) the impact of the dimensionality on the fitness function values, the previously defined fitness functions are normalized with respect to the number of features. Therefore, the fitness functions become

$$L_{nor}(\mathbf{x}|\mathbf{p}, \mathbf{P}_C) = \frac{L(\mathbf{x}|\mathbf{p}, \mathbf{P}_C)}{d(\mathbf{p})} \tag{14.6}$$

$$B_{nor}(\mathbf{p}) = \frac{B(\mathbf{p})}{d(\mathbf{p})}. \tag{14.7}$$

Moreover, because the two fitness functions above need to be maximized for best clustering performance, they will be rewritten in such a way that the maximization problem is converted into a minimization one, i.e.,

$$f_1(\mathbf{p}, \mathbf{P}_C) = |L_{nor}(\mathbf{x}|\mathbf{p}, \mathbf{P}_C)| \tag{14.8}$$

$$f_2(\mathbf{p}) = \frac{1}{B_{nor}(\mathbf{p})}. \tag{14.9}$$

The last issue to be addressed is the estimation of the number $\hat{C}$ of data classes representing the observed data since it is not known a priori. We have to resort to a technique that deals with this important issue, which is typical of mixture modeling problems. Indeed, the selection of the number of components in a mixture raises a tricky trade-off, since on one hand, the higher the number of components is, the higher the risk of data overfitting becomes, while on the other, the smaller the number of components is, the lower the model flexibility will be. In the literature, the most popular methods for automatically estimating the number of data classes are based on approximate Bayesian criteria or on information theory concepts [23]. We will use the MDL criterion, which takes origin from the information theory and is defined for a given number of classes, e.g., C classes, as [27]

$$MDL(C) = -L_{nor}(C) + \gamma \cdot K(C) \cdot \log(n) \tag{14.10}$$

where $L_{nor}(C)$ represents the normalized log-likelihood function value found at convergence of the PSO algorithm, $K(C)$ is the number of estimated statistical parameters, and γ is a constant. For the setting of γ, different values are proposed in the literature. According to [19], $\gamma = 5/2$ seems the most appropriate choice. The optimal number of data classes $\hat{C}$ is estimated by minimizing the MDL criterion,

i.e.,

$$\hat{C} = \arg \min_{C=C_{min},\ldots,C_{max}} \{MDL(C)\} \tag{14.11}$$

where C_{min} and C_{max} are predefined minimal and maximal numbers of data classes.

14.3.2 Priors Estimation Procedure

Another issue to solve is how to estimate the prior probability of each class since poor estimation of these probabilities can strongly affect the performance of the clustering process. A first strategy consists of including the probabilities as variables in the PSO particle as it is done for the mean and variance parameters. Since the sum of all prior probabilities must equal one, a constrained PSO search implementation would be needed. As an alternative, we will adopt another simpler and faster strategy, which is based on the idea of optimization by perturbing the priors outside but parallel with the PSO process. First, we start by clustering the n samples using the simple k-means algorithm [32] for getting an initial estimate of the priors. At each iteration of the PSO process, for each particle, a single prior probability $P\left(\omega_j\right)$ is selected at random. Then, its value is updated by adding a quantity Δ chosen randomly in the interval $\left[-P\left(\omega_j\right), 1 - P\left(\omega_j\right)\right]$. For the other classes, the prior probability values are updated by subtracting the amount $\Delta/(C-1)$. This way, the constraint requirement is fulfilled, and all the particles remain in the feasible region of the optimization space.

14.3.3 Algorithm Description

In Fig. 14.2 we show the flow chart of the presented MOPSO clustering methodology, which can be subdivided in six main steps. In the following, we describe the algorithm.

1. *Parameter Setting:*
   i. Choose the range of variation of the number of classes $[C_{min}, C_{max}]$.
   ii. Set C to C_{min}.
2. *PSO Initialization:*
   i. Initialize each particle position $\mathbf{p}_i$, $i = 1, 2, \ldots, S$, as follows:
      a. Run the k-means algorithm on the samples with a class number equal to C by considering only the randomly selected features encoded by $\mathbf{p}_i$.
      b. Initialize the mean and variance values coordinates of the particle using those given by the k-means algorithm.

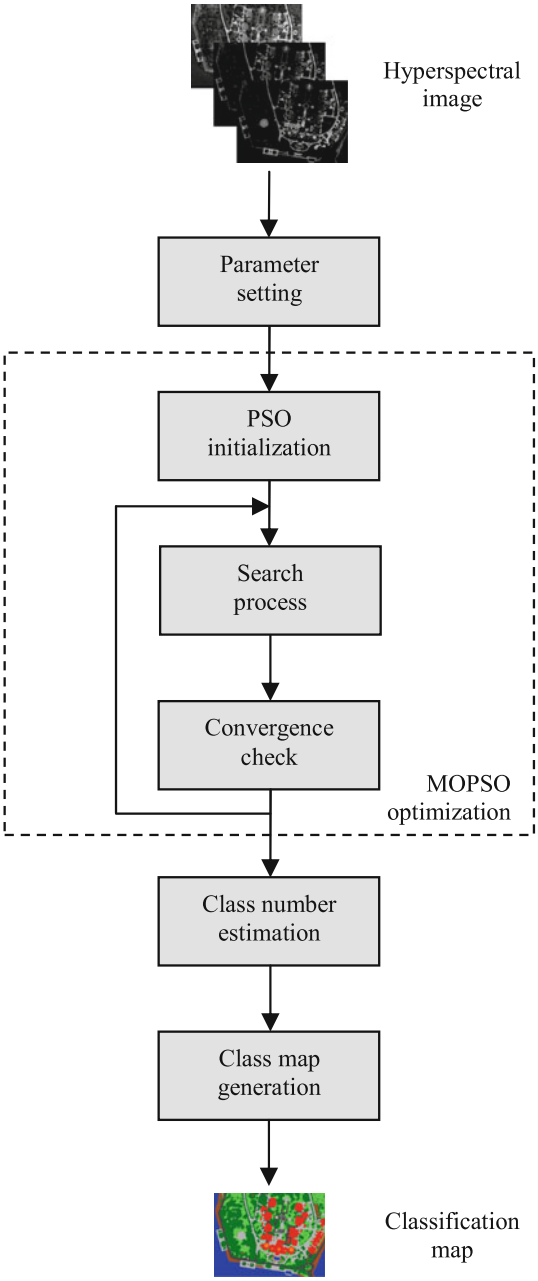


Fig. 14.2 Flow chart of the presented MOPSO clustering methodology

   c. Initialize the prior probabilities of each class by counting the sample number associated by the k-means algorithm with that class.
   ii. Set the best position of each particle with its initial position.
   iii. Set the velocity vectors $\mathbf{v}_i, i = 1, 2, \ldots, S$, that are associated with the S particles to $\mathbf{0}$.
   iv. Compute for each candidate particle $\mathbf{p}_i, i = 1, 2, \ldots, S$, its fitness functions $f_1(\mathbf{p}_i, \mathbf{P}_C)$ and $f_2(\mathbf{p}_i)$.
   v. Identify the nondominated solutions by applying the nondominated sorting algorithm described in [9] and store them in a list NL.

3. *Search Process:*
   i. Update the speed of each particle. To perform the update, the best global position $\mathbf{p}_g$ is selected from the list NL according to a tournament-based selection [9].
   ii. Update the position of each particle.
   iii. Update the prior probabilities of each class by means of the perturbation process described in Sect. 14.3.2.
   iv. Compute the fitness functions $f_1(\mathbf{p}_i, \mathbf{P}_C)$ and $f_2(\mathbf{p}_i)$ for each candidate particle $\mathbf{p}_i, i = 1, 2, \ldots, S$.
   v. Update the content of NL by inserting the current nondominated solutions. Clean NL from previous nondominated solutions, which now become dominated.
   vi. Update the best position $\mathbf{p}_{bi}$ of each particle if it is dominated by its current position $\mathbf{p}_i, i = 1, 2, \ldots, S$.

4. *Convergence Check:*
   i. Return to *Phase 3* if the convergence condition on the fitness functions or/and the maximal number of PSO iterations are not yet reached.
   ii. Increment the class number C by one, and return to *Phase 2* if C is less than or equal to the maximum class number C_{max}.

5. *Class Number Estimation:*
   i. For each class number $C \in [C_{min}, C_{max}]$, select one of the nondominated solutions $\mathbf{p}_i^*$ from the list NL. In particular, the closest solution to the origin of the performance space is selected.
   ii. Estimate the optimal number $\hat{C}$ of data classes by minimizing the MDL criterion.

6. *Classification Map Generation:* Generate a classification map by applying the MAP decision criterion [10] for each image pixel.

14.4 Experiments

14.4.1 Experimental Design

The experimental phase was performed on a real hyperspectral remote sensing image acquired in July 2002 by the ROSIS sensor over the city of Pavia, northern Italy. The image was composed of 102 spectral bands. A 400 × 400 pixel crop of the original image was used in the experiments. Some of the 102 bands are noisy and some are clean, as shown in Fig. 14.3a, b. A false color composite representation of the image is illustrated in Fig. 14.3c. Since the ground-truth is not available, we relied on the very high spatial resolution (1.2 m) of the image to assess qualitatively the classification results.

In order to evaluate the performance of the presented MOPSO clustering methodology, two sets of experiments were performed. In the first set, we assessed its capability in terms of cluster parameter estimation. For this purpose, the proposed methodology was run by minimizing only the first fitness function $f_1(\mathbf{p}, \mathbf{P}_C)$. Accordingly, the feature detection process was inhibited. Moreover, we assumed that the class number C was known and thus the MDL criterion was not used. In particular, we fixed the desired class number to ten. The method performance was evaluated through visual inspection of the clustering results.

In the second set of experiments, we intended to evaluate the performance of the entire MOPSO clustering methodology. This time, the MOPSO method was run by using both fitness functions $f_1(\mathbf{p}, \mathbf{P}_C)$ and $f_2(\mathbf{p})$ as well as the MDL criterion to estimate the number of data classes. In addition to the visual inspection of the maps, capabilities of the methodology in terms of detection of noisy features were also analyzed.

Concerning the PSO settings, in all experiments, we considered the following standard parameters: swarm size $S = 100$, inertia weight $w = 0.4$, acceleration constants c_1 and c_2 equal to unity, maximum number of iterations fixed to 100.

14.4.2 Experimental Results

In the first part of the experiments, we evaluated the behavior of the fitness function f_1 by varying the number of iterations of the optimization process. The graph of Fig. 14.4a shows a stable decreasing behavior, in which the faster decrease is verified in the first iterations. This behavior was somewhat expected, because the initial estimation of the cluster parameters done through the k-means algorithm is poor. In the second part of the plot the decrease is smoother suggesting that the fitness function is close to convergence. At convergence, we applied the MAP criterion in order to obtain the classification map shown in Fig. 14.4e. The same hyperspectral image was classified with the traditional k-means algorithm. The corresponding classification map is shown in Fig. 14.4g. From a visual inspection, the proposed strategy based

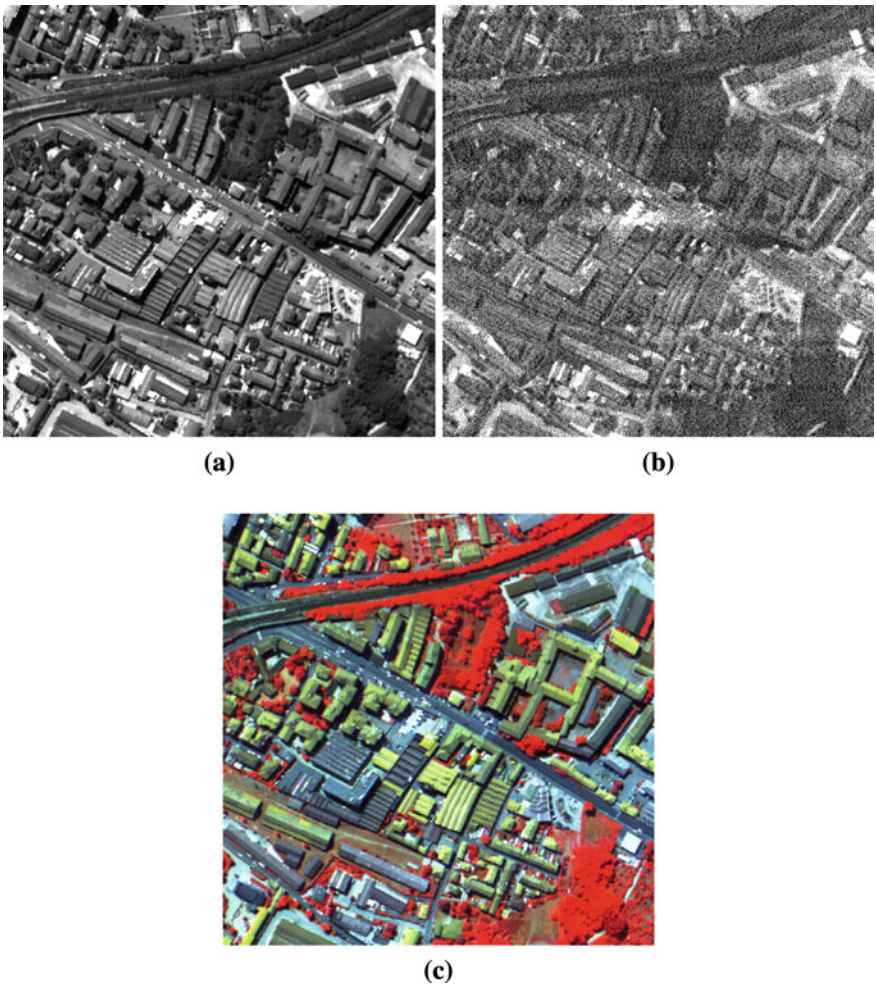


Fig. 14.3 Hyperspectral image acquired by the ROSIS sensor used in the experiments. Examples of **a** clean band (#30), **b** noisy band (#2), and **c** false color composite image (R: #100, G: #50, B: #10)

on PSO gives a map that is more coherent and thus more accurate with respect to the one provided by the k-means algorithm.

In the second part of the experiments, the entire MOPSO methodology was executed. The plot shown in Fig. 14.4b provides the values of the MDL criterion for a range of class numbers varied from three to 13 classes. Note that the minimum MDL value is obtained for a class number of ten. Considering the run associated with the ten class case, at convergence the Pareto front was composed of twelve solutions, which are depicted in Fig. 14.4c in logarithmic scale for a better visualization. Note how the different solutions are well spread along the front. From all these solutions,

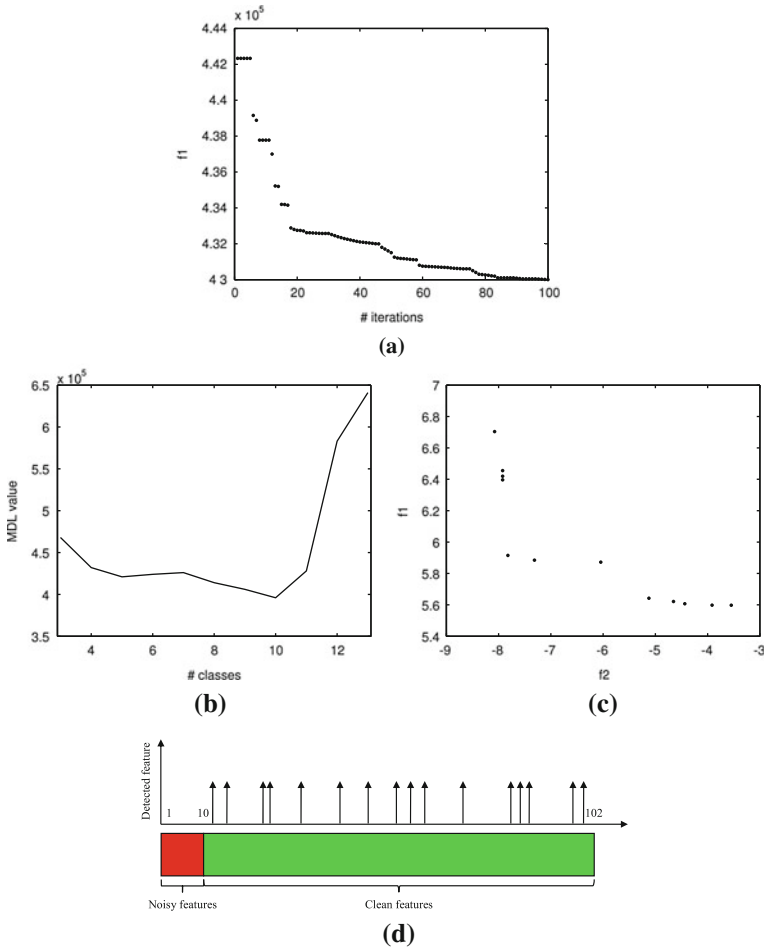


Fig. 14.4 Results obtained on the hyperspectral image acquired by the ROSIS sensor. **a** Fitness funcion value by varying the number of iterations. **b** Behavior of the MDL criterion. **c** Front of nondominated solutions. **d** Result of the feature detection process. Classification maps given by **e** the PSO-f_1, **f** the MOPSO, **g** the k-means, **h** the PCA + k-means

the closest one to the origin of the performance space is selected in order to have a tradeoff between the two different fitness functions. Considering the selected solution, the results in terms of feature detection are illustrated in Fig. 14.4d. The clean features of the hyperspectral image are represented in green, while the noisy features are depicted in red. The MOPSO detected 16 features (among 102), which are marked by arrows. From a visual inspection, all the selected features are really associated with clean bands. Thus the implemented feature detector allowed us to obtain a probability of detection of 100 % and of false alarm of 0 %. Note a good reduction ratio of around 15 % was achieved. Finally, the classification map generated using

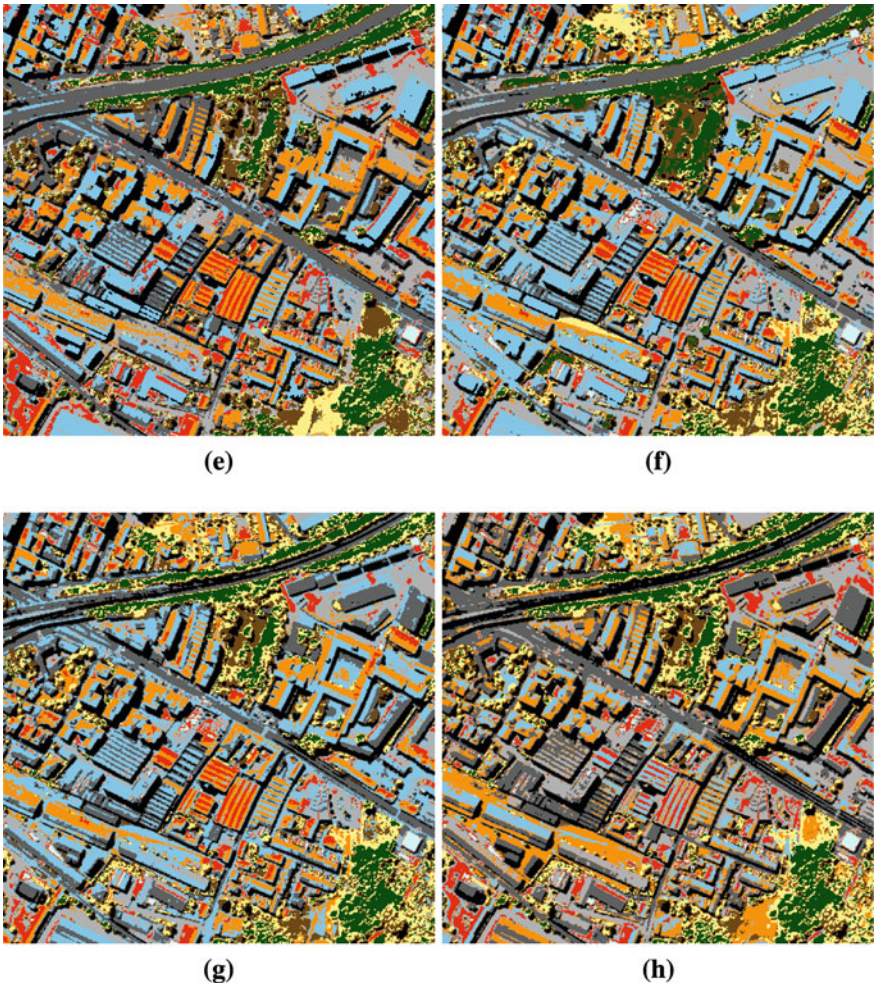


Fig. 14.4 Continued

the MAP criterion is given in Fig. 14.4f. We also implemented the clustering with the k-means method after applying a feature reduction step based on the first ten principal components produced by the well-known principal component analysis technique [26]. The resulting classification map is illustrated in Fig. 14.4h. From a visual inspection, the proposed MOPSO strategy confirms to attain the best accuracy with respect to the other methods.

14.5 Conclusion

In this chapter, we have presented a methodology for the unsupervised classification of hyperspectral images. It allows us to solve simultaneously problems of clustering, feature detection, and class number estimation in a completely automatic and unsupervised way. The proposed MOPSO solution provides an effective answer to this complex challenge, as shown by the experimental results. Indeed, it provides a very satisfactory classification accuracy while reducing drastically the number of bands used for the classification task. It yields a good estimation of the number of data classes characterizing the considered image. Such a guess, however, refers to data classes and not to thematic classes, which do not necessarily match. It could be refined in a second step by the user if it is in possession of some prior knowledge about the scene. Finally, this chapter has shown a successful application of PSO in remote sensing image analysis. Because of their capability to effectively handle complex optimization problems, it can be expected that PSO will continue to open the way to the design of attractive alternatives for many other remote sensing problems.

References

1. Bandyopadhyay, S., Maulik, U., Mukhopadhyay, A.: Multiobjective genetic clustering for pixel classification in remote sensing imagery. IEEE Trans. Geosci. Remote Sens. **45**(5), 1506–1511 (2007)
2. Barni, H., Garzelli, A., Mecocci, A., Sabatini, L.: A robust fuzzy clustering algorithm for the classification of remote sensing images. In: Proceedings of the IEEE International Geoscience and Remote Sensing Symposium 2000, Honolulu, Hawaii, vol. 5, pp. 2143–2145 (2000)
3. Bazi, Y., Melgani, F.: Semisupervised PSO-SVM regression for biophysical parameter estimation. IEEE Trans. Geosci. Remote Sens. **45**(6), 1887–1895 (2007)
4. Berge, A., Solberg, A.: Structured Gaussian components for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. **44**(11), 3386–3396 (2006)
5. Bhattacharyya, A.: On a measure of divergence between two statistical populations defined by their probability distributions. Bull. Calcutta Math. Soc. **35**, 99–109 (1943)
6. Bilgin, G., Erturk, S., Yildirim, T.: Unsupervised classification of hyperspectral-image data using fuzzy approaches that spatially exploit membership relations. IEEE Geosci. Remote Sens. Lett. **5**(4), 673–677 (2008)
7. Broilo, M., De Natale, F.: A stochastic approach to image retrieval using relevance feedback and particle-swarm optimization. IEEE Trans. Multimedia **12**(4), 267–277 (2010)
8. Clerc, M.: Particle-Swarm Optimization. ISTE, London (2006)
9. Deb, K.: MultiObjective Optimization Using Evolutionary Algorithms. Wiley, Chichester (2001)
10. Duda, R., Hart, P., Stork, D.: Pattern Classification, vol. 2. Wiley, New York (2001)
11. Dundar, M., Landgrebe, D.: A model-based mixture-supervised classification approach in hyperspectral data analysis. IEEE Trans. Geosci. Remote Sens. **40**(12), 2692–2699 (2002)
12. Eberhart, R., Shi, Y., Kennedy, J.: Swarm Intelligence. Morgan Kaufmann, San Mateo (2001)
13. Frery, A., de Araujo, C., Alice, H., Cerqueira, J., Loureiro, J., de Lima, M., Oliveira, M., Horta, M.: Hyperspectral images clustering on reconfigurable hardware using the k-means algorithm. In: Proceedings of the IEEE Symposium on Integrated Circuits and Systems Design 2003, Sao Paulo, Brazil, pp. 99–104. (2003)

14. Gaing, Z.: A particle-swarm optimization approach for optimum design of PID controller in AVR system. IEEE Trans. Energy Convers. **19**(2), 384–391 (2004)
15. Jia, X., Richards, J.: Cluster-space representation for hyperspectral data classification. IEEE Trans. Geosci. Remote Sens. **40**(3), 593–598 (2002)
16. Jia, X., Richards, J.: Efficient transmission and classification of hyperspectral image data. IEEE Trans. Geosci. Remote Sens. **41**(5), 1129–1131 (2003)
17. Kersten, P., Lee, J., Ainsworth, T.: Unsupervised classification of polarimetric synthetic aperture radar images using fuzzy clustering and EM clustering. IEEE Trans. Geosci. Remote Sens. **43**(3), 519–527 (2005)
18. Lee, S., Crawford, M.: Hierarchical clustering approach for unsupervised image classification of hyperspectral data. In: Proceedings of the IEEE International Geoscience and Remote Sensing Symposium 2004, vol. 2, pp. 941–944. Anchorage, Alaska (2004)
19. Liang, Z., Jaszczak, R., Coleman, R.: Parameter estimation of finite mixtures using the EM algorithm and information criteria with application to medical image processing. IEEE Trans. Nucl. Sci. **39**(4), 1126–1133 (1992)
20. Liu, X., Li, X., Zhang, Y., Yang, C., Xu, W., Li, M., Luo, H.: Remote sensing image classification based on dot density function weighted FCM clustering algorithm. In: Proceedings of the IEEE International Geoscience and Remote Sensing Symposium 2007, vol. 2, pp. 2010–2013. Barcelona, Spain (2007)
21. Lizzi, L., Viani, F., Azaro, R., Massa, A.: A PSO-driven spline-based shaping approach for ultrawideband (UWB) antenna synthesis. IEEE Trans. Antennas Propag. **56**(8), 2613–2621 (2008)
22. Marçal, A., Castro, L.: Hierarchical clustering of multispectral images using combined spectral and spatial criteria. IEEE Geosci. Remote Sens. Lett. **2**(1), 59–63 (2005)
23. McLachlan, G., Peel, D.: Finite Mixture Models. Wiley, New York (2000)
24. Melgani, F., Bazi, Y.: Classification of electrocardiogram signals with support vector machines and particle-swarm optimization. IEEE Trans. Inf. Technol. Biomed. **12**(5), 667–677 (2008)
25. Palubinskas, G., Datcu, M., Pac, R.: Clustering algorithms for large sets of heterogeneous remote sensing data. In: Proceedings of the IEEE International Geoscience and Remote Sensing Symposium 1999, vol. 3, pp. 1591–1593. Hamburg, Germany (1999)
26. Pearson, K.: On lines and planes of closest fit to systems of points in space. Philos. Mag. **2**(6), 559–572 (1901)
27. Rissanen, J.: Stochastic Complexity in Statistical Inquiry. World Scientific, Singapore (1989)
28. Shi, H., Shen, Y., Liu, Z.: Hyperspectral bands reduction based on rough sets and fuzzy C-means clustering. In: Proceedings of the IEEE Instrumentation and Measurement Technology Conference 2003, vol. 2, pp. 1053–1056. Vail, Colorado (2003)
29. Slade, W., Ressom, H., Musavi, M., Miller, R.: Inversion of ocean color observations using particle-swarm optimization. IEEE Trans. Geosci. Remote Sens. **42**(9), 1915–1923 (2004)
30. Tran, T., Wehrens, R., Hoekman, D., Buydens, L.: Initialization of Markov random field clustering of large remote sensing images. IEEE Trans. Geosci. Remote Sens. **43**(8), 1912–1919 (2005)
31. Tyagi, M., Bovolo, F., Mehra, A., Chaudhuri, S., Bruzzone, L.: A context-sensitive clustering technique based on graph-cut initialization and expectation-maximization algorithm. IEEE Geosci. Remote Sens. Lett. **5**(1), 21–25 (2008)
32. Webb, A.: Statistical Pattern Recognition. Wiley, Chichester (2002)
33. Wong, Y., Posner, E.: A new clustering algorithm applicable to multispectral and polarimetric SAR images. IEEE Trans. Geosci. Remote Sens. **31**(3), 634–644 (1993)
34. Xia, G., He, C., Sun, H.: A rapid and automatic MRF-based clustering method for SAR images. IEEE Geosci. Remote Sens. Lett. **4**(4), 596–600 (2007)
35. Zhong, Y., Zhang, L., Huang, B., Li, P.: An unsupervised artificial immune classifier for multi/hyperspectral remote sensing imagery. IEEE Trans. Geosci. Remote Sens. **44**(2), 420–431 (2006)

Chapter 15
A Computational Intelligence Approach to Emotion Recognition from the Lip-Contour of a Subject

Anisha Halder, Srishti Shaw, Kanika Orea, Pavel Bhowmik, Aruna Chakraborty and Amit Konar

Abstract This chapter provides an alternative approach to emotion recognition from the outer lip-contour of the subjects. Subjects exhibit their emotions through their facial expressions, and the lip region is segmented from their facial images. A lip-contour model has been developed to represent the boundary of the lip, and the parameters of the model are adapted using differential evolution algorithm to match it with the boundary contour of the lip. A support vector machine (SVM) classifier is then employed to classify the emotion of the subject from the parameter set of the subjects' lip-contour. The experiment was performed on 50 subjects in an age group from 18 to 25, and the average case accuracy in emotion classification is found to be 86 %.

A. Halder (✉) · S. Shaw · K. Orea · P. Bhowmik · A. Konar
Jadavpur University, Kolkata, India
e-mail: halder.anisha@gmail.com

S. Shaw
e-mail: srishti67@gmail.com

K. Orea
e-mail: kanor30@gmail.com

P. Bhowmik
e-mail: bpavel88@gmail.com

A. Konar
e-mail: konaramit@yahoo.co.in

A. Chakraborty
St. Thomas' College of Engineering and Technology, Kolkata, India
e-mail: aruna_stcet@rediffmail.com

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1_15, © Springer-Verlag Berlin Heidelberg 2013

15.1 Introduction

Emotion recognition is part and parcel of the next generation of human computer interactive (HCI) systems. To aid HCI systems, researchers are keen to determine emotion of the subjects from their facial expression, hand gesture and voice. For instance, Ekman and Friesen [3] proposed a scheme for recognition of facial expression from the movements of cheek, chin and wrinkles. Kobayashi and Hara [7–9] developed a scheme for recognition of human facial expression using the well known back-propagation neural network algorithm. Their scheme is capable of classifying facial expression depicting happiness, sadness, fear, anger, surprise and disgust. Yamada presented a new scheme of recognizing human emotions through classification of visual information. Frenandez-Dols et al. [31] designed a scheme for decoding emotion from both facial expression and content. Busso and Narayanan [11] compared the scope of facial expression, speech and multimodal expression in emotion recognition. Cohen et al. [19, 20] considered recognition of emotions from live video using hidden Markov Model. Gao et al.[14] proposed a technique for facial expression recognition from a single facial image using line-based caricatures. Lanitis et al. [12] proposed a novel technique for automatic interpretation and coding of facial images using flexible models. Some of the other well known works of emotion recognition from facial expressions include [1, 2, 10, 13, 15, 21–25, 30, 31].

Most of the existing works on emotion recognition employ one or more of the facial attributes for recognition and interpretation about the emotional state of the subject. However, we are afraid that there exists hardly any significant work on emotion recognition by a single facial feature. This chapter takes a serious attempt to recognize human emotion by considering the lip-contour [18, 26–29] of the subject. Although the possibility of emotion recognition from the lip-contour has already been explored in [4], we put forward our results for the following reason. In [4], the authors used a basic elliptical pattern and offered a method to tune the parameters of the ellipse to match it with the outer contour of a lip. It is apparent that the elliptical lip-contour cannot correctly capture the lip boundaries for all emotional instances.

This work overcomes this problem by judiciously selecting a six-segment lip-contour model, whose individual segments can be tuned to all typical non-overlapped lip-contours by controlling model parameters. An evolutionary algorithm is used to match the model lip-contour with the segmented lip boundary of a subject. Experiments with 50 volunteers reveal that there exists a correlation between the lip-contour pattern of the individual, and a specific emotion experienced by the subject. This observation motivates us to design a classifier to map the extracted parameters of the lip-contour model on to the emotional space. Several classifier algorithms can be utilized to study their relative performance to map the lip parameters to emotions. In this chapter we selected a support vector machine (SVM) classifier for its wide spread popularity and our own experience about its merit in emotion classification problem.

The rest of the chapter is divided into seven sections. Section 15.2 offers the modeling issues of the human lip-contour. In Sect. 15.3, segmentation methods of

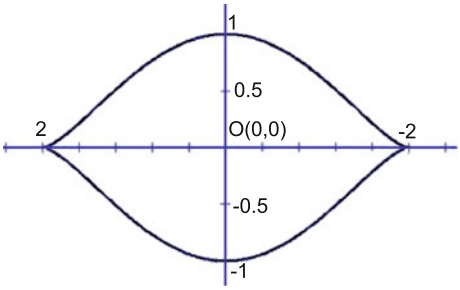


Fig. 15.1 The standard kiss curve

the lip-contour are discussed. We introduce differential evolution and demonstrate its scope in determining the best tuned parameter set of the lip-contour model in Sect. 15.4. Section 15.5 presents a scheme for classification of emotion of a subject from the exact parameters of her lip-contour. Section 15.6 deals with the experiments and results. Performance analysis is given in Sect. 15.7. The conclusion of the chapter is outlined in Sect. 15.8.

15.2 The Proposed Lip-Contour Model

At present, there is no universally accepted model of lip-contour. In this work, we start with the elementary kiss curve (Fig. 15.1), and modify it at different segments, as indicated in Fig. 15.2, to obtain an ideal model of the curve, capable of capturing most of the nonoverlapped lip-contours in different emotional states.

The basic equation of the kiss curve (Fig. 15.1) is given by

$$y^2 = (1 - x^2)^3, -1 \leq x \leq 1. \tag{15.1}$$

The curve returns both positive/negative values of y for each value of x. We, however, use the entire positive half of the curve, and a portion of the negative half. The remaining negative half is replaced by a parabola for better matching with lip profiles of the subjects. When the domain $-1 \leq x \leq +1$ is replaced by $[-l, +l]$, Eq. (15.1) is written as

$$y = \left(1 - \left(\frac{x}{l}\right)^2\right)^{\frac{3}{2}} \tag{15.2}$$

To determine all except the segment GA in Fig. 15.2, we scaled the right-hand side of Eq. (15.2) and added one or more extra terms, as needed, and determine the parameters of the new curve by setting suitable boundary conditions corresponding to the corner points and axis crossings as listed in Table 15.1. The resulting parameters are also given in the table.

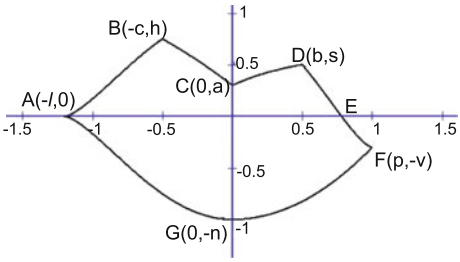
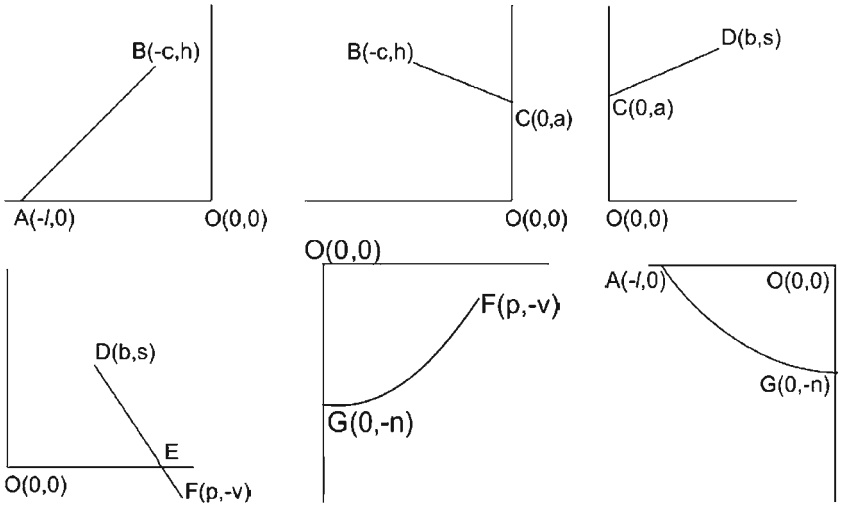


Fig. 15.2 The proposed model of the lip outer profile

Table 15.1 Proposed lip segments with boundary conditions



15.3 Segmentation of the Lip-Contour

Several algorithms for the lip segmentation are available in the literature [1, 5]. In this chapter we, however, employ a fuzzy c-means clustering algorithm to segment the lip region from the rest of the facial expression. Any pixel x_k here is considered to fall either in the lip or the nonlip region. Let $L(x_k)$ and $NL(x_k)$ be the membership of pixel x_k to fall in the lip and the nonlip regions, respectively.

A pixel in this work is denoted by five attributes: three attributes of color information $(L \times a \times b)$, and two attributes of position information (x, y). The objective of the clustering algorithm is to classify the set of five-dimensional data points into two classes/partitions – the lip region and the nonlip region. Initial membership values are assigned to each five-dimensional pixel, such that the sum of the memberships in the two regions is equal to one. That is, for the kth pixel x_k ,

$$L(x_k) + NL(x_k) = 1 \tag{15.3}$$

Given the initial membership values of $L(x_k)$ and $NL(x_k)$ for $k = 1\ to\ n^2$ (assuming that the image is of size $n \times n$) [32], we use the FCM algorithm to determine the cluster centers, V_L and V_{NL}, of the lip and the nonlip regions:

$$V_L = \frac{\sum_{k=1}^{n^2} [L(x_k)]^m x_k}{\sum_{k=1}^{n^2} [L(x_k)]^m} \tag{15.4}$$

$$V_{NL} = \frac{\sum_{k=1}^{n^2} [NL(x_k)]^m x_k}{\sum_{k=1}^{n^2} [NL(x_k)]^m} \tag{15.5}$$

Expressions (15.4) and (15.5) provide centroidal measures of the lip and non-lip clusters [33], evaluated over all data points x_k for $k = 1$ to n^2. The parameter $m(> 1)$ is any real number that affects the membership grade. The membership values of pixel x_k in the image for the lip and the non-lip regions are obtained from the following formulae:

$$L(x_k) = \left(\sum_{j=1}^{2} \left(\frac{\|x_k - v_L\|^2}{\|x_k - v_j\|^2} \right)^{\frac{1}{M-1}} \right)^{-1} \tag{15.6}$$

$$NL(x_k) = \left(\sum_{j=1}^{2} \left(\frac{\|x_k - v_{NL}\|^2}{\|x_k - v_j\|^2} \right)^{\frac{1}{M-1}} \right)^{-1} \tag{15.7}$$

where v_j denotes the jth cluster center for $j \in L, NL$.

Determination of the cluster centers (by 15.4 and 15.5) and membership evaluation (by Eqs. 15.6 and 15.7) are repeated several times following the FCM algorithm until the positions of the cluster centers do not change significantly.

Figure 15.3a presents a section of a facial image with a large mouth opening. This image is passed through a median filter and the resulting image is shown in Fig. 15.3b. Application of the FCM algorithm to the image in Fig. 15.3b yields the image in Fig. 15.4. The dark part in Fig. 15.4 represents the lip region, and the skin and teeth regions are represented by white color.

15.4 Parameter Extraction of a Given Lip-Contour Using Differential Evolution Algorithm

Evolutionary algorithms have frequently been used over the last three decades for handling unconstrained/constrained optimization problems. Given the lip-contour of

Fig. 15.3 **a** The original face, **b** the median filtered (**a**)

Fig. 15.4 The segmented mouth region obtained from 15.3b by FCM algorithm

a subject, determining the parameters of the mathematical model representing the generalized lip-contour that matches best with the actual lip-contour is the problem of concern in the present context. We consider a set of marked points of interest in a given lip outer contour, as shown in Fig. 15.2. We construct vectors of nine components, representing trial solutions for the present problem. The components of a vector here represent the control parameters of the lip-contour model, including b, c, l, p, v, n, a, h and s. Differential Evolution (DE) proposed by Storn and Price [16] is one derivative-free optimization algorithm, which offers promising solution to a global optimization problem. In this chapter, we employ differential evolution as the optimization algorithm to determine the lip parameters of a given subject carrying a definite emotion.

15.4.1 The Classical Differential Evolution Algorithm

The DE algorithm initializes a set of trial solutions, called parameter vector. The parameter vectors are evolved through a process of mutation and recombination, and a selection scheme is used to identify the better candidate between the evolved and the trial solution (parameter vector). A brief overview of the DE algorithm is available in any standard text. We here provide a simple pseudocode to the classical DE algorithm. In this problem we used a variant of DE named as $DE/rand/1/bin$.

An iteration of the classical DE algorithm consists of the four basic steps: initialization of a population of vectors, mutation, crossover or recombination and, finally selection. The main steps of classical DE are given below.

1. Set the generation number $t = 0$ and randomly initialize a population of NP individuals $P_t = \{X_1(t), X_2(t), \ldots, X_{NP}(t)\}$ with $X_i(t) = \{x_{i,1}(t), x_{i,2}(t), \ldots, x_{i,D}(t)\}$ and each individual uniformly distributed in the range $[X_{min}, X_{max}]$, where

   $X_{min} = \{x_{min,1}, x_{min,2}, \ldots, x_{min,D}\}$ and
   $X_{max} = \{x_{max,1}, x_{max,2}, \ldots, x_{max,D}\}$ with $i = [1, 2, \ldots, NP]$,

   and D denotes the dimension of an individual data point.

2. **while** stopping criterion is not reached, **do**
   for $i = 1\ to\ NP$

   a. **Mutation:**
      Generate a donor vector
      $V(t) = \{v_{i,1}(t), v_{i,2}(t), \ldots, v_{i,D}(t)\}$ corresponding to the ith target vector $X_i(t)$ by the following scheme

      $$V_1(t) = X_{r_1}(t) + F \times (X_{r_2}(t) - X_{r_3}(t))$$

      where $r_1, r_2\ and\ r_3$ are mutually exclusive random integers in the range $[1, NP]$, and F is a scale factor in [0, 2].

   b. **Crossover:**
      Generate trial vector
      $U_i(t) = \{u_{i,1}(t), u_{i,2}(t), \ldots, u_{i,D}(t)\}$ for the ith target vector $X_i(t)$ by binomial crossover as

      $$\begin{aligned} u_{i,j}(t) &= v_{i,j}(t) \quad if\ (rand(0,1)) < C_r \\ &= x_{i,j} \quad otherwise, \end{aligned}$$

      where C_r is a predefined real number in [0, 1], called the crossover rate.

   c. **Selection:** Evaluate the trial vector $U_i(t)$
      if $f(U_i(t)) \leq f(X_i(t))$, **then** $X_i(t+1) = U_i(t)$

      $$f(X_i(t+1)) = f(U_i(t)$$

      where $f(.)$ is the fitness function
      end if
      end for

   d. Increase the counter value $t = t + 1$
      end while

Table 15.2 Parametric equations for the proposed lip segments

Curve segment	Presumed equation	Boundary condition	Parameter obtained by setting the boundary conditions
AB	$y = a_1\left(1-\left(\frac{x}{l}\right)^2\right)^{\frac{3}{2}} + a_2$	$-l \le x \le -c$ $0 \le y \le h$	$a_1 = \frac{h}{\left(1-\left(\frac{c}{l}\right)^2\right)^{\frac{3}{2}}}$ $a_2 = 0$
BC	$y = a_3\left(1-\left(\frac{x}{l}\right)^2\right)^{\frac{3}{2}} + a_4 x$	$-c \le x \le 0$ $h \le y \le a$	$a_3 = a$ $a_4 = \frac{a\left(1-\left(\frac{c}{l}\right)^2\right)^{\frac{3}{2}} - h}{c}$
CD	$y = a_5\left(1-\left(\frac{x}{p}\right)^2\right)^{\frac{3}{2}} + a_6 x$	$0 \le x \le b$ $a \le y \le x$	$a_5 = a$ $a_6 = \frac{s - a\left(1-\left(\frac{b}{p}\right)^2\right)^{\frac{3}{2}}}{b}$
DEF	$y = a_7\left(1-\left(\frac{x}{p}\right)^2\right)^{\frac{3}{2}} + a_8 2$	$b \le x \le p$ $s \le y \le -v$	$a_7 = \frac{s+v}{\left(1-\left(\frac{b}{p}\right)^2\right)}$ $a_7 = 0$ $a_8 = -v$
FG	$y = a_9 x^2 + a_{10} x + a_{11}$	$p \le x \le 0$ $-v \le y \le -n$	$a_9 = \frac{n-v}{p^2}$ $a_{10} = 0$ $a_{11} = -n$
GA	$y = \pm\left(1-x^2\right)^{\frac{3}{2}}$	$0 \le x \le -l$ $-n \le y \le 0$	$y = -n\left(1-\left(\frac{x}{l}\right)^2\right)^{\frac{3}{2}}$

The parameters used in the algorithm namely scaling factor F and crossover rate C_r should be initialized before calling the while loop. The terminate condition can be defined in many ways, a few of which include:

(i) fixing the number of iterations N.
(ii) when the best fitness of population does not change appreciably over successive iterations.
(iii) Either of (i) and (ii), whichever occurs earlier.

15.4.2 System Identification Approach to Lip-Contour Detection by Differential Evolution

Given a finite set of selected points on the lip boundary of a segmented mouth region and a model lip curve, we need to match the response of the model curve with

the selected data points by varying the parameters of the model curve. The set of parameters for which the best matching takes place between the model generated data points and the selected lip boundary points are the results of a system identification procedure adopted here. The model curve is adapted by changing its parameters using a differential evolution algorithm.

Let, $y = f(x)$ be the model curve. Then for all (x, y) lying on the curve, we obtain $G(x, y) = 1$, and for all points $y \neq f(x)$, $G(x, y) = 0$. Let, $L(x, y) = 1$ for all valid data points on the outer boundary of a segmented lip. We use a performance evaluation metric J, where

$$J = \sum_{\forall x} \sum_{\forall y, y=f(x)} |G(x, y) - L(x, y)|$$

In DE algorithm, we used J as the fitness function, where we wanted to minimize J for all valid (x, y) on the lip boundary. The DE considers nine-parameter population vectors, adapts the trial population by mutation and recombination, and selects the best of the target vector and the original parameter vector to determine the parameter vector in the next iteration. This is done in parallel for NP number of parameter vectors, where NP is the population size. The algorithm is terminated when the error limit [16], defined by the difference of J's between the best of the previous and the current iteration is below a prescribed threshold. The best fit parameter vector is the parameter set of the best model lip-contour matched with a given lip boundary data points.

15.5 Emotion Classification from Measured Parameters of the Lip-Contour Model

It is noted from a large number of lip-contour instances that there exist at least two parameters of the lip model that are clearly distinctive of individual emotions. So any typical machine learning/statistical classifier can be employed to classify the different emotional status from the parameter of lip-contour. In this chapter we use a support vector machine (SVM) [2, 17] classifier for emotion classification from the lip data.

A Support Vector Machine (SVM) has successfully been used for both linear and nonlinear classification. However, as nonlinear operation yields results with lesser accuracy, in this chapter, we focus on the linear operation only. To understand the basic operation of SVM, let, X be the input vector and y be the desired scalar output that can take $+1$ or -1 values, indicating linear separation of the pattern vector X.

The function $f(X, W, b)$ can be represented as follows:

$$f(X, W, b) = sign(WX + b) \tag{15.8}$$

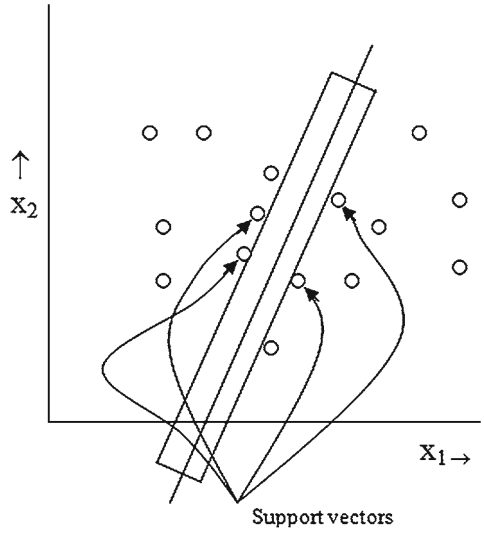


Fig. 15.5 Defining support vector for a linear SVM system

where $W = [w_1\ w_2 ... w_n]$ is the weight vector,
$X = [x_1\ x_2 ... x_n]^T$ represents the input vector, b is the bias.

The function f classifies the input vector X into two classes denoted by $+1$ or -1. The straight line that segregates the two pattern classes is usually called a hyperplane. Further, the data points that are situated at the margins of the two boundaries of the linear classifier are called support vectors. Figure 15.5 describes a support vector for a linear SVM.

Let us now select two points $X+$ and $X-$ as two support vectors. Thus by definition

$$WX^+ + b = +1 \tag{15.9}$$

$$WX^- + b = -1 \tag{15.10}$$

which jointly yields

$$W(X^+ - X^-) = 2 \tag{15.11}$$

Now, the separation between the two support vectors lying in the class $+1$ and class -1, called marginal width, is given by

$$M = \frac{\{(WX^+ + b) - (WX^- + b)\}}{\|W\|} = \frac{2}{\|W\|} \tag{15.12}$$

The main objective in a linear SVM is to maximize M, i.e., to minimize $\|W\|$, which is same as minimizing $\frac{1}{2}W^T W$. Thus, the linear SVM can be mathematically described by:

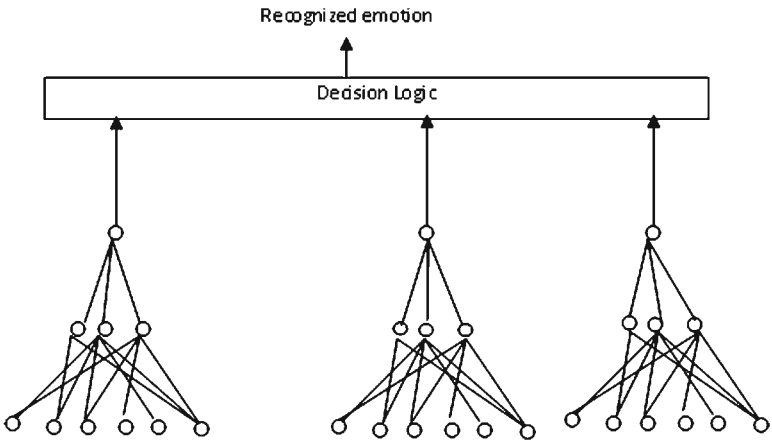


Fig. 15.6 Emotion classification by support vector machines

$$Minimize\ \phi(W) = \frac{1}{2} W^T W \tag{15.13}$$

subject to $y_i(WX_i + b) \geq 1$ for all i, where y_i is either 1 or -1 depending on the class which X_i belongs to.

Here, the objective is to solve W and b to satisfy the above equation. In this paper, we are not presenting the solution to the optimization problem, referred to above. This is available in standard texts on neural networks [11]. One important aspect of SVM is the kernel function selection. For linear SVM, the kernel K for two data points X_i and X_j is defined by

$$K(X_i, X_j) = X_i^T X_j \tag{15.14}$$

In our basic scheme, we employed five SVM networks, one each for joy, anger, sadness, fear and relaxation. The ith SVM network is trained with all of the training instances of the ith class with positive levels, and all other training instances with negative levels. The decision logic box driven by the five SVM networks ultimately recognizes the emotion corresponding to the supplied feature vector X. Figure 15.6 provides this SVM-based emotion recognition scheme.

The decision logic works in the following manner. If only one input of the decision logic is $+1$, it infers the corresponding class number at the output. For example, if the SVM-disgust only in Fig. 15.6 generates a $+1$, the output of the decision logic will be the class number for the emotion class: disgust.

When more than one input of the decision logic is $+1$, the decision is taken in two steps. First, we count the number of positive instances falling in the small neighborhood of the given pattern for each emotion class with its corresponding SVM-output $+1$. Next, the emotion class with the highest count is declared as the winner. The decision logic thus takes decision based on the principle of "majority

voting", which is realized here by the count of positive instances in the neighborhood of the given test pattern. The "neighborhood" here is defined as a nine-dimensional sphere function around a given test pattern, considering it as the center of the sphere.

The radius of the sphere is determined from the measurements of standard deviation in the individual features. The largest among the standard deviations for all the features is considered as the radius of the "neighborhood" in the data points, representing positive instances in a given emotion class. The radius of different emotion classes here is thus different. This, however, makes sense as the data density (per unit volume in nine-dimensional hyperspace) for different emotion classes is non-uniform.

To study the performance of classification using linear SVM we employed leave-one-out cross-validation . Leave-one-out classification involves using a single observation from the original sample as the classified data, and the remaining observations as the training data. This is repeated such that each observation in the sample is used once as the classified data. This is the same as a K-fold classification with K being equal to the number of observations in the original sample.

15.6 Experiments and Results

The experiment has two phases. In the first phase, we determine weight vectors of five SVM classifiers, each one for one emotion class, including anger, disgust, fear, happiness and sadness. We had 50 subjects, and for each subject we obtained ten facial expressions for ten different instances of each emotion. Thus, for five emotions, we had 50 facial expressions for individual subjects. Three out of ten instances of emotional expression are given in Tables 15.3 and 15.5 for two subjects.

Now, for each facial expression given in Tables 15.3 and 15.5, we segmented the mouth region by a fuzzy c-means clustering algorithm, and determined the optimal lip-parameters: b, c, l, p, v, n, a, h and s by adapting the model lip-contour with the outer boundary of individual segmented lips to have an optimal matching between the two. This matching was performed by classical DE algorithm. Tables 15.4 and 15.6 are the results of lip-parameters obtained from Tables 15.3 and 15.5, respectively. The weight vectors for the SVM classifiers for individual emotions of a subject are then identified. This is done by first preparing a table with ten positive and 40 negative instances for each emotion class of a subject. The weight vector for the given SVM-classifier for the emotion class is determined in a manner so that all the positive and negative instances are separated with a margin of $\frac{2}{\|W\|}$. This is done for all individual subjects separately.

The second phase of the experiment starts with an unknown facial expression of a known subject. We first obtain mouth region from the image by FCM clustering, and determine lip-parameter by DE using the model lip. Now, we feed the lip-parameters to all the five SVM classifiers for the person concerned. The output of one or more SVM-classifiers may be +1. The decision logic then determines the emotion class of the unknown pattern.

Table 15.3 Facial expression for subject 1 for different emotions

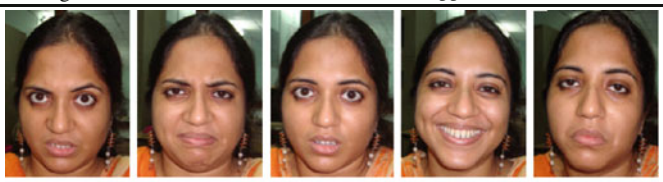
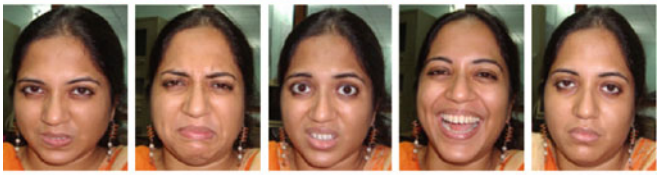
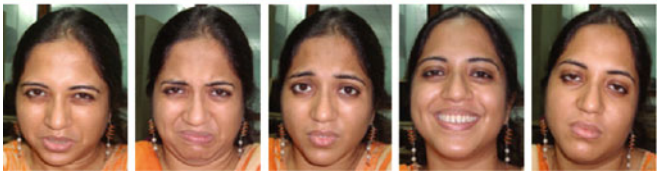
	Emotion				
Instances	Anger	Disgust	Fear	Happiness	Sadness
1.					
2.					
3.					

Table 15.4 Lip parameters for subject 1 for different emotions

Emotion	Instance	*b*	*c*	*l*	*p*	*v*	*n*	*a*	*h*	*s*
HAPPY	1	55	63	226	228	55	194	25	14	0
	2	46	80	221	249	36	241	44	52	44
	3	45	69	221	228	49	156	49	54	49
SAD	1	39	38	170	149	38	64	64	81	75
	2	37	45	171	152	38	45	45	66	53
	3	38	40	164	159	17	67	67	79	74
FEAR	1	48	49	148	142	18	116	46	60	52
	2	45	54	151	167	26	141	38	61	54
	3	44	47	143	151	29	126	36	54	44
DISGUST	1	41	40	190	168	26	96	10	24	19
	2	35	36	187	159	22	92	12	27	23
	3	39	48	176	165	30	98	12	29	18
ANGER	1	36	48	147	143	33	133	31	49	40
	2	32	48	161	168	25	134	26	45	35
	3	33	39	140	144	25	127	42	53	49

Table 15.5 Facial expression for subject 2 for different emotions

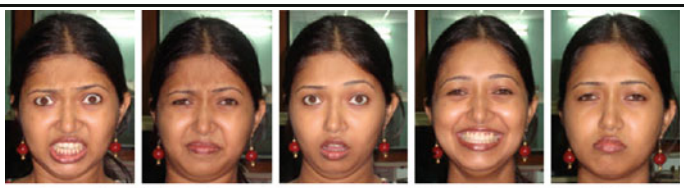
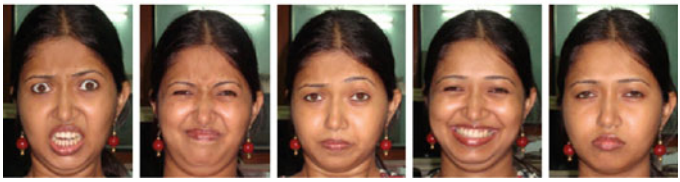
	Emotion				
Instances	Anger	Disgust	Fear	Happiness	Sadness
1.					
2.					
3.					

Table 15.6 Lip parameters for subject 2 for different emotions

Emotion	Instance	*b*	*c*	*l*	*p*	*v*	*n*	*a*	*h*	*s*
HAPPY	1	28	36	225	197	33	139	66	76	73
	2	36	46	231	223	27	179	14	25	16
	3	37	36	221	211	0	155	58	52	65
SAD	1	38	45	149	148	17	90	61	68	64
	2	35	47	145	149	14	78	59	68	65
	3	35	40	158	146	23	74	61	75	70
FEAR	1	40	50	145	145	14	115	68	73	110
	2	28	45	136	152	19	80	23	35	28
	3	40	187	138	139	10	120	73	63	74
DISGUST	1	38	54	171	145	30	82	30	41	33
	2	37	73	173	132	37	84	24	35	28
	3	44	65	179	159	36	86	18	28	21
ANGER	1	48	40	186	185	19	149	71	81	81
	2	43	34	169	154	8	150	76	83	84
	3	45	42	158	172	19	148	54	65	63

Table 15.7 Comparative study of original emotion and classified emotion for subject 1

		Original emotion				
		Anger	Disgust	Fear	Happiness	Sadness
Classified emotion	Anger	60	10	10	0	0
	Disgust	10	90	0	0	10
	Fear	30	0	90	0	0
	Happy	0	0	0	100	0
	Sad	0	0	0	0	90

Table 15.8 Accuracy of emotion classification after omitting parameters one-by-one for subject-1

Parameter	*b*	*c*	*l*	*p*	*v*	*n*	*a*	*h*	*s*
Accuracy (%)	78	88	88	90	90	70	90	86	86

15.7 Performance Analysis

The experiment presented in the last section is now repeated for 30 unknown instances of emotion taken from the first experimental subject. Table 15.7 provides the results of classification accuracy of unknown emotional instances taken for subject 1. It is clear from the table that happiness here is correctly identified in 100 % of the cases. Sadness is confused with disgust in 10 % of the cases. Disgust is confused with anger in 10 % of the cases. Anger seems to be very complex emotion. It is correctly classified in 60 % of the cases. It is sometimes mis-interpreted as disgust and fear. Disgust is correctly classified in 90 % of the cases, but in 10 % of the cases it is misinterpreted as anger.

Table 15.9 provides the results for emotion classification for subject two. Here too we considered 30 unknown instances of emotion taken for subject two, and classified the emotion into five classes. The results given in Table 15.9 indicate that here anger, disgust and happiness are classified correctly in 100 % of the cases. There are, however, confusions in sadness and fear.

One interesting experiment was to determine the significance of individual lip-parameters. We dropped one parameter at a time and designed the weight vector for SVM-classifier for different emotions. Now, we feed the lip-parameters of unknown emotional instance of the same subject and determine the classification accuracy in absence of one parameter each.

The results of classification accuracy while omitting one parameter are given in Tables 15.8 and 15.10, respectively, for subjects one and two. The most important parameter would be the one where classification falls off by a bigger margin.

From Tables 15.8 and 15.10, it is clear that parameter *b* has an important role in emotion classification, since in the absence of this, classification falls off by a large margin.

Table 15.9 Comparative study of original emotion and classified emotion for subject 2

		Original emotion				
		Anger	Disgust	Fear	Happiness	Sadness
Classified emotion	Anger	100	0	0	0	0
	Disgust	0	100	20	0	10
	Fear	0	0	80	0	0
	Happy	0	0	0	100	0
	Sad	0	0	0	0	90

Table 15.10 Accuracy of emotion classification after omitting parameters one-by-one for subject 2

Parameter	*b*	*c*	*l*	*p*	*v*	*n*	*a*	*h*	*s*
Accuracy (%)	86	94	96	94	90	96	90	92	88

In general, facial expression based emotion recognition usually has an average classification accuracy of (80–90) % [2, 3, 8, 10]. To obtain the above classification accuracy, we require a large number of facial features. The present experiment, however, attempts to use a single facial attribute –the lip. The experiments undertaken here provide us with an average classification accuracy of 86 %, which is better than the classification accuracies reported for the existing lip-contour based emotion classification schemes [6].

15.8 Conclusion

This chapter proposed a new approach to emotion classification from the lip-contour of the subjects experiencing a specific emotion. It employed lip segmentation, lip parameter evaluation and classification by SVM to determine the emotion class of the subject.

The lip-contour used here is unique and unknown to the machine intelligence community. Experiments with a large number of subjects confirm that the proposed model can capture most of the experimental lip-contours for a specific emotive experience of the subject. The DE algorithm used here is very fast and robust and thus can easily determine the parameters of the lip-contour within first 50 iterations of the execution of DE program. The SVM classifier, which is already an established tool for pattern classification with high accuracy, has been utilized here for classifying lip parameters into emotions. Experiments here too confirm that the percentage accuracy in classification of emotion on an average is 86 %, as obtained from the data set of 50 subjects, each having 10 frames per emotion.

References

1. Chakraborty, A., Konar, A., Chakraborty, U.K., Chatterjee, A.: Emotion Recognition From Facial Expressions and Its Control Using Fuzzy Logic. IEEE Trans. Syst. Man Cybern. Part A **39**(4), 726–743 (2009)
2. Konar, A., Chakraborty, A.: Emotional Intelligence: a cybernetic approach. Springer, Heidelberg (2009)
3. Ekman, P., Friesen, W.V.: Unmasking the Face: A Guide to Recognizing Emotions From Facial Clues. Prentice-Hall, Englewood Cliffs (1975)
4. Rizon, M., Karthigayan, M., Yaacob, S., Nagarajan, R.: Japanese face emotions classification using lip features. In: IEEE Computer Society Geometric Modelling and Imaging (GMAI'07), School of Mechatronics Engineering, Universiti Malaysia Perlis, Jejawi, Perlis, Malaysia (2007)
5. Bouvier, C., Coulon, P.-Y., Maldague, X.: Unsupervised lips segmentation based on ROI optimisation and parametric model. In: IEEE GIPSA_lab, INPG, CNRS, UJF, U.Stendhal 46 av. F. Viallet, France and LVSN, University Laval, Sainte-Foy, mQuebec, Canada (2007)
6. Yaling, L., Minghui, D.: Lip contour extraction based on manifold. In: IEEE Proceedings of the 2008 International Conference on MultiMedia and Information Technology, pp. 229–232, Washington, DC, USA (2008)
7. Kobayashi, H., Hara, F.: The recognition of basic facial expressions by neural network. Trans. Soc. Instrum. Contr. Eng. **29**(1), 112–118 (1993)
8. Kobayashi, H., Hara, F.: Measurement of the strength of six basic facial expressions by neural network. Trans. Jpn. Soc. Mech. Eng. (C) **59**(567), 177–183 (1993)
9. Kobayashi, H., Hara, F.: Recognition of mixed facial expressions by neural network. Trans. Jpn. Soc. Mech. Eng. (C) **59** (1993)
10. Bashyal, S., Venayagamoorthy, G.K.: Recognition of facial expressions using Gabor wavelets and learning vector quantization. Int. J. Eng. Appl. Artif. Intell. **21**, 1056–1064 (2008)
11. Busso, C., Narayanan, S.: Interaction between speech and facial gestures in emotional utterances: A single subject study. IEEE Trans. Audio Speech Lang. Process. **15**(8), 2331–2347 (2007)
12. Lanitis, A., Taylor, C.J., Cootes, T.F.: Automatic interpretation and coding of face images using flexible models. IEEE Trans. Pattern Anal. Mach. Intell. **19**(7), 743–756 (1997)
13. Zeng, Z., Fu, Y., Roisman, G.I., Wen, Z., Hu, Y., Huang, T.S.: Spontaneous emotional facial expression detection. Int. J. Multimedia **1**(5), 1–8 (2006)
14. Gao, V., Leung, M. K. H., Hui, S. C., Tananda, M.W.: Facial expression recognition from line-based caricatures. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **33**(3), 407–412 (2003)
15. Panti, M., Patras, I.: Dynamics of facial expression: recognition of facial actions and their temporal segments from face profile image sequences. IEEE Trans. Syst. Man Cybern. B Cybern. (2006)
16. Price, K. V., Storn, R. M., Lampinen, J. A.: Differential Evolution: a practical approach to global optimization. Springer, Berlin (2005)
17. Wang, L.: Support Vector Machines: theory and applications studies in fuzziness and soft computing, Springer, Berlin (2010)
18. Yang, Y., Wang, X., Qiarr, Y., Lin, S.: Accurate and real-time lip contour extraction based on constrained contour growing. In: Conferences on Pervasive Computing (JCPC), pp. 589–594 (2009)
19. Cohen, I.: Facial expression recognition from video sequences. M.S. thesis, University of Illinois at Urbana-Champaign, Department of Electric Engineering, Urbana (2000)
20. Cohen, I., Sebe, N., Garg, A., Chen, L.S., Huang, T.S.: Facial expression recognition from video sequences: Temporal and static modeling. Comput. Vis. Image Underst. **91**(1/2), 160–187 (2003)

21. Karthigayan, M., Rizon, M., Yaacob, S., Nagarajan, R., Sugisaka, M., Rozailan Mamat, M., Desai, H.: Fuzzy clustering for genetic algorithm based optimized ellipse data in classifying face emotion. In: International Conference on Control, Automation and Systems 2007 ICCAS 07, pp. 1–5 (2007)
22. Goleman, D.: Emotional Intelligence. Bantam, New York (1995)
23. Das, S., Halder, A., Bhowmik, P., Chakraborty, A., Konar, A., Nagar, A.K.: A support vector machine classifier of emotion from voice and facial expression data. IEEE World Congr. Nat. Biol. Inspired Comput. NaBIC **2009**, 1010–1015 (2009)
24. Das, S., Halder, A., Bhowmik, P., Chakraborty, A., Konar, A., Nagar, A. K.: Voice and facial expression based classification of emotion using linear support vector machine. In: Proceedings of the 2009 Second International Conference on Developments in eSystems Engineering (DESE '09), pp. 377–384 (2009)
25. Ghosh, M., Chakraborty, A., Acharya, A., Konar, A., Panigrahi, B. K.: A recurrent neural model for parameter estimation of mixed emotions from facial expressions of the subjects. In: Proceedings of the 2009 international joint conference on Neural Networks, pp. 3421–3428 (2009)
26. Hisagi, M., Saitoh, T., Konishi, R., Analysis of efficient feature for Japanese vowel recognition. In: IEEE sponsored International Symposium on Intelligent Signal Processing and Communications (ISPACS '06) (2006)
27. Gocke, R., Asthana, A.: A Comparative Study of 2D and 3D Lip Tracking Methods for AV ASR, Auditory-Visual Speech processing (AVSP). Moreton Island, Australia, pp. 235–240 (2008)
28. Revret, L., Benoot, C.: A new 3D lip model for analysis and synthesis of lip motion in speech production. In: Auditory-visual Speech Processing Workshop. Terrigal, Australia. pp. 207–212 (1998)
29. Kuratate, T., Hsu, K., Riley, M.: Creating speaker specific 3D lip models using 3D range data. Inform. Process. Soc. Jpn. **2004**(16), 19–24 (2004)
30. Yamada, H.: Visual information for categorizing facial expression of emotion. Appl. Cogn. Psychol. **7**, 252–270 (1993)
31. Fernandez-Dols, J.M., Wallbott, H., Sanchez, F.: Emotion category accessibility and the decoding of emotion from facial expression and context. J. Nonverbal Behav. **15**(2), 107–123 (1991)
32. Konar, A.: Computational Intelligence: principles techniques and applications. Springer, Heidelberg (2005)
33. Zimmermann, H.J: Fuzzy Set Theory and its Applications. Springer, Netherlands (2001)

Index

A
Adaptive fuzzy switching filter, 12
Adaptive median filter, 12
Affine multimodality, 187, 188
Affine transformation, 189, 192, 194, 196
Alpha-trimmed mean-based filter, 12
Ant colony, 211
Average distortion, 132–136, 142–145, 150

B
Bacterial foraging
optimization, 131, 133, 137, 139
Biogeography-based
optimization (BBO), 45
Bitrate, 232
Box plot, 34

C
C-means, 94
Center-weighted median filter, 4
Chemotaxis, 137, 138
CIELAB, 227, 229
Classifier, 259
Cluster, 94
Cluster center, 285
Clustering, 268
Color cast, 221
Computed tomography, 188
Computer vision, 71, 89
Content-based image retrieval, 201
Contrast enhancement, 21
Contrast limited adaptive histogram
equalization (CLAHE), 33
Correlation, 189, 193, 195
Crisp membership degree, 96
Crisp VQ, 94
Criterion, 98
Crossover, 287
Cumulative histograms, 25

D
Data density, 292
Decision tree, 161, 162
Depth of field, 229
Desaturation, 227, 228
Differential
evolution (DE), 286
Discontinuities, 27
Dispersal, 139
Distance metric, 157
Distortion, 94
Diversity, 208

E
Edge detecting
median filter, 12, 15
Edge detection, 224
Element-wise
maximization, 28
Elimination, 139
Emotion recognition, 282
Entropy, 29, 38, 161
Evolutionary, 203
Experiments
compute on demand, 102
distortion comparison, 100
literature comparison, 102
PSNR comparison, 101
utility analysis, 102
Exposure time, 229

A. Chatterjee and P. Siarry (eds.), *Computational Intelligence in Image Processing*,
DOI: 10.1007/978-3-642-30621-1, © Springer-Verlag Berlin Heidelberg 2013

F
False positive, 231
Feature selection, 269
Figures
  computational time, 102
  testing Images, 100
  transition from fuzzy to crisp mode, 97
  utility measures distributions, 102
Fitness, 30, 203
Fitness function, 287, 289
Flash, 218, 223, 229
Flash/no flash, 229
FLVQ, 95
Fractal dimension, 256
Fuzzy, 131, 133, 134, 136
Fuzzy c-means, 94, 95, 98, 174, 187, 188, 190, 192, 284
Fuzzy entropy, 37–40, 54
Fuzzy filter, 12
Fuzzy membership degree, 96
Fuzzy relation, 108, 111, 115, 117
Fuzzy relation equation, 174
Fuzzy set, 38, 40, 131, 135, 158
Fuzzy transform, 107–109, 111, 115, 117–120, 122–124, 129
Fuzzy VQ (FVQ), 94, 95

G
Gaussian, 228
Gaussian membership distribution, 159
Genetic algorithm, 187–190, 195, 211
Genetic programming, 71, 72
Gleason grading system, 254
Global histogram equalization, 25
Golden eyes, 219

H
Hierarchical feature matching, 161
Hierarchical template matching, 156
Histogram equalization, 24
HSL, 222
HSV color mode
HSV color model, 166
Hue-saturation-value (HSV), 24
Human brain, 187–190
Hyperplane, 290

I
Illuminant, 221
Image compression, 93, 131, 132, 134, 149, 232
Image segmentation, 37, 38, 40, 42, 71, 72, 75, 77, 89
Image-layer, 108, 114
Impulse detector, 4, 13
Information gain, 162
Inpainting, 228, 233

J
JPEG, 232

K
K-means, 206
Kernel function, 291
Kiss curve, 283
Kurtosis, 158

L
LBG, 94
Learning process, 98
Leave-one-out cross validation, 292
Lens aperture, 229
Levenberg-Marquardt algorithm, 10
Likelihood function, 270
Linear classifier, 290
Lip-contour, 282, 283
Lipschitz condition, 108
Lossless compression, 93
Lossy compression, 93, 84
Lukasiewicz t-norm, 174, 179

M
Mean, 157
Mean filters, 5
  alpha trimmed, 12
  minimum-maximum exclusive, 15
  signal-dependent rank-ordered, 12
Mean squared error criterion, 12
Median filters
  adaptive, 12
  center-weighted, 4
  edge detecting, 12, 15
  multistate, 12
  progressive switching, 12
  standard, 4, 12, 15
  switching, 12
  tri-state, 12
  weighted, 4
Membership, 158
Membership degrees, 97
Membership function, 133, 135

Migration, 98–100
Minimum-maximum exclusive mean filter, 15
Morphological operators, 223, 229
Multistate median filter, 12
Mutation, 287
Mutual information, 189, 192, 195

N
Noise, 229, 232

O
Object recognition, 71, 75, 89
Objective function, 93, 96

P
Pairing verification, 224
Particle-swarm optimization (PSO), 30, 204
Performance evaluation metric, 289
Postprocessing, 219
Preflash, 218
Progressive switching median filter, 12
Prostate carcinoma, 254
PSNR, 100, 108, 113, 119–122, 124, 131, 142–148, 174, 175, 182, 183

Q
Query-by-example, 208

R
Red eyes, 218
Redness map, 222
Region of interest, 187–190, 224
Registration, 187, 189, 190, 194
Relevance feedback, 208
Remote sensing, 265
Resolution, 224
RGB, 221

S
Sector-based equalization, 25
Segmentation, 190, 232
Selection, 287
Self organizing map, 206
Semantic gap, 202
Signal-dependent rank-ordered mean filter, 12
Similarity metric, 187–189, 193, 194
Skin detection, 221, 223
Standard deviation, 158
Standard median filter, 4, 12, 15
Statistical separability measures, 270
Support vector machine (SVM), 207, 289
Support vectors, 290
Swarming, 138
Switching median filter, 12

T
Tables
  distortion mean values, 100
  literature comparison results, 102
  PSNR mean values, 101
Template, 157, 224
Template matching, 156
Termination criteria, 31
Texture feature, 255
Theta, 97
Thresholding, 38, 42
Tristate median filter, 12
Type-2 fuzzy logic systems
  applications in image processing, 5
  as a noise detector, 13
  as a noise filter, 11
  definition, 3
  sugeno type, 6

U
Uniform histogram, 24
Utility, 93, 94, 98

V
Vector quantization (VQ), 93–95, 131–134, 142
Vigenère algorithm, 174, 177–179, 185

W
Watermarking, 174, 177–179, 183, 185
Weighted median filter, 4
Weighted-sum scheme, 25

Y
YCC, 222