UNIVERSITAT DE LES ILLES BALEARS
Departament de Ciències Matemàtiques i Informàtica

Analytical methods for the study of color in digital images

Ana Belén Petro Balaguer

Thesis supervisors: Bartomeu Coll Vicens,
Jose Luis Lisani Roca,
Jean-Michel Morel.

Dissertation submitted in partial fulfillment
of the requirements for the degree of

DOCTORA EN MATEMÀTIQUES

Palma de Mallorca, Febrer 2006

A en Mateu,
per demostrar-me que l'esforç
i l'amor ho poden tot.

Agraïments

Com comentava l'altre dia amb un amic meu, crec que aquesta part de la tesi és més important que qualsevol dels capítols que hi ha escrits, perquè sense tota aquesta gent aquesta tesi no hagués pogut ser una realitat. Esper no deixar-me ningú, però si ho faig que sàpiga que ell/a també es troba dins el meu cor.

Primero de todo, querría agradecer el espléndido trabajo de mis TRES directores de tesis. No es cierto eso de que tres son multitud. Para mi cada uno de ellos ha sido necesario e imprescindible y el buen entendimiento entre ellos me ha hecho poder enriquecerme de los tres. Gracias Jean-Michel por darme la oportunidad de trabajar a tu lado y poner a mi disposición tu experiencia y tus grandes conocimientos. Gracias por enseñarme lo importante que es hacer el trabajo bien hecho. Gràcies Tomeu per convidar-me a entrar al fantàstic món de la recerca i per entusiasmar-me amb les Matemàtiques Aplicades. Gràcies per creure en mi en tot moment i per ensenyar-me que val la pena anar pel "bon camí". A vegades dius que l'important a l'Universitat és tenir bons "padrins", jo crec que he tingut la sort de tenir-ne a un dels millors. Gràcies Jose Luis per tot el temps i l'esforç que has posat en la meva tasca. Gràcies per fer-me avançar dia a dia amb la teva atenció i les teves exigències, per fer-me aprendre que el "boli vermell" ha estat i serà pel meu propi bé.

En segon lloc, no puc deixar d'agrair l'acollida i el calor humà del Departament de Ciències Matemàtiques i Informàtica de la Universitat de les Illes Balears, la meva segona llar. Des que vaig arribar-hi he descobert l'important que és poder fer feina a un lloc on t'hi sents a gust. Gràcies Rut per tots aquests anys "una davant l'altra", compartint els moments bons i els dolents. Mai no t'oblidis que l'esforç d'aquests anys ha valgut la pena, almenys per compartir despatx juntes. Gràcies als meus companys/es de dinars i de parloteos. Gràcies Chus, Antonio, Guillermo, Mercè, Toni, Julian, Jose Maria, Esperança, Biel,... Gràcies per recordar-me amb cada rialla i amb cada conversa el molt que estimam la nostra tasca, gràcies per fer-ho tot més bo de dur. Gràcies a tots/es aquells/es que m'han recolzat en la meva tasca aquests anys i mitjançant la seva experiència m'han encoratjat a seguir endavant.

Merci a mes amis de l'École Normale Supérieure de Cachan. Merci Julie pour toute le travaille ensemble, pour me régaler ton idées et ton connaissement, pour faire l'effort de comprendre mon horrible français. Merci Agnès pour proportionner ton expérience a notre travaille. Merci a les doctorants de la salle de doctorants, pour me faire sentir accueilli loin de chez moi. Gracias especialmente a Diego, por llenar de buen humor y amistad mis estancias en Paris.

Gracias a Fiona Mettini por dejarme utilizar la fotografía de la mariquita. Esta fotografía ha inspirado gran parte de este trabajo, por lo que sin ella no podría haber avanzado tanto.

I qué dir-ne de tots els/les meus/ves amics/gues. Tots/es aquells/es que durant aquests anys han estat al meu costat recolzant-me i donant-me suport amb les seves preguntes, interessant-se per saber com anava el meu treball. Aquells/es que en els darrers mesos m'han regalat un gran somriure en saber que arribava a l'etapa final. Gràcies als de la Delgació, a na Marga i a na Silvia, a les nines de Mates, als amics de la carrera, a n'Imma i na Cris, a les d'AMA, als de Balèria,... Gràcies perquè sense vosaltres tot s'hagués fet més costa amunt.

En darrer lloc, no puc oblidar la meva família, la qual ha soportat i ha patit la meva tesi tots aquests anys. Gràcies per ésser-hi, per preocupar-vos-ne i per fer-me sentir que el meu treball és genial i important, encara que no hagi tingut temps d'explicar-vos qué faig. Gràcies Mateu per la teva paciència, per posar la meva tesi per damunt de moltes coses i per creure en mi en tot moment. Gràcies per omplir tot el meu treball d'amor i d'optimisme.

Contents

Introduction **1**
1 Color image description problem . . . 1
2 Plan of the thesis . . . 7

1 Color: description and representation **11**
1.1 Introduction . . . 11
1.2 How do we see color? . . . 12
1.2.1 Human vision . . . 12
1.3 Creating colors . . . 18
1.3.1 Historical notes . . . 19
1.4 Characteristics of color perception . . . 20
1.5 Color spaces . . . 23
1.5.1 RGB Color Space . . . 24
1.5.2 HSI Color space . . . 25
1.5.3 Two-dimensional spaces . . . 27
1.5.4 CIE Color Spaces . . . 28
1.6 Colors in real life . . . 31

2 Histogram segmentation **35**
2.1 Introduction . . . 35
2.2 An a contrario approach to histogram analysis . . . 37
2.2.1 Meaningful intervals and meaningful gaps of a histogram . . . 37
2.2.2 Maximal meaningful intervals and gaps . . . 39
2.2.3 Limits of the meaningful gap and meaningful interval notions . . . 40
2.2.4 Direct segmentation using a generalized entropy functional . . . 42
2.2.5 Optimal separators . . . 42
2.3 Histogram segmentation as density estimate . . . 44
2.3.1 Adequacy to a given distribution . . . 45
2.3.2 Testing the monotone hypothesis . . . 46
2.3.3 Piecewise unimodal segmentation of a histogram . . . 48
2.4 Properties and comments . . . 51
2.4.1 Detection properties . . . 51
2.4.2 Flat zones . . . 52
2.4.3 Trinomial . . . 53

3 Experiments and applications of the FTC algorithm 55
3.1 Some results on synthetic data . . . 56
3.2 On Gaussian mixtures . . . 57
3.2.1 Goodness-of-fit test . . . 57
3.2.2 Adequacy of the Gaussian mixture model . . . 60
3.3 Some experiments on grey level images segmentation . . . 63
3.4 Some experiments in document image analysis . . . 67
3.5 Sensibility of the method . . . 71
3.6 Some experiments on camera stabilization . . . 78

4 Color palette 83
4.1 Introduction . . . 83
4.2 Computer graphics color palette . . . 84
4.3 Color palette construction . . . 86
4.3.1 Quantization problems . . . 87
4.3.2 Algorithm description . . . 89
4.4 Experimental results . . . 90
4.5 Color names . . . 97
4.5.1 Experiments on naming colors . . . 98
4.6 ACoPa algorithm and Computer Graphics color palettes . . . 102
4.7 ACoPa Improvements . . . 102

5 CProcess: a software tool for the study of color 109
5.1 Introduction . . . 109
5.2 Installing CProcess . . . 109
5.3 Getting started . . . 111
5.4 Menu Options . . . 115
5.4.1 File menu . . . 115
5.4.2 Edit menu . . . 117
5.4.3 View menu . . . 118
5.4.4 Processing menu . . . 119
5.4.5 Color Spaces menu . . . 119
5.4.6 Color Quantization menu . . . 121
5.4.7 Palette Menu . . . 122
5.4.8 Operations menu . . . 123

6 Conclusions and future work 125

A EM Algorithm 127
A.1 Introduction . . . 127
A.2 Gaussian Mixtures . . . 128
A.3 The EM Algorithm . . . 128
A.4 Limitations of EM . . . 132
A.5 Goodness-of-fit Tests . . . 132
A.5.1 Kolmogorov-Smirnov test . . . 133
A.5.2 Chi-Square Test . . . 133
A.5.3 Cramer-Von Mises Test . . . 134

B Some notes on Gestalt Theory 135

C A brief review of binarization techniques **139**

Bibliography **150**

List of Figures

1 Final aim . . . 2
2 Channels in RGB cube and HSI spaces. . . . 3
3 Color clouds . . . 4
4 Histogram segmentation algorithm . . . 6
5 Color palette . . . 7

1.1 Distribution of rods and cones . . . 13
1.2 Absorption spectra of the cones . . . 14
1.3 Metameric colors . . . 15
1.4 Post-image phenomenon . . . 16
1.5 Simultaneous contrast effect . . . 16
1.6 Opponent process . . . 17
1.7 Opponent process curves . . . 17
1.8 Additive and substractive mixture . . . 19
1.9 Color constancy phenomenon . . . 21
1.10 Simultaneous contrast phenomenon . . . 22
1.11 Simultaneous contrast phenomenon . . . 22
1.12 Color-vision deficiency . . . 23
1.13 RGB color model . . . 24
1.14 HSI color model . . . 26
1.15 Maxwell triangle . . . 28
1.16 Maxwell triangle . . . 28
1.17 CIE chromatic diagram . . . 29
1.18 Red, green and blue histograms from the "Very Large Cube" . . . 32
1.19 Hue histogram from the "Very Large Cube" . . . 32
1.20 Fixed intensity planes of the "Very Large Cube" . . . 33
1.21 Removing colors in the "Very Large Cube" . . . 34

2.1 Maximal meaningful mode ang gap . . . 41
2.2 RG and RM comparison . . . 43
2.3 Optimal separators . . . 44
2.4 Rejections of the uniform law . . . 46
2.5 Grenander estimator . . . 48
2.6 FTC algorithm . . . 50
2.7 Density estimation . . . 51
2.8 Flat zones . . . 52

3.1 Segmentation by FTC of Gaussian mixtures . . . 56

3.2 Mixture of three Gaussians . . . 59
3.3 Mixture of four Gaussians . . . 59
3.4 Segmentation of grey image with FTC algorithm . . . 61
3.5 Representation of intensity histogram by Gaussian laws . . . 62
3.6 Representation of intensity histogram by Gaussian laws . . . 63
3.7 Representation of hue histogram by Gaussian laws . . . 64
3.8 Segmentation of grey image with FTC algorithm . . . 65
3.9 Grey-level histogram segmentation . . . 66
3.10 Segmentation of document images . . . 68
3.11 Segmentation of document images . . . 69
3.12 Illustration of Property (P1) . . . 70
3.13 Segmentation of document image . . . 71
3.14 Segmentation of intensity and hue histograms . . . 72
3.15 Impulse noise . . . 73
3.16 Gaussian noise . . . 74
3.17 Sensibility of the method to quantization. . . . 75
3.18 Inhomogeneous illumination . . . 76
3.19 Effects of blurring . . . 77
3.20 Orientation histogram example . . . 78
3.21 Orientation histogram example . . . 79
3.22 Quantization effects over the histogram of orientation . . . 80
3.23 FTC application to camera stabilization problem. . . . 80
3.24 FTC application to camera stabilization problem. . . . 81

4.1 Noisy peaks in the hue histogram . . . 88
4.2 Quantization problem . . . 88
4.3 Grey cylinder elimination . . . 89
4.4 Complete example of ACoPa process (I) . . . 92
4.5 Complete example of ACoPa process (II) . . . 93
4.6 Example of the ACoPa algorithm . . . 94
4.7 HSI and the cylindrical representation of $CIELab$ space comparison . . . 95
4.8 HSI and the cylindrical representation of $CIELab$ space comparison . . . 96
4.9 NBS/ISCC system modifiers . . . 97
4.10 List of color names . . . 99
4.11 X11 and NBS/ISCC dictionaries comparison . . . 100
4.12 X11 and NBS/ISCC dictionaries comparison . . . 101
4.13 Color palettes comparison . . . 104
4.14 Color palettes comparison . . . 105
4.15 Color palettes comparison . . . 106
4.16 Over-segmentation regions . . . 107
4.17 Parameter too high in erosion . . . 107
4.18 Evolution of the erosion operation . . . 108
4.19 Example of erosion . . . 108

5.1 Initial window of CProcess . . . 111
5.2 Image displayed on a CProcess window . . . 112
5.3 3D visualization on RGB space . . . 112
5.4 PCA Visualization . . . 113
5.5 3D histogram and Erase operation . . . 113

5.6 Hue-Saturation and Hue Histograms 114
5.7 Partial selection and Cut&Copy results 114
5.8 Result of the Change to 6 quantization 115
5.9 Partial results of the FTC HSI option 116
5.10 Resulting palettes of the FTC HSI option 116

B.1 Gestalt principles 136
B.2 Gestalt conflicts 136

Introduction

Even though color is of paramount importance in vision, image processing has progressed in grey levels. However, with the advances in computer world and the proliferation of image collection devices, such as digital cameras or scanners, this reality is changing. In the last three decades, color image processing and analysis has been, and now is, a very active field of research. Nowadays, the amount of color image processing and analysis methods is growing quickly. A large number of algorithms can be classified in this field, which give solution to problems such as color image segmentation [87], compression [82], enhancement [76], indexing [85],...

Some of these techniques, such as color image segmentation, are natural operations for the human visual system ([51]). This complex system is a perfect interconnection network between the environment and our brain. A basic task of this system is image description. Given an image, our visual system is able to detect the meaningful objects in the image and to describe the image by means of these elements. In the color case, color names are associated to these meaningful objects improving the description of the image.

Computationally speaking, the automatic description of a color image by means of its color names is a difficult task. It implies answering a lot of questions: what is color?, how can we distinguish between similar colors?, what is the best representation of color information?, which is the best way of naming colors?, why some colors represent better an image than others?,... In this thesis we try to give an answer to all these questions. The final goal is, given a color image, to obtain the list of meaningful colors which allow the image description. Figure 1 illustrates this goal.

1 Color image description problem

Attaining this aim supposes a long way. As we have mentioned, different questions about color, color perception, color representation,... have to be answered to face the problem computationally.

First, we need a deep knowlegde of color. How can we "look for" a property of an image when we do not have enough knowledge about it? Therefore we must study aspects of color such as color creation, color perception,... In short, an exhaustive analysis of color from different points of views. An important point in this theorethical study is the color representation analysis. A wide variety of color spaces exists in literature ([37]). Our analysis has to allow us the selection of the best color codification to reach our goal. In this sense, only three tridimensional color spaces are analysed, RGB, HSI and $CIELab$.

The RGB space is indispensable since the starting point in the proposed work is a digital color image ([60]). Digital color images consist of little squares or pixels, each one of them associated to three values, which are interpreted as color coordinates in some color

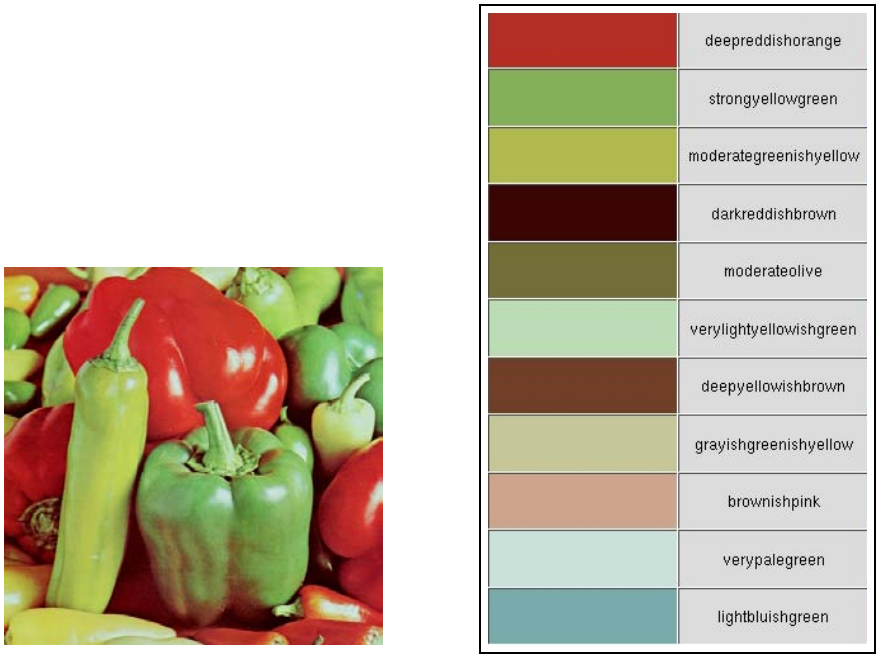


Figure 1: *The original image "Peppers" and the list of names of meaningful colors in the image, the final goal of this thesis.*

space. The RGB color space is commonly used in computer displays then, we assume that the initial data are provided in RGB color space. In order to be able to describe the image we need to use a color space closer to the human perception than RGB. We need a color space which provides the color information clearly. One of the drawbacks of the RGB space is that the color information is provided by the mixture of the three components. If we consider only the red channel, for example, we cannot distinguish one color from another (see the first row in Figure 2). From these premises we decided to use the HSI (hue, saturation and intensity) color space. Each variable in the HSI color space provides a basic information about the colors, which can be interpreted separately of the other variables. In the second row of Figure 2, a representation of the HSI variables of Figure "Peppers" is shown. We also need a color space to correctly measure the distance between colors. The RGB and HSI spaces are not appropriate for this task for this reason we consider the $CIELab$ space, the usest uniform space, to perform this operation.

The previous stage provides us with a theoretical knowledge of color. The next stage is to acquire a "practical" knowledge of color. We want to answer questions such as: what shape has the distribution of image colors (or "clouds") in the different color spaces?, what transformations can we apply to the image colors?, what components provide more information about the colors in an image?, what colors are more common in the real word?,... Therefore, we need to develop a useful color-image editing and processing tool to analyse and to understand the basis of color. The main contributions of this development are the representation of the color pixels in an image by means of different color spaces and of histograms of different color space components.

The analysis of the color clouds in the RGB cube provides us with an interessant information. In Figure 3 we can observe an example of this fact. The color clouds are the basical information about the colors in the image. In the example we can see a kernel of colors and different "branchs" corresponding to the different colors in the image. We have observed that, normally, the color clouds present these characteristic shapes, "branchs"

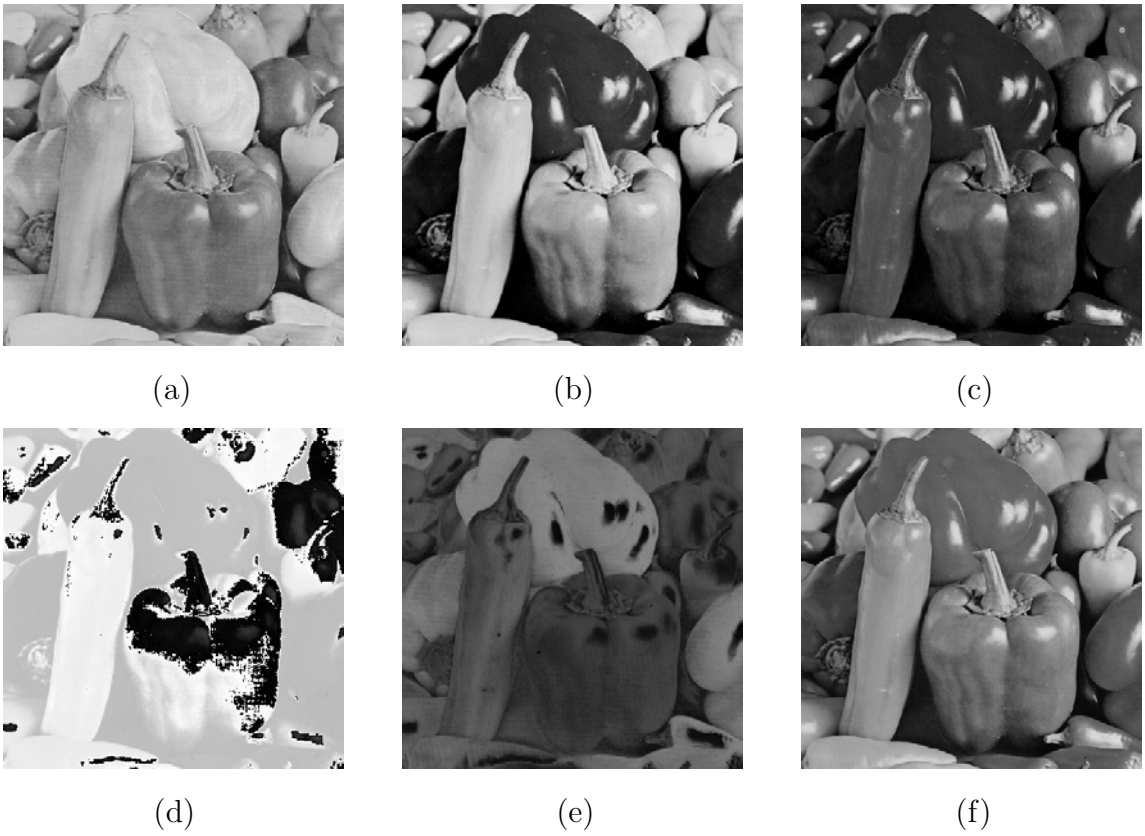
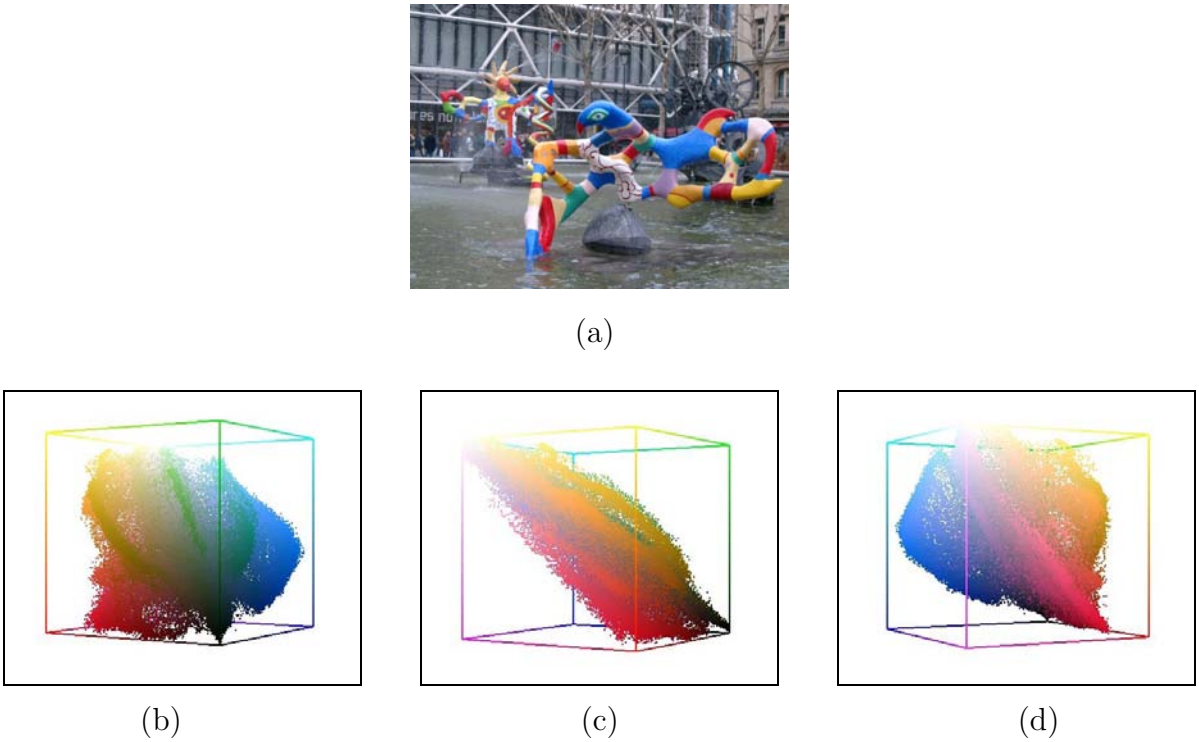


Figure 2: *Channels in RGB cube and HSI spaces. (a) R component, (b) G component, (c) B component, (d) H component, (e) S component, (f) I component. Each image of the second row provides more information about the colors individually than the ones of the first row. For example, by seeing the image of red components we cannot say that the two main peppers in front are of the same color. On the other hand, in the image of hue components this fact is clear (Taking into account that the hue component is circular and then the black values are similar to the white values)*



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

Figure 3: *Example of color clouds: (a) Original image "Stravinski", (b)–(d) Three point of view of the color clouds of the image pixels in the RGB cube.*

of pixels corresponding to a same color whose initial point is the black color and the final point is the white color. These "branchs" have a fish-like shape. The "fishes" provide a lot of visual information about color, but their extraction from the color clouds is a difficult task. The different "fishes" hide each other, the central "fishes" are hidden behind the external ones. Moreover an important information is not provided by the "fishes": the amount of pixels corresponding to each color.

Due to these drawbacks, we need a tool, different from color clouds, which provides the basic color information of the "fishes", but which reports the quantity information, too. It is worth noting the need of a good color space in which this tool can be used. It is clear that when using the HSI space we do not lose the "fishes" information. Quite the contrary, the hue component of the HSI space allows us to separate the different "fishes".

Quantitative information about the color contents of an image is obtained with the help of histograms. Histograms provide us the quantitative information clear and visually, they indicate the amount of pixels that are associated with a given color component. It is worth noting that we want to obtain the list of meaningful colors by considering only the color information of the image, we will not use spatial information. As we have mentioned, each component in the HSI space provides individually a basic information about color. For this reason 1D histograms of hue, saturation and intensity are used to analyse the image colors. Moreover, the computational processing cost is reduced with respect to the use of 3D histograms.

In the histograms, the information is given by their modes, that is, the intervals where data concentrate. The meaningful colors will be obtained from the meaningful modes in the hue, saturation and intensity histograms. Therefore, we need a histogram segmentation algorithm, which gives these meaningful modes. But, what means "meaningful"? What is a "meaningful" color? What is a "meaningful" mode? We base the proposed algorithm on the quantitative study of the Gestalt theory. From this study, proposed

first by A. Desolneux, L. Moisan and J.M. Morel in 2000 ([32], [33]), we can define the meaningfulness of different geometric objects. In our case, we develop a theory about meaningful intervals in a histogram. In a first reasoning, we define meaningful intervals and gaps and the segmentation is achieved by detection against an *a contrario* law. A second proposition defines meaningful rejections and the algorithm confirms the hypothesis of an *a contrario* law. The proposed histogram segmentation algorithm is coherent with our goal of finding the more representative colors in a image even if some of them correspond to small details. Other segmentation methods in literature ([35, 29]) are not able to detect the little peaks in the histogram which may correspond to these details. A clear example is represented in Figure 4. Reaching our goal supposes to consider the red color in the final list for this image. We observe in the hue histogram (Figure 4(b)) that the ladybird represents a little mode on the right. It is clear that this little mode has to be detected as a meaningful mode. The proposed algorithm segments the histogram into three modes, one corresponding to the red mode. Classical histogram segmentation methods, such as Gaussian Mixture estimation with EM algorithm, are not able to detect the small peak as a separate mode if we want to consider only three modes (see Figure 4(d)-(e)). On the other hand, the intensity histogram (Figure 4(d)) presents a lot of little peaks that do not correspond to meaningful modes in the histogram. The proposed algorithm segments it into six modes only. Other approaches which detect the local maxima without estimating the underlying density, such as [20], would detect too many peaks in this histogram. The new approach is a powerful tool in the sense that avoids under and over-segmentation of the histograms.

Once we dispose of a histogram segmentation algorithm and three components that provide us with the color information we are ready to face the last stage of the research project. In this last stage we have to develop the algorithm that obtains the color palette of a given image, which, similarly to the color palette of a painter, defines the colors "used" for "painting" the given image. The proposed approach considers a hierarchical ordering of the three color components, hue, saturation and intensity. Thus, the color palette algorithm has three different steps. Each step has as input the results in the previous one. Therefore, we obtain a hierarchical color palette, where the amount of color and the accuracy in the image representation grows at each step. In Figure 5 we can observe the hierarchical palette of the initial example. The final list, presented in Figure 1, is the result of associating each color in the palette to a dictionary color name.

The proposed algorithm can be considered as a segmentation algorithm, since sa egmentation is the partition of a digital image into multiple regions (sets of pixels), according to some criterion ([90]). In our case, the criterion is the meaningful colors in the image. Most of the segmentation algorithm make use of spatial information (region-based [17], edge-based [83] and hybrid [99] techniques). The proposed method would be included in the pixel-based techniques since only the color information is considered to face the segmentation. But, the goal of this kind of methods is to detect semantic visual objects in the image (see [16] for details). The goal of the proposed method is very different, in fact, in most of the resulting images the semantic visual objects are not associated to a unique color. Due to this fact, the comparison with existent segmentation methods is very difficult. Since the goals are different, the results will be different and incomparable.

On the other hand, the presented method can be considered as a quantization method, since quantization is the process of approximating a very large set of values by a relatively-small set of discrete symbols or integer values ([79]). But, in this case again, the goal of the proposed approach is very different from the goal of classical methods. In these methods the goal is to obtain a pleasant visual representation with a reduced number of

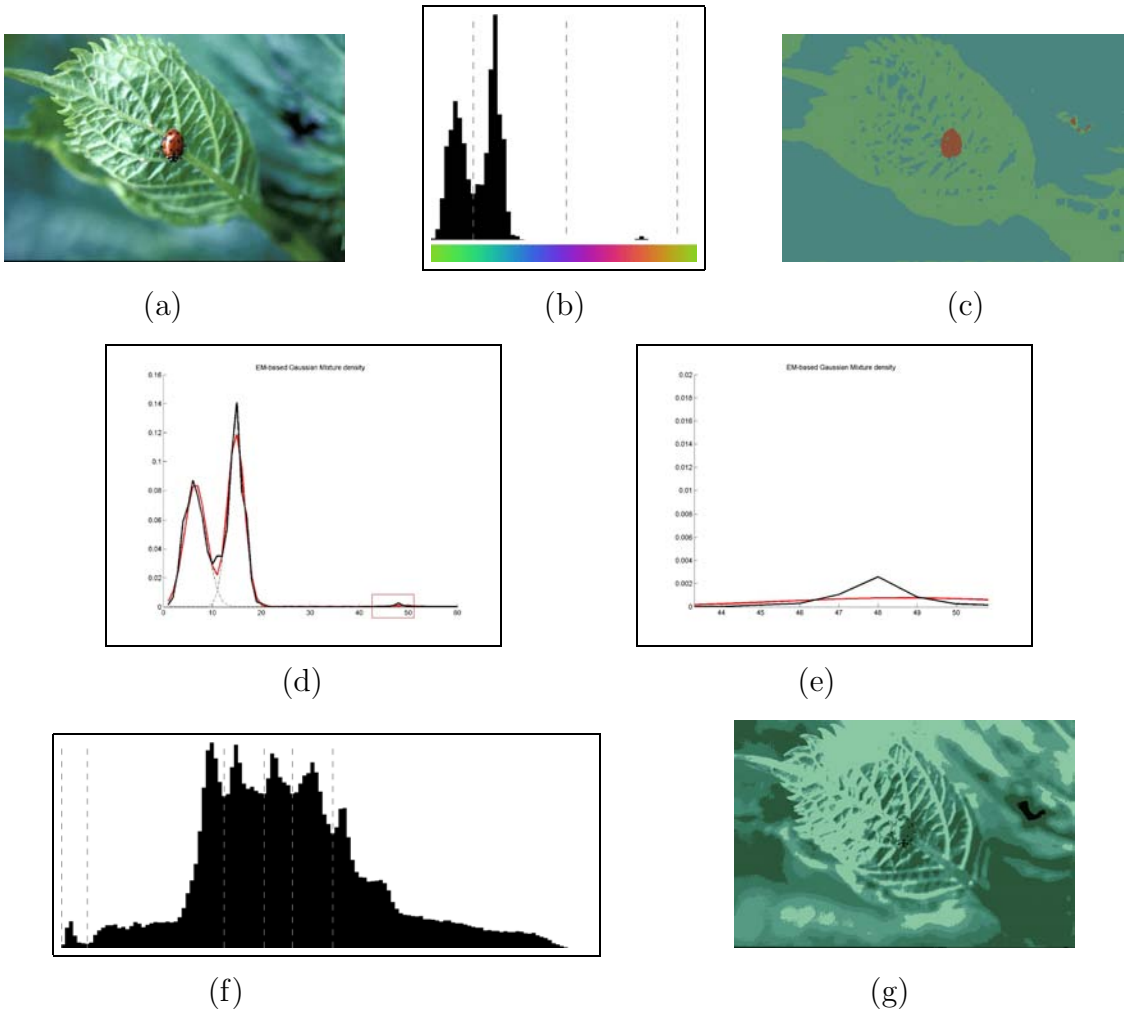


Figure 4: *Two examples of the histogram segmentation algorithm. (a) Original image "ladybird". (b) Hue histogram segmented with the proposed algorithm. We obtain three different modes, separated by dashed lines. (c) Segmented image, associating to each mode resulting in the hue histogram segmentation a different color. (d) Resulting approximation (red line) to the hue histogram with three components in a Gaussian Mixture estimation with EM algorithm, (e) Zoom of the red rectangle in image (d). The mode corresponding to the red hue is not well estimated. (f) Intensity histogram segmented with the proposed algorithm. We obtain six different modes, separated by dashed lines. (g) Segmented image, associating to each mode resulting in the intensity histogram segmentation a different color.*

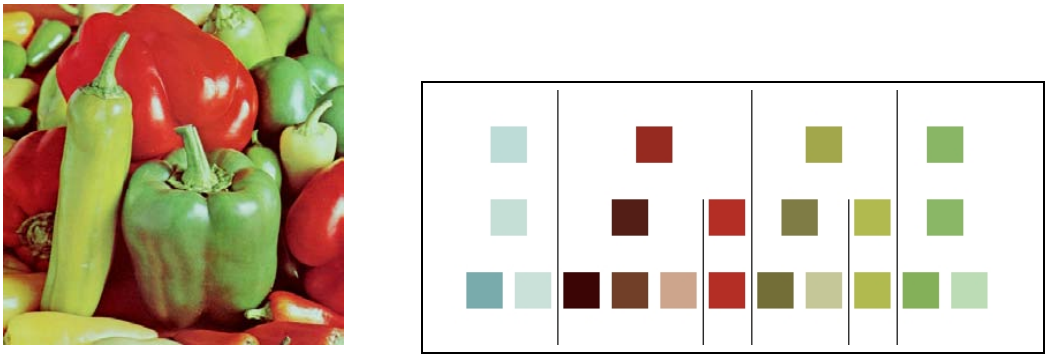


Figure 5: *Original image "Peppers" and the hierarchical palette. Each row in the palette corresponds to one of the steps in the process. At each step, the amount of colors in the palette grows.*

colors. Normally, in these methods, the amount of colors has to be selected by the user. In the proposed method, the amount of colors is computed automatically and the goal is the qualitative description of the image. The proposed algorithm is compared to two known quantization methods ([48]) for the obtention of computer graphics color palettes. We can conclude that the proposed approach does not obtain a visual representation as pleasant as the computer graphics color palettes, but the results are more accurate, allowing the representation of the little details in the image.

2 Plan of the thesis

This thesis is developed in a pedagogical way. The different stages to obtain the proposed goal are explained with detail in the different chapters.

Chapter 1 is a short introduction to the color world. The first sections of the chapter are devoted to the physical description of colors and the psychophysiological mechanisms in the human visual system involved in the perception of color. The two main mechanisms that encode the color information from the colored object to the brain together with the main characteristics of color perception are described in this chapter. Methods for creating colors from mixtures of lights or inks are also reviewed. Next, different color representation mechanisms are analysed. Three tridimensional spaces are presented. These color spaces will be the tools used in the next chapters for obtaining a description of the color images. Finally, a new research project aiming at the analysis of real life colors is briefly presented. The preliminary results of this study indicate that the more abundant colors in real life are reds, oranges and yellows.

Histograms permit a quantitative study of the color information contained in a digital image. Colors are represented in some of the 3D color spaces described in Chapter 1 and the analysis of the 1D histograms associated to each one of the space components allow us to obtain the main features characterizing the color image. A basic description of the 1D histograms is given by the list of its *modes*, *i.e.* the intervals of values around which data concentrate. In Chapter 2 we propose a new approach to segment a 1D-histogram without any *a priori* assumption about the underlying density function. The theorical justification of the approach is presented in two steps, considering two different points of view. The method is based on the definition of meaningful events, modes and gaps in a first approach, and of meaningful rejections in a second approach. The final result is a fast and parameter free algorithm, the Fine to Coarse (FTC) algorithm, which avoids

under and over-segmentation. Some properties of the method are remarked at the end of the chapter.

In Chapter 3 some results and applications of the FTC algorihtm are studied. Firstly, the FTC results are checked on synthetic data, showing that the FTC algorithm can detect the modes of a mixture of unimodal laws without any *a priori* parametric model. In the next section, the meaningful rejections, introduced in the previous chapter, are used as a goodness-of-fit test and compared to other tests of this kind. In this same section, the adequacy of Gaussian mixtures for fitting the intensity and hue histograms is questioned. In the following sections are analysed two applications of the FTC algorithm: grey level image segmentation and text documents segmentation. For both applications, some results are shown, trying to improve the existent algorithms in literature. Mainly in text documents analysis, the characteristics of the proposed algorithm (parameter-free and optimum conditions for small modes detection) allow us to obtain excellent results. The method is sensible to factors such as noise and illumination changes. These factors modify the histogram shape and the mode detections is more difficult. These factors and their influence on the FTC algorithm results are analysed. Finally, the application of the algorithm to the camera stabilization problem is studied. In this case, the studied histogram is the gradients orientation histogram and the modes detected by FTC indicate the rotation of the image with respect to the horizontal and vertical directions. The camera stabilization application, together with some of the theoretical results of the previous chapter, have been presented in [24]. Some of the experiments on the analysis of text documents and a theorical explanation represent the main part of the submitted publication [27].

Chapter 4 represents the final step in the search of a qualitative description of the image. In this chapter, a method for the automatic construction of the color palette of an image is described. This method, based on the FTC algorithm of the previous chapter and a hierarchical ordering of the components of the HSI color space, obtains automatically, in three steps, the minimum number of colors that describe an image. We call this minimal set "color palette". The experimental results seem to endorse the capacity of the method to obtain the most significant colors in the image, even if they belong to small details in the scene. The obtained palette can be combined with a dictionary of color names in order to provide a qualitative description of the image. In the chapter are studied different computer graphics color palettes found in literature, to which the proposed palette is compared. The results show that classical computer graphics palettes obtain a better visualization of the image, but they obviate the small details in the scene. To avoid some problems of over-segmentation in some images, at the end of the chapter, a modified version of the algorithm, that includes the use of morphological operators is presented. Some contents of this chapter and Chapter 2 have been presented in the VIIP [23], the IEEE ICIP05 [25] and the IbPRIA [26] conferences.

CProcess (which stands for Color Processing) is a useful color-image editing and processing software tool to analyse and to understand the basis of color. This tool has been created to cover the different needs that arise in the study of color. The first of these needs is to have a good representation of the colors in an image. This representation can be created by means of the different color spaces, or, by means of others tools such as histogram representations. In Chapter 5 we present a brief user's manual for the CProcess software. The different options are explained and some examples are shown.

Finally, the last chapter is devoted to develop some conclusions and the planning of the future research.

This thesis is completed with the inclusion of three appendices that provide a deeper

insight on some of the concepts commented in the previous chapters. The EM algorithm and some known goodness-of-fit tests are described in Appendix A. Appendix B is devoted to a brief presentation of the Gestalt theory concepts. And, finally, a brief reviewer of some binarization techniques for histogram segmentation is developed in Appendix C.

Chapter 1

Color: description and representation

Abstract. This chapter is a short introduction to the color world. The first sections of the chapter are devoted to the physical description of colors and the psychophysiological mechanisms in the human visual system involved in the perception of color. The two main mechanisms that encode the color information from the colored object to the brain together with the main characteristics of color perception are described in this chapter. Methods for creating colors from mixtures of lights or inks are also reviewed. Next, different color representation mechanisms are analysed. Three tridimensional spaces are presented. These color spaces will be the tools used in next chapters for obtaining a description of the color images. Finally, a new research project aiming at the analysis of real life colors is briefly presented. The preliminary results of this study indicate that the more abundant colors in real life are reds, oranges and yellows.

1.1 Introduction

The study of Color involves several branches of knowledge: physics, psychology, mathematics, art, biology, physiology,... Each one of these disciplines contributes with different informations, but the way towards the absolute understanding of color has just begun.

According to the *Collins* dictionary (2000): "Color is:

a. an attribute of things that results from the light they reflect, transmit, or emit in so far as this light causes a visual sensation that depends on its wavelength,

b. the aspect of visual perception by which an observer recognizes this attribute,

c. the quality of the light producing this aspect of visual perception."

This definition illustrates the complexity of the notion of color and roughly sketches the three factors on which color depends: light, physical objects and our visual system.

Color does not exist by itself; only colored objects exist ([90]). Three elements are necessary for the existence of color:

- A light source, to light the scene.
- The objects, which reflect, spread, absorb or diffract the light.
- A receptor, which captures the spectrum reflected by the object.

Color basically depends on these three elements, in such a way that, if one of them is not present, then we are not able to perceive color. On one hand, color is a physical attribute, due to its dependence on a light source and on physical characteristics of the objects; on the other hand, it is a psychophysical and physiological attribute, since it depends on our visual perception.

In the next sections, we will take a brief look at the human visual system and at the principal characteristics of color. In Section 1.5 different systems for the representation of color information are presented. Finally, the last section is devoted to present the "Very Large Cube", a starting project for the analysis of the existing colors in real life scenes.

1.2 How do we see color?

Briefly, vision is an active process depending both on the operations of the brain, performed thanks to eye information, and on the external, physical environment. The human eye and brain together translate light into color. But, how is this conversion done?

Light consists of a flux of particles called photons, which can be regarded as tiny electromagnetic waves ([96]). These waves must have a length between 380nm and 780nm to stimulate our visual system. The wavelength content of a light beam can be assessed by measuring how much light energy is contained in a series of small frequency intervals. The light can then be described by its spectral distribution of energy.

This spectral distribution characterizes each light source. For instance, the distribution of solar light is virtually flat since it has the same quantity of all visible wavelengths ([42]). The distribution of the light emitted by a light bulb or tungsten is more intense for the longer wavelengths than for the short ones. Our perception of color may change depending on the scene illumination, since the wavelengths reflected by the objects depend on the light source. Section 1.4 describes how our visual system adapts to changes of the light source.

The color of an object is defined and measured by its reflection spectrum ([96]). When light hits an object, the following three phenomena can happen: the light can be absorbed and the energy converted to heat, as when the sun warms something; it can pass through the object, as when the sun's rays hit water or glass; or it can be reflected, as in the case of a mirror or any light-colored object. Often two or all three of these phenomena occur simultaneously. Moreover, the same object can have different "colors", depending on the light source, on the geometry of the light devices, or on the change of some of its physical characteristics.

Finally, the receptor can be of different nature, for example, a photographic camera, a video camera, the eyes,... These last ones are the receptors in the human vision, they are the "gateway" for external information and this is the beginning of a large and complicated process which we explain in detail in the next subsections.

1.2.1 Human vision

In the human vision system, the information related with color perception is encoded at two fundamental levels ([69]). The first level occurs in the receptors which are located on the retina. It is a level related with the existence of three types of receptors called cones. Their responses are the starting point for the second level, which is based on the activity of three opponent mechanisms. We start by describing the first level, also called the *trichromatic level*.

In human vision, cones and rods in the eyes are the receptors of information from the external world. In particular, the rods are responsible for our ability to see in dim light and the cones are the color receptors. The number of rods and cones is different, and it differs depending on the part of the retina where they are located; rods generally outnumber cones by more than 10 to 1, except in the center of the retina, the fovea. In Figure 1.1 we can see a graphical representation of the distribution of rods and cones in the eye. The cones are concentrated in the so-called *fovea centralis*. Rods are absent there but dense elsewhere. We can note the absence of both rods and cones between -15 and -19 degrees of visual angle. This is the region of the blind spot where there are no cones or rods and the collected ganglion axons exit the retina as the optic nerve.

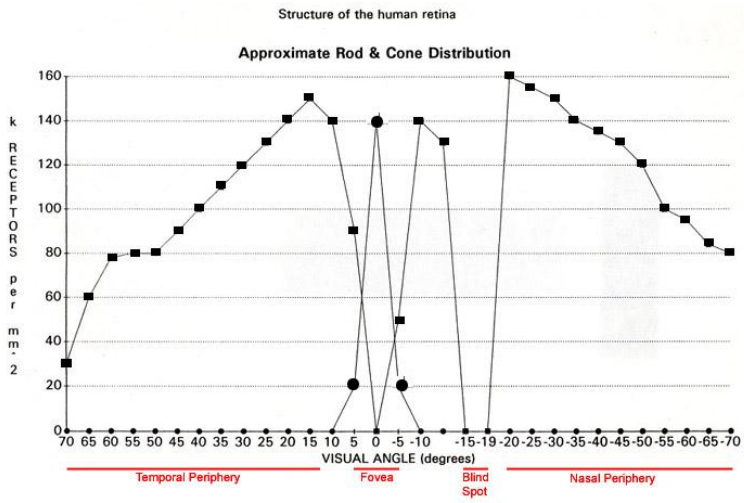


Figure 1.1: *Approximate distribution of rods and cones across the retina. Cone distribution is signified by the line with the circle markers and the rod distribution is signified by the square markers.*

Each rod or cone contains a pigment that absorbs some wavelengths better than others. Depending on the absorption peak of the pigment, we can distinguish three different types of cones: L, for the long wavelengths; M, for the medium wavelengths; and S, for the short ones (see Fig. 1.2). Due to the overlap between the curves, the wavelengths which affect one particular cone affect the other ones, too.

These cones are wrongly called blue, green and red cones, respectively, due to their absorption peaks: at 440 nm for cone S, at 545 nm for cone M and at 580 for cone L (see Fig. 1.2). These names are wrong because monochromatic lights, whose wavelengths are 440, 545 and 580 are not blue, green and red, but violet, blue-green and yellow-green. For this reason, if we were to stimulate cones of just one type, we would not see blue, green and red but, probably violet, green and yellowish-red instead.

We talk of *tristimulus values* to refer to the joint response of the three types of cones to an input light. It turns out that the human visual system cannot distinguish between two physically different color lights as long as they produce the same tristimulus values. Colors with different spectral power distribution but the same tristimulus values are known as metameric color stimuli ([19]) and they match in color for a given observer. Metamerism is what makes color encoding possible: there is no need to reproduce the exact spectrum of a stimulus. It is sufficient to produce a stimulus that is visually equivalent to the original one. This yields a principle for color reproduction, for instance, in televisions or printers (see Section 1.3).

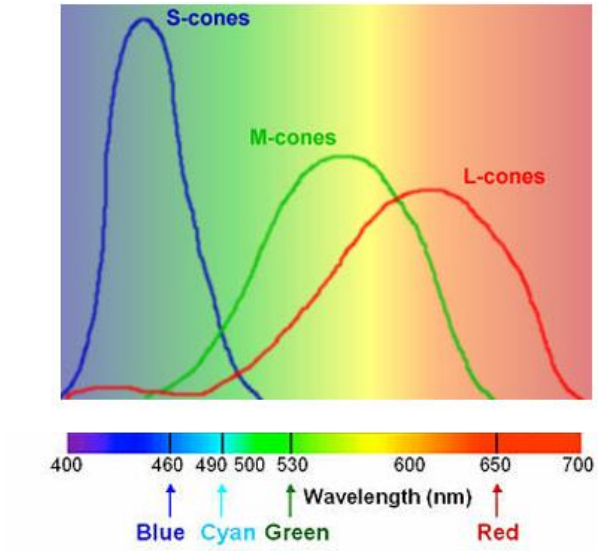


Figure 1.2: *Absorption spectra for the three classes of cones in the retina.*

The color equivalence phenomenon can be formulated in quantitative colometric terms. A stimulus $S(\lambda)$ is determined by the spectral distribution of radiation projected on the object (the light source) $I(\lambda)$ and the spectral reflectance characteristics of the object $R(\lambda)$. We can write ([67])

$$S(\lambda) = I(\lambda) \cdot R(\lambda).$$

Two metameric color stimuli whose different spectral radiant power distributions are denoted by $S_1(\lambda)$ and $S_2(\lambda)$ respectively satisfy the following equations:

$$\begin{aligned}
\int_\lambda S_1(\lambda)l(\lambda)d\lambda &= \int_\lambda S_2(\lambda)l(\lambda)d\lambda, \\
\int_\lambda S_1(\lambda)m(\lambda)d\lambda &= \int_\lambda S_2(\lambda)m(\lambda)d\lambda, \\
\int_\lambda S_1(\lambda)s(\lambda)d\lambda &= \int_\lambda S_2(\lambda)s(\lambda)d\lambda,
\end{aligned}$$

where $l(\lambda)$, $m(\lambda)$ and $s(\lambda)$ represent the spectral sensibility of cones L, M and S respectively. We can conclude that two metameric colors stimulate the long, medium and short wavelengths receptors in the same proportion, since the integral $\int_\lambda S(\lambda)x(\lambda)d\lambda$ represents the response of the cone of type x for stimulus $S(\lambda)$. An example can be seen in Figure 1.3.

There are two important points to consider concerning Figure 1.2. First, the fact that the spectral responses of the medium and long cones are very similar. They broadly overlap ([37]). Second, the range of the three spectra is very wide. It fills almost the whole visible spectrum. The absorption spectrum for the three types of cones explains the resulting color that we perceive in front of a determined mixture of colored light. For example, we perceive the yellow color when the medium and long cones have a high activation and the short cones have a small response (see [42] and [51] for more details). If we consider monochromatic light, we perceive yellow with a light of approximately 600nm. This explains why our perception of red is always non-luminous, because the

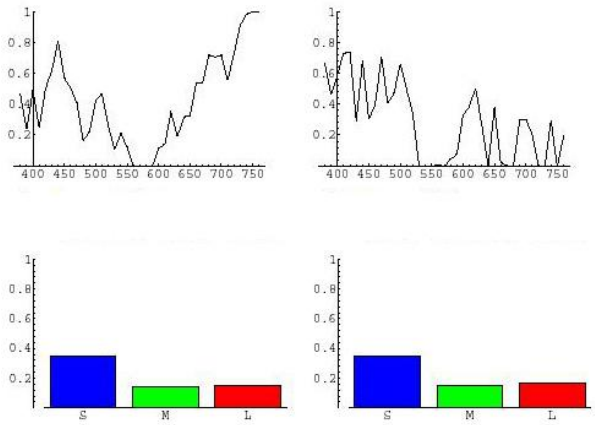


Figure 1.3: *Spectral distribution of two metameric stimuli, and the level response in the S, L and M cones.*

light of pure red is the light of 680nm and it activates the long cones in the right extreme of the spectrum, where the response is very small. On the contrary, we can perceive bright yellow or bright blue, since the monochromatic light of these colors falls over the peak of maximum activity. We want to emphasize that the sensation of "white" (the result of an approximately equal stimulation of the three cones) can be brought about in many different ways: by using broad-band light or by using a mixture of narrow-band lights, such as yellow and blue or red and blue-green.

Finally, we have to consider that the number of each type of cones in our retina is different. As a consequence, the eye response to blue light is much weaker than its response to red or green, because almost 2/3 of the cones process the longer light wavelengths (reds, oranges and yellows).

This theory about the three types of receptors in our retina does not explain some psychophysical observations, which can be understood by considering the existence of a second level of information processing provided by the cones. The psychophysical observations that support the existence of this second perception level are ([61]):

- There are certain color combinations that cannot exist in the phenomenological world. Blue and yellow cannot exist together in the same color (can we imagine a yellowish blue?) The same happens with red and green colors.

- *Post-image.* When we look at a red square for a long time and then we fix our eyes on a white area, we can see a post-image of the square, but its color is green. The same happens when we look at the sun directly, after what we see blue patches. We can observe this phenomenon by looking at Figure 1.4 for 45 seconds. Then, if we gaze at a white area, we will see the same image but the colors are changed (the red zone will be changed with the green zone, and the blue zone will be changed with the yellow zone).

- *Simultaneous contrast.* If a red background surrounds a grey material, the material seems green. In the same way a blue background converts the grey into a yellowish color. Figure 1.5 shows this phenomenon.

- *Color blindness.* As we will see later, red-blind people are green-blind too; and yellow-blind people are blue-blind too.

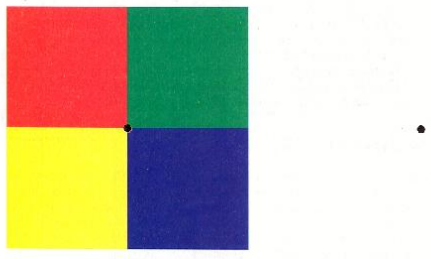


Figure 1.4: *Experiment of post-image phenomenon.*

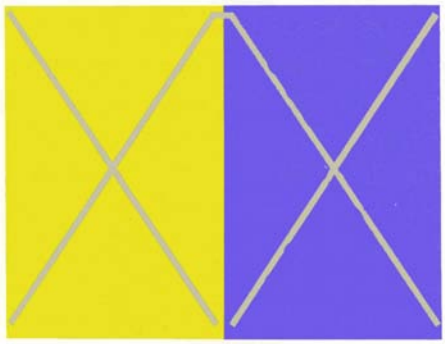


Figure 1.5: *Simultaneous contrast example. The diagonal bars over the blue background seems more yellowhish than that over the yellow background.*

All these observations have led to a theory called the *opponent process theory* which proposes that trichromatic signals from the cones feed into a subsequent neural stage and exhibit three major classes of information processing in the superior cells of the retina. Three different biochemical mechanisms occur at this level, which respond in an opponent way to different wavelengths ([68]). The first mechanism is red-green, which responds positively to red and negatively to green; the second one is yellow-blue, which responds positively to yellow and negatively to blue; and, the third one is white-black, which responds positively to white and negatively to black. Positive response means a biochemical process in the receptors that produces a chemical substance; whereas, negative response means the breakdown of this chemical molecule.

In Figure 1.6 we can see the connection between the trichromatic phase and the opponent process phase. There, we can observe how the S, M and L cones are connected to the bipolar cells to produce the opponent responses. The blue-yellow bipolar cells are excited by the medium and long cones and inhibited by the short cones. The red-green cells are excited by the long and short cones and inhibited by the medium cones.

The opponent response mechanism can be studied with the help of the so-called opponent process curves, which were first used by Hurvich and Jameson ([53]). The first step in their experimental work was to identify the shortest wavelength that represents the pure blue. Then yellow light was added to that wavelength until the blue hue was canceled out. The same operation was repeated with other wavelengths. At about 500nm

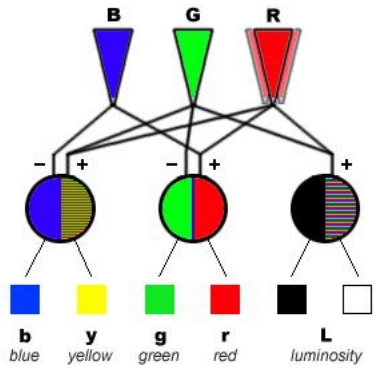


Figure 1.6: *Diagram of the connection between the three cones and the opponent cells.*

the light was neither blue nor yellow. From this wavelength, more blue light was added for canceling out yellow light. The results of this experiment were represented with a graph where the wavelength values were placed in an axis and the amount of blue or yellow in the other axis. The same experiment was done with red and green light. The resulting curves, shown in Figure 1.7, have three peaks: two for red (on the right and on the left) and one for green (in the middle).

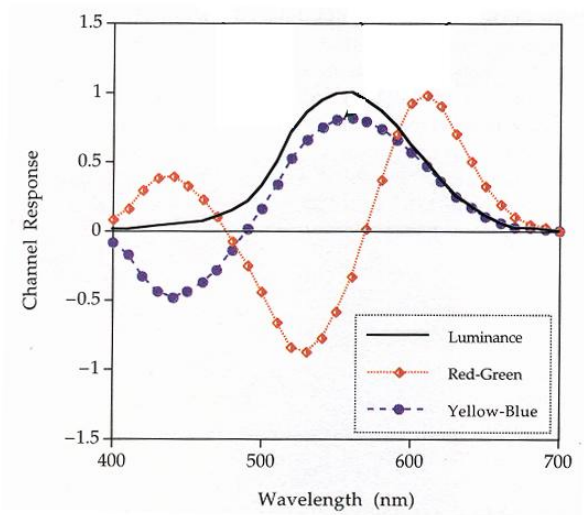


Figure 1.7: *Opponent process curves.*

In Figure 1.7 the positive and negative values are arbitrarily assigned. They are assigned to represent the opponent characteristics of colors. The chromatic values represent the amount of color seen at different wavelengths. For example, at 450 nm the blue value is approximately -1.0 and the red value is approximately 0.2. These values indicate that the perceived color should be blue with a small red component. From the opponent process curves, we can observe that pure red cannot be obtained with a single wavelength. In fact, to obtain pure red one must take a long wavelength (e.g. 700nm) and mix it with a little bit of blue to cancel out the perception of yellow.

1.3 Creating colors

There exist two different methods of mixing colors ([59]): the first one is additive mixture, which is the addition of different wavelengths and is produced by light mixture. The second one is subtractive mixture, which is the subtraction or cancelation of bands of wavelengths by the combination of light absorbing materials. It is produced by mixing paints.

In additive mixing ([65]), we start the mixture with the absence of color stimuli, and we add three lights ("colors") of different wavelengths, that cover different parts of the spectrum, to stimulate the S, M and L cones receptors in the eye. Any three colors that are linearly independent (i.e., none of them can be obtained as a mixture of the other two) can be considered in order to obtain an additive mixture. These three colors are called primary colors. Red, green and blue are the most commonly used as additive primary colors (see Fig. 1.8). In this type of mixture, the black color is the result of no colors mixed at all. On the contrary, the addition of these three primary colors yields the white color.

By means of this kind of mixture, color is produced on a color display such as a television or computer monitor. The surface of a color display is made up of hundreds of tiny dots of phosphor ([68]). Phosphors are chemical substances that emit light when they are bombarded with electrons and the amount of light given off depends on the strength of the electron beam. The phosphors on the screen are in groups of three, one bluish (for the short wavelengths), one greenish (for the medium wavelengths) and one reddish (for the long wavelengths). By varying the intensity levels of the phosphors, different levels of lightness are achieved. The human eye does not have enough resolution to distinguish each isolated element. For this reason the perception is a light beam with the three primary colors blending at each point, creating any hue.

Subtractive mixing is the process of filtering parts of the spectrum of the reflected light. It is based on the capacity of the surface to reflect some wavelengths and absorb others ([65]). When a surface is painted with a pigment or dye, a new reflectance characteristic is developed based on the capacity of the pigment or dye to reflect and absorb the different wavelengths of light. An example shall help to better understand this kind of mixture. Consider a surface painted with a yellow pigment which reflects wavelengths 570-580nm and another surface painted with cyan pigment which reflects 440-540nm. The color resulting from the mixture of both pigments will be green. This is because the yellow pigment absorbs the shorter wavelengths and the cyan pigment absorbs the entire longer wavelengths. As a consequence the only reflected wavelengths are some medium wavelengths, which create the sensation of green. Yellow, cyan and magenta are the most commonly used subtractive primary colors (see Fig. 1.8). As mentioned above, the yellow ink absorbs the short wavelengths and the cyan ink absorbs the long wavelengths, while the magenta ink absorbs the medium wavelengths. They are called the secondary colors and they are obtained by adding two primary colors. Yellow is obtained from red and green, cyan from green and blue, and magenta from red and blue. Here white is the result of no mixing (the entire spectrum is reflected) whereas mixing the three subtractive primary colors yields no reflection of light, i.e., black.

Some mechanical devices, such as printers, use the secondary colors to obtain the rest of colors ([74]). In printers, the original image is separated into its cyan, yellow and magenta components. A film is made for each separation and then a plate is produced from the film. White paper (which reflects all wavelengths) is run through the stations of a color press to accept layers of ink from each plate. The different quantity of ink mixed

produces a higher or lower level of absorption in the short, medium and long wavelengths. In theory, it is possible to create any reflective color by mixing a combination of cyan, magenta and yellow. In practice, however, the inks that printers use are not perfect. This becomes more obvious when all three colors are mixed to obtain black color. The color that results is muddy brown, due to the impurities in the inks. That is why printers use black ink to get the best results.

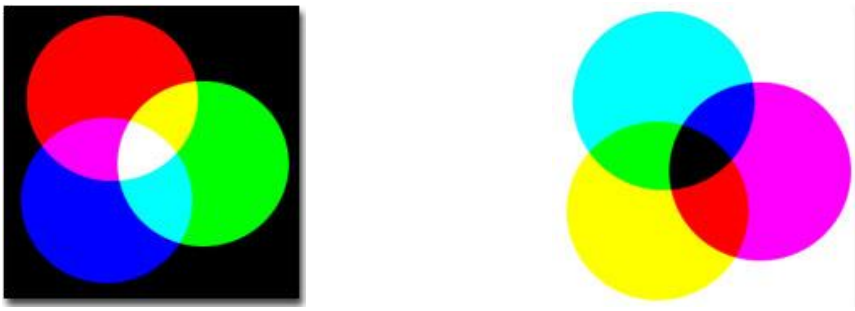


Figure 1.8: *Additive and subtractive mixture.*

These two mixture processes can be represented with color algebra. If we consider the three primary colors, we can write the following formulas:

	Additive mixture (Light)	**Subtractive mixture (Ink)**
Red	$+r$	$-g-b$
Green	$+g$	$-r-b$
Blue	$+b$	$-r-g$
Cyan	$+g+b$	$-r$
Magenta	$+r+b$	$-g$
Yellow	$+g+r$	$-b$
White	$+r+g+b$	0
Black	0	$-r-g-b$

where the symbol $-$ in front of some color means that this color is absorbed, and the symbol $+$ means that the color light is projected. In the substractive mixture the white light is diffused. We can obtain all colors from these combinations, obtaining the level of intensity from the mixture level.

By looking at these formulas we can observe that the addition of blue and yellow light results in white light, and their subtractive mixture results in black ink. This fact can surprise us because we have the idea that blue plus yellow is green. The problem starts in childhood, when we are taught that yellow plus blue equals green. The mixing only works because the colors produced by pigments have a broad spectral content and the used paints are not monochromatic yellow and blue colors. Then, yellow and blue paints used together absorb everything in the light except greens, which are reflected ([51]).

The traditional primary colors that painters have used are red, yellow and blue. This confusion comes from 19th century theories and it has lasted for nearly two centuries.

1.3.1 Historical notes

The study of the perception of color is historically intertwined with the study of the physical nature of light. The early discoveries in optics were made on the basis of direct observations, confounding the effects of perception with the physical nature of light.

The first studies about color were done by Newton ([61]). He noted that light passing through a prism was affected in such a way that different colors of light were refracted at different angles and he deduced that the white light is composed of various colors. Newton also suggested that the perception of color was due to some differential impact of something like the frequency of light upon the eye. Furthermore, he observed that by mixing yellow powder and blue powder together one obtained an apparently green powder. He thus confused additive and subtractive colors.

This confusion persisted among the next color researchers. One of them, Thomas Young observed that "The sensation of different colors depends on the different frequency of vibrations, excited by light in the retina"([97]). Furthermore, he suggested that the retina might be only sensitive to three principal colors (red, yellow and blue) and that the whole appearance of color might be attributable to varying degrees of excitation of these three receptors. Then, David Brewster exposed that each portion of the spectrum was actually composed of three individual types of light which had the primary colors of red, yellow and blue ([9]).

Finally Hermann von Helmholtz brought to an end both confusions, the confusion of additive versus subtractive colors and the confusion of the perceptual phenomenon of color. He demonstrated that green and red spectral colors added to make yellow, while yellow and blue added to make white ([92]). He further observed that white light could be composed from choices of three spectral colors.

Even though after the Helmholtz theory red, yellow and blue are no more considered primary colors, they are still considered as such by the painters.

1.4 Characteristics of color perception

The human visual system does not discriminate all the colors with the same efficiency and if two colors are very close perceptually, they can be confused. In the perceptual system, the sensitivity to the distance between colors is not related to the photoreceptors behavior in an uniform way ([90]).

Dynamic adaption mechanism

One of the main characteristics of color perception is the dynamic mechanism of adaptation that serves to optimize the visual response to the particular viewing environment. Three different mechanisms of adaptation are of particular relevance to the study of color appearance, the dark, the light and the chromatic adaptation.

The dark and light adaptations are the changes in visual sensitivity that occur when the prevailing illumination level is decreased or increased suddenly, respectively. In this process, the visual system response becomes more sensitive, depleting or regenerating the photopigment in the cones and the rods. The rods have a longer recovery time than the cones and, as a consequence, the adaptation to bright light is faster than the adaptation to a dark enviroment (see [37] and [42]).

Chromatic adaption

In reference to the chromatic adaptation, the visual system has the capacity to adapt to the average luminosity of the environment. This phenomenon can be seen in Figure 1.9. In this figure there are three images. The first image is the original image with a yellow towel. In the second image a blue filter has been put only on the towel. Then we can see

that the color is now green. In the third image the same filter as in the second image has been put on the whole image. Then we again see the towel with a yellow color, because our visual system has adapted to the luminosity change. This phenomenon, called color constancy, can be described as the fact that if a scene is lighted up from a given source and this light source is changed, then the object colors are not changed for the observer. On the contrary, if the illumination change only affects one object, then the object color is changed. In other words, an observer adapts his colors perception in reference to the whole visual field.

Figure 1.9: *Color constancy phenomenon.*

The color constancy phenomenon can also be observed by examining a white piece of paper under various types of illumination, for example, daylight and tungsten light. We have mentioned before that the spectral distribution of these two illuminants is very different. Thus, the paper color should look different under the different illuminants. It turns out that the paper retains its white appearance under both light sources. This is due to the fact that the cones adapt their response to the reflectance level in the environment. In the case of a tungsten light, the light has a higher intensity in the long wavelengths than in the short ones. Thus, the objects that are illuminated by tungsten light reflect more long light wavelengths than those illuminated by daylight. The eye adapts to the long wavelengths, which are more present, and this adaptation decreases the sensibility of the eye to reds and yellows. Then, the white paper, which should look yellow, is perceived as white. We can conclude that the visual mechanism to perceive the contrast of the colors depends on the relative proportions between the different illuminations in the image ([42]).

This phenomenon does not affect the display devices, since they emit their own light, unless there is a colored light source illuminating them.

Simultaneous contrast phenomenon

Another characteristic of the visual system is the simultaneous contrast phenomenon, which is apparent in different cases, for instance ([90]):

1. Two colors, one in front and the other behind, seem more contrasted than if they are seen separately.

2. The same color, put on different backgrounds, tends to be perceived as different colors.

3. Two colors, put on the same background, tend to seem more contrasted when their surface grows.

This phenomenon can be seen in Figures 1.10 and 1.11.

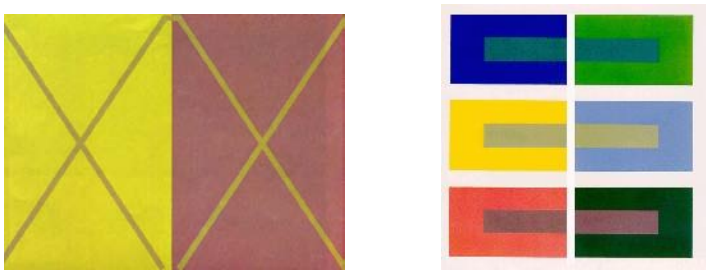


Figure 1.10: *Simultaneous contrast phenomenon. In the first image there are only three colors, the colors in the diagonal bars are the same in the two rectangles. The background color changes their perception. In the second image, the fact that the color contrast depends on the background is patent.*

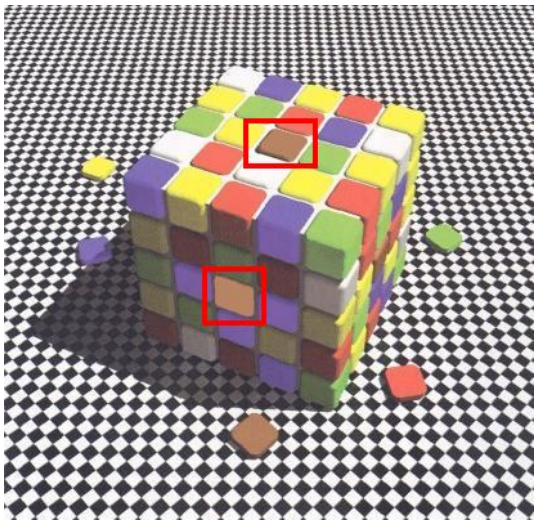


Figure 1.11: *Simultaneous contrast phenomenon. The little brown square in the middle of the superior face (marked with a red rectangle) has exactly the same color as the little brown square in the middle of the frontal face (marked with a red rectangle). The intensity neigborhood difference produces an color effect different in both squares.*

Color-vision deficiencies

Some color-vision deficiencies are caused by the lack of a particular type of cone photopigment. Since there are three types of cone photopigments, there are three general classes of these color-vision deficiencies: protanopia, deuteranopia and tritanopia ([42] and [61]). The protanopes and the deuteranopes are also called daltonic.

A subject with protanopia is missing the L-cone photopigment and therefore is unable to distinguish between reddish and greenish hues, due to the fact that the red-green opponent mechanism does not have the correct information. A deuteranope is missing the M-cone photopigment and cannot distinguish reddish and greenish hues, either. Protanopes and deuteranopes can be distinguished by their relative luminous sensitivity. Finally, a tritanope is missing the S-cone photopigment and cannot distinguish between yellowish and bluish hues due to the lack of a yellow-blue opponent mechanism.

There are other types of color-vision deficiencies, such as some cases of cone monochromatism (people that have only one cone type) or rod monochromatism (no cone responses).

An 8% of the male population and slightly less than 1% of the female population suffer from some of these color deficiencies. This disparity between the occurrence in males and females can be traced to the genetic basis of such deficiencies. Given the fairly high rate of occurrences of color-vision deficiencies, various tests are available to make critical color-appearance or color-matching judgements. In Figure 1.12, we can see one of these tests.

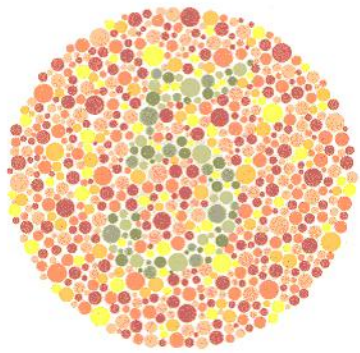


Figure 1.12: *Subjects with normal vision are capable to see a green number in the image center. Instead protanopes and deuteranopes do not see the number, due to their impossibility to distinguish reddish and greenish hues.*

1.5 Color spaces

In order to use color as a visual cue in image processing, an appropriate method for representing the color signals is needed. The different color specification systems or color models (colors spaces or solids) address this need. Color spaces provide a rational method to specify, order, manipulate and effectively display the object colors taken into consideration. The color space choice depends on the previous knowledge that we have about "color" and on the application that we want to give to this information (for example, defining colors, discriminating between colors, judging similarity between colors,...).

The color models are normally three-dimensional spaces, due to the fact that our perception of color is trichromatic. The different spaces differ in the choice of the coordinate system, which defines the space. In the classical literature, four basic color model families can be distinguished ([76]):

1. **Physiologically inspired color models**, which are based on the three primary colors which stimulate the three types of cones in the human retina. The RGB color space is the best-known example of a physiologically inspired color model.

2. **Colorimetric color models**, which are based on physical measurements of spectral reflectance. Three primary color filters and a photometer are usually needed to obtain these measurements. The CIE chromacity diagram is an example of these models.

3. **Psychophysical color models**, which are based on the human perception of color. Such models are either based on subjective observation criteria or are built by taking into account the human perception of color.

4. **Opponent color models**, which are based on perception experiments, utilizing mainly pairwise opponent primary colors.

In this chapter, we present three color spaces which are the most commonly used in the image processing field. Each one of them has certain essential characteristics, in contrast with the other one's.

1.5.1 *RGB* Color Space

The red, green and blue receptors in the retina define a trichromatic space whose basis is composed by pure colors in the short, medium and high portions of the visible spectrum. As we have mentioned in Section 1.3, it is possible to reproduce a large number of colors by additive mixture using the three primary colors.

The *RGB* model is the most natural space and the most commonly used in image processing, computer graphics and multimedia systems. This is due to the fact that display devices use the addition mixture and the three primary colors to reproduce and "encode" the different hues (see Section 1.3). This model is based on the cartesian coordinate system ([59]). The pixel's red, green and blue values in a digital color image are the three coordinates of the model and they are represented by a cube, since the three values are non-negative and they are considered as independent variables. Pure red, green and blue are situated in three vertices of the cube, while the other three vertices correspond to pure yellow, cyan and magenta (see Fig. 1.13). Black has coordinates $(0, 0, 0)$ and, at the opposite vertex, stands the white color. We call the line that joins the black vertex to the white vertex the *grey axis*, in which the three coordinate values are equal. In mathematical terms, the *RGB* cube is made up of all points of three coordinates, (r, g, b), with $0 \leq r,\ g,\ b \leq M$. The range of the three coordinates is usually from 0 to 255. The space is sometimes normalized so that $M = 1$.

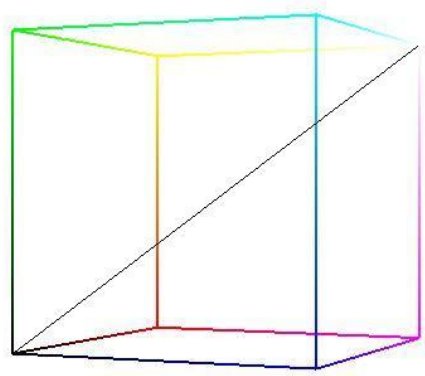


Figure 1.13: *The RGB color model.*

The main disadvantage of this space is the perceptual non-uniformity, *i.e.* the low correlation between the perceived difference of two colours and the Euclidean distance in this space. Its psychological non-uniformity is another problem. in this space, the information about the chromaticity (hue and saturation) and intensity components of color is not independent. Finally, considering the representation of a natural image in

the RGB space, we observe a powerful correlation between the different coordinates. It is due to the special distributions of the sensors in the camera and to the post-processing operations. Therefore independent operations over each coordinate are not possible.

1.5.2 HSI Color space

The RGB space is not easy for user specification and recognition of colors. The user cannot easily specify a desired color in the RGB model. Models based on lightness, hue and saturation are considered to be better suited for human interaction. These three color features are defined as ([96]):

- **Intensity or lightness** is the visual sensation through which a surface that is illuminated by a given luminous source seems to project more or less light. It corresponds to light, dark or faint terms. In some sense, lightness may be referred to as relative brightness.

- **Hue** is the visual sensation that corresponds to the color purity. The hue is defined by the dominant wavelength in the spectral distribution.

- **Saturation** measures the proportion on which the pure color is diluted with white light. It corresponds to pale, faded or brilliant terms.

The HSI color model owes its usefulness to two main facts. First, the intensity component is decoupled from the chrominance information represented as hue and saturation. Second, hue is invariant with respect to shadows and reflections. This is due to the fact that hue is the dominant wavelength in the spectral distribution and is independent on the intensity of the white light. We would wish the three variables to be independent. However, saturation depends on the intensity. If the intensity level is very high or very low, then, the saturation variable can only take very low values.

The HSI color space can be described geometrically from the RGB cube (Fig. 1.14). It is constructed by placing an axis between the black point and the white point in the RGB space, that is the diagonal of the RGB cube. As we have mentioned before, this axis is often referred to as the *grey axis*. Any color point on the HSI space is defined by its three components with respect to this axis: hue, which is the angle between a reference line and the line passing through the color point and orthogonal to the grey axis; saturation, which is the radial distance from the point to the grey axis; and intensity, which is the magnitude of the orthogonal projection of the color point onto the grey axis.

The main drawback of this representation in the continuous case is that the hue has a nonremovable singularity on the grey axis. This singularity occurs whenever $R = G = B$.

We have indicated in Section 1.2.1 that the perception of the white color is the result of an approximately equal stimulation of the S, M and L cones. We know that the grey axis is composed of the colors which have the three variables identical, i.e. $R = G = B$. Then, we can conclude that the greys are not colors, or that they are colors without hue, since the white and the grey are just differentiated by their intensity level.

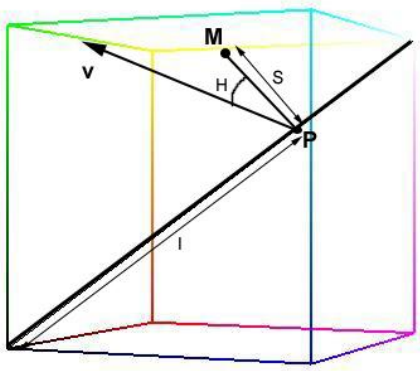


Figure 1.14: *Representation of the HSI space.*

Conversion Formulas

In the literature, there are various formulas of conversion from RGB to HSI. The more usual ones are ([72]):

$$\begin{aligned} H &= arccos\left[\frac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-G)^2+(R-B)(G-B)]^{1/2}}\right], \\ S &= 1-\frac{3}{R+G+B}[min(R,G,B)], \\ I &= \frac{R+G+B}{3}, \end{aligned} \tag{1.1}$$

where RGB and SI values range between 0 and 1 and $H = 360° - H$, if $B/I > G/I$. Hue is not well defined when the saturation is zero. Similarly, saturation is undefined if intensity is zero.

We note that these formulas are not adequate because they are defined with respect to the Maxwell triangle, defined in the next section, wich is not an adequate representation of the color, as we will discuss later. Moreover, they do not take into account the definitions of hue, intensity and saturation given at the beginning of Section 1.5.2. Other color models use similar parameters, HSV or HLS for example, but they have inherent problems in describing saturated pixels ([58]). We propose a definition of the HSI space (similar to the one presented in [58]).

We will redefine the conversion formulas with respect to the definitions of hue, saturation and intensity given at the beginning of this section. We denote by O the origin point in the RGB cube, and by $M = (R, G, B)$ the point of color in the cube (see Fig. 1.14). The intensity component is the arithmetic mean of the R, G, and B values, obtaining the same formula as before,

$$I = \frac{R+G+B}{3}.$$

Now, consider the projection point from M to the grey axis, $P = (I, I, I)$, where $P = (\overrightarrow{OM} \cdot \vec{U}) \cdot \vec{U}$, and $\vec{U}$ is the unitary vector of the grey axis $\vec{U} = (\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}})$. The saturation component is the distance of the point M to its projection P, that is, the module of the vector $\overrightarrow{MP}$,

$$S = ||\overrightarrow{OM} - \overrightarrow{OP}|| = \sqrt{(R-I)^2 + (G-I)^2 + (B-I)^2}.$$

Finally, the hue component is calculated as the angle between the vector difference $\overrightarrow{OM} - \overrightarrow{OP}$ and an orthonormal vector to the grey axis, for example, the unitary vector $v = (0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$. Thus,

$$H = \arccos\left(\frac{(G-I)-(B-I)}{S\cdot\sqrt{2}}\right).$$

If we compare the new formulas for H, S and I with the formulas (1.1), we observe that the expressions of the components H and S are different. A simple computation shows that the formula for H is basically the same with the only difference consisting in the fact that in (1.1) the reference vector is $(2, -1, -1)$ while in our expression it is $(0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$.

In the case of the inversion formulas, we want to obtain the value of point M, that is, the position in the RGB cube from the point that is defined by the H, S and I values. To get the point M is equivalent to find the vector $\overrightarrow{OM}$. Using the projection point P on the grey axis, it follows that $\overrightarrow{OM} = \overrightarrow{OP} + \overrightarrow{PM}$. By hypothesis, we have that $\overrightarrow{OP} = (I, I, I)$ and therefore, we only need to compute the vector $\overrightarrow{PM}$. If we consider the vectors $\vec{\imath}$ and $\vec{\jmath}$ as two unitary vectors orthonormal between them and orthonormal to the grey axis, we can define $\overrightarrow{PM} = S\cdot\cos(H)\vec{\imath} + S\cdot\sin(H)\vec{\jmath}$. The vector $\vec{\imath}$ has to be the orthonormal vector used in the conversion formulas from RGB to HSI, that is, $\vec{\imath} = (0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$. As the vector $\vec{\jmath}$ has to be orthonormal to $\vec{\imath}$ and to grey axis and unitary, then $\vec{\jmath} = (-\frac{2}{\sqrt{6}}, \frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}})$. Finally, the conversion formulas from HSI to RGB are

$$\begin{aligned}
R &= I + S\cdot\left(\frac{-2\cdot\sin(H)}{\sqrt{6}}\right),\\
G &= I + S\cdot\left(\frac{\cos(H)}{\sqrt{2}} + \frac{\sin(H)}{\sqrt{6}}\right),\\
B &= I + S\cdot\left(\frac{-\cos(H)}{\sqrt{2}} + \frac{\sin(H)}{\sqrt{6}}\right).
\end{aligned}$$

1.5.3 Two-dimensional spaces

Working with three-dimensional spaces presents some difficulties, as their representation, interpretation, storing, processing,... To solve these problems, different two-dimensional spaces have been proposed. We can distinguish two main representation systems: the Maxwell triangle and the HS space.

The Maxwell triangle is the plane of the RGB cube on which all color points satisfy the equation $R+G+B = a$, where $a = \{1, 255\}$, depending on whether the data is normalized or not ([76]). This triangle joins the three primary colors in the cube. Moreover, all points in this plane have the same intensity, as, also, in all the planes parallel to it. In Figure 1.15 we can see the triangle location in the RGB cube and, in Figure 1.16, its representation.

The main disadvantage of the Maxwell triangle is that the primary color area is bigger than the secondary one. This model attaches much more importance to the primary colors than to the secondary colors. For example, it is difficult to see in Figure 1.16 the pixels related to the yellow area. We conclude that this representation is not accurate enough and inappropriate to recognize the colors. For this reason the classical conversion formulas between HSI and RGB spaces, which are based on the Maxwell triangle, are not suited for our purposes, as pointed out in the previous section.

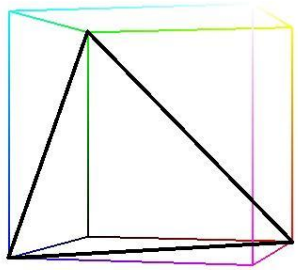


Figure 1.15: *Maxwell triangle in the cube RGB.*

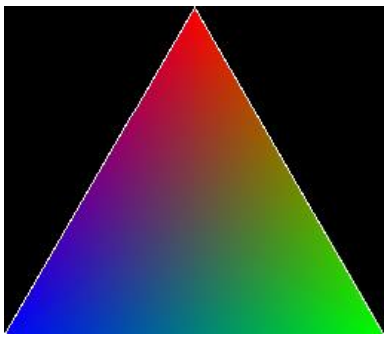


Figure 1.16: *Representation of the Maxwell Triangle.*

The HS space is the conversion from the HSI space to a two-dimensional space. This two-dimensional space is defined as a plane of the HSI space. There are different ways to create this space, either by considering the pixels belonging to a plane with a fixed intensity, or by projecting all the color points in this same plane.

However, this system does not give an accurate representation, either because a lot of information is lost or because different color points can be represented by the same coordinates in the space.

1.5.4 CIE Color Spaces

In 1931, the Commission Internationale de L'Eclairage (CIE) created a new color space called $CIE\ XYZ$ color space. XYZ is a device independent color space that is exceptionally useful for color management purposes. It is based on direct measurements of the human eye. In this space the coordinates X, Y and Z are called the tristimulus values, which are also roughly red, green and blue, respectivaly. Any color can be described by the XYZ coordinates ([96]). The CIE color specification system is based on the description of color as the luminance component Y and the chromaticity information, given by two additional components x and y, which are functions of all three tristimulus values X, Y and Z :

$$x = \frac{X}{X+Y+Z}, \qquad y = \frac{Y}{X+Y+Z}.$$

Figure 1.17 shows the related chromaticity diagram. The outer curved boundary is the spectral locus, with wavelengths shown in nanometers. Note that the chromaticity diagram is a tool to specify how the human eye will experience light with a given spectrum. This diagram represents the x and y coordinates with Y constant. Less saturated colours

appear in the interior of the figure, with white at the centre. If one chooses any two points on the chromaticity diagram, then all colors that can be formed by mixing these two colors lie between those two points, on a straight line connecting them. It follows that the gamut of colors must be convex in shape. All colors that can be formed by mixing three sources are found inside the triangle formed by the source points on the chromaticity diagram, and so on for multiple sources. Moreover, the mixture of two equally bright colors will not generally lie on the midpoint of that line segment. In more general terms, a distance on the xy chromaticity diagram does not correspond to the degree of difference between two colors. Other color spaces ($CIELuv$ and $CIELab$ in particular) have been designed to solve this problem.

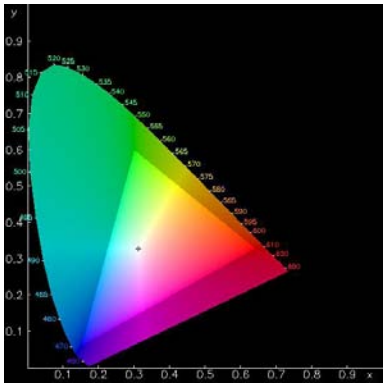


Figure 1.17: *Representation CIE chromaticity diagram.*

The linear RGB color space can be transformed to and from the $CIE\ XYZ$ color space by using a simple linear transform (see next section).

In image processing, it is of particular interest the use of perceptually uniform color spaces where a small perturbation in a component value is approximately equally perceptible along the range of that value. The color specification systems discussed until now are far from this uniform property. In 1976, CIE standardized two spaces, Lab and Luv, as perceptually uniform ([52]). Both spaces work well in perceptual uniformity and provide very good estimates of color differences between two color vectors, because this difference is calculated with the Euclidean distance.

In both spaces, the coordinate L represents the luminance axis, which is equivalent to the grey axis. The coordinates (a, b) and (u, v) represent the chrominance, considering a constant luminance.

The axis a corresponds to the antagonist pair green-red and the axis b corresponds to the pair blue-yellow.

A disadvantage of these uniform color systems is that both color spaces are computationally expensive to transform to and from the linear as well as non-linear color spaces. Moreover, the different conversion formulas present singularities and rounding problems, where the formulas are not well defined.

Conversion Formulas

Conversion from RGB to $CIELab$ or $CIELuv$ consists of two steps. In a first step, the RGB components are transformed to $CIE\ XYZ$ using the appropriate matrix. Then, the particular formulas to transform from $CIE\ XYZ$ to $CIELab$ or $CIELuv$ are used. These formulas depend on the physical luminance of the white reference point.

The transformation matrix between XYZ and RGB is ([36]):

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 3.240479 & -1.537150 & -0.498535 \\ -0.96926 & 1.875992 & 0.041556 \\ 0.055648 & -0.204043 & 1.057311 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

The inverse transformation matrix, i.e. from RGB to XYZ, is:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

The coordinates of a XYZ color point in the $CIELab$ space can be obtained by using the following relations ([96]):

- If $\frac{Y}{Y_n} > 0.008856$, then:

$$\begin{aligned} L^* &= 116\left(\frac{Y}{Y_n}\right)^{1/3} - 16 \\ a^* &= 500\left(\left(\frac{X}{X_n}\right)^{1/3} - \left(\frac{Y}{Y_n}\right)^{1/3}\right) \\ b^* &= 200\left(\left(\frac{Y}{Y_n}\right)^{1/3} - \left(\frac{Z}{Z_n}\right)^{1/3}\right) \end{aligned}$$

- If $\frac{Y}{Y_n} \leq 0.008856$, then:

$$\begin{aligned} L^* &= 903.3\left(\frac{Y}{Y_n}\right) \\ a^* &= 7.787\left(\frac{X}{X_n} - \frac{Y}{Y_n}\right) \\ b^* &= 7.787\left(\frac{Y}{Y_n} - \frac{Z}{Z_n}\right) \end{aligned}$$

In both cases X_n, Y_n and Z_n are the standard values of X, Y and Z for the white color.

The inverse transform from $CIELab$ to RGB is only allowed when $\frac{Y}{Y_n} > 0.008856$, and it can be calculated using the next formulas:

$$\begin{aligned} X &= X_n(P + \tfrac{a^*}{500})^3 \\ Y &= Y_n P^3 \\ Z &= Z_n(P - \tfrac{b^*}{200})^3 \end{aligned}$$

where $P = \frac{L^*+16}{116}$.

Normally, $CIELab$ coordinates are converted to classical HSI coordinates, since the conversion formulas are much simpler

$$\begin{aligned} I &= L^* \\ H &= arctan\left(\frac{a^*}{b^*}\right) \\ S &= \sqrt{(a^*)^2 + (b^*)^2} \end{aligned}$$

The $CIELuv$ space coordinates of a XYZ color point are computed as

$$\begin{aligned} L^* &= 116\left(\frac{Y}{Y_n}\right)^{1/3} && if \quad \frac{Y}{Y_n} > 0.008856 \\ L^* &= 903.3\frac{Y}{Y_n} && if \quad \frac{Y}{Y_n} < 0.008856 \\ u^* &= 13L^*(u' - u'_n) \\ v^* &= 13L^*(v' - v'_n) \end{aligned}$$

where

$$u' = \frac{4X}{X + 15Y + 3Z} \qquad v' = \frac{9Y}{X + 15Y + 3Z}$$

and u'_n, v'_n are the corresponding values of u' and v' for X_n, Y_n and Z_n.

In the same way than for the $CIELab$ transformation, we use an inverse transformation from $CIELuv$ space to HSI space, obtaining the same equations as in the previous case by replacing the coordinate a by the coordinate u and b by v.

1.6 Colors in real life

We cannot finish this chapter without talking about the colors that we find around us. What colors are predominant in the nature? Are there any colors which we cannot find around us? What proportion of red exists in the real life? Answering these questions and other similar is the aim of the project "Very Large Cube" (VLC), which is still in progress, but whose preliminar results are presented in this section.

The idea of this study is to accumulate the colors of a lot of pictures in a same RGB color cube. These pictures come from "real life" scenes: landscapes, photographies of animals, people, daily life, objects,.. That is, varied photographies with different conditions of illuminants and different characteristics. Different frames of films can be considered as pictures too. We do not consider synthetic pictures, that is, digital pictures whose colors have been normally selected by an artist or graphic designer. The RGB values of the pixels in each picture are stored in the same data file (the "Very Large Cube" file). Currently, we have accumulated values for over 15.800 images. Our aim is obtaining at least 30.000 pictures.

When the "Very Large Cube" is completely filled, we will be able to study the theorical distribution of the colors that are around us and to use this information to analyse the color images.

From this partial work, we already can extract some conclusions. Firstly, we can observe in Figure 1.18 the red, green and blue histograms extracted from the "Very Large Cube". We can observe that the three histograms have a similar shape. The histograms present high peaks in the extreme values, that is in the lighter and darker colors. Moreover, we observe that the darker colors are more abundant than the lighter ones. We observe in the three histograms little peaks that are repeated periodically. This phenomenon requires a more profound study, but we can hypothesize that they are due to the JPG compression effect.

The hue histogram from the "Very Large Cube", Figure 1.19, provides us a very important information. We can observe that the more abundant hue among these 15.800 pictures is the red-orange hue, followed by the blue hues. We can also observe that the magenta and cyan hues are presented in relatively few pixels.

Finally, we have visualized the 3D distribution of the colors of the VLC in the RGB space. We have observed that this distribution almost fills the color space, except for a few gaps in the saturated colors. The direct observation of the cube is a hard task, since the exterior colors in the cube hide the interior information. Due to this fact, we have decided to study the cube by representing the color information for different fixed intensity planes, see Figure 1.20. In this image we can observe nine planes of the Cube. We observe that almost all the colors in the planes of low and high intensity are present in the Cube. While the other planes present gaps in some saturated colors, such as magentas, cyans or some greens.

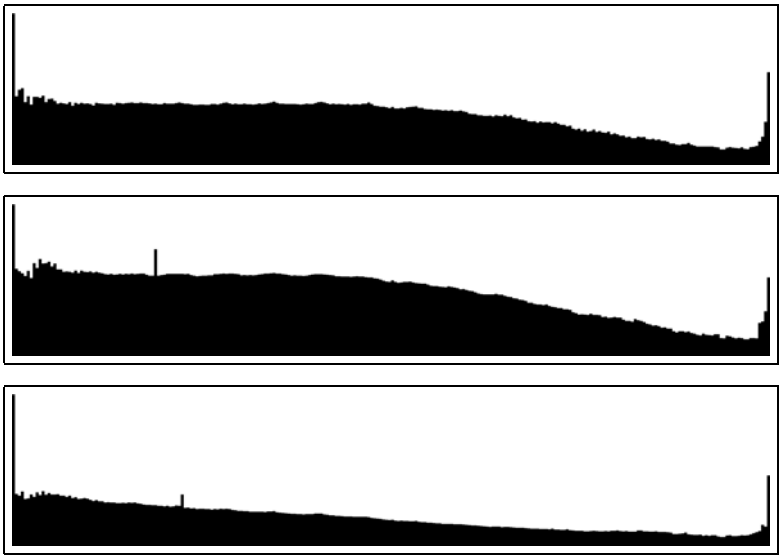


Figure 1.18: *Red, green and blue histograms from the "Very Large Cube", respectively.*

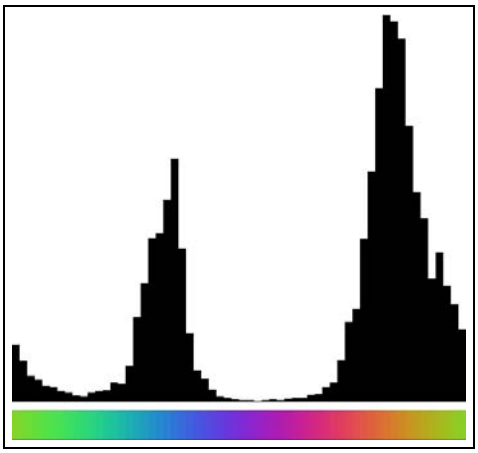


Figure 1.19: *Hue histogram from the "Very Large Cube".*

Finally, we can analise the Cube by removing the colors that have a small value in the histogram of the "Very Large Cube". These colors appear in few pictures and are not very meaningful around us. The results of this action can be observed in Figure 1.21. In the first row we can see the "Very Large Cube" from three different viewpoints. Only the colors with a histogram value higher than 100 are displayed. In the second row the colors that have a histogram value lower than 1000 have been eliminated , and in the third row the threshold for the histogram value is set to 10000. It is clear that the colors with high histogram values are the low saturated colors, the colors close to the grey cylinder. Moreover, this figure underlines the gaps in the Cube observed in the previous figure. By adding more pictures we will obtain a more uniform cube.

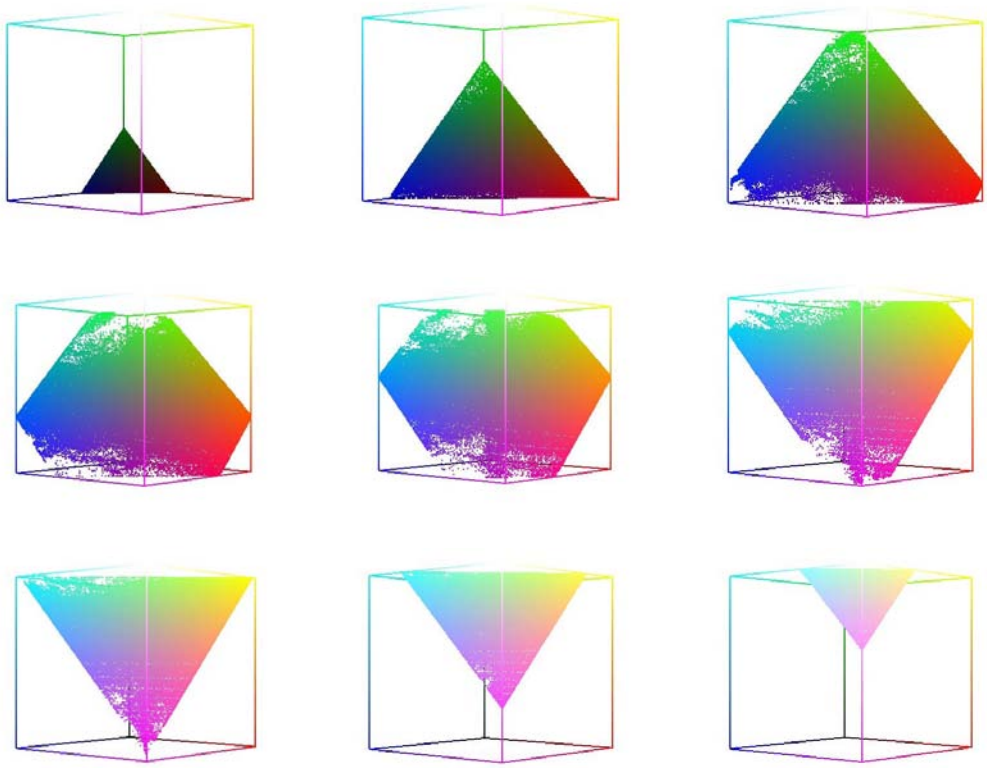


Figure 1.20: *Fixed intensity planes of the "Very Large Cube". These nine planes represent clearly the "Very Large Cube". We can observe the gaps in the Cube, that is, the colors that are not present around us (at least in a large set of images from the real world).*

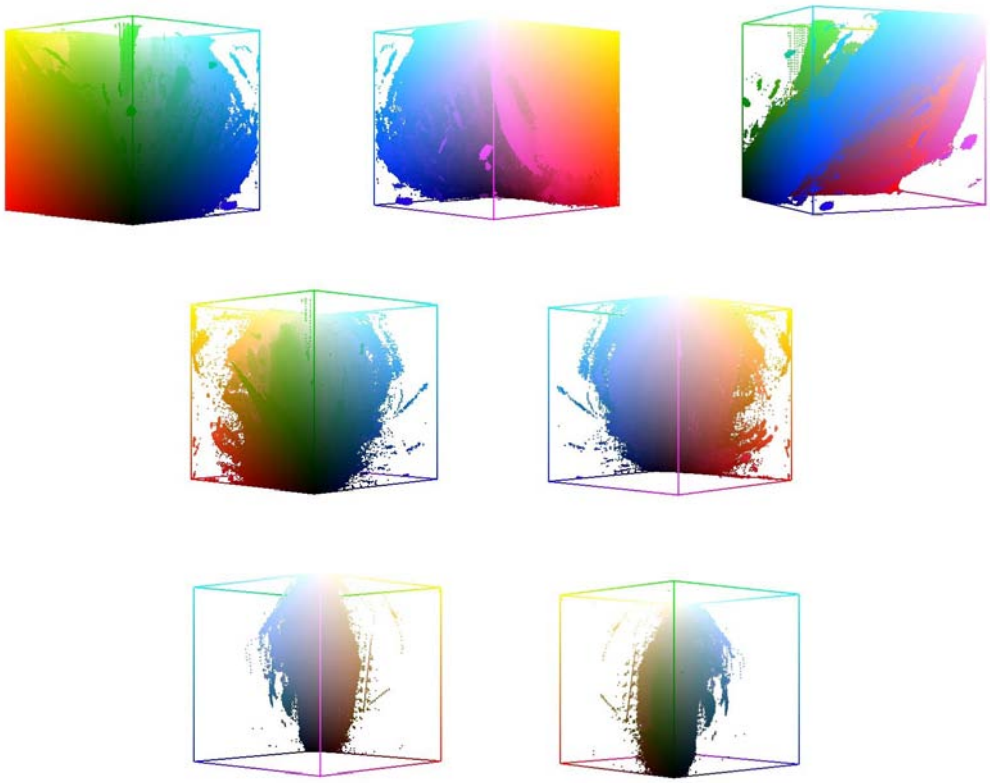


Figure 1.21: *Results of removing colors in the "Very Large Cube". The first row represents three different points of view of the colors in the "Very Large Cube" with histogram values higher than 100. In the second row we have two points of view of the colors in the "Very Large Cube" with histogram values higher then 1000. And, finally, in the third row we display the colors with histogram values higher then 10000. We observe that in the third row the remaining colors are the ones with low saturation values.*

Chapter 2

Histogram segmentation

Abstract. Histograms permit a quantitative study of the color information contained in a digital image. Colors are represented in some of the 3D color spaces described in Chapter 1 and the analysis of the 1D histograms associated to each one of the space components allow us to obtain the main features characterizing the color image. A basic description of the 1D histograms is given by the list of its *modes*, *i.e.* the intervals of values around which data concentrate. In this chapter we propose a new approach to segment a 1D-histogram without *a priori* assumption about the underlying density function. The theorical justification of the approach is presented in two steps, considering two different points of view. The method is based on the definition of meaningful events, modes and gaps in a first approach, presented in the first section of this chapter, and of meaningful rejections in a second approach, and in the second section. The final result is a fast and parameter free algorithm, the Fine to Coarse algorithm, which avoids under and over-segmentation. Some properties of the method are remarked at the end of the chapter.

2.1 Introduction

Histograms have been extensively used in image analysis and more generally in data analysis, mainly for two reasons. They provide a compact representation of large amounts of data and it is often possible to infer global properties of the data from the behavior of their histogram. One of the features that better describes a 1D-histogram is the list of its *modes*, *i.e.* the intervals of values around which data concentrate. For example the histogram of hues or intensities of an image made of different regions can exhibit different peaks, each one of them ideally corresponding to a different region in the image. In this case, a proper segmentation of the image can be obtained by computing the appropriate thresholds that separate the modes in the histogram. However, it is not always easy to quantify the amount of "data concentration" in an interval, and hence to separate modes.

Among the algorithms proposed for 1D histogram segmentation we can distinguish two principal classes of methods. First, parametric approaches (see [35]) assume the set of points to be samples of mixtures of k random variables of given distributions, as in the Gaussian Mixture Models (see Appendix A). If k is known, optimization algorithms such as the EM algorithm [29] can estimate efficiently the parameters of these distributions. The estimated density can then be easily segmented to classify the original data. The main objection to this approach is that histograms obtained from real data cannot always be modeled as mixtures of Gaussians. This is for example the case for luminance histograms,

as we shall see in Section 3.2. Other approaches give up any assumption on the underlying density. Among them, bilevel or multilevel thresholding approaches intend to divide the histogram into several segments in an optimal way according to some energy criterion (variance ([2]), entropy of the histogram ([55]), entropy of the co-ocurrence matrix ([14]), *etc...*).

Other approaches (for instance, mean shift [20]) find peaks (local maxima) of the histogram without estimating the underlying density. These methods tend to detect too many peaks in histograms coming from real data when they are noisy. Some criterion is therefore needed to decide which of these peaks correspond to true modes ([93]). Indeed, one of the main challenges of histogram analysis is the detection of small modes among big ones (see, for example, Fig. 2.1).

In all of these cases, the choice of the number of segments is crucial. If the histogram is constant, a bilevel thresholding based on entropy divides it at the median value. Yet we know that a constant histogram is not bimodal. In the opposite way, modes can be missed if the number given by the user is too small. This number can be specified *a priori* and becomes a method parameter. It can also be estimated if its *a priori* distribution is hypothesized. Generally, *ad hoc* procedures are used to estimate the actual number of modes. For example, in energy-based approaches, they can consist in adding a term penalizing the number of segments in the energy criterion, in order to reach a compromise between the segmentation length and the local energy of the modes. Now, the choice of the penalizing term and of its relative importance in the energy criterion is still user-dependent.

The decision parameter can also be inherent to the method. In [40], Frederix *et al.* try to fit the simplest density function compatible with the data. The repartition function F_n of the data being known, they look for the repartition function F which minimizes $\int (F''(x))^2 dx$, under the condition that F_n is compatible with F for a given statistical test of significance level α (they use in their paper the Kolmogorov-Smirnov and Cramer-von Mises tests). This method is globally quite convincing, but the choice of the data-compatibility threshold α is again not formalized, and only justified by experiments.

The limitations observed in the previous histogram thresholding methods and the advantage of Gestalt Theory (see Appendix B) for automatic detection have motivated the development of a new non-parametric approach for histogram segmentation. This method satisfies the following properties:

- it is parameter-free: given any 1D histogram, it automatically computes the number and the locations of its modes,
- it is robust: small variations in the histogram due to noisy data or due to the sampling procedure are not detected as modes,
- it is local, in the sense that it is able to detect as modes small concentrations of values.

The obtention of this method can be done from two different points of view. In a first approach, we consider an *a contrario* theory, in which the aim is the detection of meaningful modes and meaningful gaps against an *a contrario* uniform law. From this first step we obtain three algorithms, which segment adequately the histogram but yield under-segmentation. The second point of view is related to the problem of density estimation. A estimated density will be considered admissible if no meaningful gaps or modes can be detected on the histogram where the density is hypothesized. Thus, the second point of

view involves the same arguments as the first, but the detection method is used to accept or to reject an hypothesized density. The result of this second approach is the Fine to Coarse algorithm, that yields an admissible segmentation of the histogram.

The plan of the chapter is as follows. In the next section, we present the *a contrario* approach, defining the concepts of meaningful and maximal meaningful mode and gap, detailing the approach limitations and presenting the different results. In Section 2.3 the density estimation approach is explained in three steps; and, finally, some properties and new concepts are commented in Section 2.4.

2.2 An a contrario approach to histogram analysis

The approach that we present here is based on a method for the computation of locations of interest (modes or gaps) of a histogram, recently proposed in [30]. It is based on the idea that any event (either a geometric structure in an image or a mode in a histogram) is meaningful if it is very unlikely to happen under a given *a contrario* hypothesis. For a histogram, this hypothesis can be for example that the values of the data set follow a uniform law: in this sense, the fact that a high percentage of the values concentrate in a certain interval indicates that this interval is meaningful.

In the rest of the chapter, we will consider a discrete histogram $h = (h_i)_{i=1\dots L}$, with N samples on L bins $\{1, \dots L\}$. The number h_i is the value of h in the bin i. It follows that

$$\sum_{i=1}^{L} h_i = N.$$

For each discrete interval $[a, b]$ of $\{1, \dots L\}$, let $r(a, b)$ be the proportion of points in $[a, b]$,

$$r(a, b) = \frac{1}{N}\left(\sum_{i=a}^{b} h_i\right).$$

Definition 1. *Let h be a histogram on $\{1, ..., L\}$. A* ***segmentation*** *S of h is a partition $1 = s_1 < s_2 < \cdots < s_n = L$. The number $n - 1$ is termed* ***length****, or number of segments of S.*

2.2.1 Meaningful intervals and meaningful gaps of a histogram

Suppose a distribution function p, hypothesized over $\{1, ..., L\}$. As planed, we want to test the adequacy of the histogram h to the density p, where p is called the *a contrario* distribution. The null hypothesis is that the N samples are distributed independently on $\{1, ..., L\}$ with the fixed distribution law p. For each interval $[a, b]$ of $\{1, ..., L\}$, we denote $p(a, b)$ the probability for a point to be in that interval.

The rejection of the null hypothesis can be tested efficiently by considering all possible excess or deficit intervals for h with respect to p. Assume for instance that some interval $[a, b]$ has a density $r(a, b)$ larger than $p(a, b)$, is this excess meaningful or not? This can be evaluated with the *a contrario* probability that $[a, b]$ contains at least $Nr(a, b)$ points, among the N points sampled independently with law p, given by the binomial tail $\mathcal{B}(N, Nr(a, b), p(a, b))$, where

$$\mathcal{B}(n, k, p) = \sum_{j=k}^{n} \binom{n}{j} p^j (1 - p)^{n-j}.$$

Since, we perform multiple testing, a probability value is not adequate and we have to control the number of false alarms of an interval $[a, b]$ ([34]) defined by

$$\mathrm{NFA}([a,b]) = \frac{L(L+1)}{2}\mathcal{B}(N, Nr(a,b), p(a,b)).$$

Fixing a maximal value for the false alarms, ε, we say that an interval is interesting if

$$\mathrm{NFA}([a,b]) \leq \varepsilon.$$

With such a definition, the expectation of the number of interesting intervals in $\{1, \dots L\}$ under the null hypothesis, the law p, is smaller than ε ([34]).

Theorem 1. *Let R the random variable representing the exact number of interesting intervals in $\{1, \dots L\}$ under the null hypothesis, then*

$$E(R) \leq \varepsilon.$$

Before to give the proof, we recall without proof the following classical lemma,

Lemma 1. *Let X be a real random variable and be $H(X) = P(X \geq x)$. Then for all $t \in [0, 1]$*

$$P(H(X) < t) \leq t.$$

Proof: Let $m(L) = \frac{L(L+1)}{2}$ be the total number of intervals in $\{1, \dots L\}$. For $0 \leq i \leq m(L)$, let e_i be the following event: "the i-th interval is interesting" and χ_{e_i} denotes the characteristic function of the event e_i. We have

$$P[\chi_{e_i} = 1] = P[NFA(i) \leq \varepsilon],$$

where $NFA(i)$ is the number of false alarm of the i-th interval. Since $R = \sum_{i=1}^{m(L)} \chi_{e_i}$, the expectation of R is

$$E(R) = \sum_{i=1}^{m(L)} E(\chi_{e_i}) = \sum_{i=0}^{m(L)} P[\chi_{e_i} = 1] = \sum_{i=0}^{m(L)} P[NFA(i) \leq \varepsilon]$$

By definition we have

$$\begin{aligned} P[NFA(i) \leq \varepsilon] = & \; P\left[\tfrac{L(L+1)}{2}\mathcal{B}(N, Nr(i), p(i)) \leq \varepsilon\right] \\ = & \; P\left[\mathcal{B}(N, Nr(i), p(i)) \leq \tfrac{2\varepsilon}{L(L+1)}\right]. \end{aligned}$$

If we consider X as the random variable representing the number of points in an interval, then $\mathcal{B}(N, Nr(i), p(i)) = P(X \geq Nr(i))$. From Lemma 1, we have

$$P\left[\mathcal{B}(N, Nr(i), p(i)) \leq \frac{2\varepsilon}{L(L+1)}\right] \leq \frac{2\varepsilon}{L(L+1)}.$$

So that,

$$E(R) \leq \sum_{i=0}^{m(L)} \frac{2\varepsilon}{L(L+1)} = \varepsilon.$$

□

In the same way, we can consider that an interval $[a, b]$ is an interesting "gap" if it contains "fewer points" than the expected average in the sense that

$$\mathcal{B}(N, N(1 - r(a, b)), 1 - p(a, b)) < \frac{2\varepsilon}{L(L + 1)}.$$

In practice, we will always use $\varepsilon = 1$, which means that we want on the average at most one "false detection". Now, these binomial expressions are not always easy to compute, specially when N is large. The large deviation theory tells us (see [28] for example) that for $r > p$,

$$\frac{1}{N} \log \mathcal{B}(N, Nr, p) \xrightarrow[N\to\infty]{} -r \log \frac{r}{p} - (1 - r) \log \frac{1 - r}{1 - p}.$$

Moreover, Hoeffding's inequality ensures that $\mathcal{B}(N, Nr, p) \leq e^{-N\left(r \log \frac{r}{p} + (1-r) \log \frac{1-r}{1-p}\right)}$ for $r > p$. In practice, we adopt this large deviation estimate, which has been proven to be a good approximation to define meaningful intervals and gaps (see [32]).

Definition 2. *The* ***relative entropy*** $H_p([a, b])$ *of an interval* $[a, b]$ *related to the* a contrario *distribution* p *is defined by*

$$H_p([a, b]) = r(a, b) \log \frac{r(a, b)}{p(a, b)} + (1 - r(a, b)) \log \frac{1 - r(a, b)}{1 - p(a, b)}.$$

Remark: We note that $H_p([a, b])$ is the Kullback-Leibler distance between the two Bernoulli distributions of respective parameters $r(a, b)$ and $p(a, b)$ (see [21]).

Then, meaningful intervals and gaps can be defined by:

Definition 3. *An interval* $[a, b]$ *is said to be a* ***meaningful interval*** *(resp. a* ***meaningful gap****) if* $r(a, b) > p(a, b)$ *(resp.* $r(a, b) < p(a, b)$*) and if its relative entropy* $H_p([a, b])$ *is such that*

$$H_p([a, b]) \geq \frac{1}{N} \log \frac{L(L + 1)}{2}$$

By Hoeffding's inequality, an interval is interesting if it is meaningful and the inverse is, by large deviation, almost true when $N \to \infty$.

2.2.2 Maximal meaningful intervals and gaps

If an interval $[a, b]$ has been detected as meaningful and contains a lot of points, it is clear that larger intervals containing $[a, b]$, or smaller intervals contained in $[a, b]$, will also be counted as meaningful. The presence of an excess of points somewhere induces the existence of several meaningful intervals with regard to this zone. In this case, a natural economy principle is to only keep the best interval, which stands for all the others concerned. This leads to introduce the notions of "maximal meaningful interval" and "maximal meaningful gap". The proofs of the theorems of this part can be found in A. Desolneux's PhD Thesis [30].

Definition 4. *An interval* $I = [a, b]$ *is a* ***maximal meaningful interval*** *(resp.* ***maximal meaningful gap****) if it is a meaningful interval (resp. meaningful gap) and if*

$$\forall J \subset I, \qquad H_p(J) \leq H_p(I) \qquad \text{and} \qquad \forall J \supsetneq I, \qquad H_p(J) < H_p(I).$$

One of the main interest of the previous definition is that maximal intervals and gaps follow a natural exclusion principle: two maximal intervals (resp. two maximal gaps) never intersect.

Theorem 2. *Let I_1 and I_2 be two meaningful intervals (or two meaningful gaps) such that $I_1 \cap I_2 \neq \emptyset$. Then*

$$\max(H_p(I_1 \cap I_2), H_p(I_1 \cup I_2)) \geq \min(H_p(I_1), H_p(I_2))$$

and the inequality is strict when $I_1 \cap I_2 \neq I_1$ and $I_1 \cap I_2 \neq I_2$.

The main consequence of this theorem is the aforesaid exclusion principle:

Proposition 1. *Let I_1 and I_2 be two different maximal meaningful intervals (or two different meaningful gaps). Then*

$$I_1 \cap I_2 = \emptyset.$$

The maximality implies other properties which will be useful later. The following theorem concerns the structure of maximal intervals and gaps.

Theorem 3. *Let h be a histogram defined on a finite set of values $\{1, ..., L\}$. If $[a, b]$ is a maximal meaningful interval such that $1 < a < b < L$, then*

$$r(a-1) < r(a) \quad \text{and} \quad r(b+1) < r(b), \quad r(a) > r(b+1) \quad \text{and} \quad r(b) > r(a-1).$$

If $[a, b]$ is a maximal meaningful gap such that $1 < a < b < L$, then

$$r(a-1) > r(a) \quad \text{and} \quad r(b+1) > r(b), \quad r(a) < r(b+1) \quad \text{and} \quad r(b) < r(a-1).$$

In the case of maximal meaningful intervals (resp. gaps), the first assertion means that the histogram is increasing (resp. decreasing) at its left bound, and decreasing (resp. increasing) at its right bound. The second assertion implies that the research of maximal intervals and gaps can be restricted to intervals of the level sets of the histogram.

For example, we can see on Figure 2.1 the maximal meaningful intervals and gaps of a given histogram when we choose for p the uniform law. This experiment shows that the uniform *a contrario* hypothesis leads to a subsegmentation: it only permits a rough first sketch. This histogram is constituted of three modes. Two of them are predominant and contiguous and the third one is isolated. If we take a rough model like the uniform hypothesis, only the larger mode is detected and the smaller one is obscured by the others, since it does not contain enough points to be globally meaningful.

All of these definitions and properties can be extended to the case of a continuous histogram, *i.e.* a density function f on $[0, 1]$ such that $\int_0^1 f = 1$. We will not develop this formalism here, but it can be useful and it can be found in [30].

2.2.3 Limits of the meaningful gap and meaningful interval notions

The notions of meaningful interval and gap give us a strong indication of the global deviations from a given hypothesis. They tell us whether there are intervals with a meaningful excess or a meaningful lack of points, and where they are. Now, these indications do not give a segmentation of the histogram. Moreover, the meaningfulness of an interval

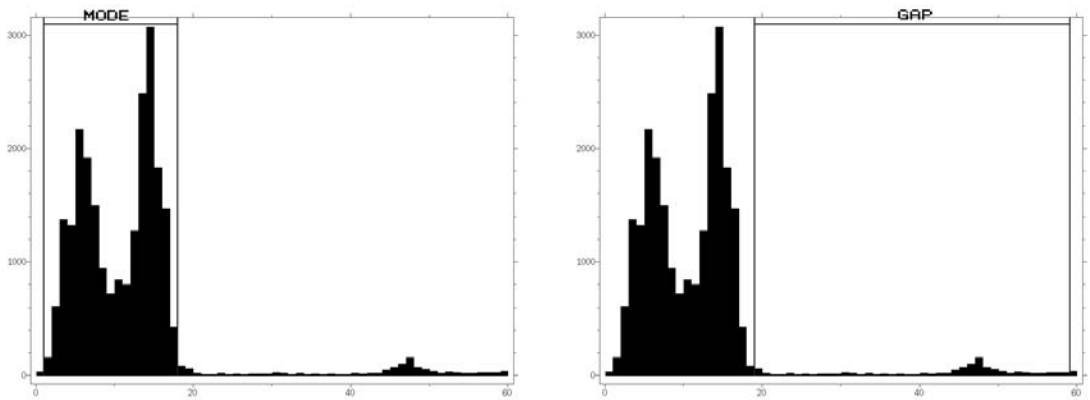


Figure 2.1: *Histogram of* $N = 240000$ *samples distributed on* $L = 60$ *bins. Left: maximal meaningful interval* $[3, 18]$ *of the histogram for the uniform law* p*. Right: maximal meaningful gap* $[19, 60]$ *of the same histogram.*

or gap depends of the total histogram, and this property has two main drawbacks. First, the width of a maximal meaningful interval or gap depends on the rest of the histogram. Secondly, a small mode is obviously not detected as meaningful if there are too many points somewhere else in the histogram.

The first of these drawbacks can be expressed more precisely. If there is a maximal meaningful interval (or gap) somewhere, the more the total weight of the histogram grows outside of the gap, the more the maximal meaningful gap will be large. On the contrary, a maximal meaningful interval becomes narrower when the total weight of the histogram grows outside of it. Next proposition ([30]) states this fact.

Proposition 2. *Let* $[c, d]$ *be an interval, and* $[a, b] \subset [c, d]$ *be a maximal meaningful interval such that* $1 \leq c < a < b < d \leq L$*. Assume also that there is no other maximal interval in* $[c, d]$*. If the proportion of points inside* $[c, d]$ *decreases, and if there is still only one maximal interval* $[a', b']$ *inside* $[c, d]$*, then* $[a', b'] \subset [a, b]$*. The inclusion becomes strict if the proportion of points decreases enough.*

Consequently, the exact bounds of a maximal interval are not a robust information, and they just indicate reliably the location where we can find the most meaningful "excess" of points. Now, when a mode is detected as meaningful, we would like to detect "the whole mode" with precise limits, and not just part of it. More precisely, we would like to find the "optimal separators" between the modes of the histogram, because the information given by the maximal intervals and gaps is not directly exploitable to create such a segmentation.

The other drawback already announced is that some modes, even if they seem concentrated, isolated and well defined, cannot be detected if they are too small. However, in the middle of an almost empty interval, such a mode can clearly be locally considered as a positive deviation from randomness. If we consider the histogram of Figure 2.1, this histogram is quite clearly constituted of three modes. Two are predominant and contiguous and the third is isolated. If we take a rough model like the *a contrario* uniform hypothesis, then only the big mode is detected and the small mode is put in the shade by the others: it does not contain enough points to be globally meaningful. Moreover, in the gap complementary to the two continous modes, this little mode is locally very meaningful. We distinguish it because it represents a concentrated group of points in a narrow zone, in the middle of an almost empty interval. In order to find it, our research must be refined, from global to local.

2.2.4 Direct segmentation using a generalized entropy functional

A direct generalization of the previous theory, inspired by the energy minimizing methods (see [14] for example), would be to segment our histogram by maximizing the global entropy functional

$$\begin{aligned} H_{p_i}(a_1, b_1, ..., a_n, b_n) &= r_1 \log(\frac{r_1}{p_1}) + r_2 \log(\frac{r_2}{p_2}) + ... + r_n \log(\frac{r_n}{p_n}) + \\ &\quad +(1 - r_1 - ... - r_n) \log(\frac{1 - r_1 - ... - r_n}{1 - p_1 - ... - p_n}) \end{aligned}$$

where r_i is the proportion of points contained in $[a_i, b_i]$ and p_i the measure of this interval.

A specific point has taken us away from this method: the number of segments has to be introduced as a variable in the functional. Now, when the number of intervals n increases, the entropy always increases (this is a consequence of the log sum inequality, cf. [21]). The values taken by the functional are not comparable when the number of segments changes. This drawback can be avoided if we minimize the number of false alarms of the segmentation

$$\mathrm{NFA}(a_1, b_1, ..., a_n, b_n) = C_{L+n}^{2n} e^{-NH_p(a_1, b_1, ..., a_n, b_n)},$$

where C_{L+n}^{2n} is the number of choices of n non-empty and disjoint intervals in $\{1, ...L\}$. However, it seems clear that in the case of the histogram of Figure 2.1 with a priori distribution p taken to be uniform, such a functional will prefer to create a huge gap in the second part of the histogram, instead of distinguishing a little mode in the middle of it.

All of these observations lead to elaborate a coarse to fine method. The global results of meaningfulness, because of their importance and reliability, and in spite of their lack of precision, can be used as starting points.

2.2.5 Optimal separators

Our first step has been to take interest in the maximal meaningful gaps and modes obtained previously and to try to refine them. For that, we established two algorithms. One considers the maximal meaningful modes in the histogram, called *Recursive Mode (RM)*, and the other considers the maximal meaningful gaps, called *Recursive Gap (RG)*. The structure of these algorithms is very similar and the comparison of their results will allow us to define the best "tool" for the histogram segmentation. In all that follows the a priori distribution p on each interval is assumed to be uniform. The algorithms are:

Recursive Mode (RM):

1. *Detect the list M=$\{I_1, I_2, ... \}$ of maximal meaningful modes of the histogram.*
2. *Repeat:*

   For each i, if I_i contains relative meaningful mode, replace I_i in M by the list of the maximal ones.
   Stop when there are no more modes detected for every i.

Recursive Gap (RG):

1. *Detect the list G=$\{I_1, I_2, ... \}$ of maximal meaningful gaps of the histogram.*

2. *Repeat:*

   For each i, if I_i contains relative meaningful gaps, replace I_i in G by the list of the maximal ones.
   Stop when there are no more gap detected for every i.

Comparing the results, we can conclude that the maximal meaningful gaps algorithm obtains a better histogram segmentation. In Figure 2.2 we can observe a example of this fact. The application of the RG algorithm segments the histogram in three intervals, the two gaps in the histograms are considered as maximal meaningful. On the other hand, the result of the RM algorithm obtains only two modes, because the second part of the histogram, after the first gap, is considered as a maximal meaningful mode and is not separated.

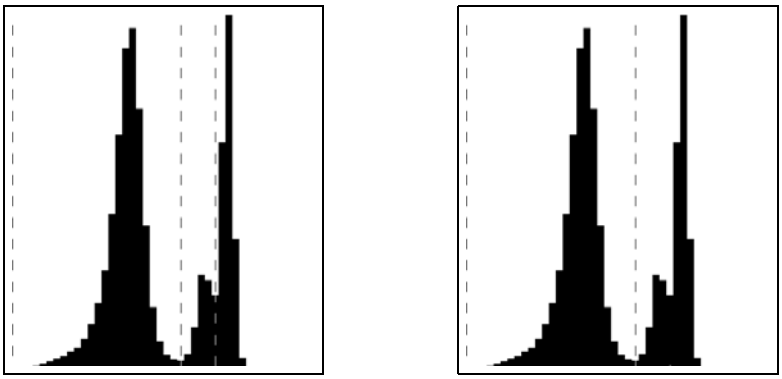


Figure 2.2: *Left: Result of the RG algorithm Right: Result of the RM algorithm.*

One of the main qualities of the RG algorithm is that it converges obviously towards a union of intervals. In fact, at each step, the total length of the gaps decreases. In the discrete case, the number of points being finite, it is clear that the algorithm converges in a finite number of steps. The limit is a finite union of disjoint intervals $[a_1, b_1] \cup [a_2, b_2] \cup ... \cup [a_n, b_n]$. Each of them contains no relative gap, and is contained in one of the maximal gaps of the beginning. Furthermore, each one of these limit gaps has been detected as a maximal gap of a previous interval. Hence its bounds are part of the same level set of the histogram. The connected parts of their complementary in $\{1, 2..., L\}$ can be considered as modes of the histogram.

However, it is clear that even if these separators are justified, they are not sufficient. Indeed, they are all contained in the first maximal gaps. Now, we can easily imagine cases where modes are clearly separated without the presence of a maximal gap between them. This is the case of the histogram of Figure 2.1, where in the first big mode we can distinguish two modes, but they are not separated by a maximal gap. In order to find those separators, we added a refinement to the RG algorithm, which we call *Gap Scale Space (GSS) algorithm*:

Gap Scale Space (GSS):

1. Detect the list $\{I_1,\ I_2,...\ \}$ of maximal meaningful gaps of the histogram. The connected parts which remain are called "modes" and form a second list $\{J_1, J_2, \ldots\}$.

2. Repeat :

   For each i, detect the maximal meaningful gaps of I_i. For each k, detect the maximal meaningful gaps of J_k. Update the list of gaps and the list of modes.
   Stop when there are no more gaps detected.

The obtained gaps contain no relative gaps, they are almost flat, without fluctuation. Consequently, their points have no reason to be attributed to the left or the right mode. In practice, we choose as a convention to split the gap in the middle and to attribute the points of the gap to the left or the right mode according to their position with respect to this point.

Definition 5. *We call* ***optimal separators*** *the middle points of the limit intervals obtained by the RG or GSS algorithm.*

For example, we can see on Figure 2.3 the limit gaps and optimal separators obtained for the histogram used previously.

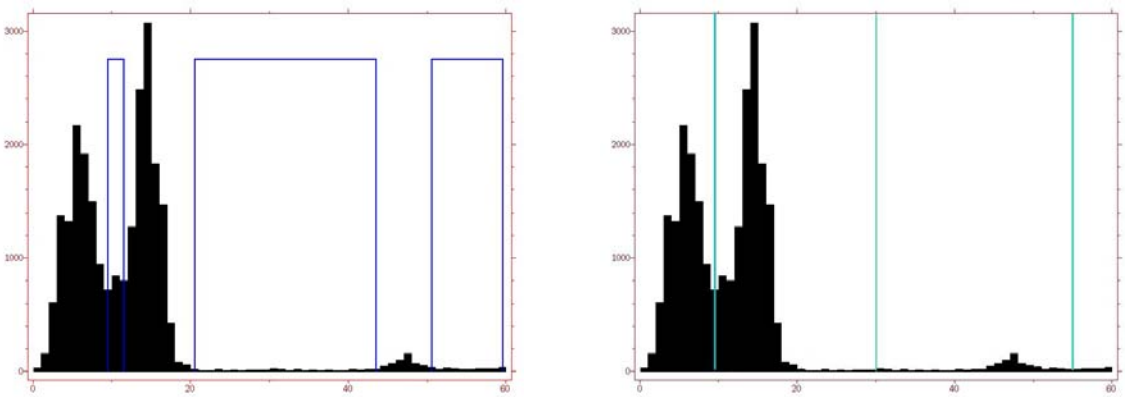


Figure 2.3: *Left: limit gaps obtained by the GSS algorithm for the histogram of Figure 2.1. Right: Optimal separators of the same histogram.*

The GSS algorithm finds always more divisions than the previous one, because the gaps found by the RG algorithm are obviously all found here. For example, in the histogram of Figure 2.1, using the GSS algorithm, we find a additional separator in the middle of the big mode (see Figure 2.3).

Even though the GSS algorithm seems to provide an optimal segmentation, we want to obtain a simpler method. It seems clear from the complexity of GSS that we have reached a limit in the use of uniform *a contrario* distribution. GSS involves multiple *a contrario* uniform densities depending on the considered interval. In the next section, instead of detecting gaps and modes *a contrario*, we shall reverse the point of view. We wish to estimate directly the underlying density for the histogram. A segmentation will be defined as any underlying density with a finite number of modes, as small as possible, such that no gap or mode can be found when it is taken as *a contrario* assumption. Thus, we shall use the same detection tools as above for a quite different aim. We pass from "detection against an *a contrario* law" to "confirmation of the *a contrario* law".

2.3 Histogram segmentation as density estimate

A density f is said to be **unimodal** on some interval $[a, b]$ if f is increasing on some $[a, c]$ and decreasing on $[c, b]$. In a color image, the pixel hues of a given object are naturally distributed according to a unimodal law around the average hue of the object. It seems appropriate to segment a histogram by looking for segments on which it is "likely" that the histogram is the realization of a unimodal law. On such intervals we will say that the histogram is "statistically unimodal" (this expression will be precisely defined later).

Obviously, such a segmentation is generally not unique. In particular the segmentation defined by all the local minima of the histogram has this property. However, small variations due to the sampling procedure should clearly not be detected as modes. In order to get a "minimal" division of the histogram, these fluctuations should be neglected. We arrive at two requirements for an admissible segmentation:

- in each segment, the histogram is "statistically unimodal",
- there is no union of several consecutive segments on which the histogram is "statistically unimodal".

What are the right tests to decide whether a histogram is "statistically unimodal" on an interval or not? In a non parametric setting, any unimodal density on the considered interval should be hypothesized and the compatibility between this density and the observed histogram distribution should be tested. Unfortunately, this leads to a huge number of tests and this is therefore impossible. There is, however, a way to address this question by testing a small number of adequate unimodal laws. In [45], this problem was solved for the case of decreasing laws. Our purpose here is to extend this method to the segmentation of any histogram into meaningful modes. We shall treat the problem in three stages in the next three subsections:

- Step A: testing a histogram against a fixed hypothesized density
- Step B: testing a histogram against a qualitative assumption (decreasing, increasing)
- Step C: segmenting a histogram and generating an estimate of its underlying density.

2.3.1 Adequacy to a given distribution

Consider the hypothesis $\mathcal{H}_0$ that the histogram h is originated from the law p. In other words, the N samples of the histogram h have been sampled independently on $\{1, \dots L\}$ with law p. A simple way to accept or to reject $\mathcal{H}_0$ is to test for each interval $[a, b]$ the similarity between $r(a, b)$ and $p(a, b)$. This similarity can be tested using the meaningful gaps and intervals as confirmation tools.

Definition 6. *An interval $[a, b]$ is said to be an ε-**meaningful rejection** of $\mathcal{H}_0$ if*

$$H_p([a, b]) \geq \frac{1}{N} \log \frac{L(L+1)}{\varepsilon}$$

Proposition 3. *Under the hypothesis $\mathcal{H}_0$, the expectation of the number of ε-meaningful rejections among all the intervals of $\{1, \dots L\}$ is smaller than ε.*

The proof of Proposition 3 is given in [34] and uses a Bonferroni argument, taking into account the number of tests $\frac{L(L+1)}{2}$ (the number of different intervals in $\{1, \dots L\}$). This means that testing a histogram h following a law p will lead on the average to less than ε wrong rejections. It may be asked how reliable this estimate is. In [46], Grompone and Jakubowicz have shown that the expectation of ε-meaningful events could be approximated by $\varepsilon/100$. As before, in practice, we fix $\varepsilon = 1$ and just talk about meanigful rejections.

Definition 7. *We say that a histogram h* **follows the law** *p on $[1, L]$ if h contains no meaningful rejection for $\mathcal{H}_0$.*

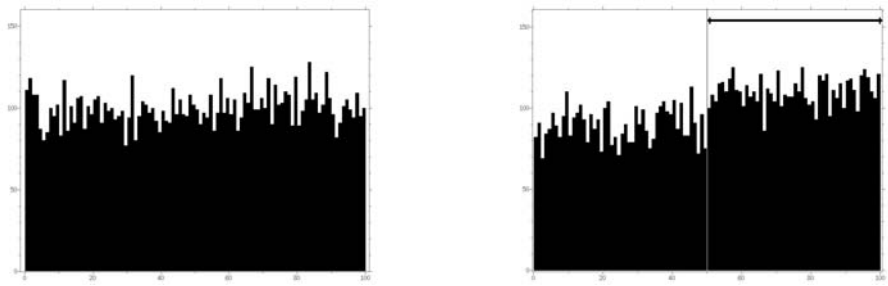


Figure 2.4: *Histograms of* $N = 10000$ *samples distributed on* $L = 100$ *bins, tested against the unifom law on* $[1, 100]$*. Left: The first histogram is a realization of the uniform law on* $[1, 100]$*. No meaningful rejection is found. Right: The second histogram is a realization of a mixture of two uniform laws, one on* $[1, 50]$ *with a weight* 0.45*, and another on* $[51, 100]$ *with weight* 0.55*. Several rejections of the uniform law on* $[1, 100]$ *are found. The rejection with the lowest* NFA_p *(the interval* $[50, 100]$*) is shown here.*

Figure 2.4 shows two histograms which have been tested against the uniform law on $[1, 100]$. The first one is a realization of this law, and no rejection is found. The second is a mixture of two uniform laws on different intervals. In this case, the uniform law on $[1, 100]$ is rejected.

2.3.2 Testing the monotone hypothesis

In the previous subsection, we explained how to detect meaningful rejections against a given law p. Now, what can we do if we know that the observed objects follow a decreasing (or increasing) distribution? For example, we know from the work of Yann Gousseau ([43]) that the area distribution of "objects" in an image, defined as the connected components of bilevelsets, is very close to the function $\frac{C}{x^\alpha}$, where α is a real number around 2. In this kind of case, it is absurd to look for the deviations from the uniform hypothesis. We would always "discover" that there is a large quantity of small objects in the images, and a few big ones. This monotonicity information being known, we still want to detect groups of objects which contradict it. Thus, let us define the meaningfulness of a rejection against the decreasing hypothesis (the definitions and results for the increasing hypothesis can be deduced by symmetry). This definition will be useful later in order to give a suitable meaning to the expression "being statistically unimodal on an interval". We will call $\mathcal{D}(L)$ the space of all decreasing densities on $\{1, 2, ..., L\}$ and $\mathcal{P}(L)$ the space of probability distributions on $\{1, 2, ..., L\}$, *i.e.* the vectors $r = (r_1, ..., r_L)$ such that:

$$\forall i \in \{1, 2, ..., L\}, \quad r_i \geq 0 \quad \text{and} \quad \sum_{i=1}^{L} r_i = 1.$$

If $r \in \mathcal{P}(L)$ is the normalized histogram of our observations, we need to estimate the density $p \in \mathcal{D}(L)$ with respect to which the empirical distribution has the "less meaningful" rejections. This can be summed up by the optimization problem

$$\tilde{r} = \arg\min_{p \in \mathcal{D}(L)} \max_{[a,b] \in \{1,2,...,L\}} H_p([a, b]), \tag{2.1}$$

where H_p is the entropy function defined before. In other words, $\tilde{r}$ is the decreasing distribution, which gives rejections with minimal NFA. If there is any meaningful rejection

for $\tilde{r}$, there will be meaningful rejections for every other hypothezised decreasing distribution. The meaningfulness of rejections can then be defined relatively to this distribution $\tilde{r}$. Note that the uniform distribution is a particular case of decreasing density, which means that this formulation strengthens the previous theory: if there is no meaningful rejections against the uniform hypothesis, then there will be no meaningful rejections against the decreasing hypothesis.

However, the optimization problem (2.1) is not easy to solve. We chose to simplify it by approximating $\tilde{r}$ by the Grenander estimator $\bar{r}$ of r. If $r \in \mathcal{P}(L)$ is the normalized histogram of our observations, let $\bar{r}$ be the Grenander estimator of r. Introduced by Grenander in 1956 ([45]), this estimator is defined as the non-parametric maximum likelihood estimator restricted to decreasing densities on the line.

Definition 8. *The histogram $\bar{r}$ is the unique histogram which achieves the minimal Kullback-Leibler distance from r to $\mathcal{D}(L)$,* i.e.

$$KL(r||\overline{r}) = \min_{p \in \mathcal{D}(L)} KL(r||p),$$

where $\forall p \in \mathcal{D}(L)$, $KL(r||p) = \sum_{i=1}^{L} r_i \log \frac{r_i}{p_i}$.

Grenander shows in [45] (see also [6]) that $\bar{r}$ is merely "the slope of the smallest concave majorant function of the empirical repartition function of r". The estimator $\bar{r}$ also achieves the minimal L^2-distance from r to $\mathcal{D}(L)$. It can easily be derived from r by an algorithm called "Pool Adjacent Violators" (see [5, 8]). The procedure is: if r is increasing on successive bins, replace its value by the mean value on these bins; repeat the procedure. This algorithm, more precisely described below, leads to a unique decreasing step function $\bar{r}$.

Pool Adjacent Violators

Let $r = (r_1, ..., r_L) \in \mathcal{P}(L)$ be a normalized histogram. For $r \in \mathcal{P}(L)$, and for each interval $[i, j]$ on which r is increasing, i.e. *$r_i \leq r_{i+1} \leq ... \leq r_j$ and $r_{i-1} > r_i$ and $r_{j+1} < r_j$, consider the operator $D : \mathcal{P}(L) \to \mathcal{P}(L)$ defined by*

$$D(r)_k = \begin{cases} \frac{r_i + ... + r_j}{j-i+1} & \text{for } k \in [i, j] \\ r_k & \text{otherwise.} \end{cases}$$

This operator D replaces each increasing part of r by a constant value (equal to the mean value on the interval). After a finite number (less than the size L of r) of iterations of D we obtain a decreasing distribution denoted $\overline{r}$:

$$\overline{r} = D^L(r).$$

An example of histogram and its Grenander estimator is shown on Figure 2.5, where we can see the transformation of the observed histogram into a monotone one.

The previous definitions of meaningful rejections can obviously be applied to this case by taking $p = \bar{r}$ in the hypothesis $\mathcal{H}_0$, with $\bar{r}$ the Grenander estimator of $r = \frac{1}{N}h$.

Definition 9. *Let h be a histogram of N samples and $\overline{r}$ the Grenander estimator of $r = \frac{1}{N}h$. An interval $[a, b]$ is said to be a* ***meaningful rejection for the decreasing hypothesis*** *if*

$$H_{\overline{r}}([a, b]) \geq \frac{1}{N} \log \left(L(L+1)\right)$$

where $H_{\overline{r}}([a, b])$ is defined for any density law p in Definition 2.

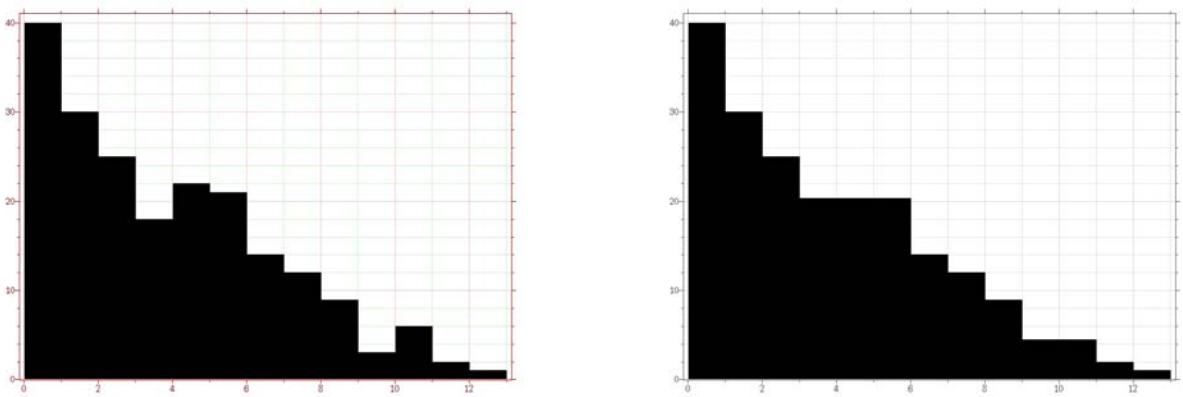


Figure 2.5: *The right histogram is the Grenander estimator of the left histogram, computed by the "Pool Adjacent Violators" algorithm.*

We can note that if a histogram r is an empirical trial of the uniform hypothesis, its Grenander estimator will be almost flat, and the previous test will be almost equivalent to the uniform case test.

Definition 10. *We say that a histogram h* **follows the decreasing hypothesis** (*resp. the increasing hypothesis*) *on an interval $[a, b]$ if the restriction of the histogram to $[a, b]$* (i.e. $h_{|[a,b]} = (h_a, h_{a+1}, \dots h_b)$) *contains no meaningful rejection for the decreasing (resp. increasing) hypothesis.*

2.3.3 Piecewise unimodal segmentation of a histogram

Definition 11. *We say that a histogram h* ***follows the unimodal hypothesis*** *on the interval $[a, b]$ if there exists $c \in [a, b]$ such that h follows the increasing hypothesis on $[a, c]$ and h follows the decreasing hypothesis on $[c, b]$.*

This notion defines precisely what we called in the introduction "to be statistically unimodal". Such a segmentation exists. Indeed, the segmentation defined by all the minima of the histogram as separators follows obviously the unimodal hypothesis on each segment. Yet if there are small fluctuations it is clear that it is not a reasonable segmentation (see the left part of Figure 2.6). Fortunately, a segmentation following the unimodal hypothesis on each segment is generally not unique. We can build one by refining the RG method: once the first separators are created by the Recursive Gap algorithm, we only have to check that the histogram, between two separators, follows the unimodal hypothesis. If it doesn't, we split the histogram at the location of the more meaningful of the maximal meaningful gaps and we repeat the procedure. This procedure gives the following coarse to fine algorithm:

Coarse to Fine (CTF) Segmentation Algorithm:

1. *Apply the RG algorithm to the histogram. Let $1 = s_1 < s_2 < ... < s_n = L$ be the limit list S of the obtained separators.*

2. *Repeat:*

   For each i, check if $[s_i, s_{i+1}]$ follows the unimodal hypothesis. If not, let m_i be the point of $[s_i, s_{i+1}]$ where the histogram is maximal, apply RG with respect to the increasing (resp. decreasing) hypothesis on $[s_i, m_i]$ (resp. $[m_i, s_{i+1}]$) and define new

separators. Update S.
Stop when the unimodal hypothesis is followed on each of the intervals of the segmentation.

This procedure ensures that we find a segmentation much more reasonable than the segmentation constituted of all the minima. Indeed, if a fluctuation is not meaningful, no new separator will be created. However, in a way, we would like to be sure to build a minimal segmentation, in terms of numbers of separators. Moreover, this algorithm involves concepts from two different approaches (*a contrario* analysis and density estimation), therefore it is not consistent. We need a new algorithm considering the tools as confirmation of the a contrario law only. The obtaining of this algorithm leads us to introduce the notion of "admissible segmentation".

Definition 12. *Let h be a histogram on $\{1, ..., L\}$. We will say that a segmentation $S = \{s_1, \ldots, s_n\}$ of h is* ***admissible*** *if it satisfies the following properties:*

- *h follows the unimodal hypothesis on each interval $[s_i, s_{i+1}]$,*
- *there is no interval $[s_i, s_j]$ with $j > i+1$, on which h follows the unimodal hypothesis.*

The first requirement avoids under-segmentations and the last one is imposed in order to avoid over-segmentations. It is easy to see in the discrete case that such a segmentation exists. From a limit segmentation containing all the minima of h, we can merge recursively the consecutive intervals until both properties are satisfied. This is the principle used in the algorithm presented underneath, called "fine to coarse segmentation algorithm". Now, there is no reason for such a segmentation to be unique. We can easily imagine some symmetric case where several admissible segmentations would coexist.

Fine to Coarse (FTC) Segmentation Algorithm:

1. *Define the initial segmentation, $S = \{s_1, \ldots, s_n\}$, as the finest segmentation given by the list of all the minima, plus the endpoints 1 and L of the histogram.*
2. *Repeat:*

   Choose i randomly in $[2, length(S)]$. If the segments on both sides of s_i can be merged in a single interval $[s_{i-1}, s_{i+1}]$ following the unimodal hypothesis, group them. Update S.

   Stop when no more pair of successive intervals follow the unimodal hypothesis.
3. *Repeat step 2 with the unions of j segments, j going from 3 to $length(S)$.*

The random initialization is obviously not very satisfactory from a theoretical point of view. Thus, the previous description of the proposed algorithm is correct, but it is not optimum and it is not robust. A better solution is to merge successive intervals in a specific order, starting with the intervals where the histogram best follows the unimodal hypothesis. The final algorithm is:

Fine to Coarse (FTC) Segmentation Algorithm:

1. *Define the initial segmentation, $S = \{s_1, \ldots, s_n\}$, as the finest segmentation given by the list of all the minima, plus the endpoints 1 and L of the histogram.*
2. *Repeat:*

(a) *For each* i, $i = 1, \ldots, n-1$, *compute the cost of the union of* s_i *with its successive interval, that is, compute the value*

$$C_i = C(s_i \cup s_{i+1}) = \min_{[a,b] \in s_i \cup s_{i+1}} NH_{\overline{r}}([a,b]) - \log\left(L(L+1)\right),$$

defined as the lowest rejection against the unimodal hypothesis on their union. Obtain the list $L = \{C_1, \ldots, C_{n-1}\}$.

(b) *Sort the list of costs* L, *in decreasing order.*

(c) *Assume that the higher cost is* C_j. *If* C_j *is higher than 0, then merge the intervals* s_j *and* s_{j+1}.

(d) *Update* S.

Stop when no more pair of successive intervals follows the unimodal hypothesis, that is, when the higher cost is lower than 1.

3. *Repeat step 2 with the unions of* j *segments,* j *going from* 3 *to* $length(S)$.

Note that the algorithm is much more coherent, and we are sure to always obtain the same segmentation for a given histogram (it was not the case with the 'random merging', even if the uniqueness was satisfied most of the time...). The result of this algorithm on the histogram of Figure 2.1 is shown on Figure 2.6. It presents small oscillations which create all the local minima that we can observe on the left part of the figure. The final result is shown on the right; only three modes are detected.

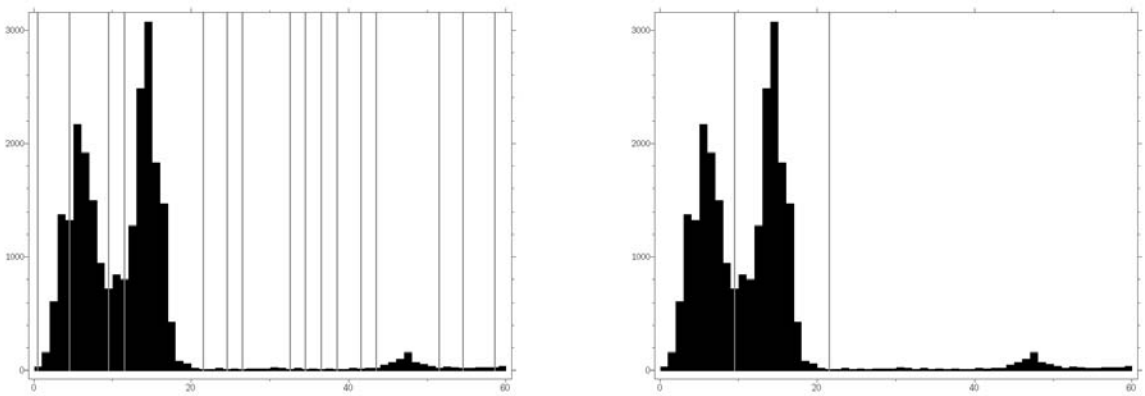


Figure 2.6: *Left: initialization of the algorithm from all the minima of the histogram. The histogram presents small oscillations, which create several local minima. Right: remaining minima after the fine to coarse algorithm.*

If we use an energy minimizing algorithm on the histogram of Figure 2.6, for example the one presented in [2], we find approximately the same segmentation if we specify three segments "a priori". The separator between the second and third modes is not located exactly at the same place, but this variation has a negligible effect on the classification, since very few points are represented in this zone of the histogram. If we ask for only two segments, the second and third modes are merged. It is interesting to note that for the energy defined in [2], the bimodal segmentation has almost the same energy as the three-modal segmentation. This implies that if we add a term penalizing the number of segments in the energy, the bimodal segmentation will certainly be chosen instead of the

three-modal one. Therefore, the small mode will never be found by this kind of method, even if it appears as very meaningful in the middle of a large and almost empty interval.

Figure 2.7 shows the result of the FTC algorithm on a more oscillation histogram. Let us remark that the proposed method also provides an estimation of the underlying density of the data (right of Figure 2.7). This density is computed by replacing the original histogram by its best unimodal fit on each interval of the segmentation. The fact that the observed histogram on the left follows a seven mode density has been checked by NFA and crossvalidated by a Chi-Square and Kolmogorov-Smirnov test (see Appendix A).

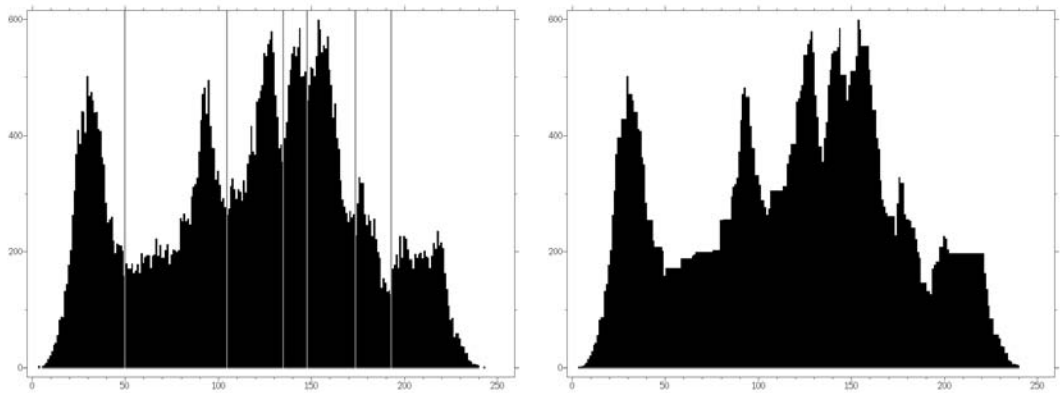


Figure 2.7: *Left: result of FTC algorithm on a very oscillating histogram. In the initialization, the histogram presented 60 local minima among 256 bins. The segments were merged as much as possible in order to follow the Definition 12 of an admissible segmentation. Right: density estimation corresponding to the computed segmentation (best unimodal fit on each interval). The estimated multimodal density has seven modes.*

2.4 Properties and comments

2.4.1 Detection properties

The following properties give us indications about how the above theory reacts in very simple situations. Assume that the histogram is roughly constituted of two modes, a small one in $[a, b]$ and a large one in $[b, c]$. This is often the case with intensity histograms of written documents, see Section 3.4. In this case, we can expect that the larger the big mode is in comparison to the small one, the less this theory will separate them. Indeed, the theory separates them if the small mode cannot be statistically considered as a "piece of the tail" of the large mode. Assume that the total number of points in the large mode increases, and that the number of points in the small mode does not change. Assume also that the Grenander estimator remains the same on $[a, b]$. The number of points in the whole interval $[a, c]$ is then multiplied by a factor λ, with $\lambda > 1$. Since the number of points does not change in $[a, b]$, this implies that for every interval I contained in $[a, b]$, $r(I)$ and $p(I)$ are divided by λ, and the number of false alarms of I becomes

$$\frac{L(L+1)}{2} e^{-\lambda N H(r(I)/\lambda, p(I)/\lambda)}.$$

The next proposition shows that the meaningfulness of those intervals decreases (the NFA increases) when λ increases.

Proposition 4. *The function* $f : \lambda \to \lambda H(\frac{r}{\lambda}, \frac{p}{\lambda})$ *is decreasing on* $[1, +\infty[$.

Proof: Follows immediately from the fact that $\log x \leq x - 1$ on $\mathbb{R}_+^*$. □

Now, assume on the contrary that the proportion of points in both modes remains the same, when the total number of points increases, multiplied by λ. Then, the NFA of any interval I becomes

$$\frac{L(L+1)}{2} e^{-\lambda N H(r(I), p(I))}$$

and this function decreases with λ. Thus, we obtain:

Property (P1): The tendency to separate the two modes increases with λ.

We will interpret these two monotonicity properties for text segmentation in the next chapter.

2.4.2 Flat zones

The output of the proposed FTC algorithm is an admissible segmentation of a histogram. Such a segmentation tells us how to split a histogram in modes. However, we can be interested in only detecting the "meaningful peaks" in the histogram. In this case, it can be useful to know precisely what are the uncertain zones between these modes, specially when the modes are very concentrated and well separated from each other. In this perspective, we define the "flat zones" of an admissible segmentation. Let $S = \{s_1, \ldots, s_n\}$ be an admissible segmentation of the histogram h on $\{1, \ldots L\}$.

Definition 13. *For each separator* s_i*, we call* **flat zone** *the largest neighborhood* I_i *of* s_i *in* $\{1, \ldots L\}$*, such that for all* x *in* I_i*, the segmentation obtained by replacing* s_i *by* x *in* S *is still admissible.*

The Figure 2.8 shows the flat zones of a histogram. In such a zone, we are unable to decide wether the points belong to the mode on the left, to the mode on the right, or to a background noise. We can see them as uncertainty zones. The definition of these zones allows to tighten the bounds around each mode. This can be very useful for instance if we want to compute precisely the means or standard deviations of the modes.

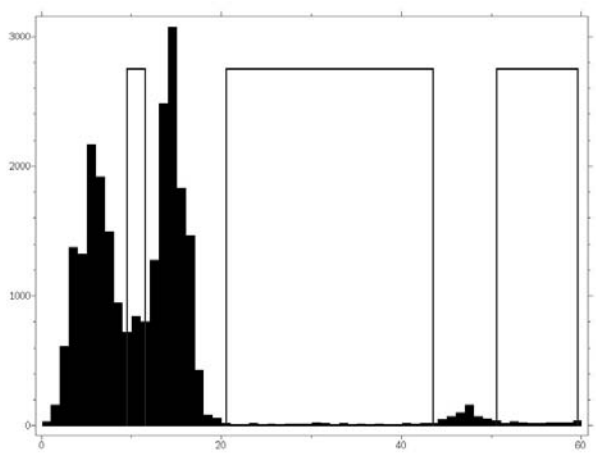


Figure 2.8: *Flat zones of a histogram, where the flat zones are represented by the white rectangles. The flat zones represent the uncertainty zones between the modes. We cannot tell wether their points belong to the left or to the right mode.*

2.4.3 Trinomial

We can consider that the monotone hypothesis is not enough discriminating, and we can propose a new vision of this hypothesis. Then, we define a decreasing hypothesis violation as a excess of points followed by a deficit.

Let $[a, b]$ be an interval and we denote by I_1 the excess interval, I_3 the deficit interval and I_2 the complementary interval, that is, $I_2 = [a, b] - I_1 - I_3$. In the same way, we denote r_i the density of the interval I_i, $i = 1, 2, 3$. In this case, the null hypothesis is given by the trinomial tail and we can define the number of false alarms as:

$$\text{NFA} = \frac{1}{2}\frac{N(N-1)(N-2)}{6}\mathcal{B}(N, k_1, k_2, p_1, p_2),$$

where k_1 and k_2 are the amount of points in the interval I_1 and I_2, respectively, and p_1 and p_2 are the value of the *a contrario* distribution on the same intervals. We can prove that the large deviation theory applies in the trinomial case, too. In the next proposition, we give an upper bound for the trinomial expression.

Proposition 5. *Let N, k_1 and k_2 be three positive integers such that $k_1 + k_2 \leq N$. Let p_1 and p_2 be two positive numbers such that $p_1 + p_2 \leq 1$. If $k_1 > Np_1$ and $k_2 > Np_2$, then*

$$\mathcal{B}(N, k_1, k_2, p_1, p_2) \leq e^{-N\, H\left(\frac{k_1}{N}, \frac{k_2}{N}, p_1, p_2\right)},$$

where

$$H(r_1, r_2, p_1, p_2) = r_1 \log \frac{r_1}{p_1} + r_2 \log \frac{r_2}{p_2} + (1 - r_1 - r_2) \log \frac{1 - r_1 - r_2}{1 - p_1 - p_2}.$$

Proof: We can write

$$\begin{aligned}
\mathcal{B}(N, k_1, k_2, p_1, p_2) &= \sum_{i=k_1}^{N} \sum_{j=k_2}^{N-i} \frac{N!}{i!j!(N-i-j)!} p_1^i p_2^j (1 - p_1 - p_2)^{N-i-j} \\
&= \sum_{i=k_1}^{N} \binom{N}{i} p_1^i (1 - p_1)^{N-i} \sum_{j=k_2}^{N-i} \binom{N-i}{j} \left(\frac{p_2}{1 - p_1}\right)^j \left(\frac{1 - p_1 - p_2}{1 - p_1}\right)^{N-i-j} \\
&= \sum_{i=k_1}^{N} \binom{N}{i} p_1^i (1 - p_1)^{N-i} \mathcal{B}(N - i, k_2, \frac{p_2}{1 - p_1}) \\
&\leq \mathcal{B}(N, k_1, p_1) \max_{k \geq k_1} \mathcal{B}(N - k, k_2, \frac{p_2}{1 - p_1}).
\end{aligned}$$

Now, we know that Hoeffding's inequality ensures that as soon as $\frac{k}{N} > p$,

$$\mathcal{B}(N, k, p) \leq e^{-NH(k/N, p)}.$$

From the assumptions about the constants, we get that $k_1 > Np_1$ and $k_2 > \frac{p_2}{1-p_1}(N - k_1)$. Then, we can write

$$-\log\left(\mathcal{B}(N, k_1, k_2, p_1, p_2)\right) \geq NH(\frac{k_1}{N}, p_1) + \min_{k \geq k_1} \left((N - k) H\left(\frac{k_2}{N - k}, \frac{p_2}{1 - p_1}\right)\right).$$

The function $x \to (N-x)H\left(\frac{k_2}{N-x}, \frac{p_2}{1-p_1}\right)$ is increasing on $[N - \frac{1-p_1}{p_2}k_2, N-k_2]$. Thus, as we know that $k_1 > N - \frac{1-p_1}{p_2}k_2$, we have

$$\begin{aligned} -\log\left(\mathcal{B}(N,k_1,k_2,p_1,p_2)\right) &\geq NH\left(\frac{k_1}{N}, p_1\right) + (N-k_1)H\left(\frac{k_2}{N-k_1}, \frac{p_2}{1-p_1}\right) \\ &\geq N\ H\left(\frac{k_1}{N}, \frac{k_2}{N}, p_1, p_2\right). \end{aligned}$$

□

Then, we can use the large deviation estimate for computing the NFA. But, this approach is equivalent to the binomial one, since, in general, the value $\frac{k_2}{Np_2} = 1$. This fact is due to the Grenander estimator definition. In fact, the binomial tail is a "wavelet" detector. The trinomail could be a useful test when we search a gap between two modes or a mode between two gaps.

Chapter 3

Experiments and applications of the FTC algorithm

Abstract. In this chapter some results and applications of the FTC algorihtm are studied. Firstly, the FTC results are checked on synthetic data, showing that the FTC algorithm can detect the modes of a mixture of unimodal laws without any *a priori* parametric model. In next section, the meaningful rejections, introduced in the previous chapter, are used as a goodness-of-fit test and compared to other tests of this kind. In this same section, the adequacy of Gaussian mixtures for fitting the intensity and hue histograms is questioned. In the following sections are analysed two applications of the FTC algorithm: grey level image segmentation and text documents segmentation. For both applications, some results are shown, trying to improve the existent algorithms in literature. Mainly in text documents analysis, the characteristic of the proposed algorithm (parameter-free and optimum conditions for small modes detection) allow us to obtain excellent results. The method is sensible to factors as noise and illumination changes. These factors modify the histogram shape and the mode detections is more difficult. These factors and their influence on the FTC algorithm results are analysed in Section 3.5. Finally, the application of the algorithm to the camera stabilization problem is studied. In this case, the studied histogram is the gradients orientation histogram and the modes detected by FTC indicate the rotation of the image with respect to the horizontal and vertical directions.

In the previous chapter we have presented the Fine to Coarse (FTC) algorithm, a method for histogram segmentation. Even though the aim of the FTC algorithm is color segmentation, we can find different applications of the algorithm, considering different histograms in different fields. Moreover, it is necessary to analise the presented method, analising its behaviour on synthetic data and in front of different changes in the histogram. This chapter is spent in these two tasks. The chapter begins analysing some results on synthetic data. In this first section, the FTC algorithm is applied on different know distributions laws, demonstrating its capacity to segment histograms without a priori assumption. Section 3.2 is devoted to the Gaussian mixtures. In a first time, the rejection theory is used to define a new goodness-of-fit test and to compare it to other classical tests. In a second time, the adequacy of the Gaussian mixtures to fit the hue and intensity histograms is questioned. Follow, the FTC algorithm is applied on the intensity histograms. segmenting grey level images and text document images. In Section 3.5 the FTC results in front of different disruption elements (noise, illumination,..) are studied. Finally, some results on camera stabilization are presented.

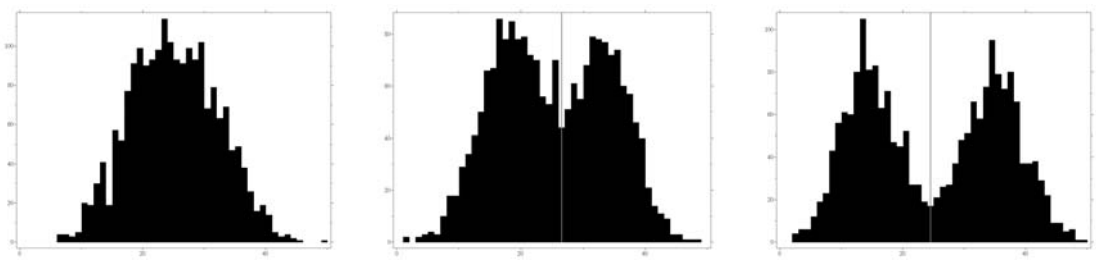
3.1 Some results on synthetic data

The better way for testing an algorithm is to experiment it on synthetic data. In this section, we observe the results of the FTC algorithm on different known distribution laws.

Table 3.1: *Number of segments found by the FTC algorithm for 100 samples of size 2000 of different laws.*

	unif.	gauss.	mix. of 2 Gaussian laws		
			$d = 2\sigma$	$d = 3\sigma$	$d = 4\sigma$
1 segment	99	100	100	24	0
2 seg.	1	0	0	76	100
3 seg.	0	0	0	0	0

From a given distribution law, we generated 100 samples of size $N = 2000$, quantized on 50 bins. For each sample, we registered the number of segments found by the FTC algorithm. Table 3.1 shows, for different classical laws, the number of samples among the 100 leading to 1, 2 or 3 segments. The laws used here are: the uniform law, a Gaussian distribution of standard deviation 10, and mixtures of two Gaussian functions $\frac{1}{2}\mathcal{N}(\mu, \sigma) + \frac{1}{2}\mathcal{N}(\mu + d, \sigma)$, with $\sigma = 5$ and $d = 2\sigma$, 3σ or 4σ.



(a) (b) (c)

Figure 3.1: *Samples of Gaussian mixtures of the form* $\frac{1}{2}\mathcal{N}(\mu, \sigma) + \frac{1}{2}\mathcal{N}(\mu + d, \sigma)$, *where* $\sigma = 5$. *(a) Sample with* $d = 2\sigma$, *(b) Sample with* $d = 3\sigma$, *(c) Sample with* $d = 4\sigma$. *For* $d = 2\sigma$, *the FTC algorithm finds no separator. For the other mixtures, the vertical line indicates the segmentation found.*

For a uniform or a Gaussian law, the number of segments is almost always found to be 1. For Gaussian mixtures, the results are closely related to the distance d between the means of the Gaussian distributions, as we also could expect. When $d = 2\sigma$, the FTC algorithm always find a single segment. It begins to find two segments when $d \simeq 2.5\sigma$, and finds two segments in 99% of the cases as soon as $d \geq 3.4\sigma$. Obviously, we see on Figure 3.1 that these results correspond to intuition. When $d = 2\sigma$, the two Gaussian functions cannot be distinguished, whereas the mixture clearly shows two modes when $d \geq 3\sigma$. These results also obviously depend on the number N of points. The larger N is, the more each sample looks like the real mixture law, and the sooner the algorithm finds two segments. Of course, segmenting Gaussian mixtures can be made much more efficiently by dedicated algorithms. But this example shows that we can detect the modes of a mixture of unimodal laws without any *a priori* parametric model.

3.2 On Gaussian mixtures

As mentioned in the beginning of Section 2.3, segmenting a histogram consists of two steps: 1. choosing a set of possible densities; 2. looking for the simplest of these densities in adequacy with the histogram for some statistical test. In the FTC algorithm, the densities proposed are a set of mixtures of unimodal laws, constructed from local Grenander estimators of the histogram. The test consists in looking for meaningful rejections. Another option is to use an EM algorithm (see Appendix A) to look for the best mixture of k Gaussian laws fitting the histogram. It is an extensively used method but it presents a hard drawback. It is necessary to specify the number k of gaussians which better fits the given histogram. Firstly, in this section we analyse the performance of the theory presented in the previous chapter as a tool to overcome this drawback, considering the NFA as a statistic in a goodness-of-fit test (see Appendix A). Secondly, in this section, we perform a study about the adequacy of the Gaussian mixture model to fit intensity and hue histograms.

3.2.1 Goodness-of-fit test

In this subsection, we consider NFA as a tool to answer to the question: are two distributions different? Answering to this question will allow us to know the number of gaussians necessary to represent a given distribution. The idea lies in aproximating the given distribution by means of the EM algorithm with a fixed number k of gaussians and comparing the obtained result with the given distribution by means of some goodness-of-fit test. The processing is repeated, changing the value of k, until the test verifies that the two distributions are not different.

In Appendix A we present three known goodness-of-fit tests. Taking into account the theory of the previous chapter, we can consider a new goodness-of-fit test. We consider two samples, $(X_1, \ldots, X_{n_1})$ and $(Y_1, \ldots, Y_{n_2})$, of two variables X and Y. Taking the null hypothesis as in Appendix A ($\mathcal{H}_0 : X$ and Y come from the same distribution), we reject $\mathcal{H}_0$ if there exist meaningful rejections between X and Y.

We want to compare the proposed goodness-of-fit test (called NFA test) with Kolmogorov-Smirnov (K-S), Chi-Square (χ^2) and Cramer-Von Mises (C-VM) tests. We perform this study by considering synthetic data and by searching the test that obtains the best result. We generate different samples of a gaussian mixture distribution with a fixed number k of gaussians. For these synthetic data the EM algorithm for the estimation of the Gaussian mixture is applied, for different values of the k parameter. The adequacy of the obtained estimation is assessed by using the previous goodness-of-fit tests (with significance level 0.95). The Tables 3.2 to 3.8 show how many of the data samples where accepted by these tests.

In a first experiment the synthetic data consisted of 25 samples of a mixture of three Gaussian functions $\frac{1}{3}\mathcal{N}(\mu, \sigma) + \frac{1}{3}\mathcal{N}(\mu + d, \sigma) + \frac{1}{3}\mathcal{N}(\mu + 2d, \sigma)$. Each sample consists of N ($N \in \{1000, 10000\}$) points quantized on 100 bins and the distance between the Gaussian components is variable ($d \in \{2\sigma, 3\sigma\}$ with $\sigma = 5$). The results are shown in Table 3.2, 3.3, 3.4 and 3.5. In the two first tables the size of the samples is 1000. In this case the results are very similar for the four tests (Table 3.2 and 3.3). If we consider the distance among the Gaussians as 2σ, then we obtain that all the samples can be estimated by a mixture of two Gaussians. This fact is clear if we observe the representation of a sample in Figure 3.2(a). The results are different if we consider a larger size, $N = 10000$. We observe that with this size the Kolmogorov-Smirnov and Cramer-Von Mises tests need

more than ten Gaussians to fit the samples. The results of NFA and χ^2 tests are more similar, but the first indicates the correct number in more cases.

Comparison between different goodness-of-fit tests on 25 samples of a mixture of three Gaussian functions

Number of Gaussians (k)	NFA	K-S	χ^2	C-VM
2	25	25	25	25

Table 3.2: *Size* $N = 1000$ *and* $d = 2\sigma$.

Number of Gaussians (k)	NFA	K-S	χ^2	C-VM
2	0	0	0	5
3	25	25	25	20

Table 3.3: *Size* $N = 1000$ *and* $d = 3\sigma$.

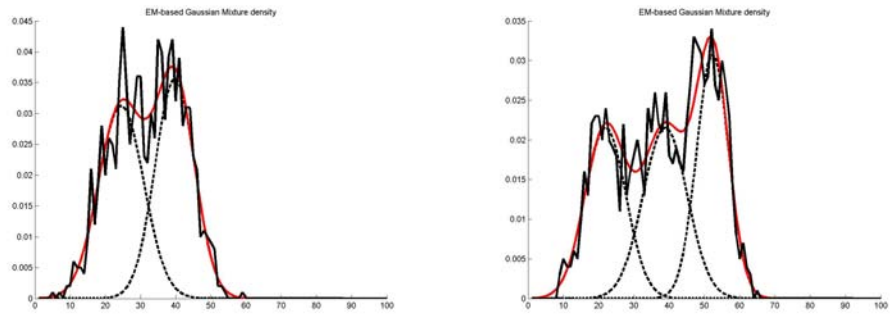
Number of Gaussians (k)	NFA	K-S	χ^2	C-VM
2	0	0	4	0
3	21	0	19	0
4	2	0	1	0
5	2	0	1	0
>10	0	25	0	25

Table 3.4: *Size* $N = 10000$ *and* $d = 2\sigma$.

Number of Gaussians (k)	NFA	K-S	χ^2	C-VM
3	25	0	23	0
>10	0	25	2	25

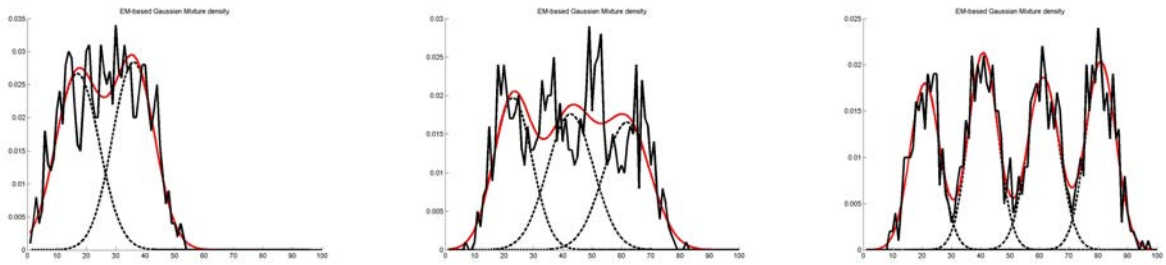
Table 3.5: *Size* $N = 10000$ *and* $d = 3\sigma$.

In a second experiment, we generate 25 samples of the mixture of four Gaussian functions $\frac{1}{4}\mathcal{N}(\mu,\sigma)+\frac{1}{4}\mathcal{N}(\mu+d,\sigma)+\frac{1}{4}\mathcal{N}(\mu+2d,\sigma)+\frac{1}{4}\mathcal{N}(\mu+3d,\sigma)$. Each sample consists of N ($N \in \{1000, 10000\}$) points quantized on 100 bins and the distance between the Gaussian components is variable ($d \in \{2\sigma, 3\sigma, 4\sigma\}$ with $\sigma = 5$). The results are shown in Table 3.6, 3.7, 3.8 and 3.9. In these case we consider three different distances with size $N = 1000$ (Tables 3.6, 3.7 and 3.8). When the distance is large, $d = 4\sigma$, most of the tests have no problem at pointing out the correct number of Gaussians. On the other hand, when the distance is very small, $d = 2\sigma$, the tests do not achieve the correct answer. In these case the K-S test obtains slightly better results. Finally, in the case $d = 3\sigma$ the results are very varied, but the NFA test is the test with better results. An example of these samples is shown in Figure 3.3. Finally, we study only one case with size $N = 10000$. This case (Table 3.9) verifies the fact that the K-S test is not good with samples of large size.



(a) (b)

Figure 3.2: *Samples of Gaussian mixtures of the form* $\frac{1}{3}\mathcal{N}(\mu,\sigma)+\frac{1}{3}\mathcal{N}(\mu+d,\sigma)+\frac{1}{3}\mathcal{N}(\mu+2d,\sigma)$*, where* $\sigma = 5$*. (a) Sample with* $d = 2\sigma$*, (b) Sample with* $d = 3\sigma$*. The red line indicates the EM algorithm result and the dashed line the obtained Gaussians with this algorithm. The EM algorithm obtains a good estimation of the first sample using only two Gaussians. Three Gaussians are needed for the estimation of the second sample.*



(a) (b) (c)

Figure 3.3: *Samples of Gaussian mixtures of the form* $\frac{1}{4}\mathcal{N}(\mu,\sigma)+\frac{1}{4}\mathcal{N}(\mu+d,\sigma)+\frac{1}{4}\mathcal{N}(\mu+2d,\sigma)+\frac{1}{4}\mathcal{N}(\mu+3d,\sigma)$*, where* $\sigma = 5$*. (a) Sample with* $d = 2\sigma$*, (b) Sample with* $d = 3\sigma$*. (c) Sample with* $d = 4\sigma$*. The red line indicates the EM algorithm result and the dashed line the obtained Gaussians with this algorithm. The EM algorithm obtains a good estimation of the first sample using only two Gaussians. Three Gaussians are needed for the estimation of the second sample. The third example fits for four Gaussians, since the distance among them is very large.*

Comparison between different goodness-of-fit tests on 25 samples of a mixture of four Gaussian functions

Number of Gaussians (k)	NFA	K-S	χ^2	C-VM
1	17	5	22	22
2	8	20	3	3

Table 3.6: *Size* $N = 1000$ *and* $d = 2\sigma$.

Number of Gaussians (k)	NFA	K-S	χ^2	C-VM
2	0	5	1	25
3	15	20	17	0
4	10	0	6	0
5	0	0	1	0

Table 3.7: *Size* $N = 1000$ *and* $d = 3\sigma$.

Number of Gaussians (k)	NFA	K-S	χ^2	C-VM
3	0	0	0	17
4	25	25	25	8

Table 3.8: *Size* $N = 1000$ *and* $d = 4\sigma$.

Number of Gaussians (k)	NFA	K-S	χ^2	C-VM
4	25	20	25	25
6	0	5	0	0

Table 3.9: *Size* $N = 10000$ *and* $d = 3\sigma$.

3.2.2 Adequacy of the Gaussian mixture model

As we have mentioned before,the Gaussian mixture model is one of the most popular methods to estimate a given distribution. In this section we want to check if the Gaussian mixture model is a good model for hue and intensity histograms estimation. With this aim, we will use the EM algorithm to estimate the best mixture of k Gaussian laws which fits the intensity or hue histogram. For each k, the adequacy of the mixture to the histogram is measured by a Chi-Square test (see Appendix A). The final segmentation is then defined by all the local minima of the selected mixture. This can be tested on the "Lena" histogram (see Figure 3.4). The EM algorithm is initialized by a k-means algorithm. For a significance level of 5% the first value of k leading to an acceptable mixture is $k = 14$ (the p-value for this mixture is 0.053). The adequacy between this mixture and the histogram is confirmed by a Cramer von Mises test (see Appendix A) at a significance level of 5% (p-value = 0.0659). Figure 3.4 (d) shows this best mixture of 14 Gaussian laws and indicates its local minima. Observe that by applying the FTC algorithm we segment the histogram in seven modes (see Figure 3.4(b)). The best mixture of 14 Gaussian laws, found by the EM algorithm, is constituted of seven modes (given by the local minima of this mixture), that correspond exactly to the modes found using the

FTC algorithm.

With more demanding tests (a Kolmogorv-Smirnov test, or NFA test), all mixtures are rejected until $k = 20$ (the modes found in this case are still the 7 same modes). This illustrates the discrepancy between the number of Gaussians needed to correctly represent the law and the actual number of modes. This discrepancy can be explained by the following observation: when a digital picture is taken, the sensors of the camera and the post-processing that is used to store the picture into a file are non-linear. Even if the real intensity values of a given object followed a Gaussian law, the intensity distribution of this object on the picture would not be well represented by a Gaussian function. In particular, the corresponding mode on the histogram can be highly non-symmetric (see e.g. Figure3.4). Such a mode needs several Gaussian laws to be well represented, whereas a unique unimodal law fits it. As a consequence, looking for a mixture of unimodal laws is more adapted in this case than looking for Gaussian mixtures.

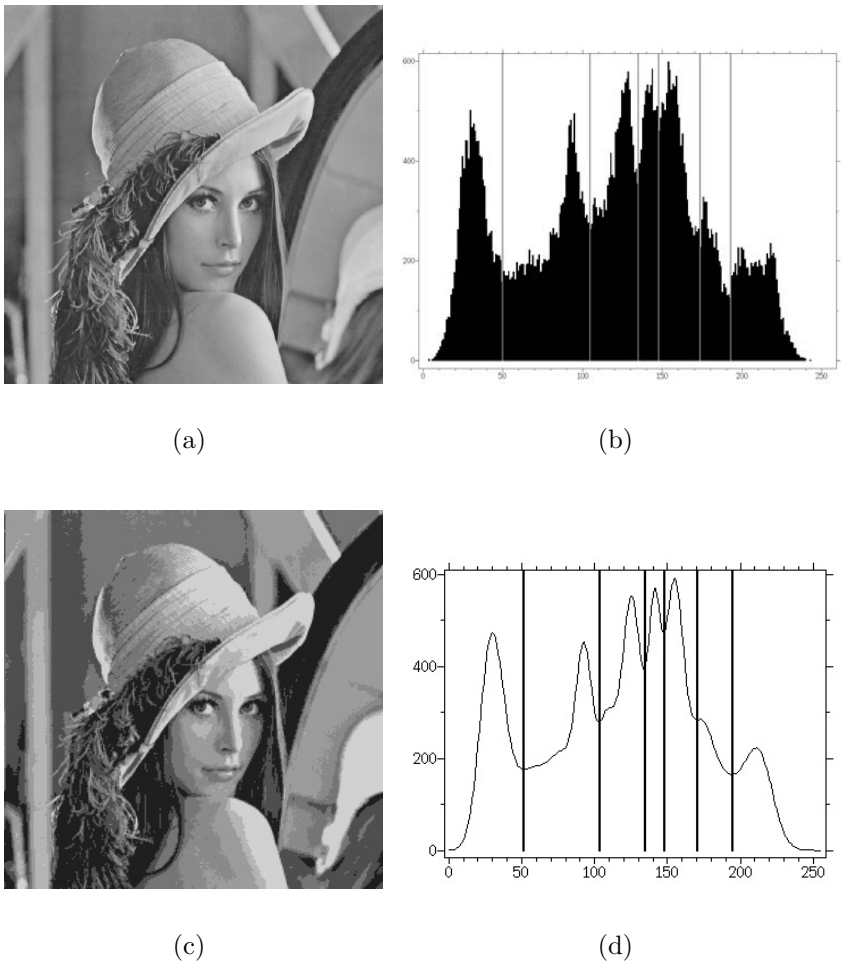


Figure 3.4: *(a) The original image (256 × 256 pixels) "Lena". (b) Its intensity histogram segmented into 7 modes by the FTC algorithm. Observe that this histogram presents strong oscillations. In the initialization, the histogram presented 60 local minima among 256 bins. The segments have been merged until they follow the definition 12 of an admissible segmentation. (c) The image "Lena" quantized on the 7 levels defined by the histogram segmentation shown in (b). (d) Best mixture of 14 Gaussian laws for the histogram (b), found by an EM algorithm. The local minima of this mixture, indicated by the vertical lines, correspond almost exactly to the separators found in (b).*

Other experiments confirm the previous assertions. For example, for the intensity histogram in Figure 3.5, the test rejects the proposed mixture for any value k. We see in Figure 3.5 the cases $k = 6$, 7 and 8. The found Gaussians are represented by a dashed line. The histogram is clearly not well modeled by any mixture of Gaussian functions. The FTC algorithm divides the histogram into seven modes.

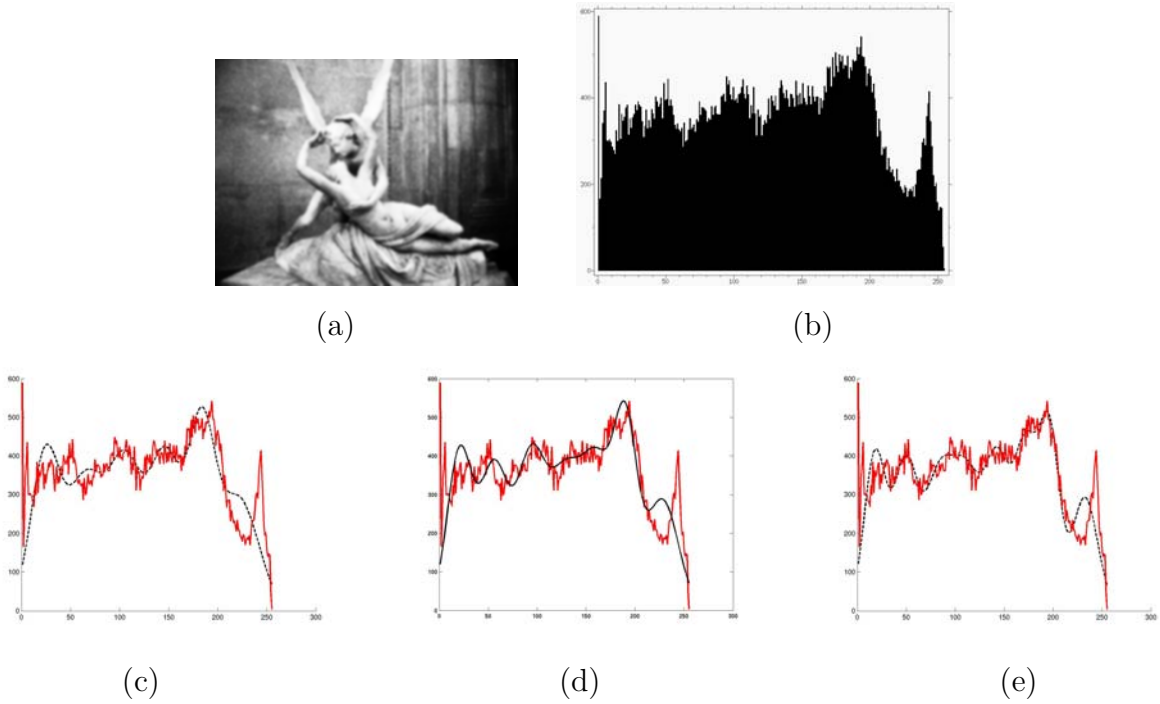


Figure 3.5: *(a) Original image of the sculpture "Amour and Psyché" by Canova. (b) Intensity histogram of the image. (c) Intensity histogram of the image, and best corresponding mixture of six Gaussian laws (dashed line) for the EM algorithm. (d) Best mixture of seven Gaussian laws for the same histogram.(e) Best mixture of eight Gaussian laws for the same histogram.*

A second example is shown in Figure 3.6. In this case the histogram present some very small modes which cannot be correctly modeled by the mixture of Gaussians.

In the case of color images the same observation hold. We can simply argue that the histogram of an image presenting a color gradation between two given hue values does not follow a Gaussian law. But this observation is still true if we consider that the original hue values of the objects are distributed with Gaussians laws.

This can be verified with a simple experiment. Let us consider the hue histogram of the "Madrid" image, and use the EM algorithm to look for the best mixture of k Gaussians representing the histogram. For each value of k, we assessed the adequacy of the best found mixture with a Chi-Square test (see Figure 3.7 for the case $k = 6$). The test always rejects the proposed mixture. These results are easy to understand : the histogram contains a large number of samples (1.3 millions), and it is very unlikely that these points have been sampled from a mixture of Gaussians laws. We see on Figure 3.7 that the modes of the hue histogram are sharper than Gaussian laws (they are much higher than a Gaussian of the same width). This observation is confirmed if we make the same test on the hue histogram of a given coherent object.

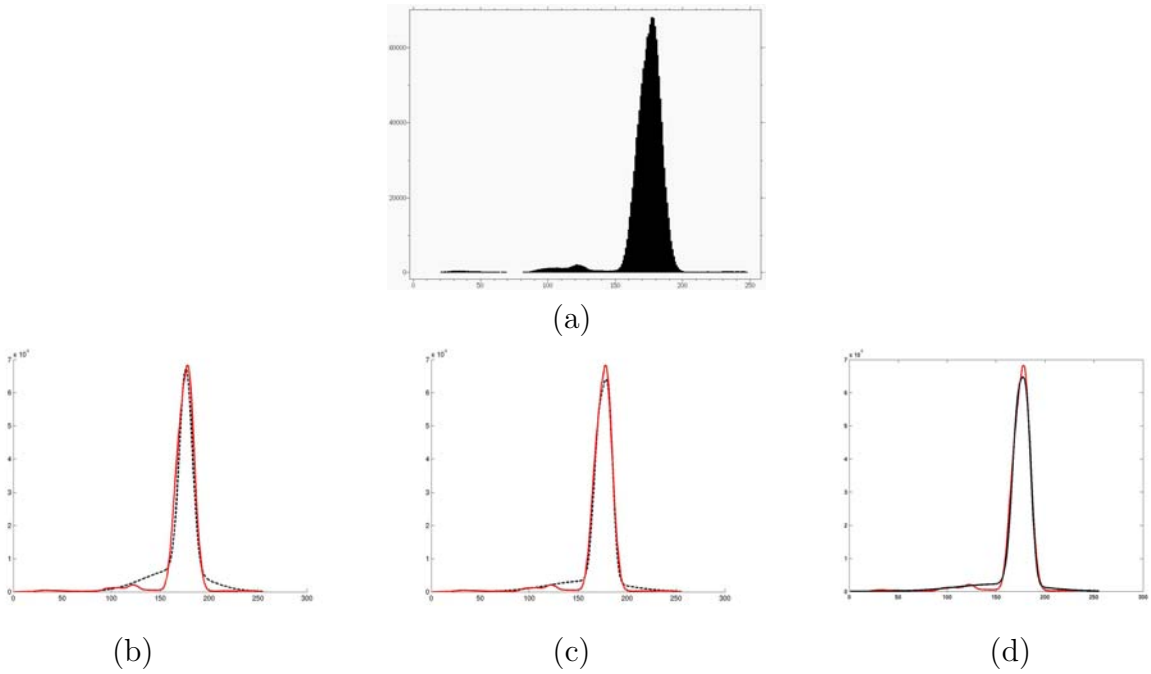


Figure 3.6: *(a) Intensity histogram of the image in Fig. 3.8 (top left). (b) Intensity histogram of the image, and best corresponding mixtures of two Gaussian laws (dashed line) for the EM algorithm. (c) Best mixture of three Gaussian laws for the same histogram. (d) Best mixture of four Gaussian laws for the same histogram. The larger mode of the histogram is not well modeled by a Gaussian law. As a consequence, even after a k-means initialization, the EM algorithm uses two Gaussians to better represent this large mode. A Gaussian-based segmentation would therefore divide this large mode into two parts.*

3.3 Some experiments on grey level images segmentation

One of the most important operations in Computer Vision is segmentation. Image segmentation is a process of dividing an image into a set of regions which are visually distinct and uniform with respect to some property, such as grey level or texture. The problem of segmentation has been, and still is, an important research field and many segmentation methods have been proposed in the literature (see surveys [73, 57]). The methods can be categorised as:

- **Histogram thresholding,** assumes that images are composed of regions with different grey ranges, and separates it into a number of peaks, each corresponding to one region,
- **Edge-based approaches,** use edge detection operators such as Sobel or Laplacian. Resulting regions are connected set of pixels bounded by edge pixels.
- **Region-based approaches,** based on similarity of regional image data. Resulting regions are (maximal) connected sets of pixels for which a uniformity condition is satisfied.
- **Hybrids,** consider both, edges and regions.

As we have mentioned, the approach presented in the previous chapter can be classified in the histogram thresholding techniques. Even though the aim of this approach is not

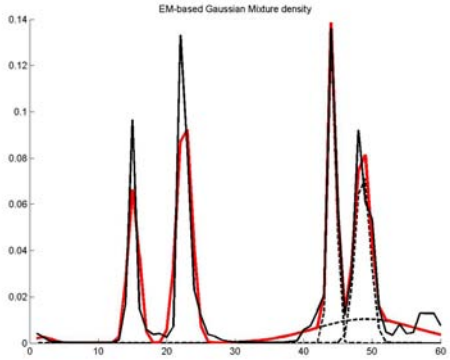


Figure 3.7: *Hue histogram of the "Madrid image", and best corresponding mixture of six Gaussians laws (red line) for the EM algorithm. We observe that the histogram is not well modeled by this mixture of Gaussians.*

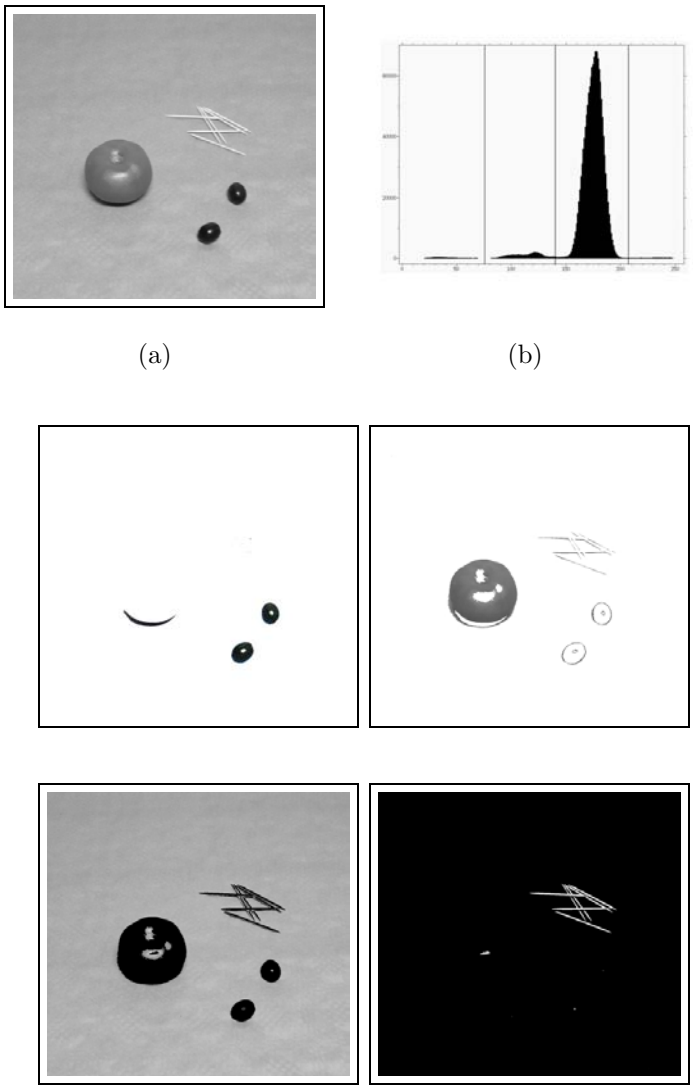
image segmentation, it can be applied in this sense to the intensity histogram. In this section we present some examples of this application. We have to note again that the presented approach is automatic and parameter free.

Figure 3.8 displays an image that contains an uniform background and a set of small objects of different intensities. The intensity histogram shows a large peak corresponding to the background and very small groups of values corresponding to the objects in the foreground. The FTC algorithm successfully segments this histogram into four modes. The associated regions are shown in Figure 3.8 (middle and bottom rows), each one of them corresponding to a different object in the scene. Remark that the histogram cannot be properly modeled by a mixture of Gaussians, as discussed in the previous section (see Fig. 3.7).

Modes of a grey level histogram do not necessarily correspond to semantical visual objects. In general modes simply correspond to regions with uniform intensity and segmenting the histogram boils down to quantizing the image on the so-defined levels. Quantization is acheived by assigning the same grey level to all the pixels contributing to the same mode of the histogram. This common grey level can be either the mean or the median value of the grey levels in the mode. An example of such quantization is shown in Figure 3.4. The histogram of "Lena" is automatically divided into seven modes, and the corresponding image quantization is shown in Figure 3.4 (c). Remark that no information about the spatial relations between image pixels is used to obtain the segmentation. Recent works (e.g. [81], [18]) propose the use of spatial and histogram-based information to improve the image segmentation results. The combination of spatial features with the FTC histogram segmentation algorithm is an interesting problem and it will be addressed in our future research.

We can observe another example in Figure 3.9. In this case the intensity histogram cannot be modeled by a Gaussian mixture (see Fig. 3.5) and, moreover, due to its oscillating behavior, methods such as mean shift would fail to properly segment it, since many peaks would be detected. The FTC algorithm permits to segment the image into six regions.

As remarked in the previous example, in general the modes of the intensity histogram have no semantical meaning. However, there are situations where histogram analysis can be used to discern between background and foreground information. This is the case of document analysis, discussed in the following section.



(a) (b)

(c)

Figure 3.8: *(a) Original image (399 × 374 pixels) "table", (b) its intensity histogram segmented into four modes by the FTC algorithm. (c) Regions of the image corresponding to the four obtained histogram modes (in decreasing level of intensity, the background is either white or black depending on the mean intensity of the represented mode).*

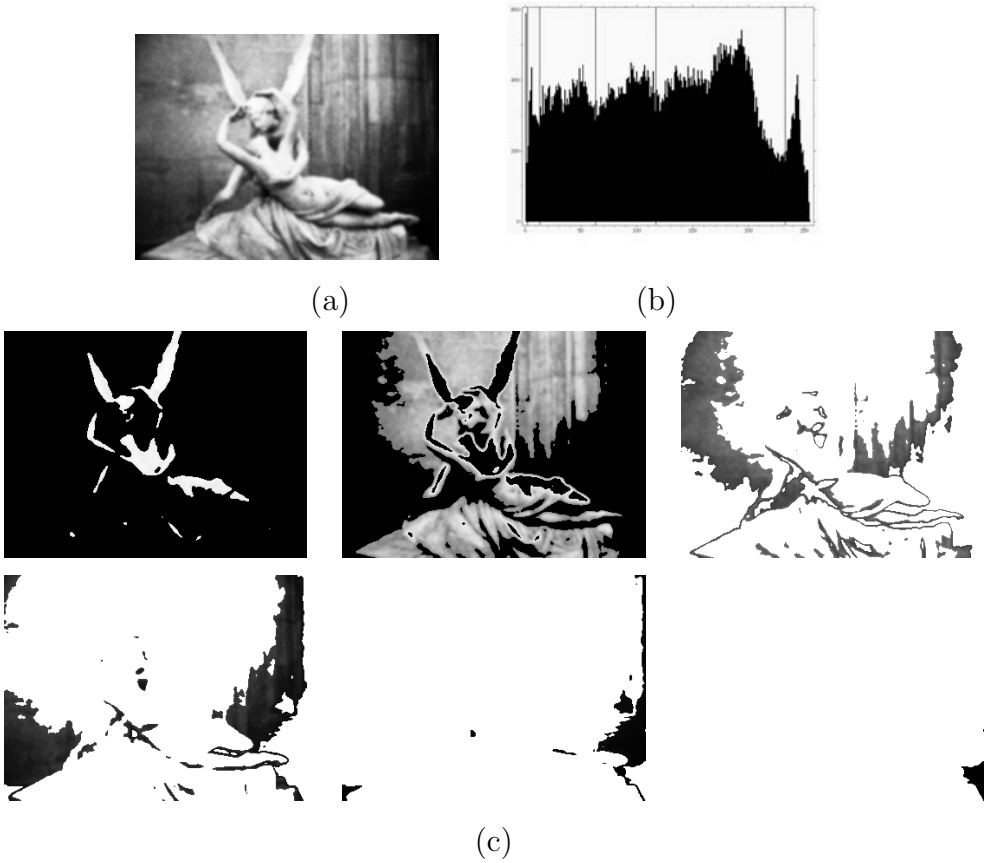


Figure 3.9: *(a) Image (350 × 251 pixels) of the sculpture "Amour and Psyché" by Canova, (b) Its intensity histogram segmented into six modes, (c) Regions of the image corresponding to the obtained histogram modes (in decreasing level of intensity, the background is either white or black depending on the mean intensity of the represented mode).*

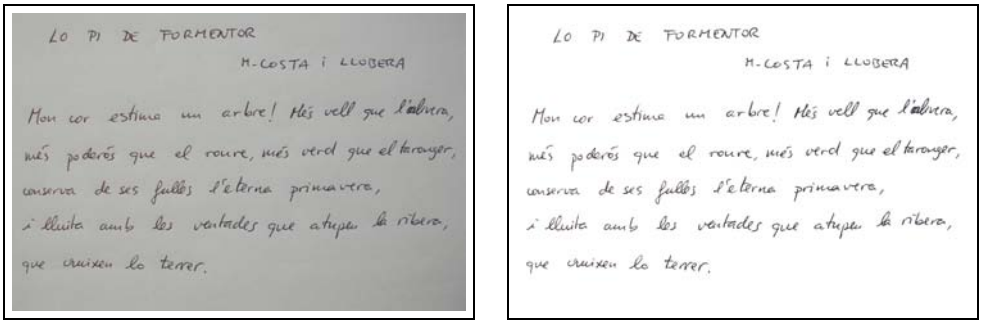
3.4 Some experiments in document image analysis

Histogram thresholding is widely used as a pre-processing step for document understanding and character recognition. Its main use in this domain is to sort out the background and the characters in scanned documents. In this kind of documents, the intensity histogram generally presents two different modes: one large mode that represents the background, and another one, much smaller, corresponding to the text. Many different binarization methods have been proposed (see Appendix C for a review) to find the best histogram thresholds for greyscale images. Nowadays, different methods are still studied, using simple spatial features ([22]), texture features ([63]) or mathematical morphology information ([12]).

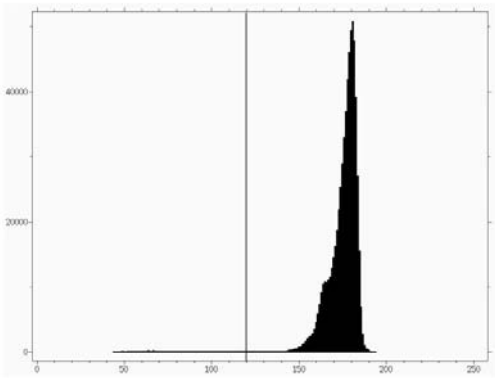
However, binarization methods present two drawbacks. First, when the foreground region (the text here) is too small in comparison to the background (see Fig. 3.10), the position of the threshold becomes arbitrary and the foreground may not be well detected. Second, binarization methods are not adapted to any kind of written documents (see Fig. 3.13). When the background pattern is complicated, or when different inks are used in the text, segmenting the histogram in only two modes is not a good solution. Finding automatically the number of modes in the histogram allows one to get more than two modes when necessary. The first drawback can be overcome by considering that a threshold between two modes is kept if none of the modes can be neglected in comparison to the other one, which gives a strong probabilistic significance to the final thresholds of the segmentation. The FTC algorithm presents these two characteristics, by which we can consider it as a good tool for segmentation between background and text in document images. The following examples illustrate these properties.

In simple written documents, where only one ink has been used, the segmentation found by the FTC algorithm is bimodal. This is the case of the example shown in Figure 3.10. The histogram is segmented into two modes, one of them corresponding to the text characters (see Fig. 3.10(b)). This example shows that the FTC algorithm is able to find very small modes when they are isolated enough. In Figure 3.11, although the image presents several different grey shades, the FTC algorithm also segments the histogram into two modes (Fig. 3.11(c)), separating clearly the characters in the check from the background, as we can see in Figure 3.11(b).

Text extraction in document images can be achieved by treating the whole image, or by considering it as a collection of subimages. It must be underlined that the size of the images can interfere in the results. The properties stated at the end of Chapter 2 (Proposition 4 and Property (P1)) are useful to choose the shape and the size of the subimages. In a text image, the larger the proportion of background pixels, the more difficult it becomes to extract a text mode from the histogram, since it tends to become negligible in front of the background one. Consequently, considering Proposition 4, it is more judicious to use a portion of document well tight around the text than one with a large piece of white background. This fact can be observed in Table 3.10. We have considered the image of a text with a great proportion of background pixels (Figure 3.12(a)). From this image, we have created six subimages with the same number of text pixels but with a smaller proportion of background pixels (we have cut out the original image in smaller images, keeping the top part). Thus, we have the hypothesis of Proposition 4. In Table 3.10 we have indicated the size of the different images, the number of background pixels in each image and the value of $\log(NFA)$ for the small mode (the one corresponding to the text pixels) of the intensity histogram of each one. The smaller the size of the image, the smaller the value of λ, (see definition in Section 2.4.1), and the smaller the NFA. This



(a) (b)



(c)

Figure 3.10: *Results of the proposed approach: (a) Original image (1010 × 661 pixels), (b) Extraction of the text, (c) Intensity histogram of the original image and corresponding segmentation.*

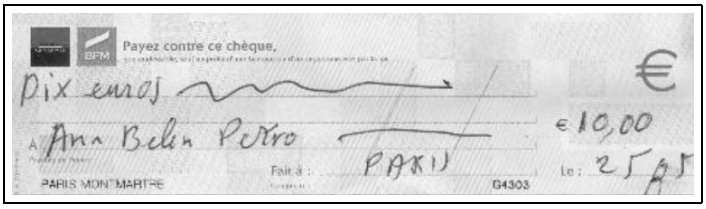
fact confirms the conclusion of Proposition 4. Then, we can observe that the NFA is lower when the background pixels quantity is lower, from which we can conclude that we segment the histogram better in this case.

Figure 3.12 illustrates Property (P1): if the proportion between the text and the background is fixed, then the larger the document, the better the separation will be. We see on this figure that the three histograms, corresponding respectively to the whole text, to a line of text, and to a word, have very similar shapes, but the total number of points decreases from one histogram to the other. The last one is no more segmented.

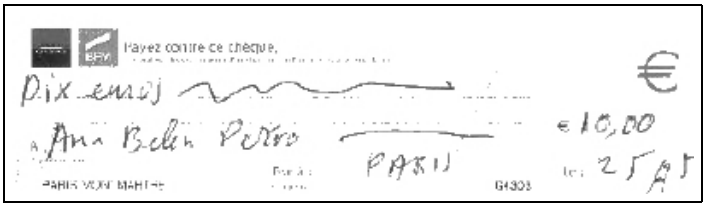
Table 3.10: *Relation between the background pixels in a image and the NFA value.*

Size	Background pixels	$log(NFA)$
1062 × 1518	1598353	-44.44
1037 × 1333	1368558	-46.40
1031 × 1181	1203848	-47.43
1020 × 1029	1035817	-48.22
945 × 531	488033	-52.16
908 × 504	443870	-54.18

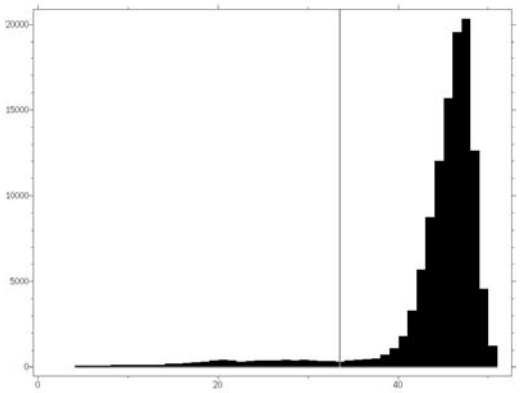
As we have mentioned, most of the text extraction methods achieve a binarization of the image. In some cases where the background pattern is complicated or different



(a)



(b)



(c)

Figure 3.11: *Results of the proposed approach: (a) Original image, (b) Extraction of the text, (c) Intensity histogram of the original image and corresponding segmentation. This histogram presents several local minima, but the final segmentation is bimodal.*

inks are used in the text, the histogram segmentation into only two modes is not a good solution. The FTC algorithm is an automatic method and it segments the histogram using the appropriate number of modes. Thus, the image of the Figure 3.13 is separated into three regions because the intensity histogram presents three different modes. The first mode represents the band in the bottom of the image, the second mode corresponds to the text and the stars of the image, and the third one is the background. It is clear that using the bilevel histogram thresholding we cannot obtain the text separated from the background and the lower band simultaneously.

In the document image where the text presents different inks, sometimes the intensity histogram is insufficient to separate it. For example, in Figure 3.14 we represent a handwritten text where each paragraph is written with a different ink (the first one in red ink, the second one in blue ink, and the thirst in green). The intensity histogram (Figure 3.14(c)) allow us to segment the text from the background and from the noise, but does not divide the three inks. We can obtain this separation thanks to the hue histogram of the extracted text. If we apply the FTC algorithm to this histogram we obtain three modes, corresponding to each paragraph.

Even though the FTC algorithm obtains good results in the different application fields, it presents a huge sensibility to such elements as illumination or noise. These sensibilities

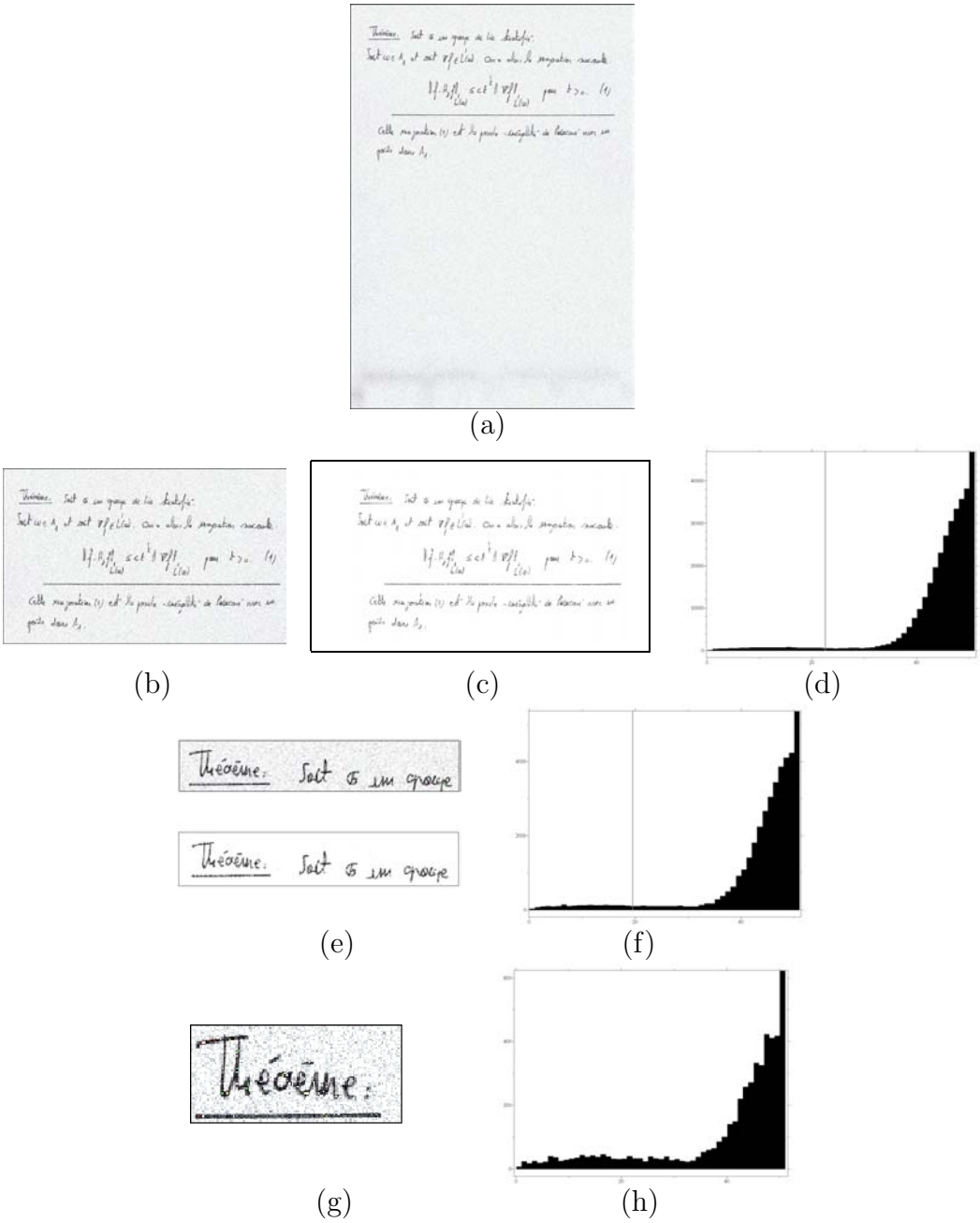
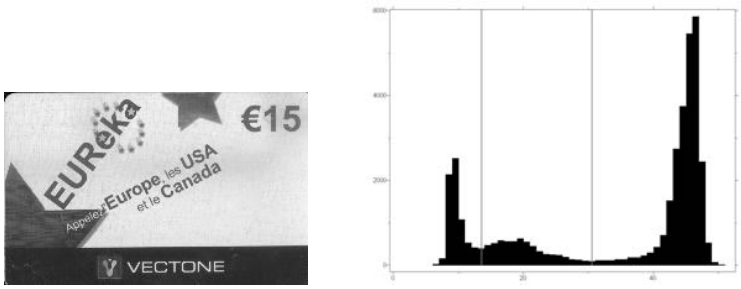


Figure 3.12: *(a) Original image, (b) Region extracted from the original image, (c) Image corresponding to the left mode in the histogram segmentation of the previous image, (d) Intensity histogram of the previous region with the segmentation obtained. The left mode, which is small and quite flat, corresponds to the text (see (c)). The large mode in the right (which presents a saturation peak at the end) corresponds to the background. (d) Line extracted from the original image and image corresponding to the little mode in the histogram segmentation, (e) Intensity histogram of the text line and corresponding segmentation, (f) Word extracted from the line, (g) Histogram of the word: the histogram is very similar to the others, but it is not segmented because it has not enough points.*



(a) (b)

Figure 3.13: *Results of the proposed approach: (a) Original image, (b) Intensity histogram with the three modes obtained. The first mode on the left corresponds to the lower and darker band of the image, the middle mode corresponds to the text, and the last mode is the background.*

are analysed in the next section.

3.5 Sensibility of the method

Generally, two factors can influence the segmentation: the noise in images and the histogram quantization noise.

Concerning the effect of noise in the segmentation results we have performed two tests. The first one (shown in Fig. 3.15) consists in adding impulse noise to the image in Figure 3.10(a). In the second case we add Gaussian noise to the same image (see Fig. 3.16). We observe that impulse noise is uniformly distributed over the dynamic range of the image, therefore both modes in the obtained histogram segmentation contain it. As a consequence, the extracted text is noisy.

In the case of Gaussian noise, it produces a smoothing effect on the shape of the histogram (compare histograms in Figures 3.10(c) and 3.16(b) and (e)), which may lead to the detection of an unique mode (see Fig. 3.16).

Theoretically if we add a noise b to an image u, its intensity distribution becomes $h_u * h_b$, where h_u is the grey level histogram of u, and h_b the noise histogram. This results in a blur in the histogram. If the image has N pixels, and if the noise is an impulse noise, added to $p\%$ of the pixels, then $h_u * h_b = (1-p)h_u + p\frac{N}{256}\mathbf{1}_{[0,255]}$, which is not really disturbing for the FTC algorithm (adding a uniform noise on a histogram does not change its unimodality property on a given interval). Moreover this kind of noise can be easily removed by a median filter on the image. Indeed, observe in the example of Figure 3.15 that the histogram segmentation is not affected by this kind of noise. However, as could be expected of any histogram-based segmentation, the extracted text is noisy. In the case of Gaussian noise, the operation smoothes the shape of the histogram h_u. As a consequence, the number of modes found can decrease when the standard deviation of the noise increases too much. However, this kind of image noise can be efficiently handled for example by the NL-means algorithm [10] before computing the histogram.

The real noise in histograms is the quantization noise, coming from the fact that the histograms have a finite number N of samples. If a histogram h originates from an underlying density p, the larger N, the more h looks like p. When $N \to \infty$, a segmentation algorithm should segment the histogram at each local minimum of its limit p. Consider the example of Figure 3.17. An image of size 1600x1200 is subsampled several times

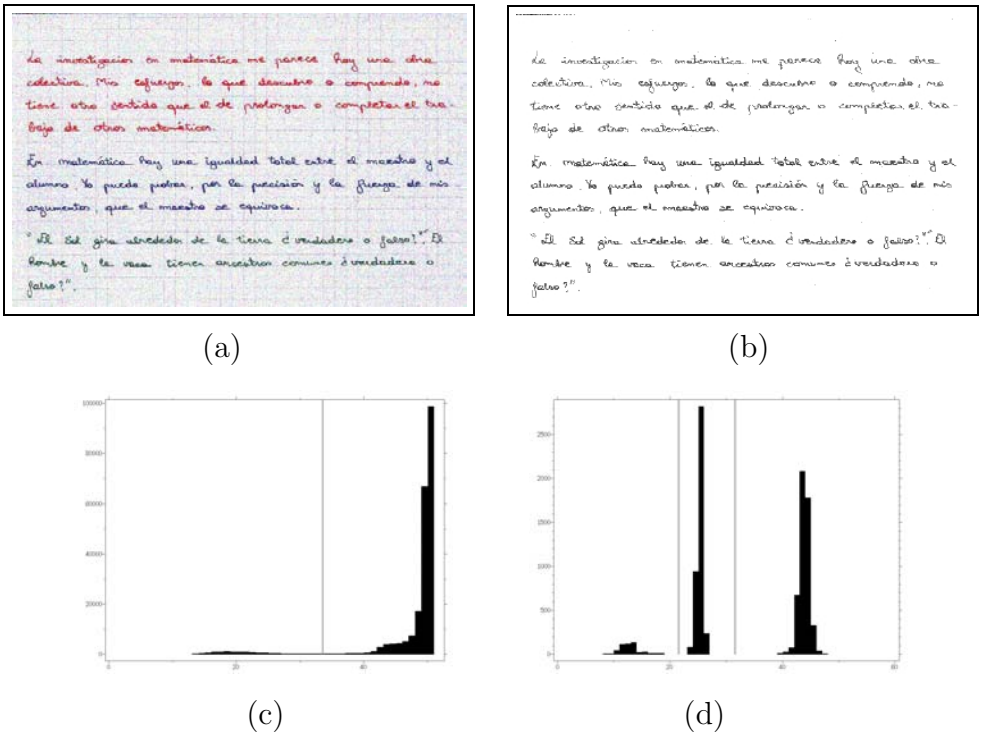


Figure 3.14: *Results of the proposed approach: (a) Original image,(b) Extraction of the text, (c) Intensity histogram with the two obtained modes, (d) Hue histogram with the three obtained modes.*

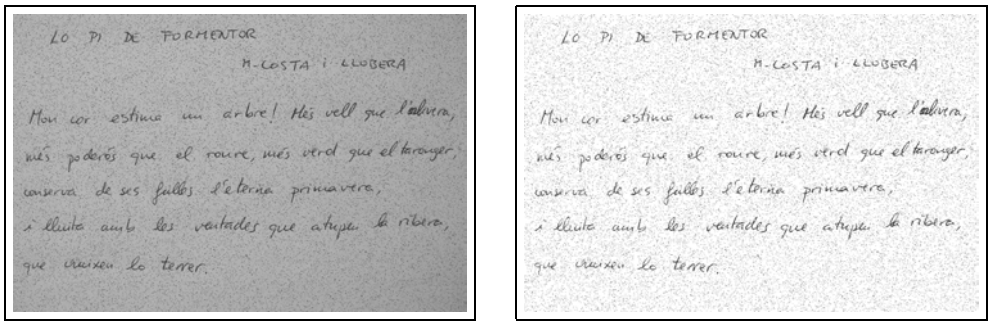
by a factor 2. Each intermediate image yields a histogram. These histograms can all be considered as realizations of the density given by the histogram of the original image. The smaller the number of samples, the less information we have, and the less the histogram can be segmented with certainty. Figure 3.17 shows that the number of segments found by the FTC algorithm increases with N. The separators tend towards the separators of the deterministic histogram of the continuous underlying image.

The performance of any image segmentation method based on histogram thresholding can be severely affected by changes in the image that modify the shape of its histogram. The following figures illustrate the effect of some of these changes on the results obtained in Figure 3.10.

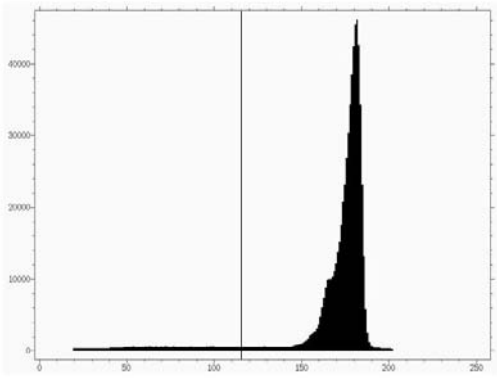
A non uniform illumination of the image produces a significant modification of its histogram (compare Figures 3.10(c) and 3.18(c)). As a consequence, the grey level of the characters in the text may no longer be concentrated in an unique mode. This is the case in Figure 3.18, where a part of the text is missing in the detection.

Figure 3.19 displays the results of the segmentation algorithm on a blurred version of image 3.10(a). In this case the dynamic range of the image is reduced and therefore its histogram shrinks (Fig. 3.18(d)). Moreover, some new grey levels appear, with an intermediate value between the bright background and the dark characters. As a consequence a new mode appears in the histogram, corresponding to these new intermediate values (see Fig. 3.18(c)).

It is clear from the previous experiments that such a simple technique as histogram thresholding is not robust to severe changes in the image. The use of spatial information is needed in order to improve the results. Techniques such as the ones reported in [81] and [18] provide better results.



(a) (b)



(c)

Figure 3.15: *(a) Image degraded with impulse noise (*10% *of the pixels are affected). (b) Extraction of the text, observe as noisy pixels appear in the detection. (c) Intensity histogram of the image and corresponding segmentation.*

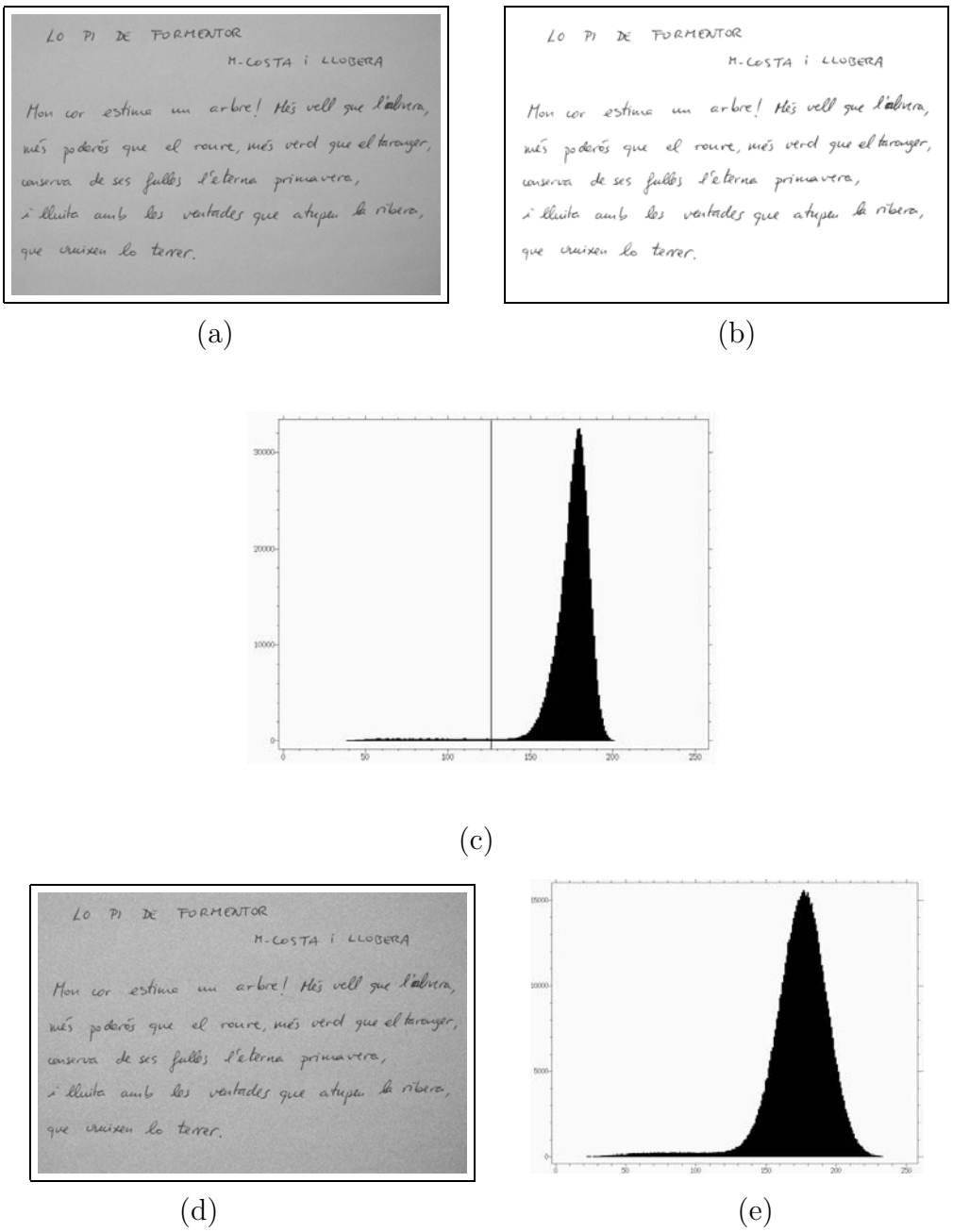
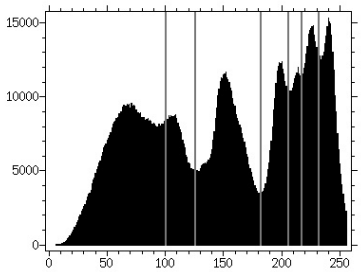
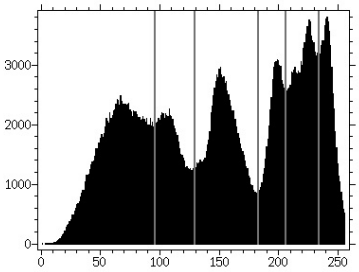


Figure 3.16: *(a) Image corrupted with Gaussian noise (standard deviation 5).(b) Extracted text. (c) Intensity histogram of the image and corresponding segmentation. Remark that even in the presence of noise two modes are correctly detected. (d) Image degraded with Gaussian noise (standard deviation 15). (e) Intensity histogram of the image and corresponding segmentation. In this case an unique mode is detected, therefore the method is unable to separate text from background.*

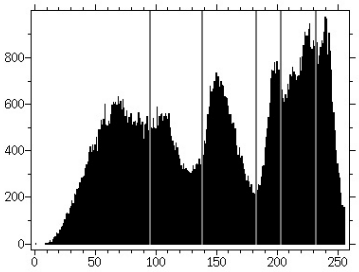
(a) Original image, 1600x1200 wide



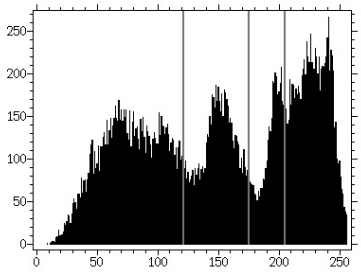
(b) Histogram of the original image (1920000 samples).



(c) Histogram of the image subsampled by a factor 2 (480000 samples).

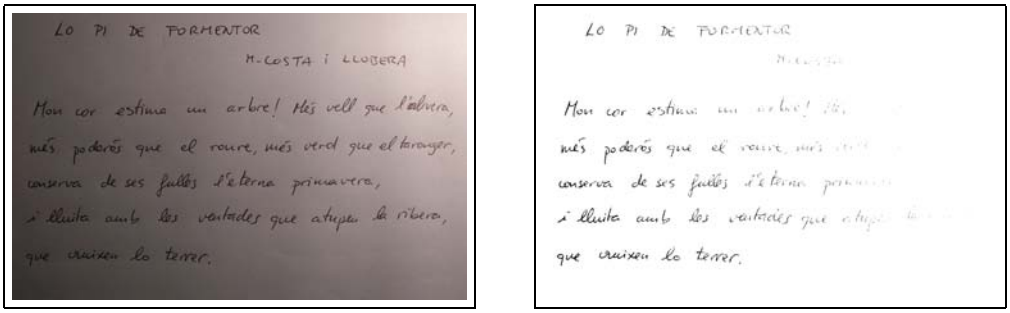


(d) Histogram of the image subsampled by a factor 4 (120000 samples).

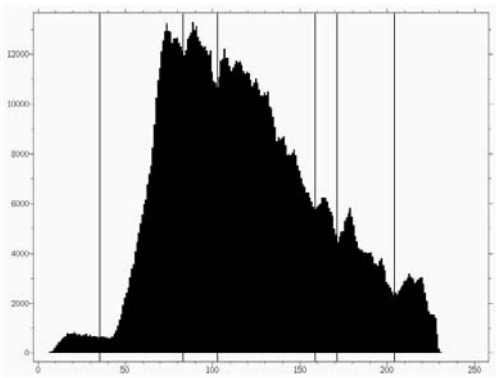


(e) Histogram of the image subsampled by a factor 8 (30000 samples).

Figure 3.17: *Sensibility of the method to quantization. The larger the number of samples, the more certain the segmentation. It follows that the histogram is more and more segmented when N increases. The segmentation tends towards the segmentation of the deterministic histogram of the continuous underlying image.*

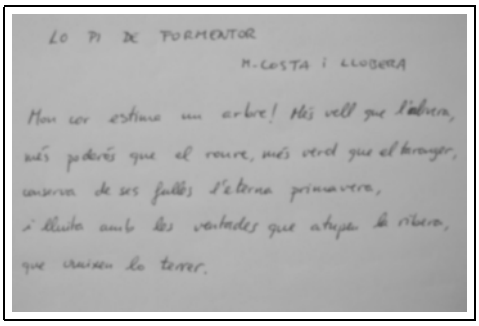


(a) (b)

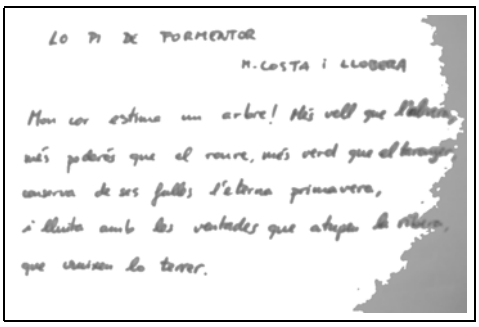


(c)

Figure 3.18: *(a) Image showing the effect of inhomogeneous illumination. (b) Extraction of the text. Observe that a part of the text is missing due to the non-uniform illumination. (c) Intensity histogram of the image and corresponding segmentation.*

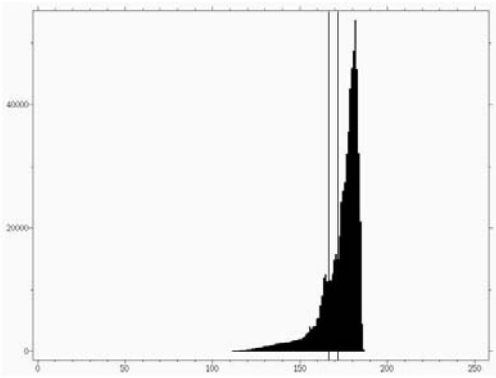


(a)



(b)

(c)



(d)

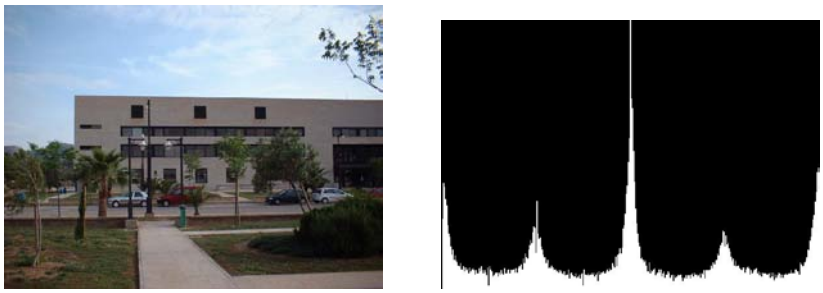
Figure 3.19: *(a) Image blurred with a Gaussian kernel. (b) and (c) Extraction of the text corresponding to the left-most and central modes in the histogram, respectively. (d) Intensity histogram of the image and corresponding segmentation.*

3.6 Some experiments on camera stabilization

By camera stabilization we refer to the problem of compensating all undesirable motions that affect a video camera, such as small rotations or translations due to vibrations of the camera. This means that stabilization shall not modify the motion of the objects in the scene or the correct motion of the camera (e.g. for a panning camera, stabilization should not affect the panning motion).

In order to stabilize the camera some reference system needs to be defined. Most stabilization algorithms use as a reference some frame in the video sequence ([98]). In the case of dynamic video sequences, where background changes with time, this reference frame must be periodically updated ([66]). Once the reference frame has been chosen, the motion between this frame and any given frame is computed, usually by applying some correlation matching strategy. The estimated motion is then used to compensate the motion of the frame.

In these experiments, we propose to use, for stabilization, the natural reference system given by the horizontal and vertical directions that are dominant in most video sequences. Indeed, it is remarkable that in most urban scenes there is a large set of horizontal and vertical contours, mostly corresponding to buildings. If we compute the histogram of orientations of all the pixels of an image, the previous assertion becomes crystal clear. Let us observe, for example, images in Figure 3.20. The left image is a typical urban scene, where some building appears in the background of the video sequence. The histogram of pixels orientations can be seen on the right.



(a) (b)

Figure 3.20: *(a) Example of an image in an urban scene. (b) Its gradient orientation histogram. Horizontal and vertical directions are dominant. The histogram is periodic, and four modes, centered at values* 0°, 90°, 180° *and* 270°, *are clearly observed.*

The orientation of a pixel can be defined from its gradient vector. The gradient vector indicates the direction of maximum variation of the image grey levels. For a horizontal contour the direction of the gradient vector is either 90º or 270º, and for a vertical contour it is 0º or 180º. Therefore, we can define the **pixel orientation** as the angle of the 90º rotation of the gradient vector [31]:

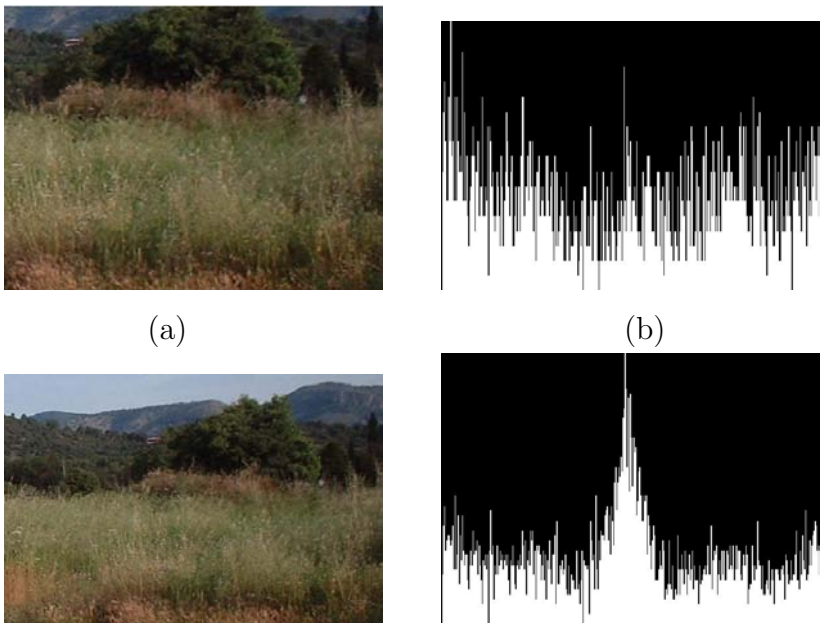
$$\mathrm{O}_{i,j} = \mathrm{angle}(\vec{D}_{i,j})$$

$$\vec{D}_{i,j} = \frac{1}{2}\begin{pmatrix} -(u_{i,j+1} + u_{i+1,j+1}) + (u_{i,j} + u_{i+1,j}) \\ (u_{i+1,j} + u_{i+1,j+1}) - (u_{i,j} + u_{i,j+1}) \end{pmatrix}$$

The histogram of orientations being circular, the first and last columns of the histogram represent the same information. In Figure 3.20-right four modes are clearly observed,

which are centered at 0º, 90º, 180º, and 270º, thus corresponding to the horizontal and vertical directions in the image.

It must be remarked that not all images contain these dominant directions, particularly those corresponding to rural scenes, such the one in Figure 3.21(a). In these cases, either no meaningful modes are detected, or the number of modes is different from four, which indicates that there are not two dominant directions. It is interesting however to remark that, even in these cases, some dominant directions may be detected (see e.g. Figure 3.21(c)(d)), usually corresponding to the horizon line.



(a) (b)

(c) (d)

Figure 3.21: *(a) Original image "vegetation". It does not contain any dominant direction (b) Its orientation histogram. No meaningful modes are detected in it. (c) Original image "vegetation+mountain", (d) Its orientation histogram. In this case, a meaningful mode is detected (centered at 0º), which corresponds to the approximately horizontal orientation of the mountain in the background of the scene.*

A straightforward application of the existence of these dominant directions in the scene is a fast algorithm for camera rotation stabilization. The algorithm consists of three steps:

1. Compute the histogram of orientations of all the pixels in the image.
2. Segment the histogram, using FTC algorithm, and taking into account the periodic aspect of the histogram.
3. If four equispaced modes are detected, compute the minimum circular translation that centers the modes at positions 0º, 90º, 180º, and 270º. This translation gives a rough estimate of the rotation of the image with respect to the horizontal and vertical directions.

The orientation of pixels with small gradient is highly sensitive to the quantization of the image grey levels. As a result, the histogram of the orientations for these pixels displays a characteristic pattern as the one in Figure 3.22.

In order to reduce this effect, only pixels with a large enough gradient are used to compute the histogram of orientations. In all the displayed histograms only pixels whose

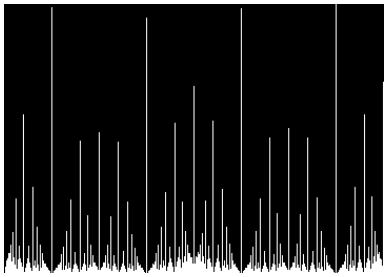
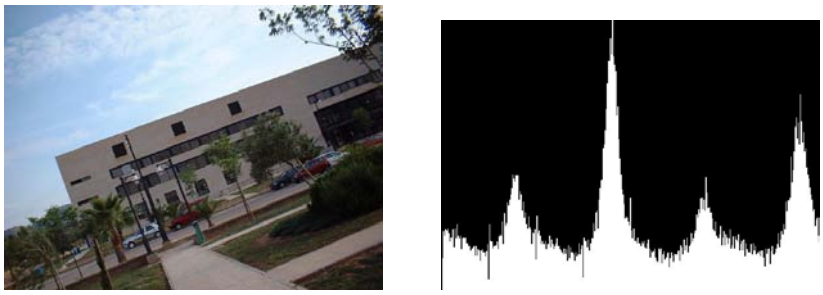


Figure 3.22: *Histogram of orientations for image 3.20(a). All the pixels have been considered in order to compute the gradient. The histogram presents the quantization effects on the pixels with small gradient. Then, we use only pixels with a large enough gradient. The histogram using only pixels whose gradient magnitude is above* 25 *of this image is shown in Figure 3.20.*

gradient magnitude is above 25 are used for the computations. A more elegant and parameter-free method to reduce this quantization effect is described in [32] and it consists of computing the $\frac{1}{2}$ pixel translate of the original image by using the FFT algorithm.

An application of the algorithm is illustrated in Figure 3.23. This figure shows a frame displaying the same scene as in Figure 3.20(a) after a camera rotation. The histogram of orientations of the image is segmented into four modes, centered at positions 70º, 160º, 250º, and 340º. The comparison of these values with the ideal values 0, 90, 180 and 270 allows us to estimate the rotation of the camera (approximately 20º). The same experiment is repeated for the image of Figure 3.6. In this case, the transformation between the images is a combination of a translation and a rotation. The angle of the rotation is correctly estimated (approximately 30º).



(a) (b)

Figure 3.23: *(a) Image similar to the one in Figure 3.20(a). (b) Its gradient orientation histogram. The rotation of the camera is reflected on the histogram. Four dominant modes are detected, centered at values* 70°, 160°, 250° *and* 340°. *From these locations, a rough estimation of the rotation angle is obtained (approximately* 20° *in this example).*

The algorithm is very fast and it can be used as a first step for the computation of more complex motions.

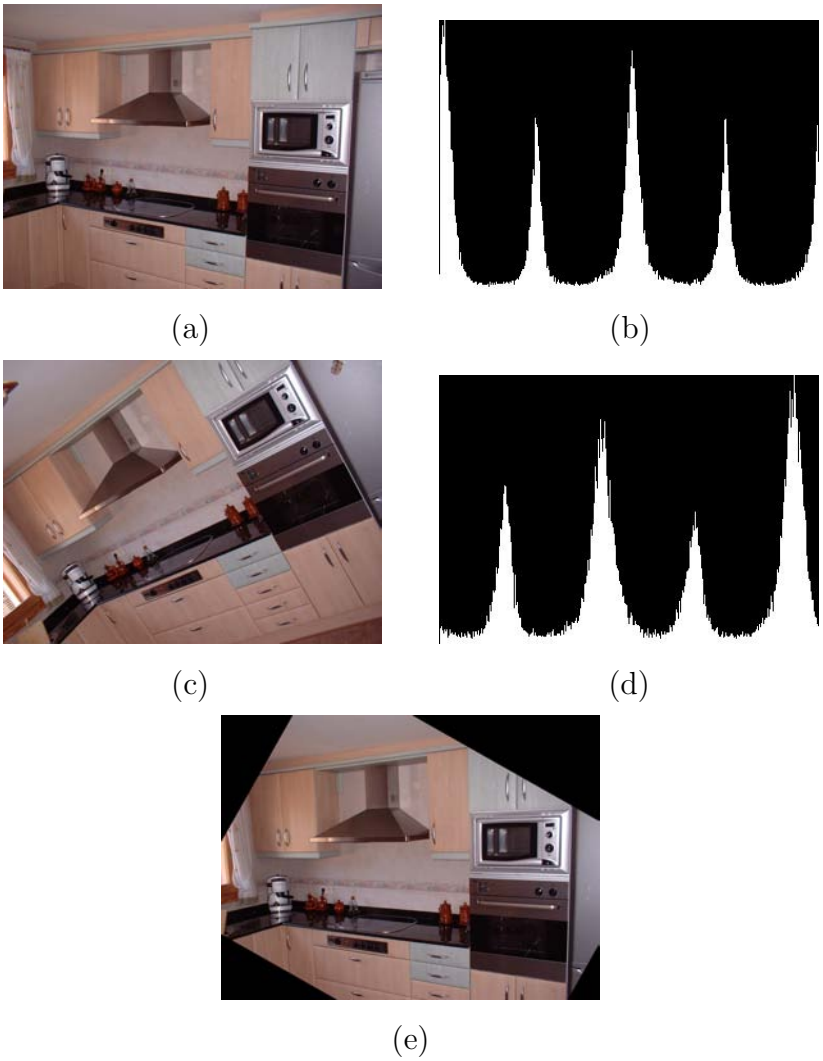


Figure 3.24: *(a) Image of a kitchen interior, (b) Corresponding histogram of gradient orientations, (c) Same image than (a) after a camera rotation, (d) Its orientation histogram, (e) Image in (c) transformed by the inverse of the estimated rotation.*

Chapter 4

Color palette

Abstract. This chapter represents the final step in the search of a qualitative description of the image. In this chapter, a method for the automatic construction of the color palette of an image is described. This method, based on the FTC algorithm of the previous chapter and a hierarchical ordering of the components of the HSI color space, obtains automatically, in three steps, the minimum number of colors that describe an image. We call this minimal set "color palette". The experimental results seem to endorse the capacity of the method to obtain the most significant colors in the image, even if they belong to small details in the scene. The obtained palette can be combined with a dictionary of color names in order to provide a qualitative description of the image. In the chapter are studied different computer graphics color palettes, to which the proposed palette is compared. The results show that classical computer graphics palettes obtain a better visualization of the image, but they obviate the small details in the scene. To avoid some problems of over-segmentation in some images, at the end of the chapter, a modified version of the algorithm, that includes the use of morphological operators is presented.

4.1 Introduction

The human visual system (HVS) is a complex and precise entity that is able both to distinguish millions of colors and to describe an image by naming just a few ones of them. This last characteristic derives from the HVS ability of grouping colors with similar tonality and of assigning a unique name to each group. Humans perform this process effortless but, computationally, it is not an easy task. We aim at the automation of such a process.

In this chapter we propose a method to automatically obtain a qualitative description of the colors in the image. That is, we want to automatically determine what is the minimum number of colors that permits to describe the image and to assign a color code (RGB value) and a name to each one of these colors. We call *color palette*, by analogy to the palette of a painter, this minimum set of colors.

It must be remarked that the term "color palette" is usually used in computer graphics to refer to the look-up table that maps the continuous RGB color space into the finite number of colors displayable by the display hardware. Even if both definitions are related, they involve different concepts, as we will see in the next section.

The proposed method for the obtention of the color palette tries to mimic the way the human visual system classifies colors. In a first step, color information is decomposed

into hue, saturation and illumination, which are magnitudes that can be interpreted intuitively and have a physical meaning ([96]). Then, color clusters are obtained by applying the following hierarchical algorithm: initially, colors are discriminated by their hue; next, colors with similar hue are discriminated by their saturation; finally, colors with similar hue and saturation are discriminated by their illumination. At each step of the algorithm the discrimination between different values of a given magnitude (hue, saturation or illumination) is performed by analysing its associated 1D histograms. The FTC histogram segmentation algorithm described in Chapter 2 is used to parse these histograms. It is worth noting that the presented approach does not consider the spatial information of the image, it only uses the color information.

The paper is organized as follows: Section 4.2 gives an overview of classical methods for the construction of computer graphics color palettes and states the main differences between these palettes and the one proposed in this thesis. In Section 4.3, the hierarchical algorithm for the construction of the color palette is explained in detail. Section 4.4 is devoted to several examples of the application of the proposed algorithm to different images. Section 4.5 presents an improvement of the obtained palette. In this section the palette is combined with a dictionary of color names, which allows a qualitative description of the image and a reduction in the color palette. Section 4.6 describes an application of the obtained palette to the automatic computation of the computer graphics color palette. Finally, a method for reducing some colors in the palette is explained in Section 4.7.

4.2 Computer graphics color palette

The goal of the computer graphics color palette (CG color palette, from now on) is to select a user-defined number of colors among all the possible RGB values, in order to produce a visually pleasant *representation* of the image in the computer display. In this case, the term color palette (also kown as *colormap*) refers to the look-up table that maps the continuous RGB color space into this finite number of colors displayable by the hardware.

The color palette must contain the smallest number of colors that allow an acceptable image representation. Color quantization or color clustering techniques are often used for the obtention of the color palette for a given image.

Quantization is the process of mapping a continuous variable to a discrete set of values [48]. Color quantization commonly refers to the problem of selecting k colors from a color space to represent n ($k < n$) colors from the same space, such that some error function is minimized. This error is usually evaluated by computing the sum of the squared differences between the original pixel colors and the ones obtained from the quantization.

The color image quantization task may be divided into four phases:

1. Sampling the original image for color statistics.
2. Choosing a colormap based on the color statistics.
3. Mapping the original colors to their nearest neighbors in the colormap.
4. Quantizing and redrawing the image (with optional dither).

In general, algorithms for color quantization can be classified into two categories: uniform and tapered. In uniform quantization the color space is divided into k regions of the same size; while in tapered quantization the regions have unequal size. In the latter

case, the color space is divided according to the statistical distribution in the image. Algorithms for uniform quantization are, in general, simpler, but their results are not good. Due to this fact, most of the common quantization methods perform a tapered quantization.

Among the most popular approaches we can cite:

- **Popularity algorithm** ([48]), is one of the simplest forms of tapered quantization. The palette is composed of the most frequent colors in the image (the so-called more popular colors). Every other color is then mapped to the closest popular color. The quantization is performed by creating a histogram (a frequency count of each color) and by sorting the colors in descending order in the histogram. Then, the first k sorted colors are selected. One drawback of this method is that it gives too much importance to the large and uniform colored regions. Thus, the algorithm performs poorly on the images with a wide range of colors, since the details in the image are lost.

- **Median-Cut algorithm** ([48]), whose premise is that each color in the color palette represents the same number of pixels in the original image. This algorithm repeatedly subdivides the color space into smaller and smaller rectangular boxes, until there are as many boxes as requested colors in the final palette. The initial step is to consider the original distribution of color points in the RGB color space. The quantization process is as follows. First, the smallest box that encloses all the colors is computed. Then, the enclosed colors are sorted according to the longest axis of the box. After that, the box is split into two new boxes by means of the median value of the sorted list. This process continues until the colors are distributed into as many boxes as colors requested. The color palette for the output image consists in the weighted average color of each final box. Finally, each original color in a final box is replaced by the weighted average color of the box. This algorithm produces better results than the Popularity algorithm but it is slower.

- **Octree algorithm** ([41]), like the Median-Cut algorithm, it groups the colors into boxes, but the way how the boxes are constructed is different. In this case, the colors are added one by one, sequentially. The algorithm utilizes a spatial data structure that uses the principle of recursive descomposition to represent the data, an octree of depth 8. An octree is a 8-way tree; that is, each node has up to eight children. The first step consists in placing k colors in the tree. Then, the others colors are introduced in the tree following the next rules:

  1. If there are less than k leaves in the tree, then the color is filtered down the tree until either it reaches some leaf node that has an associated representative color or it reaches the leaf node representing its unique color.
  2. If there are more than k leaves in the tree some set of leaves in the tree must be merged (their representative colors averaged) together and a new representative color stored in their parent.

  The reduction process starts on the nodes that have the largest depth in the tree. Once the entire image has been processed in this manner the color map consists of the representative colors of the leaf nodes in the tree. The Octree method can return fewer colors than requested. The computation time with this algorithm is bigger than with the others, but the results are better in some images. The principal

drawback of this method is that two very close colors can be classified in two different nodes.

All these methods partition the color space into a number of clusters defined by the user. When quantizing to very few colors the results can be visually unsatisfactory. In these cases a dithering technique is often used to improve the images. Dithering is the process of juxtaposing pixels of two colors to create the illusion that a third color is present ([48]). To that end the spatial information of the image is used and the basic strategy is to minimize the quantization error. The Floyd-Steinberg dithering algorithm ([39]) generates the best results among all the classical methods. It is based on error dispersion: for each point in the image, first find the closest representative color. Then, it calculates the difference between the value in the image and this representative color and it divides up these error values, distributing them over the neighboring pixels.

Other methods for the computation of the color palette are devised to obtain not only a nice visual representation with a minimum number of colors but also to reduce redundancy in the image data and thus to be used for image compression. In this case, the color palette becomes the codebook ([50]) of the compression method. It is specified in the header of the compressed image and each pixel just references the index of a color in the codebook. In this type of application, the color palette is generated by using some clustering algorithms which makes use of the spatial relations between image pixels. The iterative clustering proposed by Linde, Buzo and Gray (LBG), in [62], is usually used in this context. In this algorithm, an initial set of centers is selected and each pixel is assigned to its nearest center. The next step is to compute the centroids of each cluster so obtained, and to use these centroids as the centers for the next iteration. These algorithm presents two drawbacks: the final centers depend on the initial centers that are chosen, and the LBG algorithm requires a large amount of computation. In literature, there are different improvements of this method, for instance, [62] or [15], where the LBG algorithm is used for improving an initial color palette. In [75] a survey of this techniques is presented.

Contrary to the methods presented in the previous paragraphs, the goal in this thesis is not to obtain the minimum set of colors that provides a satisfactory visual representation of the image (the CG palette) but rather to obtain the minimum set of colors that describe all salient objects in the image. Therefore, the presented palette must be thought of in terms of image *description*, which implies the following main differences:

- Rare colors. Colors belonging to small details in the image, which would probably be dropped from the CG color palette, should be kept in the palette since their presence may be very significant in terms of image understanding.
- Rejection of colors. Colors that enhance the visual perception of the image but that have a small contribution to its description should be rejected. This permits the obtention of a minimal palette.

Moreover, as remarked above, the number of colors in the CG palette is user-defined, while the aim here is to estimate it automatically. In the next section, the method for the construction of such a palette is described. In the experimental section the results obtained with both palettes are compared.

4.3 Color palette construction

As pointed out in the previous section our goal is, given any color image, to automatically generate a palette that contains the "most significant" colors in the image. We want

to obtain these colors by means of the acceptable segmentation of the "appropriate 1D histograms". Then, the next step in the process is the selection of the correct color space whose histograms contribute the essential information to obtain those "most significant" colors. Moreover, the variables of this color space have to be independent, since we have decided to make use of 1D histograms, and they must have a hierarchical order. It is clear that we cannot use the RGB space, because the red, green and blue histograms are strongly correlated.

By using the HSI space (explained in Section 1.5.2), we fulfill the previous requirements. In this space the intensity of the light is decoupled from the chrominance information, which is represented by the parameters hue and saturation. All of these magnitudes can be easily computed from the RGB representation, can be interpreted intuitively and have a physical meaning ([96]). Moreover, the 1D histogram of each variable provides us with important information about the color. In literature, we can find other works which use this space for color quantization and they demostrate the good results in front of RGB color space utilization, for instance [95], where the quantization is computed by segmenting the HSI space in almost equal divisions.

However, prior to the utilization of this color space for the construction of a color palette some remarks must be made:

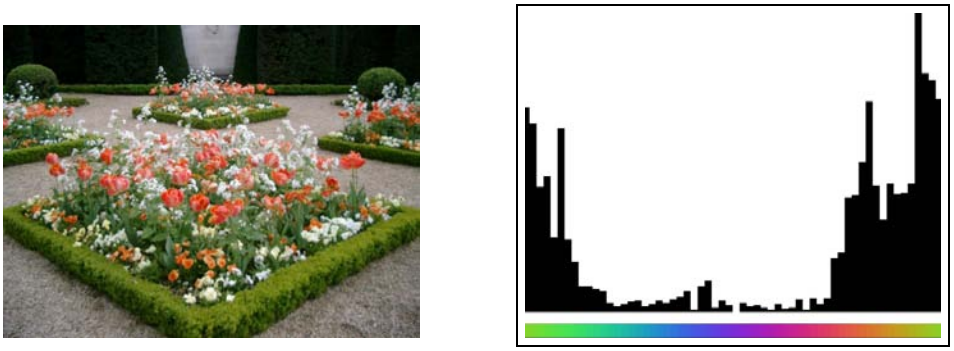
- Not all the color components (hue, saturation and intensity) have the same relevance when comparing two colors: it is known that a slight variation in the hue component may produce a big change in the perception of the color, while a variation of intensity may be slightly perceived. Furthermore, the hue is highly invariant with respect to shadows and reflections. This is the reason why the proposed palette construction algorithm follows a hierarchical order: colors are first grouped according to their hues, then according to their saturations and, finally, according to their illuminations. The algorithm is described in Section 4.3.2.

- An important drawback of the HSI representation is that the hue component is not defined for zero-saturated colors. These colors are located in the line from pure black to pure white in the RGB color cube (the so-called "grey axis"). In practice, and due to numerical limitations in the representation of the color components, this problem also arises when dealing with low saturated colors. Section 4.3.1 proposes a method to overcome this problem.

4.3.1 Quantization problems

As we mentioned before, we consider that the hue component is the variable which provides the pure color information. In practice, we encounter a problem which is illustrated in Figure 4.1: the hue histogram of any natural image always presents little noisy peaks which do not correspond to any of the colors perceived in the image. These peaks are due to the numerical instability of the hue component value in the pixels with low saturation. These pixels are called achromatic (pixels with saturation equal to 0) or near-achromatic.

Many authors have remarked this problem and they have presented different solutions. For example, in [47], Hanbury proposes saturation-weighted hue histograms as a solution to exclude achromatic and near-achromatic pixels in the hue histogram. The basic idea is that the higher saturated pixels receive higher weighting in the hue histogram than the lower saturated ones. Tico et *al.* propose in [88] a new method of color histogram creation based exclusively on the hue component in the chromatic image region and on



(a) (b)

Figure 4.1: *(a) Original image "Bagatelle"; (b) Hue histogram with all the pixels of the image. It exhibits a lot of little peaks corresponding to no hue present in the image.*

the intensity component in the achromatic image region, since they consider that the hue component is meaningless in the achromatic regions.

An accurate study of the problem has led us to conclude that it is related to the quantization level of the hue values, as illustrated in Figure 4.2. Consider the problem of assigning quantized hue values to color points in the neighbourhood of the grey axis; when the saturation (i.e. the distance from the color point to the grey axis) is large, several hue values are valid for the given saturation; however, when the saturation decreases, the set of valid hues also decreases. In general, for a given saturation r, the approximate maximum number of color points at a distance r from the grey axis is $2\pi r$. Therefore, if Q is the number of quantized hue values, then

$$Q < 2\pi r$$

or, equivalently, $r > \frac{Q}{2\pi}$. This means that, in order to prevent quantization problems, we must not consider points whose saturation is smaller than $\frac{Q}{2\pi}$. This requirement defines a cylinder in the HSI color space called the *grey cylinder*. All the points contained in it will be considered as grey values and treated as such in the algorithms.

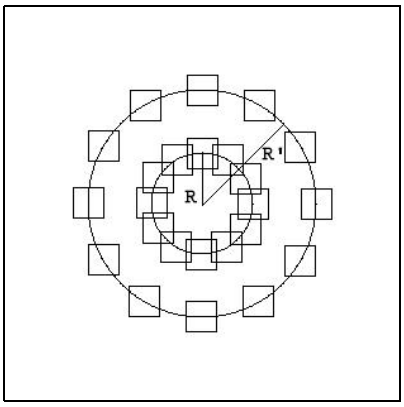
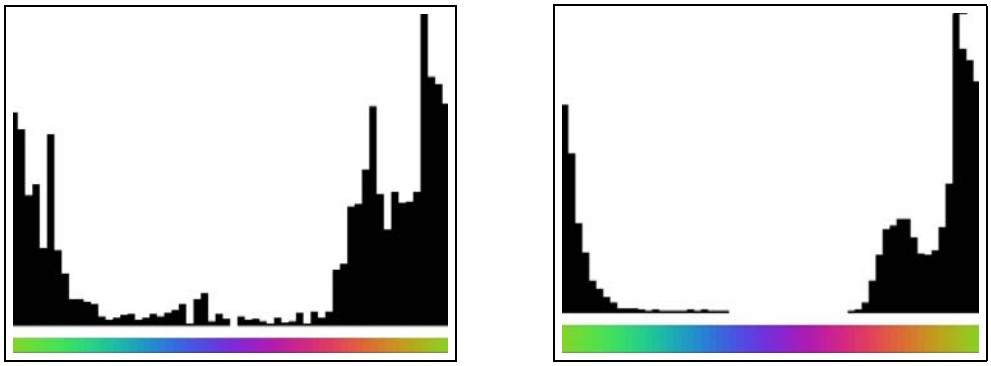


Figure 4.2: *Quantization problem. The angles at distance R' can be represented with 12 values, but at distance R the quantization in 12 values is excessive (there are less than 12 color points at this distance).*

The proposed solutions in [47] and [88] manage to remove the quantization problem spikes, since they eliminate the grey cylinder, whose pixels have a very low saturation. In

fact, removing the grey cylinder, directly without computation, is an easier way to solve the problem. The resulting histogram after the grey cylinder elimination can be seen in Figure 4.3.



(a) (b)

Figure 4.3: *(a) Hue histogram with all the pixels in the image. (b) Hue histogram after the grey cylinder elimination. This histogram is more regular than the previous one, and it does not present noisy peaks.*

In our experiments we use a quantization step of 6° (approximately 0.1 radians) for the hue component, which means that $Q = 60$. Thus, the grey cylinder has a radius approximately equal to 10. This means that, when computing the hue histogram of an image, only hues of points having a saturation larger than 10 are taken into account for the computation of the hue histogram.

However, we observe that, even after applying the previous constraint, some quantization effects still appear. These effects are due to spatial quantization problems: if the image is not accurately sampled (according to Shannon theorem), the representation of an edge between two objects of different colors or hues induces naturally a quantization of the hues of these objects. In practice, this effect apparently disappears if we increase the threshold from the value $r = 10$ to $r = 18$.

4.3.2 Algorithm description

Finally, the automatic color palette (ACoPa) algorithm can be described by the following steps:

Automatic Color Palette (ACoPa):

1. *Apply the FTC algorithm on the hue histogram of the image. Let S be the obtained segmentation.*

2. *Link each pixel of the grey cylinder to its corresponding interval $S_i = [s_i, s_{i+1}]$, according to its hue value.*

3. *For each i, construct the saturation histogram of all the pixels in the image whose hue belongs to S_i. Take into account the pixels of the grey cylinder. Apply the FTC algorithm on the corresponding saturation histogram. For each i, let $\{S_{i,i_1}, S_{i,i_2}, \ldots\}$ be the obtained segmentation.*

4. *For each i and each j, compute and segment the intensity histogram of all the pixels whose hue and saturation belong to S_i and S_{i,i_j}, including those in the grey cylinder.*

Observe that the number of colors obtained from the first step of the method increases when we add successively the saturation and intensity informations. As a result of the algorithm we end up with a minimum set of color clusters. A color can be chosen as a representative for a given cluster by simply computing the mean value of the HSI components of the pixels contributing to the cluster. This minimum set of colors forms the final color palette.

It is worth noting that the hue histogram is circular, which means that the hue value 0° is identified to the hue value 360°. Then, the obtained modes of the hue histograms are also circular. For example, in the hue histogram of Figure 4.4(b), we can see one circular mode from 318° to 84°.

4.4 Experimental results

We begin this section with a complete and detailed example of the ACoPa algorithm. In Figure 4.4 we observe the original "Bagatelle" image (Fig. 4.4(a)) and its hue histogram (Fig. 4.4(b)), which is segmented into three modes using the FTC algorithm. Figure 4.4(c) displays the original image segmented into three regions as the result of assigning a common color to all the pixels that contribute to the same hue mode. This common color is computed as the mean RGB value of the pixels in the mode. The saturation histogram is computed and segmented for each one of the obtained regions in the previous step. Again, a common mean color is assigned to all the pixels contributing to the same hue and saturation mode and the resulting image, which contains five different regions, is shown in Figure 4.4(d). Finally, the intensity histogram is computed and segmented for all of these regions and the colors in Figure 4.4(d) are further divided to obtain Figure 4.4(e). In Figure 4.4(f) we show the hierarchical palette, each row of the palette represents the set of colors obtained at each step of the algorithm, the last row being the final color palette composed by ten colors.

Figure 4.5 displays how the original image in Figure 4.4(a) is split into different regions using the ACoPa method. Figures 4.5(a), 4.5(b) and 4.5(c) show each one of the individual regions obtained after segmenting the hue histogram (pixels contributing to the region are shown with their original color and the rest are set either to black or white). Saturation information is then used to segment each one of these regions: Figure 4.5(b) is split into Figure 4.5(e) and Figure 4.5(f), Figure 4.5(c) is split into Figure 4.5(g) and Figure 4.5(h), while Figure 4.5(a) is not further divided. Finally, intensity information is used to segment the five regions obtained in the previous step. As a result, ten different regions are obtained. The images corresponding to each region are not displayed but the final segmentation is shown in Figure 4.4(e).

A second example of the ACoPa algorithm is shown in Figure 4.6. In this case just the original image and the final segmentation are displayed, together with the obtained hierarchical color palette. We observe that the final palette, the last row of Figure 4.6(c), presents similar colors, particularly black and whites. A method for merging the similar colors in the palette will be presented in the next section. The method lies in associating each final color palette to a name from a color name dictionary. Thus similar colors will be associated to the same name. The resulting palette is shown in Figure 4.6(d).

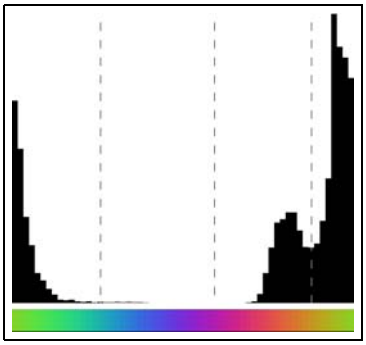
In the following examples we want to show the application of the hierarchical method to the obtention of a color palette using the cylindrical representation of the $CIELab$ space. Indeed this space, which is known for providing a color system in which color distances are proportional to perceptual distances, shares some common characteristics with the

HSI space: intensity information (represented by the L component) is decoupled from the chrominance information (represented by hue and saturation components, computed from the a and b components of the $CIELab$). Therefore, the same reasons that lead us to the utilization of the HSI color system can be invoked to justify the use of the Lab system in the same hierarchical way. The ACoPa algorithm is simply modified by replacing L by I, and by using the hue and saturation computed from the a and b components. In general, the results obtained with both color representations are similar (see Figure 4.7).

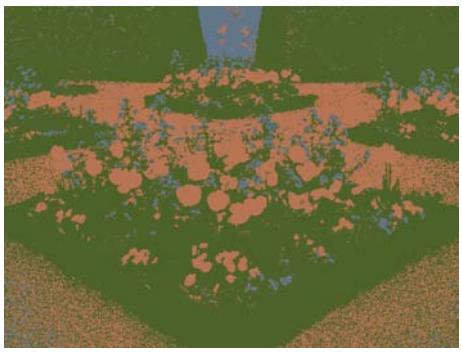
Nevertheless, in some images, we can perceive important differences. This is the case of the example in Figure 4.8. We can observe that the results with the hue, saturation and intensity of the $CIELab$ space are worse: we obtain fewer colors and the resulting image presents little color differences with the original, for instance, the little round pills of "wheat" color are represented with other colors in the resulting image. It is clear that, depending of the color palette application, the final result with $CIELab$ can be more appropriate, for example if we need a small amount of colors to represent the image.

We cannot conclude that the HSI space is better in all cases, it depends on each image. In the cases where we obtain different results with the two spaces, the main differences are in the amount of colors in the final list. But, even in these cases the two spaces correctly represent the main colors in the image.

(a)



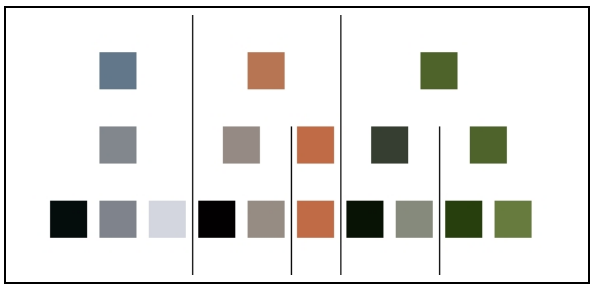
(b)



(c)

(d)

(e)



(f)

Figure 4.4: *Complete example: (a) Original image "Bagatelle", (b) Hue histogram segmentation with the 3 resulting modes, (c) Resulting image with 3 colors after hue segmentation, (d) Resulting image with 5 colors after hue and saturation segmentation, (d) Final image with 10 colors after the hue, saturation and intensity segmentation, (f) Hierarchical color palette: each row displays the colors obtained at each step of the algorithm. The last row corresponds to the final color palette. Similar colors, particularly blacks and whites, will be reunited by using, in a final step, a color dictionary. See text for further details.*

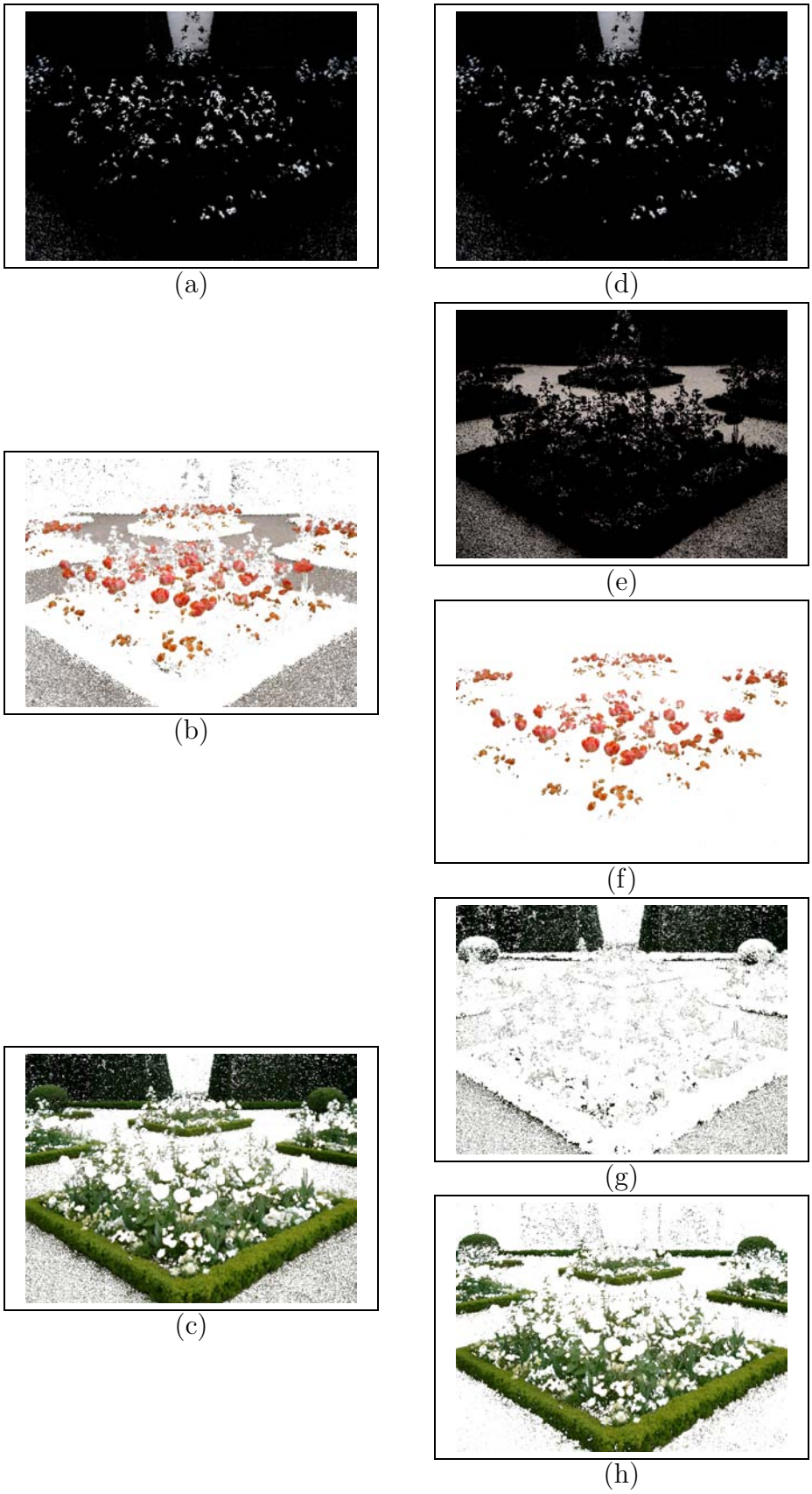
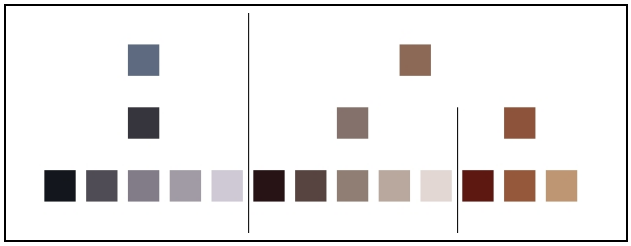


Figure 4.5: *Left column, each one of the individual regions obtained after segmenting the hue histogram of image 4.4(a). Right column, regions obtained after using saturation information: image (b) is split into (e) and (f), image (c) is split into (g) and (h), while (a) is not further divided and the image in (d) is the same than in (a). See text for details.*

(a)

(b)



(c)

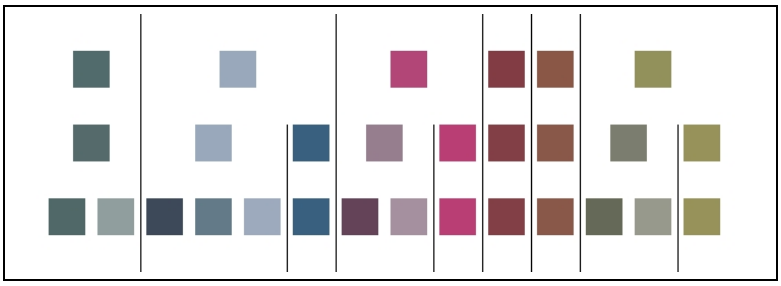
(d)

Figure 4.6: *Example of the ACoPa algorithm: (a) Original image "Girls". (b) Final segmentation. (c) Color palette (13 colors). (d) Names associated to the color palette, ten names (see Section 4.5 for more details).*

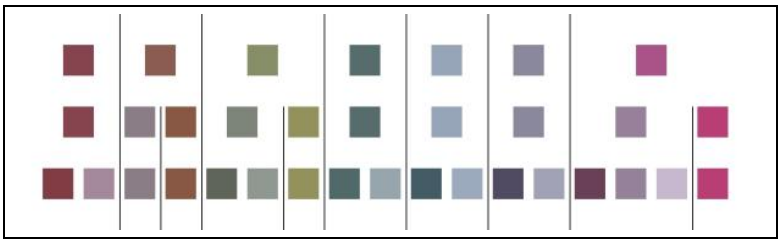
(a)

(b)

(c)



(d)



(e)

Figure 4.7: *Comparison between the results of ACoPa for HSI space and the cylindrical representation of CIELab space: (a) Original image "pills"; (b) Resulting image of ACoPa using HSI space; (c) Resulting image of CIELab space; (d) Color palette in the first case; (e) Color palette in the second case. The second color palette contains three more colors, but the resulting colors are quite similar in both palettes. The resulting segmented images are also very similar.*

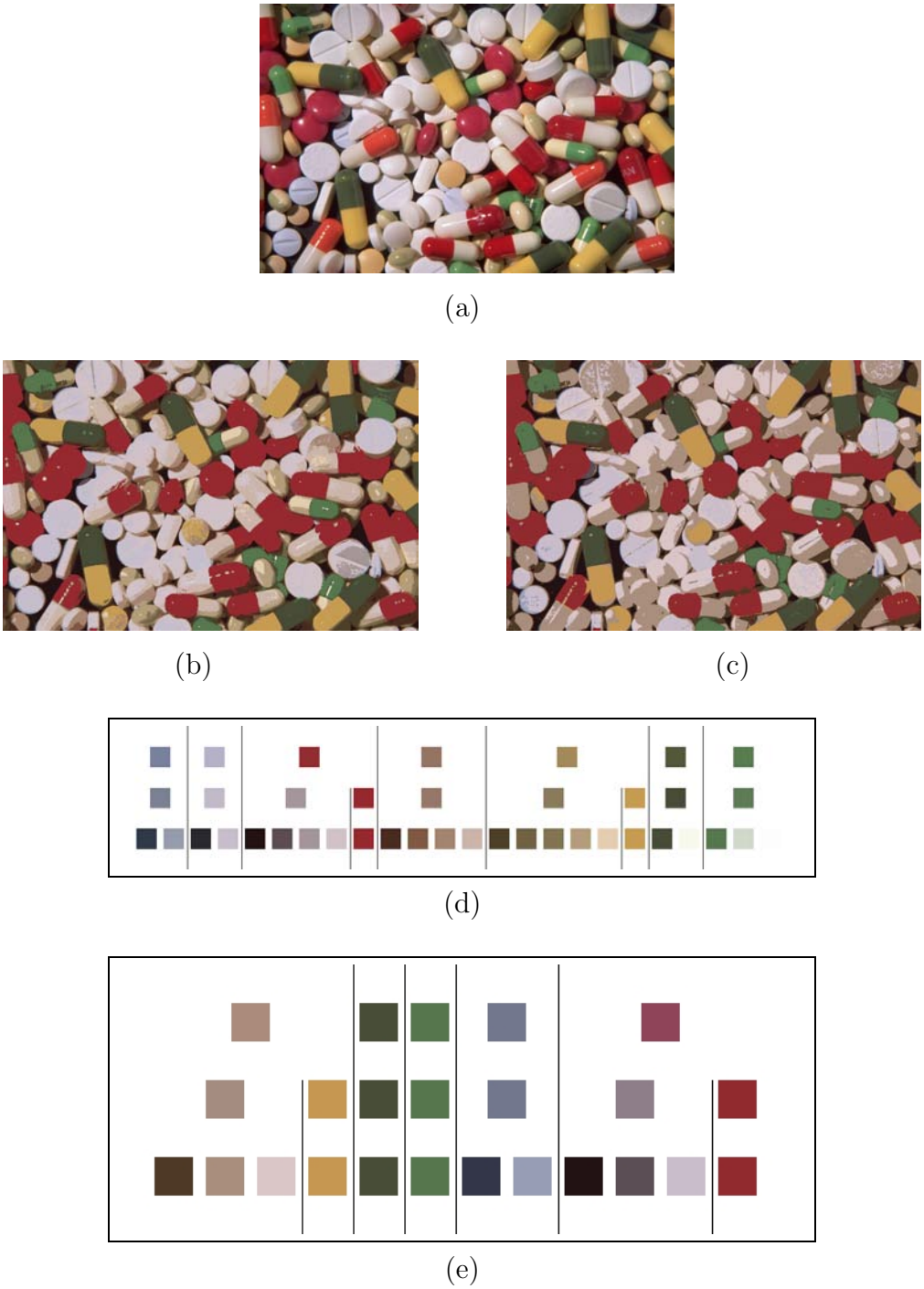


Figure 4.8: *Comparison between the results of ACoPa for HSI space and the cylindrical representation of CIELab space: (a) Original image "pills"; (b) Resulting image of ACoPa using HSI space; (c) Resulting image of CIELab space; (d) Color palette in the first case; (e) Color palette in the second case. The results with the two spaces are different, because the first palette presents a larger list of final clusters. Due to this fact, the resulting image (c) is less similar to the original one.*

4.5 Color names

In some of the previous examples some colors in the final palettes look quite similar, which suggests that they could be merged into a single color without perturbing much the image description. In this section, the combination of the obtained palette with a dictionary of color names shall allow the reduction of the colors in the palette.

The ultimate goal of the proposed method is to describe qualitatively an image by associating a name to each color in the resulting palette. To achieve this goal, we seek for a dictionary of color names, with a good representation and distribution of the millions of existing colors and containing a not excessive but sufficient amount of them.

In the literature there are a lot of color-names dictionaries (see for example [56]). We have limited our search to the ones published in the world wide web and we have considered two of them: the X11 rgb and NBS/ISCC Centroids dictionaries. X11 is the most replicated dictionary among the ones published online and its last modification took place in 1994. However, this dictionary presents some drawbacks: it assigns names to RGB colors outside the sRGB gamut (i.e. the range of colors that can be physically represented) and severely under-represents the darkest octant of the RGB color cube. The NBS/ISCC Centroids is an improvement of a NBS/ISCC dictionary, which tries to be the ideal source for surface color names.

The NBS/ISCC system is a standarized set of color terms. It defines a set of 276 color centroids. This system considers ten base terms (pink, red, orange, brown, yellow, olive, green, blue, violet and purple) and some combination of them (for instance, reddish orange or bluish green), building twenty-eight hues. Then, using eight adjectives, the degrees of saturation and brightness are specified. For example, in Figure 4.9, we can observe the complete set of modifiers for the hue purple. The combination between each hue and some adjectives permits to construct 267 names of colors. Because of non-linearities in our visual system and irregularities in our natural-language system of color names, not every hue has the full complement of modifiers. For instance, Very Light Black or Greenish Red do not exist.

Figure 4.9: *NBS/ISCC system modifiers for the hue purple [70].*

The advantages of the NBS/ISCC Centroids dictionary can be summarized into the following items:

- it has no name conflicts,
- it has colors distributed to equalize perceptual distances between them,

- it is derived from matching physical samples.

The X11 dictionary presents 450 names of colors, and the NBS/ISCC Centroids presents 250.

Independently of the used dictionary, the process that associates the color in the palette to its name in the dictionary is always the same. We use the Euclidean distance and the $CIELab$ coordinates for measuring the difference between the representative color of the dictionary name and the colors in the palette. Thus, we will associate a color of the palette to the name of the dictionary whose representative color is at the minimal Euclidean distance, considering the $CIELab$ coordinates of both colors. The final result is a list of names where the colors of the palette are represented by the associated name in the dictionary and the representative color of the name. The colors are ordered by the amount of pixels that contribute to each cluster. Thus, the first name in the list is the most frequent color in the image and the last one the less frequent.

The association of palette colors to color names usually permits to reduce the final number of colors needed to describe the image: if different colors in the palette are associated to the same name, then the colors are merged into a common cluster and the name appears only once in the final list.

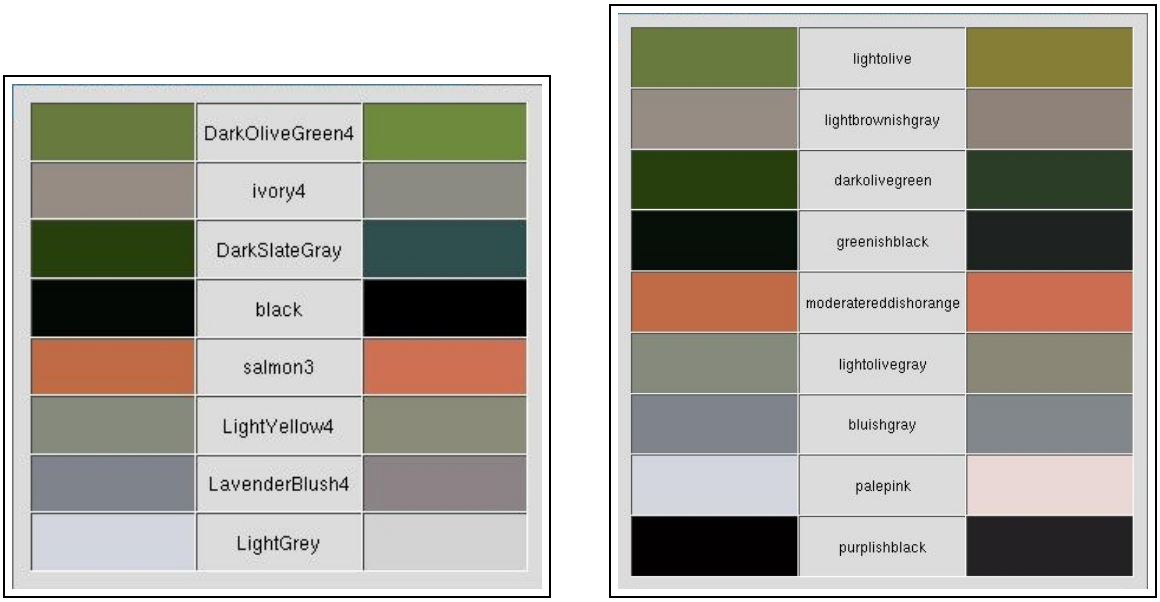
Some examples of the combination of the ACoPa palette with the X11 and the NBS/ISCC Centroids dictionaries are presented in the next section.

4.5.1 Experiments on naming colors

Figure 4.10 displays the lists of color names associated to the color palette in Figure 4.4. Together with the color name it is also displayed, on the left, the representative color of the name in the dictionary, which can be different from the color in the palette (observe, for instance, the third color in the list of Figure 4.10(a)). The lists obtained for the two tested dictionaries are displayed. In this figure we can observe the merging carried out. Thus, in Figure 4.10(a) three colors in the palette are associated to the name "black" and the final list presents only eight colors.

We can observe another example in Figure 4.11. It displays the original image "flowers"(Fig. 4.11(a)), the final segmentation obtained using the ACoPa algorithm (Fig. 4.11(b)) and the hierarchical color palette (Fig. 4.11(c)). The colors in the last row of this image form the final color palette (10 colors). These colors are named according to the procedure described in the previous section and the results are shown in Figure 4.11(d) and Figure 4.11(e) (X11 and NBS/ISCC Centroids dictionaries respectively).

We can compare the different results using both dictionaries. It is clear that the final lists of names obtained using both dictionaries are different since the names are different, but the associated representative colors for each name must be similar. We can compare the results by observing the amount of colors in the lists and the different "mergings" achieved in each case. Moreover, comparing the two columns in the list we observe the adequacy of the dictionary to represent the colors (in the left column the color palette and in the right column the representative color in the dictionary). By observing the results, we can conclude that the X11 dictionary merges more colors than the NBS/ISCC one, despite having more color names. Moreover, in the results we can observe the aforementioned drawback that the X11 dictionary severely under-represents the darkest octant of the RGB color cube. We can note this fact in Figure 4.11. In the list made from the X11 dictionary (Figue 4.11(d)) the three dark colors are represented by only one color (black). On the other hand, in the other list (Figure 4.11(e)) these colors are represented by three different names (greenishblack, oliveblack and brownishblack).



(a) (b)

Figure 4.10: *Lists of color names of Figure 4.4. (a) List computed with the X11 dictionary. (b) List computed with the NBS/ISCC Centroids dictionary.*

In these experiments we have observed that the color is better represented by the representative colors in the NBS/ISCC dictionary. In Figure 4.12 we can see this fact again. It is clear, by comparing the right and the left column, that the represented colors in the X11 dictionary list are less similar to the palette color. The X11 dictionary list merges the first, the third, the fifth, the eleventh and the fifteenth colors in the palette in one single color, the DarkSlateGray, but in the other list each one of them is associated to a different name, except the third and the fifteenth colors, which are merged into an unique color.

In general, the reduction in the number of colors is bigger when using the X11 dictionary, but the assigned color representatives are not always correct. For this reason we can conclude that the NBS/ISCC dictionary is better suited for our application: it permits a reduction of the colors in the palette while allowing a qualitative description of a extensive gamut of colors.

It is worth noting that the reduction, that is carried out by this method, is not excessive, it reduces at most five colors from the final ACoPa. In most part of the images, it only reduces two or three colors. It means that the ACoPa discrimination is very coherent. Most of the obtained colors in the ACoPa correspond to the representative colors in the color names dictionaries. Then the “segmentation” performed by ACoPa algorithm is not finest than the “segmentation” in the dictionaries.

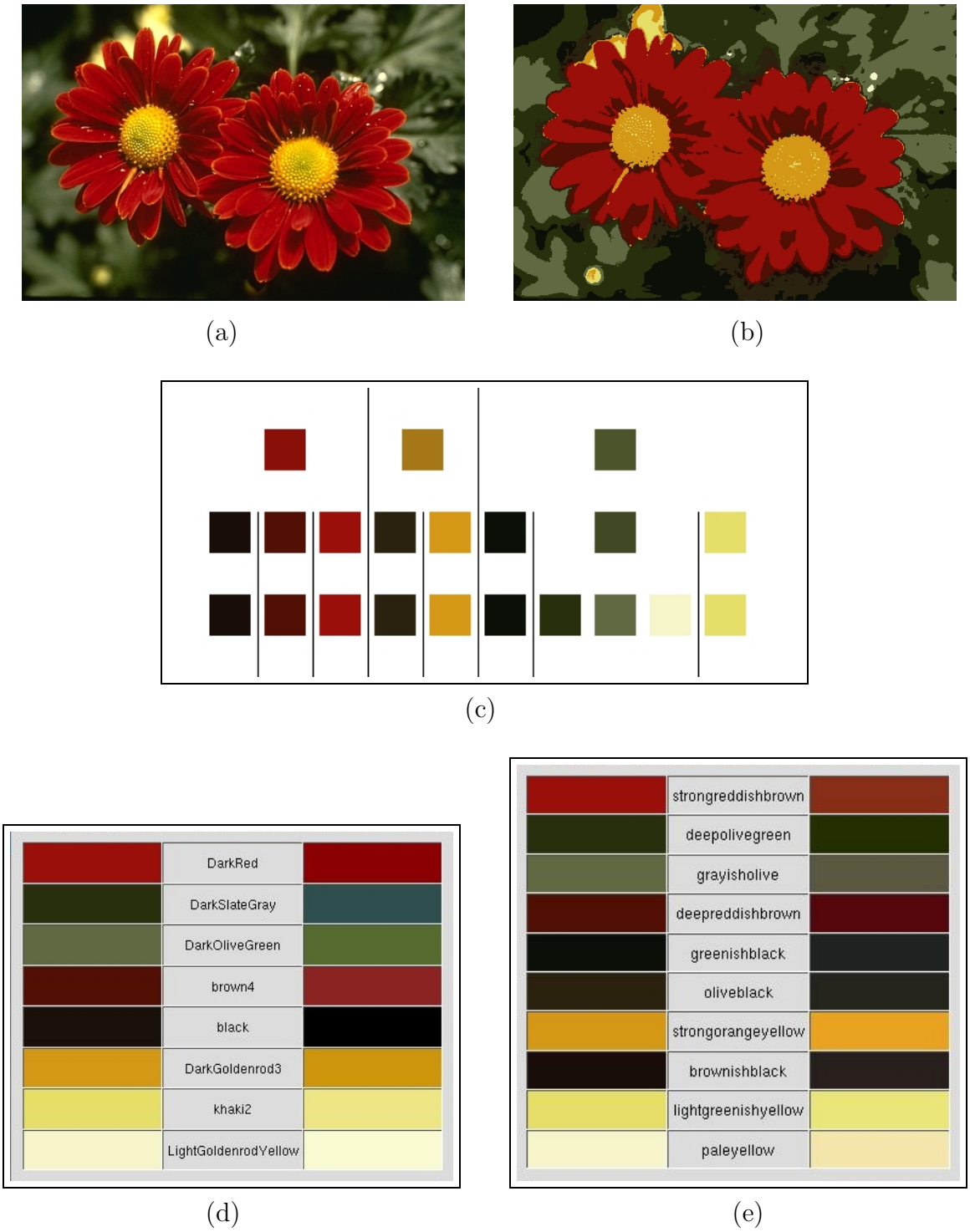
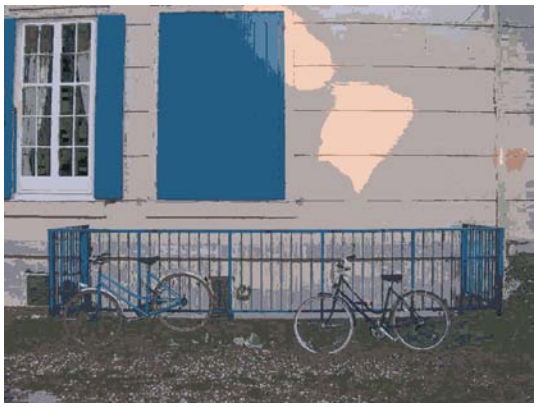
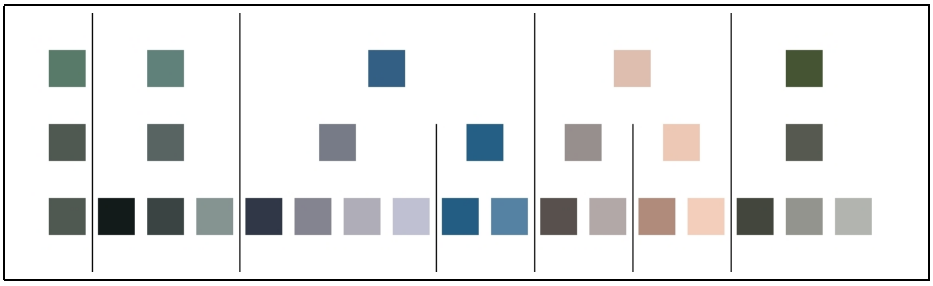


Figure 4.11: *Comparison of the X11 and NBS/ISCC dictionaries: (a) Original image "flowers"; (b) Final segmentation; (c) Color palette; (d) List of names using the X11 dictionary; (e) List of names using the NBS/ISCC dictionary. The list of the X11 presents fewer colors, all the dark ones are merged in one single color. See text for details.*

(a)



(b)



(c)

(d)

(e)

Figure 4.12: *Comparison of the X11 and NBS/ISCC dictionaries: (a) Original image "Bici"; (b) Resulting image; (c) Color palette; (d) List of names using the X11 dictionary; (e) List of names using the NBS/ISCC dictionary. The list of the X11 dictionary represents worst the colors of the palette and performs an excessive merging.*

4.6 ACoPa algorithm and Computer Graphics color palettes

Color palettes for image representation, which we call Computer Graphics (CG) color palettes, have been commented in Section 4.2. In this section we want to compare the ACoPa results with the ones obtained by using the Popularity and the Median-Cut methods, two of the most important techniques for the obtention of CG color palettes.

One of the advantages of the proposed approach with respect to other methods is that it automatically computes the number of colors in the final palette. Moreover, it is able to include in the palette the colors corresponding to small details in the image.

In the experiments, we impose to the Median-Cut and to the Popularity method a number of colors equal to the number obtained with the ACoPa method. Moreover a second parameter is added to the Popularity algorithm for the construction of the histogram of color frequencies. It consists of a quantization parameter for the histogram. If a small value is used, the most popular values tend to concentrate around a small set of colors corresponding to uniform colored regions of the image. In all the experiments, this parameter has been fixed to 16 for each color coordinate.

In the first example (Figure 4.13), the Popularity and the Median-Cut methods cannot detect the color of the little ladybird, since the ladybird represents a little portion of the pixels in the image. The AcoPa method detects it. Even if the AcoPa algorithm does not find a good representation of the colors of the leaf, the background colors are more similar to the original ones than with the other methods.

In the next example, Figure 4.14, we have an image that contains several objects with different colors. The ACoPa palette detects as many as thirty different colors. The Median and Popularity palettes, even when containing the same number of colors, are unable to include the cyan and the yellow colors. However, some objects are better represented with these palettes, as for example the buildings façades. It is also worth remarking that many of the colors in the Popularity palette have a similar shade.

Finally, we present a third example in Figure 4.15. In this example, the ACoPa method gives the best palette, since only in this one we can find the colors of the clothes (orange, yellow and pale green).

From these experiments we can conclude that, even if sometimes the colors in the AcoPa palette are good enough for image visualization (see Figure 4.15(d)), in other cases classical algorithms (specially Median-Cut) provide better results. The reason is that some of the colors in the AcoPa palette correspond to small details in the image, which do not contribute so much to its visualization. Therefore, classical algorithms for CG color palette construction tend to replace these colors by other one's that produce a nicer visual effect (see Figures 4.13 and 4.14).

4.7 ACoPa Improvements

As it has been shown in the previous section, one of the main advantages of the proposed color palette with respect to classical ones is that it is able to display rare colors, that is, colors corresponding to small objects in the image. These colors are detected as small but meaningful modes in the 1D histograms of hue, saturation or intensity. Ideally, one would expect that such small modes would always correspond to actual objects in the scene, but this is not always the case. For example, images in Figure 4.16 display the pixels associated to two of the colors in the palette shown in Figure 4.14(e). These pixels are

sparsely distributed in the image and they belong to the edges between different image regions. Along the edges there are sudden color changes and some of these colors cannot be associated to neither of the limiting regions. As a consequence, these colors tend to form a small peak in the histogram that can eventually be detected as meaningful by the ACoPa algorithm. Since these colors do not correspond to any perceptual object in the scene it is possible to eliminate them from the palette without reducing the accuracy of the image description.

Then, our goal is to reject the colors in the palette that represent a non-compact region in the image. The simplest way to perfom this task is by applying a morphological filter called **erosion** on the image pixels. By definition, the erosion of a set $X \subseteq \mathcal{R}^2$, with structuring element $B \subset \mathcal{R}^2$, is the set

$$E_B(X) = \{x \in X : B_x \subseteq X\}.$$

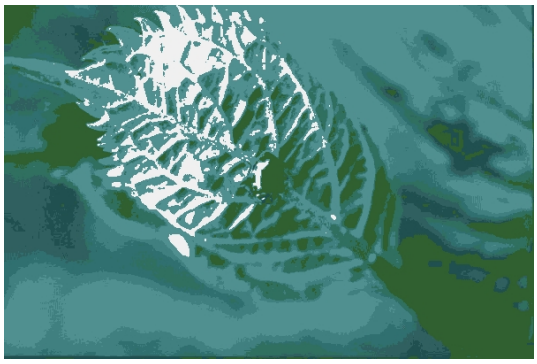
The erosion is defined as the locus of points x such that B is included in X when its origin is placed at x. Normally, we consider the structuring element as an open ball. The radius of this ball determines the amount of filtering performed by the erosion operation. A bigger radius implies that small isolated sets of points will be removed from set X.

The previous definition applies to binary images, on which X represents the set of white (or black) pixels. In our case, for each color in the palette a binary image is created. This image contains all the pixels contributing to the selected color. If for a given erosion radius all the pixels in the binary image are filtered out, then the color is removed from the palette. It is clear that the erosion radius is an important parameter of the method. On the one hand, a high value of the parameter implies an excessive rejection of colors. See, for example, Figure 4.17. In this case, we have applied an erosion with a ball of radius 5 to the image of Figure 4.15(a). This radius is too big and then, the orange color of the clothes was eliminated. On the other hand, the value of the parameter has to be large enough to remove unwanted colors. Experimentally, the optimal value is 2. In the previous example, by applying a radius 2, we reject only one color.

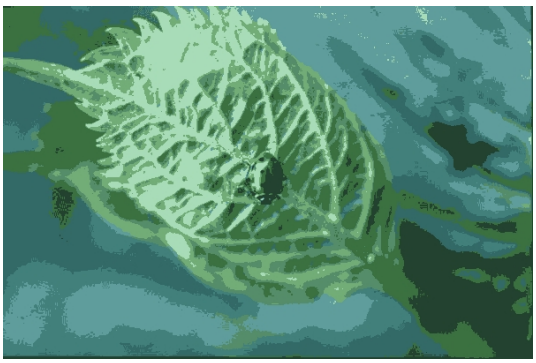
In Figure 4.18 we can observe the evolution of the erosion effect on the image "Madrid" (Fig. 4.14(a)), considering different values of the radius. The original ACoPa result has thrirty colors, which are successively rejected when increasing the radius value.

We can see a last example in which we apply an erosion with a radius value equal to 2 (Figure 4.19). This example is simpler than the image "Madrid". The ACoPa is composed of thirteen colors and after the erosion operation we obtain a palette with nine colors. This example seems to confirm that 2 is the optimal value for the radius, since a higher value would reject the eleventh color of the ACoPa, which corresponds to the branch on the bottom of the image, an important detail.

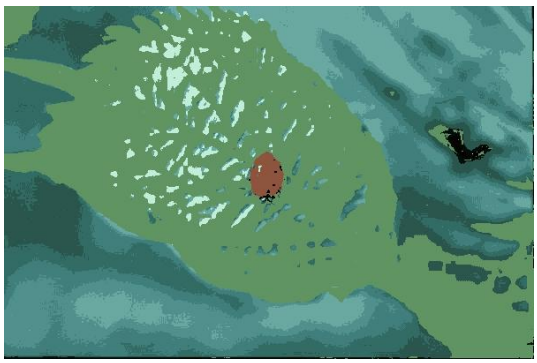
(a)



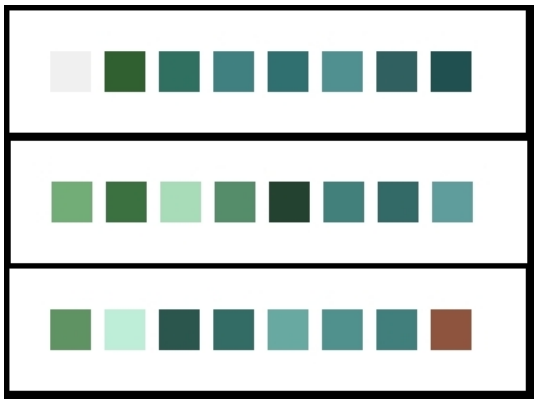
(b)



(c)



(d)



(e)

Figure 4.13: *Color palettes comparison: (a) Original image "ladybird", (b) Resulting image with the Popularity method; (c) Resulting image with the Median-Cut method; (d) Resulting image with the AcoPa method. (e) The three resulting palettes. The representations using the Median-Cut method and the Popularity method are more similar to the original image, but these method's representation presents an important drawback: the color of the ladybird is lost, since it corresponds to a small detail in the image.*

Figure 4.14: *Color palettes comparison: (a) Original image "Madrid", (b) Resulting image with the Popularity method; (c) Resulting image with the Median-Cut method; (d) Resulting image with the AcoPa method. (e) The three resulting palettes. The representation of the image with the Popularity and Median-Cut palettes are more realistic, but some of the image colors are lost.*

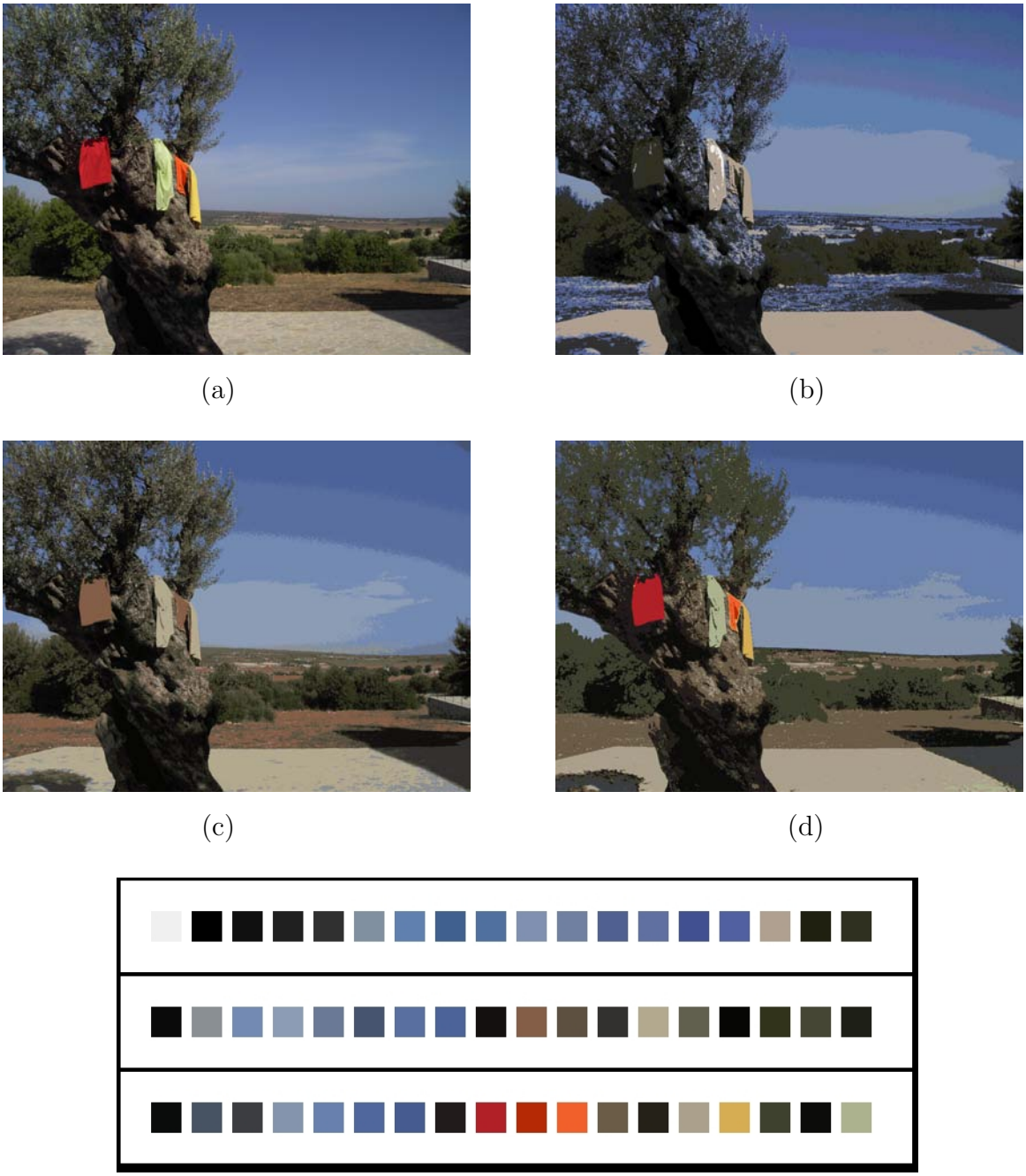


Figure 4.15: *Color palettes comparison: (a) Original image "Olive", (b) Resulting image with the Popularity method; (c) Resulting image with the Median-Cut method; (d) Resulting image with the AcoPa method. The AcoPa algorithm results in a palette of 18 colors. In this case the visual result using the AcoPa palette seems to be better than the rest. Moreover the AcoPa assigns different colors to the different shirts in the tree, but the others do not.*

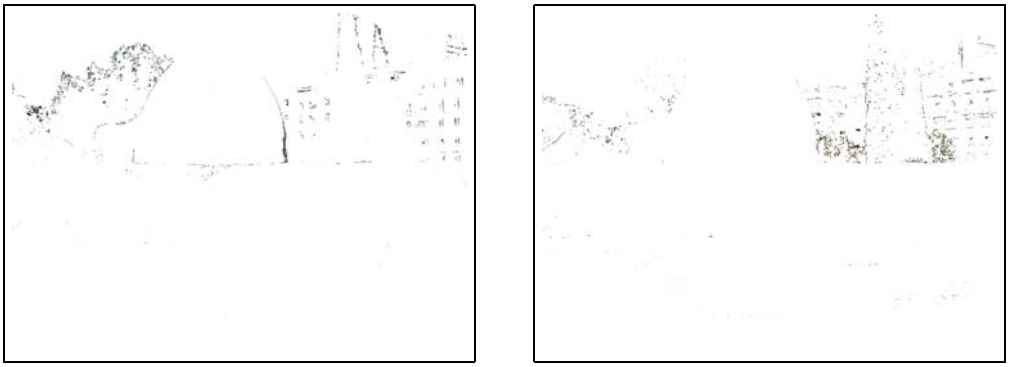
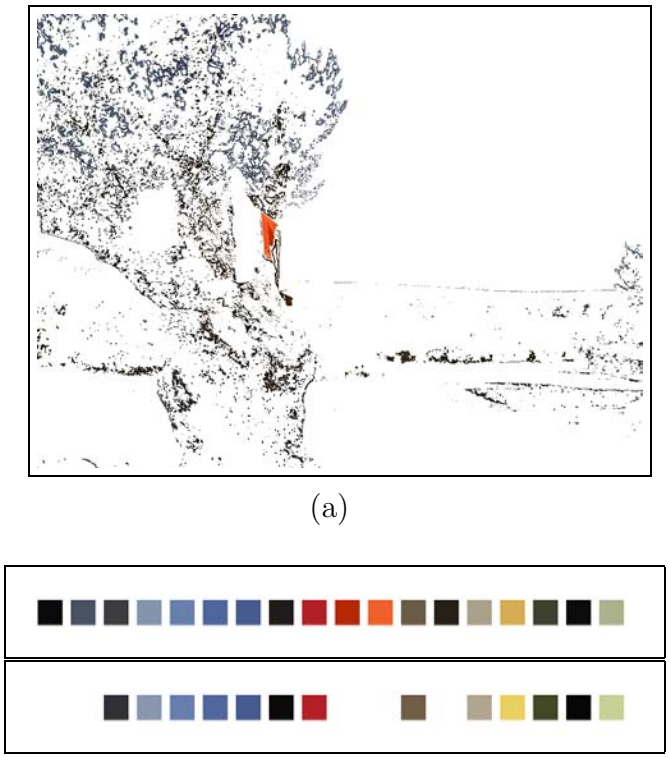


Figure 4.16: *Representation of two regions corresponding to the colors in the ACoPa of the image in Figure 4.14(a). In both cases the pixels are sparsely distributed in the image and they do not correspond to any perceptual object.*



(a)

(b)

Figure 4.17: *Example of the effect of a too high parameter in the erosion. (a) Rejected regions with a erosion of radius 5 in the image "olive". (b) Initial ACoPa and resulting palette after erosion (the rejected colors are displayed in white color in the final palette). We can observe that after erosion the eighteen initial colors are reduced to thirteen.*

Figure 4.18: *We can compare the rejected regions with the erosion operation in the same image, considering different values of the erosion parameter, 2, 3 and 4 respectively. In the initial ACoPa we obtain thirty colors, which are reduced to twenty-five in the first case, twenty in the second case and eighteen in the third one.*

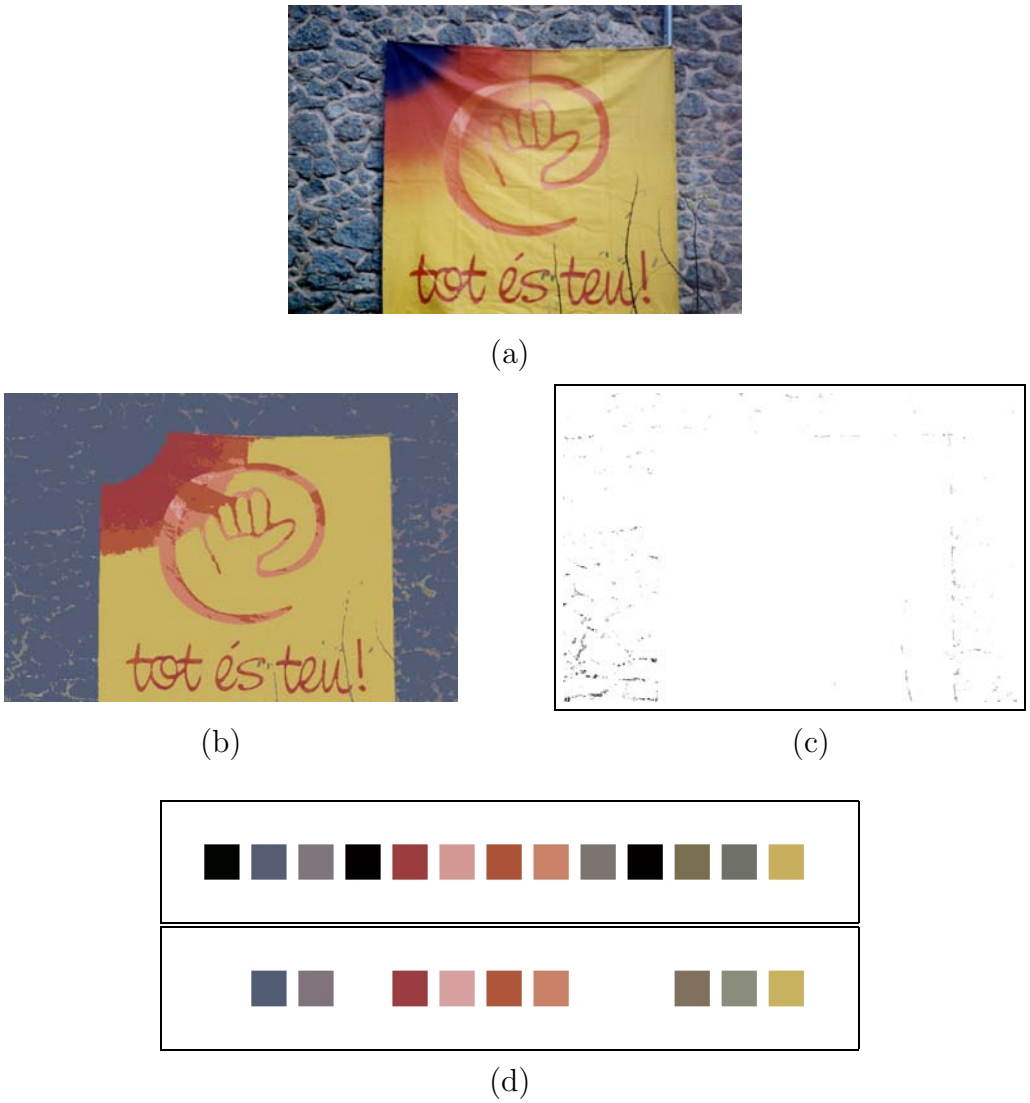


Figure 4.19: *Example of erosion operation: (a) Original image "poster", (b) Resulting image of the ACoPa algorithm, (c) Rejected regions after the erosion operation with radius 2, (d) Comparison between the ACoPa result and the palette after erosion operation.*

Chapter 5

CProcess: a software tool for the study of color

Abstract. CProcess (which stands for Color Processing) is an user-friendly software tool for the visualization, analysis and processing of the color information contained in a digital image. This tool has been created to cover the different needs that arise in the study of color. The first of these needs is to have a good representation of the colors in an image. This representation can be obtained by means of different color spaces, or, by means of others tools such as histogram representations. In this chapter we present a brief user's manual for the CProcess software. The different options are explained and some examples are displayed.

5.1 Introduction

CProcess is an user-friendly software tool for the **visualization**, **analysis** and **processing** of the color information contained in a digital image.

Its visualization utilities include an OpenGL-based tool for 3D visualization of color information in the classical RGB, HSI and $CIELab$ color spaces. 2D representations are also available: projections of the color distribution onto some fixed plane of the color space, projections onto the planes obtained from a Principal Components Analysis (PCA) of the color data, visualization of colors on the Maxwell triangle, etc.

Color information can be analised with the help of 3D, 2D and 1D histograms. 3D histograms show the number of occurrences of each color in some of the aforementioned 3D spaces, while 2D and 1D histograms display this number of occurrences for a pair of color magnitudes (e.g. intensity-saturation histograms) or for a single magnitude (e.g. hue histogram).

Finally, CProcess offers several processing tools, mainly for color quantization. Some of these tools permit to obtain a "color palette" containing the most meaningful colors in the image.

The chapter begins explaining the installation process for the CProcess software. Section 5.3 is a first contact with the capabilities of the software, displaying different examples of utilization. Finally, the different options of the software are explained with detail in the last section.

5.2 Installing CProcess

All the utilities provided by CProcess are operated through a user-friendly graphical interface that runs under GNU/Linux. The main algorithms in CProcess have been pro-

grammed in C++, while the Qt library has been used to build the graphical interface. Qt (http://www.trolltech.com) is a complete C++ development framework which includes a class library and tools for windows management. It also features a unique communication system between classes which permits CProcess to be an interactive tool. Moreover, the API (Application Program Interface) OpenGL has been used for developing 2D and 3D graphical applications.

From the previous comments, it follows that the basic requirements for intalling CProcess are:

- any Linux Distribution that contains libraries for the development of C++ applications (e.g. Linux Mandrake 10.1)

- the full Qt library (including header files and binaries), either versions 2 or 3 (version 4 has not been tested, but it is very likely to work fine).

- the OpenGL library (including header files and binaries)

It must be noted that, usually, any Linux distribution containig the software development packages already includes all the libraries needed for installing CProcess.

Once these requirements are met, the installation process consists of 5 steps:

1. download cprocess.tar.gz from http://dmi.uib.es/~lisani/ana/cprocess.tar.gz

2. copy cprocess.tar.gz to the desired folder (e.g. /home/username/myCProcess), uncompress it (gunzip cprocess.tar.gz) and untar it (tar -xvf cprocess.tar)

3. define the QTDIR environment variable so it contains the path of the Qt library (e.g. QTDIR=/usr/lib/qt3). If using a bash shell, it suffices to add the following lines to the .bashrc file:

   QTDIR="/usr/lib/qt3"

   export QTDIR

4. from the CProcess folder, execute the following command:

   /home/username/myCProcess> source intall.sh

   This command should compile all the files and create the executable file cprocess in the current directory

5. add the CProcess folder to your path. If using a bash shell, it suffices to add the following lines to the .bashrc file:

   PATH=$PATH:/home/username/myCProcess

   export PATH

To check that the installation worked fine, run the cprocess command from any folder in the hard drive:

> cprocess

A window such as the one displayed in Figure 5.1 should open. Enjoy!!

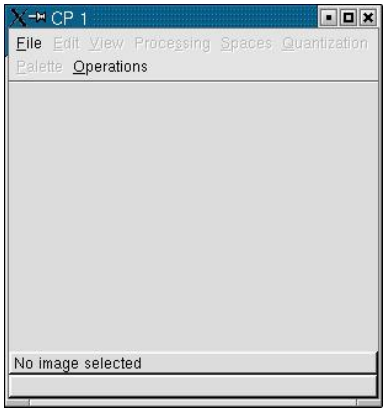


Figure 5.1: *Initial window of CProcess.*

5.3 Getting started

This tutorial-like section presents a complete example of application of many of the CProcess features. In this section we will learn how to: open an image, magnify or reduce the display size, visualize the distribution of colors in RGB space, display the distribution of these colors on the PCA planes, visualize the histogram of RGB color values, display the histogram of hue values and the joint histogram of hue and saturation values, crop a portion of the image and copy it in a new window, quantize colors using the Change-to-Six algorithm, segment an image using the ACoPa algorithm, obtaining the associated color palette, and store the processing results in a new image. For details about the rest of CProcess capabilities refer to Section 5.4.

Let's start by opening an image: go to the File menu and select the Open option. Navigate through the folders tree and select image toypark.png in the /home/username/myCProcess/images folder (we are assuming that CProcess was installed in /home/username/myCProcess). The image is displayed on the screen and some parameters such size and display scale are also shown. It is useful to remark that, by default, the maximum size of the display window is 300×300. Therefore, images bigger than this size are rescaled to fit into this window. The scale parameter is automatically computed and it is the value displayed at the bottom of the window. By using keys P and M the displayed image can be magnified or reduced, respectively. Image size can be reset to its original scale by using the Auto Scale Display option in the View menu. Once the image has been open, the position and RGB and HSI values of the pixel pointed out by the mouse cursor are shown at the bottom of the CProcess window (see Figure 5.2).

In order to visualize the distribution of colors in RGB space, select the RGB Cube option in the Spaces menu. A new window opens and a 3D representation of the colors in this space is shown. This representation can be viewed from different angles by using the window slides. In Figure 5.3 we can observe an example of this action over the previous image toypark.png. When clicking the PCA button the projection of the colors on the three principal planes of the space (obtained by Principal Components Analysis of the color data) is displayed. An example of this new displaying is shown in Figure 5.4. It is also possible to visualize the number of occurrences of each color by clicking the Histogram button. A new 3D window opens and the grey level of each color point in the RGB cube indicates the number of occurrences of this color in the image (from white for 0 occurrences to black to the maximum number of occurrences). A threshold on the minimum number of occurrences of the displayed color points can be set by using the Erase button. The

Figure 5.2: *Image* toypark.png *displayed on a CProcess window. Information about image size, display scale, pixel position and color value is shown at the bottom of the window.*

Figure 5.3: *Three viewpoints of 3D visualization of the image colors on RGB space.*

color histogram of the image toypark.png (Figure 5.2) is shown in Figure 5.5, as well as different results of the Erase operation.

Joint histograms of two color magnitudes and histograms for a single magnitude can also be visualized. For instance, select the Histogram option in the View menu. Several possibilities are available. For example, choose the Histogram Hue-Saturation or the Histogram Hue options. In both cases a new dialog box appears, asking for two parameters. The first one is the threshold that defines the "grey cylinder" (see Section 4.3.1) in the HSI color representation. The second parameter is the quantization step for the histogram computation. The parameters are set by default to their optimal values, so just click OK to confirm the selection. A new image containing the chosen histogram appears. This image can be stored using the Save Image option in the File menu. Just select an image format and a name and confirm with button Save. An example of both histograms is shown in Figure 5.6.

In some cases it is useful to analyse just a small portion of the image. This can done by cropping a rectangular window containing the desired portion: place the mouse cursor on the top-left corner of the chosen window, click the mouse left button and drag the mouse until the bottom-right corner of the window. The chosen window is highlighted in red on the image. This selection can also be performed with the Select option in the Edit menu. Once the window has been selected it can be copied as a new image with the Copy option in the Edit menu. This image can be stored using the Save Image option in the File menu. Figure 5.7 presents an example of these actions.

Let's now give two examples of color processing with CProcess. First use the Change

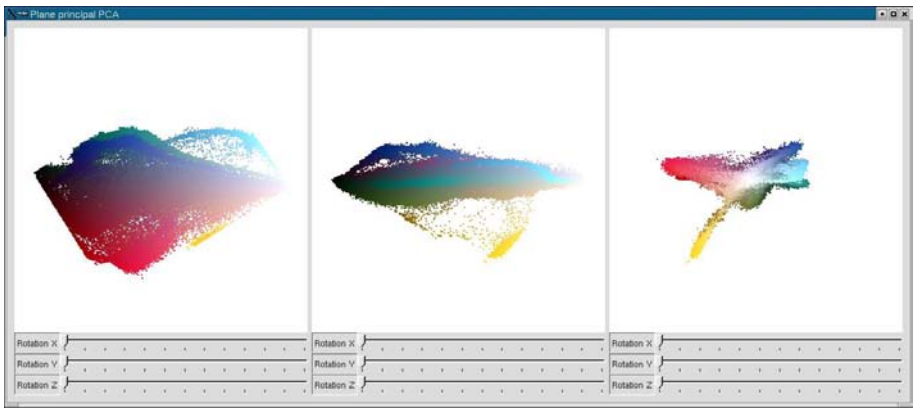
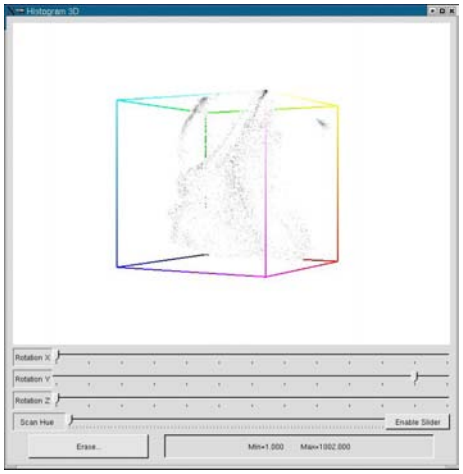
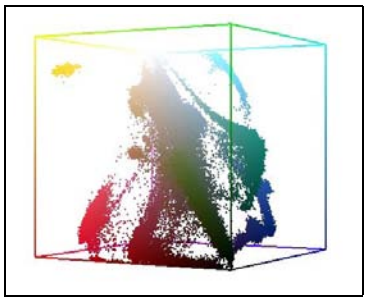


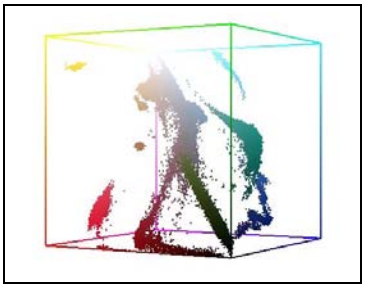
Figure 5.4: *Projection of the image colors on the 3 principal planes of the color distribution.*



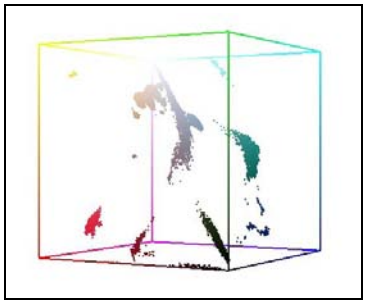
(a)



(b)

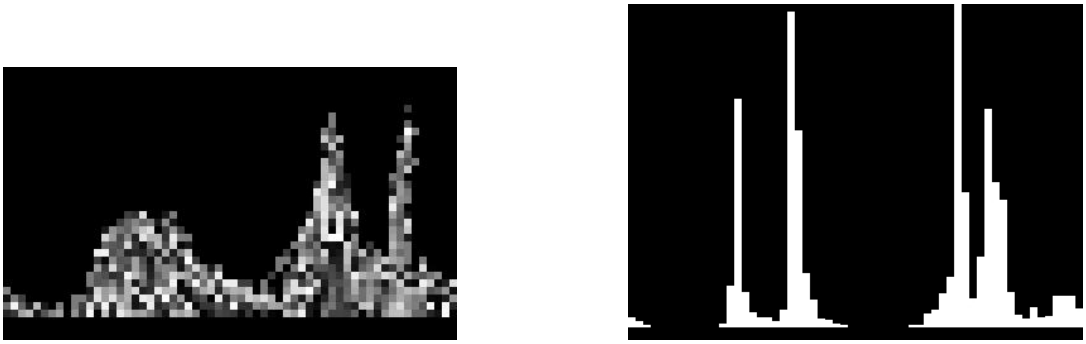


(c)



(d)

Figure 5.5: *(a) 3D histogram of colors in RGB space of the image* toypark.png*. The exterior points of the histogram hide the interior points. The* Erase *tool allows us the observation of the interior of the histogram. (b) RGB cube with the colors with an occurrence higher than 1. (c) RGB cube with the colors with an occurrence higher than 3. (d) RGB cube with the colors with an occurrence higher than 10. The last three images are the result of using three different thresholds in the* Erase *tool.*



(a) (b)

Figure 5.6: *(a) Hue-Saturation histogram for the image* toypark.png*, the saturation values correspond to the vertical axis and the hue values correspond to the horizontal axis. The selected parameters are 18 and 6. The level grey represents the number of ocurrences, from black for 0 occurrences to white to the maximum number of occurrences. (b) Hue histogram for the same image, the horizontal axis represents the hue values and the vertical one the occurrence values. The selected parameters are the same as before.*

Figure 5.7: *Partial selection and* Cut&Copy *results.*

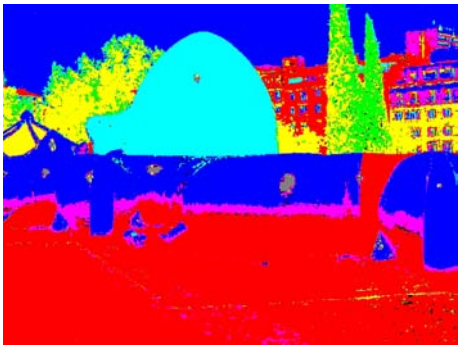


Figure 5.8: *Result of the* Change to 6 *quantization. of* toypark *image. The image is reduced to only six colors. Even though the result is very different from the original, the main elements of the image can be detected.*

to 6 option in the Quantization menu to quantize the image colors to just 6 colors: red, blue, green, cyan, magenta and yellow. A new image is created containing the quantization result (see Figure 5.8). Next, select the FTC HSI (ACoPa algorithm, described in Chapter 4) option of the Palette menu. A dialog box appears asking for the threshold parameter for the HSI "grey cylinder" and the quantization steps for hue, saturation and intensity histograms. Accept the default values and click OK. Several images appear, displaying the segmented hue histogram and the partial results of each segmentation step, the hierarchical palette containing the more significant colors in the original image, the final resulting palette ordered by hue and the list of colors and names associated to this final palette, respectively (see Figures 5.9 and 5.10).

Information about the segmentation process can be obtained interactively by clicking on these images. When clicking on a mode of the hue histogram image, a new image displaying the pixels that contribute to the mode is shown. When clicking on one color of the color palette, two images are shown: one displaying the histogram of values at the selected level (hue for the first row of colors, saturation for the second row and intensity for the third row). In these histograms, the mode to which the selected color belongs is marked in grey. The second image displays the pixels that contribute to this mode. Finally, when clicking in any image containing the partial or final results of the segmentation, a new image appears showing the pixels in the original image whose color has been grouped together with the color of the selected pixel.

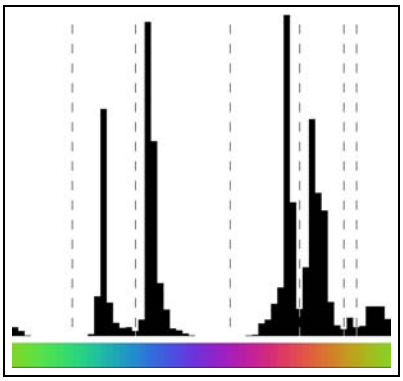
5.4 Menu Options

After a firs contact with the software, this section is devoted to explain with detail all the options in the different menus of CProcess.

5.4.1 File menu

New window
Opens a new CProcess window.

Open
This option is only active when the display window is empty. In order to open a new image, use the New window option first. This option permits to navigate through the



(a)

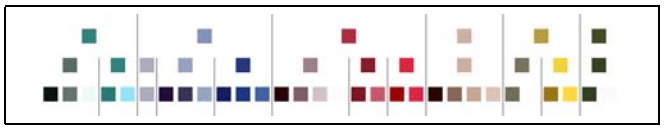
(b)

(c)

(d)

Figure 5.9: *Partial results of the* FTC HSI *option for the* toypark *image. (a) Hue histogram segmented. (b) Resulting segmentation based on hue values. (c) Resulting segmentation based on hue and saturation values. (d) Final segmentation based on hue, saturation and intensity values. At each step the amount of color grows and the image definition improves.*

(a)



(b)

(c)

Figure 5.10: *Resulting palettes of the* FTC HSI *option for the* toypark *image. (a) List of colors and names ordered by number of occurrences. (b) Hierarchical palette, each row corresponds to one step of the process. (c) Final resulting palette, ordered by hue.*

folders tree, to select an image file and to load it into the display window. Available image formats are: BMP, JPEG, PBM, PGM, PNG, PPM, XBM and XPM.

Save Image
This option is only active when the display window is non empty. This option permits to navigate through the folders tree, to select a folder and to save a copy of the current image (which is not necessarily the same that the displayed image, see the description of the Save Display option for further details). Available image formats are: BMP, JPEG, PBM, PGM, PNG, PPM, XBM and XPM.

Save Display
This option is only active when the display window is non empty. This option permits to navigate through the folders tree, to select a folder and to save a copy of the displayed image current image. Remark that this image may be a modifyied version of the one loaded originally: it may have a different scale, it may display different information (e.g. only luminance information when option Display Intensity in the View menu is selected), and can even show a red rectangle selection after using the Select option in the Edit menu. Available image formats are: BMP, JPEG, PBM, PGM, PNG, PPM, XBM and XPM.

Exit
Exits CProcess.

5.4.2 Edit menu

Duplicate Image
It creates a duplicate of the current CProcess window (a clone).

Duplicate Display
It opens a new CProcess window which is loaded with the image displayed in the current window (see the description of the Save Display option in the File menu for further details on the differences between displayed images and loaded images).

Select
This option permits to select a rectangle in the current image. The top-left and bottom-right coordinates of the rectangle must be passed as parameters. Another method for selecting the rectangle is by means of the mouse cursor: place the mouse cursor on the top-left corner of the rectangle, click the mouse left button and drag the mouse until the bottom-right corner. In both cases, once the selection is made the rectangle is highlighted in red in the displayed image.

Copy
This option is only active if some rectangle has been selected on the image. It creates a new CProcess window loaded with the selected rectangle.

Cut
This option is only active if some rectangle has been selected on the image. It creates a new CProcess window that displays the original image whith the pixels belonging to the rectangle in black.

Cut&Copy
This option is only active if some rectangle has been selected on the image. It is a combination of the two previous options.

5.4.3 View menu

Auto Scale Display
By default, the maximum size of the display window is 300×300. Therefore, images bigger than this size are rescaled to fit into this window. The scale parameter is automatically computed and displayed at the bottom of the window. This automatic resizing can be de-activated (to display the image with its original size, that is, scale 1) or re-activated with this menu option.

Display Intensity
Displays a monochrome image created by assigning to each pixel its intensity value.

Display Saturation
Displays a monochrome image created by assigning to each pixel its saturation value.

Display Hue
Displays a monochrome image created by assigning to each pixel its hue value.

Intensity Symmetry
Displays a new image created by replacing the (H, S, I) value of each pixel by $(H, S, |255 - I|)$.

Histogram

Histogram Hue
Displays the histogram of hue values. This option has two parameters: the threshold that defines the "grey cylinder" in HSI space (see Section 4.3.1 for details), and the quantization step of the histogram. Only pixels outside of the "grey cylinder" are used to compute the histogram.

Histogram Saturation
Displays the histogram of saturation values. The same remarks about the histogram parameters stated in the Histogram Hue option apply in this case.

Histogram Intensity
Displays the histogram of intensity values. The same remarks about the histogram parameters stated in the Histogram Hue option apply in this case.

Histogram Hue-Saturation
Displays the joint histogram of hue and saturation values. The histogram values are represented with a grey-level code: high values in the histogram are displayed in bright grey (with white corresponding to the maximum histogram value), while small values are represented with dark pixels (with dark corresponding to 0). Concerning the histogram parameters, the same remarks as in in the **Histogram Hue** option apply in this case. The same quantization step is applied to both variables.

Histogram Hue-Intensity
Displays the joint histogram of hue and intensity values. Same comments as in the previous option.

Histogram Saturation-Intensity
Displays the joint histogram of saturation and intensity values. Same comments as in the Histogram Hue-Saturation option.

Histogram L
Displays the histogram of L values. The default quantization step is 6.

Histogram a
Displays the histogram of a values. The default quantization step is 6.

Histogram b
Displays the histogram of b values. The default quantization step is 6.

5.4.4 Processing menu

Zoom
Creates a new CProcess window containing a scaled version of the current image. Scale parameters can be different for both image dimensions (non-uniform zoom). Linear interpolation is used when the scale parameter is bigger than 1. If the parameter is smaller than 1, low-pass filtering followed by subsampling is used.

Quantize
Displays a new image created by replacing the (R, G, B) value of each pixel by $(q \times [R/q], q \times [G/q], q \times [B/q])$, where [.] denotes the "integer part" operator and q is the quantization step parameter.

Equalize
Displays a new image created by replacing the (H, S, I) value of each pixel by (H, S, I'), where I' represents the new intensity values after intensity histogram equalization.

Specification
Displays a new image created by replacing the (H, S, I) value of each pixel by (H, S, I'), where I' represents the new intensity values after intensity histogram specification. The specified histogram can be obtained from another image or it can be a Gaussian function, depending on the user's choice.

Change Saturation
Displays a new image created by replacing the (H, S, I) value of each pixel by (H, S', I) where S' represents a new saturation value. The pixels in the "grey cylinder" (see Section 4.3.1) are not modified, since a little change in their saturation transforms them in color pixels. This option has two parameters: the threshold that defines the "grey cylinder" in HSI space, and the ratio of saturation change, that is, the increase or decrease grade of saturation.

5.4.5 Color Spaces menu

Maxwell Triangle
Displays a new image that shows the projection of the image colors onto the Maxwell Triangle. This triangle is formed by the intersection of a plane of constant intensity containing the primary colors (R, G and B) and the RGB cube (see Section 1.5.3 for more details). The projection of any color point C onto this plane is given by the intersection of the line containing vector $\overrightarrow{OC}$ with the plane, where $0 = (0, 0, 0)$.

Maxwell Triangle Cursor
Displays a new image that shows the Maxwell Triangle. Whenever the user clicks an image pixel, the projection of the pixel color onto the triangle is marked. The marks can be removed with the Reset option in the Operations menu.

HS Plane

Displays a new image that shows the colors contained in a plane of constant intensity in the HSI space. The intensity level of the plane is specified by the user.

HS Plane Cursor

Displays a new image that shows a plane of constant intensity in the HSI space. The intensity level of the plane is 128. Whenever the user clicks an image pixel, the projection of the pixel color onto the plane is marked. The projection of any color point C onto this plane is given by the intersection of the line containing vector $\overrightarrow{OC}$ with the fixed intensity plane, where $0 = (0, 0, 0)$. The marks can be removed with the Reset option in the Operations menu.

RGB Cube

Opens a new window that displays the distribution of image colors in RGB space. To help visualization, the edges of the RGB cube (see Section 1.5.1 for more details) are outlined. The color of the points along these edges changes accordingly to their position in the cube[1].

Three sliders (Rotation X, Rotation Y and Rotation Z) serve to view the RGB cube from different viewpoints. Scan Intensity, when enabled, permits to visualize the colors having a given intensity level, specified by the user through the slider. To help visualization a new window opens, displaying the plane of constant intensity. Scan Hue works similarly, displaying planes of constant hue.

The Histogram button opens a new window that displays the 3D histogram of RGB color values. The grey level of each color point indicates the number of occurrences of this color in the image (from white, for 0 occurrences, to black, for the maximum number of occurrences). The minimum and maximum number of occurrences of the image colors are displayed at the bottom of the window. A threshold on the minimum number of occurrences of the displayed color points can be set by using the Erase button. Sliders (Rotation X, Rotation Y and Rotation Z) permit to visualize the histogram from different viewpoints. As in the RGB cube window, Scan Hue, when enabled, displays the colors shown in the 3D histogram having a given hue level, specified by the user through the slider. To help visualization a new window opens, displaying the plane of constant hue.

The Convolution Histogram button works similarly to Histogram, with the only difference that the displayed 3D histogram has been previously filtered by convolution with a 3D gaussian kernel. The standard deviation of the kernel is specified by the user.

The PCA button opens a new window showing the projection of the colors on the three principal planes of the space (obtained by Principal Components Analysis of the color data).

RGB Cube Cursor

This option is similar to RGB Cube but it just displays the points in RGB space selected by the user by clicking an image pixel. Several points can be displayed in the same window by successive clicking. The Reset button permits to reset the selection.

HSI

This option is similar to RGB Cube, but in this case the image colors are displayed in HSI space. In the representation the I axis is always displayed.

HSI Cursor

This option is similar to RGB Cube Cursor, but for colors in HSI space.

[1]Remind, for example, that Red= $(255, 0, 0)$, Green= $(0, 255, 0)$, Blue= $(0, 0, 255)$, Black= $(0, 0, 0)$, and White= $(255, 255, 255)$.

CIELab
This option is similar to RGB Cube, but in this case the image colors are displayed in $CIELab$ space. The three axes are displayed in the representation.

CIELab Cursor
This option is similar to RGB Cube Cursor, but for colors in HSI space.

5.4.6 Color Quantization menu

Change to 6
Applies the Change to 6 algorithm to the current image and opens a new CProcess window to display the result. Change to 6 quantizes the images in just 6 colors: red, green, blue (primary colors) and yellow, magenta, cyan (secondary colors). Each (R, G, B) value is assigned one of these colors according to the following algorithm:

1.- Order the R, G and B values in decreasing order and find the *maximum*, *medium* and *minimum* values.

2.- If $maximum - medium \geq maximum - medium$ assign the color corresponding to the maximum value (i.e. red if *maximum*=R, green if *maximum*=G, blue if *maximum*=B).

If $maximum - medium < maximum - medium$ assign the color corresponding to complementary of the minimum value (i.e. cyan if *minimum*=R, magenta if *minimum*=G, yellow if *minimum*=B).

Change to 6+grey
Similar to the previous option with a slight variation of the algorithm:

If $maximum - medium = maximum - medium$ assign a grey value=$\frac{R+G+B}{3}$.

Change for max
Opens a new window displaying an image created by replacing the (R, G, B) value of each pixel by $(255 \times \frac{R}{\max\{R,G,B\}}, 255 \times \frac{G}{\max\{R,G,B\}}, 255 \times \frac{B}{\max\{R,G,B\}})$.

Change to saturate (Maxwell Triangle)
Opens a new window displaying an image created by replacing the (R, G, B) value of each pixel by a color in the Maxwell Triangle having the same hue (computed with the classical formulas, Section 1.5.2) and maximum saturation, that is, lying on the edge of the triangle.

Change to faces of cube
Opens a new window displaying an image created by replacing the (R, G, B) value of each pixel by a color having the same hue and illumination (computed with the new formulas, Section 1.5.2) and maximum saturation, that is, lying on the faces of the RGB cube.

Fine to Coarse (FTC)
Applies the FTC algorithm to the 1D histogram of some color magnitude (the components H, S or I for the HSI space, the components L, a or b for the $CIELab$ space and the cylindrical components hue and chroma for the $CIELab$ space). The magnitude is selected by using a dialog box, together with the quantization step of the histogram and the grey level threshold that defines the "grey cylinder" (only needed for the HSI components). Two new CProcess windows open, displaying the segmented histogram and the quantized image, where the color of each pixel has been replaced by the mean color of all the pixels belonging to the same histogram mode.

Information about the segmentation process can be obtained interactively by clicking on these images. When clicking on a mode of the segmented histogram, a new image displaying the pixels that contribute to the mode is shown. When clicking in the quantized image, a new image appears showing the pixels in the original image whose color has been grouped together with the color of the selected pixel.

Coarse to Fine (CTF)
Similar to Fine to Coarse (FTC), but applying the CTF algorithm to segment the histogram.

5.4.7 Palette Menu

Popularity
Displays the Computer Graphic palette obtained with the Popularity algorithm (see Section 4.2 for more details) and the image quantized with the palette colors. The number of colors in the palette is selected by using a dialog box.

Median Cut
Similar to Popularity option, but applying the Median Cut algorithm (see Section 4.2) to obtain the palette.

FTC HSI
Applies the hierarchical ACoPa segmentation algorithm in HSI space (described in Section 4.3) to the current image. First, a dialog box appears asking for the threshold parameter for the HSI "grey cylinder" and the quantization steps for hue, saturation and intensity histograms. The segmented hue histogram and the partial results of each segmentation step are displayed in new CProcess windows. Moreover, the hierarchical palette, the list of resulting color names and the final palette ordered by hue are shown in new windows.

Information about the segmentation process can be obtained interactively by clicking on these images. When clicking on a mode of the hue histogram image, a new image displaying the pixels that contribute to the mode is shown. When clicking on one color of the color palette, two images are shown: one displaying the histogram of values at the selected level (hue for the first row of colors, saturation for the second row and intensity for the third row). In these histograms, the mode to which the selected color belongs is marked in grey. The second image displays the pixels that contribute to this mode. Finally, when clicking in any image containing the partial or final results of the segmentation, a new image appears showing the pixels in the original image whose color has been grouped together with the color of the selected pixel.

CTF HSI
Similar to the previous option, but applying the CTF algorithm to segment the histograms.

FTC CIELab
Similar to the FTC HSI option, but applying the ACoPa algorithm in $CIELab$ space.

CTF CIELab
Similar to the previous option, but applying the CTF algorithm to segment the histograms.

5.4.8 Operations menu

Most of these options can be used without loading an image in the CProcess window, they are used to study general properties of the color spaces.

Reset
Removes the marks in the Maxwell Triangle and in the HS Plane done with the Maxwell Triangle Cursor and HS Plane Cursor options, respectively.

Planes Fix Intensity
Opens a new window that displays planes of constant intensity level (selected by the user with the slider) on the classical HSI cylinder and on the RGB cube. In the later case, the cube can be visualized from different viewpoints, by using the Rotation X, Rotation Y and Rotation Z sliders.

Planes Hue
Opens a new window that displays planes of constant hue level (selected by the user with the slider). The HSI values are computed using the new formulas computed in Section 1.5.2.

Planes CIELab
Display a representation of the $CIELab$ space. Three sliders (Rotation X, Rotation Y and Rotation Z) serve to view the $CIELab$ space from different viewpoints.

Gaussian noise
Displays a new CProcess with the image corrupted with a Gaussian noise (standard deviation 2).

VLC (Very Large Cube)

VLC Computation
Opens a set of color images and adds the pixels to a data base containing all the RGB colors in these images. This data base is called "Very Large Cube" (see Section 1.6 for more details). The database permits to know the number of occurrences of each color in the set of images.

VLC Visualization
Opens a new window that displays the distribution of colors in the VLC data base in RGB space, with the same tools that the RGB Cube option.

Red Histogram VLC
Histogram of Red values stored in the VLC database.

Blue Histogram VLC
Histogram of Blue values stored in the VLC database.

Green Histogram VLC
Histogram of Green values stored in the VLC database.

Hue Histogram VLC
Histogram of Hue values stored in the VLC database.

Saturation Histogram VLC
Histogram of Saturation values stored in the VLC database.

Intensity Histogram VLC
Histogram of Intensity values stored in the VLC database.

Segmentation VLC
Applies the ACoPa algorithm to the values in the VLC database.

Chapter 6

Conclusions and future work

In this work we have presented a new method, the ACoPa algorithm, for the qualitative description of a color image. The result of the method is a list of the color names that allow us to describe the color image. The advantages of the method are:

- **Automation.** The amount of colors in the final result is a data computed by the method. By means of the segmentation histogram method, the different meaningful colors in each step are detected automatically. In the classical computer graphics color palettes, such as Median Cut, this number is an input data. Moreover, the segmentation histogram method is a parameter-free algorithm, then, the only parameters used in the color palette method are the quantization parameters.

- **Accuracy.** The qualitative description of a color image done by the human visual system is very precise. The little details in a image, such as a ladybug in a leaf (Figure 4), are perceived clearly by our visual system. The goal of the presented method has been to imitate this accuracy. Thus, the method is able to detect colors which are present in a few pixels in the image, such as the "ladybug" example. These meaningful but little details the more are isolated, in reference to color information, the more are detected. Other computer graphics color palettes or segmentation methods give up these details in order to obtain a pleasant visual result.

- **Speed.** The ACoPa method obtains the results in real time. Computation time is very fast, due to the simplicity of the operations on 1D histograms and to the hierarchical organization of the algorithm.

- **Flexibility.** The ACoPa method makes use of the HSI color space, but the process is similar if we want to use a different color space. The requirement is only that the color space components have a hierarchical order. In fact, some results using the cylindircal representation of the $CIELab$ space are presented in this thesis. The method is also independent of the color names dictionary, therefore it is flexible to different dictionaries. In the thesis are compared the results using two different color names dictionaries.

- **Spatial independency.** As we have mentioned before, the ACoPa results are obtained using only the color information. The effectiveness of the method is surprisingly good considering that the method uses only the histograms of the different components of the color space. Even though the results are acceptable, we present an improvement of the method using spatial information.

An important result of this thesis is a new histogram segmentation algorithm, called FTC algorithm. FTC algorithm is the basis of the process in the ACoPa method and it

segments the histogram without a priori assumptions about the number of modes. This method is based on the idea that the meaningful modes follow an unimodal hypothesis, that is, they are increasing in a first part and decreasing in a second part. Therefore a new theory is developed with the aim of defining an "admissible segmentation". In a first approach, the method is based on the notion of meaningful event, detecting meaningful gaps and modes against an *a contrario* uniform law. We observe that the unimodal hypothesis is easier to test using a confirmation model. Then, in a second approach, we introduce the meaningful rejection notion and we will decide that the estimation is correct if meaningful rejections do not exist. The "admissible segmentation" allows us to develop the FTC algorithm, obtaining an algorithm automatic and parameter-free, that avoids under and over segmentation.

The different experiments and applications of the FTC algorithm presented in this thesis demonstrate the usefulness and accuracy of the algorithm.

Future work

The main project in our future work is to complete the "Very Large Cube" study. As we have explained before, this project intends to compute the statistical distribution of the real life colors, that is, the colors that exist around us. In this thesis we have presented some preliminary conclusions, but a deeper and comprehensive analysis is necessary. The first step to complete this study is the introduction of more images in the database. Now, we dispose of 15800 images and the final goal is 30000 images. When the database is complete, the different histograms will be analysed and a statistical distribution to fit the database will be computed. This processing will allow us to conclude if a given color image has a "normal" distribution of colors, where "normal" distribution will be defined as the distribution close to this statistical distribution computed with the "Very Large Cube".

Other project in the future work is the introduction of the spatial component in the color palette method. As we have mentioned before, the proposed method only takes into account the color information. Then, the method can be improved considering the spatial information. A first approximation of this idea is presented in this thesis using the erosion filter. The goal in the future is the utilization of this spatial information to detect the structural objects in the image and to obtain a more precise qualitative description of the image.

Appendix A

EM Algorithm

A.1 Introduction

We say that a random variable has a finite mixture distribution when its density function can be written as a finite weighted sum of known densities. Finite mixture distributions have been used as models for estimating an unknown probability density function. A multitude of fields of application, which demonstrates the mixture models usefulness, exists ([89]): fisheries research, sedimentology, medical diagnosis,... These applications are characterized by the presentation of a set of experimental units, which can be measured, but whose individual class-memberships are unavailable. One of the main problems of the finite mixture distribution is to make inferences about its unknown parameters. Different estimation algorithms exist in the literature. They can be classified into six methods ([89]):

- Graphical methods,
- Method of moments,
- Maximum likelihood,
- Bayesian methods,
- Minimum distance estimation.

The Gaussian mixture model is one of the best known models, due to its good approximation properties. In this case, the unknown parameters are means and covariances of the different gaussians. Most often, the estimation of the parameters of the gaussian mixture is carried out by the maximum likelihood method, aiming at maximizing the likelihood of a set of samples drawn independently from the unknown density.

One of the algorithms often used for Gaussian mixture modeling is the expectation-maximization (EM) algorithm, a well-known statistical tool for maximum likelihood problems [29]. The popularity of EM is due to its simple implementation together with the guaranteed monotone increase of the likelihood of the training set during optimization.

In this appendix we present the mathematical model for Gaussian mixtures, describe the EM algorithm, explain the different limitations of the algorithm and, finally, detail three different statistical tests for distributions comparison.

A.2 Gaussian Mixtures

Consider a mixture model with $M > 1$ components in $\mathcal{R}^n$ for $n \geq 1$:

$$p(x;\Theta) = \sum_{m=1}^{M} \alpha_m p_m(x;\theta_m), \qquad \forall x \in \mathcal{R}^n,$$

where $\alpha_1, \ldots, \alpha_M$ are the mixing proportions, each p_m is a density function parameterized by θ_m, where θ_m is the set of parameters defining the mth component, and $\Theta \equiv \{\theta_1, \ldots, \theta_M, \alpha_1, \ldots, \alpha_M\}$ is the complete set of parameters needed to specify the mixture. Being probabilities, the α_m must satisfy:

$$\alpha_m \geq 0, \quad m = 1, \ldots, M, \quad \sum_{m=1}^{M} \alpha_m = 1$$

For the Gaussian mixture, each component density $p_m(x;\theta_m)$ is a normal distribution:

$$p_m(x;\theta_m) = \frac{1}{\sigma_m\sqrt{2\pi}}\ exp\left[\frac{-(x-\mu_m)^2}{2\sigma_m^2}\right],$$

and $\theta_m = (\mu_m, \sigma_m)$.

Thus, the gaussian mixture model can be described as:

$$p(x;\Theta) = \sum_{m=1}^{M} \alpha_m G(x;\mu_m,\sigma_m),$$

where $G(x;\mu_m,\sigma_m)$ is a Gaussian distribution with mean μ_m and covariance σ_m.

For the estimation problem we assume a training set $X = \{x_1, \ldots, x_N\}$, of N independent and identically distributed samples with normal distribution p. Therefore, the resulting density for the samples is a function of Θ given by

$$\mathcal{L}(\Theta|X) = p(X;\Theta) = \prod_{i=1}^{N} p(x_i;\Theta).$$

This function $\mathcal{L}(\Theta|X)$ is called the likelihood function. In the maximum likelihood problem, the goal is to find the optimum vector Θ^* of the $3M$ parameters of the mixture,

$$\Theta^* \equiv \{\mu_1^*, \ldots, \mu_M^*, \sigma_1^*, \ldots, \sigma_M^*, \alpha_1^*, \ldots, \alpha_M^*\},$$

that maximizes the likelihood function

$$\Theta^* = arg\ max_{\Theta}\mathcal{L}(\Theta|X).$$

Often we maximize $\log(\mathcal{L}(\Theta|X))$ instead because it is analytically easier.

A.3 The EM Algorithm

The commonly used approach for determining the parameters Θ of a Gaussian mixture models from a given dataset is to use the maximum likelihood method estimation. The EM algorithm is a general iterative technique for computing maximum likelihood when

the observed data can be regarded as incomplete. The usual EM algorithm consists of an E-step and an M-step. In this section we develop the computation of these two steps (extracted from [7]).

The incomplete-data log-likelihood expression from the data X is given by:

$$\log(\mathcal{L}(\Theta|X)) = \log \prod_{i=1}^{N} p(x_i;\Theta) = \sum_{i=1}^{N} \log \left(\sum_{j=1}^{M} \alpha_j p_j(x_i;\theta_j) \right)$$

which is difficult to optimize because it contains the log of the sum. If we consider X as incomplete, however, and there are an unobserved data items $Y = \{y_i\}_{i=1}^{N}$ whose values inform us which component density "generated" each data item, the likelihood expression is significantly simplified. That is, we assume that $y_i \in \{1, \ldots, M\}$ for each i, and $y_i = k$ if the ith sample was generated by the kth mixture component. Thus, we can define a new likelihood function,

$$\mathcal{L}(\Theta|X,Y) = P(X,Y;\Theta),$$

where $P(X,Y;\Theta)$ is the joint density function.

At each E-step the algorithm computes the expected complete data log-likelihood. We define

$$Q(\Theta,\Theta^{(t)}) = E\left[\log P(X,Y|\Theta)|X,\Theta^{(t)}\right]$$

where $\Theta^{(t)}$ are the current parameters estimates that we used to evaluate the expectation and Θ are the new parameters that we optimize to increase Q. The evaluation of this expectation is called the E-step of the algorithm.

With the previous assumptions, we assume Y is a random vector and then we can compute the value of $Q(\Theta,\Theta^{(t)})$.

We must first derive an expression for the distribution of the unobserved data. Let's first guess that $\Theta^g = \{\theta_1^g, \ldots, \theta_M^g, \alpha_1^g, \ldots, \alpha_M^g\}$ are the appropriate parameters for the likelihood $\mathcal{L}(\Theta^g|X,Y)$, that is, we guess at parameters for the mixture density. Given Θ^g, we can easily compute $p_j(x_i;\theta_j^g)$ for each i and j. In addition, the mixing parameters, α_j can be thought of as prior probabilities of each mixture component, that is $\alpha_j = p(\textit{component } j)$. Therefore, using Baye's rule, we can compute:

$$p(y_i|x_i;\Theta^g) = \frac{\alpha_{y_i}^g p_{y_i}(x_i;\theta_{y_i}^g)}{p(x_i;\Theta^g)} = \frac{\alpha_{y_i}^g p_{y_i}(x_i;\theta_{y_i}^g)}{\sum_{k=1}^{M} \alpha_k^g p_k(x_i;\theta_k^g)}$$

and

$$p(\mathbf{y}|X;\Theta^g) = \prod_{i=1}^{N} p(y_i|x_i;\Theta^g)$$

where $\mathbf{y} = (y_1, \ldots, y_N)$ is an instance of the unobserved data independently drawn.

In this case, considering Υ the space of values $\mathbf{y}$ can take on, the expected complete

data log-likelihood takes the form:

$$\begin{aligned}
Q(\Theta,\Theta^g) &= \sum_{\mathbf{y}\in\Upsilon}\log(\mathcal{L}(\Theta|X,\mathbf{y}))p(\mathbf{y}|X;\Theta^g)\\
&= \sum_{y_1=1}^{M}\sum_{y_2=1}^{M}\cdots\sum_{y_N=1}^{M}\sum_{i=1}^{N}\log(\alpha_{y_i}p_{y_i}(x_i;\theta_{y_i}))\prod_{j=1}^{N}p(y_j|x_j;\Theta^g)\\
&= \sum_{y_1=1}^{M}\sum_{y_2=1}^{M}\cdots\sum_{y_N=1}^{M}\sum_{i=1}^{N}\sum_{l=1}^{M}\delta_{l,y_i}\log(\alpha_l p_l(x_i;\theta_l))\prod_{j=1}^{N}p(y_j|x_j;\Theta^g)\\
&= \sum_{l=1}^{M}\sum_{i=1}^{N}\log(\alpha_l p_l(x_i;\theta_l))\sum_{y_1=1}^{M}\sum_{y_2=1}^{M}\cdots\sum_{y_N=1}^{M}\delta_{l,y_i}\prod_{j=1}^{N}p(y_j|x_j;\Theta^g).
\end{aligned}$$

In this form $Q(\Theta,\Theta^g)$ looks fairly daunting, yet it can be greatly simplified. We first note that for $l\in\{1,\ldots,M\}$,

$$\begin{aligned}
&\sum_{y_1=1}^{M}\sum_{y_2=1}^{M}\cdots\sum_{y_N=1}^{M}\delta_{l,y_i}\prod_{j=1}^{N}p(y_j|x_j;\Theta^g) =\\
&= \left(\sum_{y_1=1}^{M}\cdots\sum_{y_{i-1}=1}^{M}\sum_{y_{i+1}=1}^{M}\cdots\sum_{y_N=1}^{M}\prod_{j=1,j\neq i}^{N}p(y_j|x_j;\Theta^g)\right)p(l|x_i;\Theta^g)\\
&= \prod_{j=1,j\neq i}^{N}\left(\sum_{y_j=1}^{M}p(y_j|x_j;\Theta^g)\right)p(l|x_i;\Theta^g) = p(l|x_i;\Theta^g)
\end{aligned}$$

since $\sum_{i=1}^{M}p(i|x_j;\Theta^g)=1$. Using these last expressions, we obtain:

$$Q(\Theta,\Theta^g) = \sum_{l=1}^{M}\sum_{i=1}^{N}\log(\alpha_l p_l(x_i;\theta_l))p(l|x_i;\Theta^g). \tag{A.1}$$

In this way we can compute the E-step of the algorithm. The second step (the M-step) is to maximize the expectation we computed in the first step. This step finds the (t+1)th estimation, $\Theta^{(t+1)}$, of Θ in terms of the parameters in the previous step.

To maximize the expression of $Q(\Theta,\Theta^g)$, we can write the expression (A.1) as:

$$Q(\Theta,\Theta^g) = \sum_{l=1}^{M}\sum_{i=1}^{N}\log(\alpha_l)p(l|x_i;\Theta^g)+\sum_{l=1}^{M}\sum_{i=1}^{N}\log(p_l(x_i;\theta_l))p(l|x_i;\Theta^g).$$

Thus, we can maximize the term containing α_l and the term containing θ_l independently since they are not related.

To find the expression for α_l, we introduce the Lagrange multiplier λ with the constraint that $\sum_l\alpha_l=1$, and solve the following system of equation:

$$\frac{\partial}{\partial\alpha_l}\left[\sum_{l=1}^{M}\sum_{i=1}^{N}\log(\alpha_l)p(l|x_i;\Theta^g)+\lambda\left(\sum_l\alpha_l-1\right)\right]=0$$

or

$$\sum_{i=1}^{N}\frac{1}{\alpha_l}p(l|x_i;\Theta^g)+\lambda=0.$$

Summing both sizes over l and fixing i we get that $\lambda = -N$ resulting in:

$$\alpha_l = \frac{1}{N}\sum_{i=1}^{N} p(l|x_i;\Theta^g).$$

For some distributions, it is possible to get an analytical expression for θ_l as function of everything else. For example, if we assume p is a 1D Gaussian distributions with mean μ and covariance σ, *i.e.*, $\theta_m = (\mu_m, \sigma_m)$ then

$$p_l(x;\theta_l) \quad = \quad \frac{1}{\sigma_l\sqrt{2\pi}}\exp\left[\frac{-(x-\mu_l)^2}{2\sigma_l^2}\right].$$

By taking the log of $p_l(x;\theta_l)$, ignoring any constant terms (since they dissapear after taking derivaties), then, the likelihood function $Q(\Theta,\Theta^g)$ is:

$$\begin{aligned}
&\sum_{l=1}^{M}\sum_{i=1}^{N}\log(p_l(x_i;\theta_l))p(l|x_i;\Theta^g)\\
&=\sum_{l=1}^{M}\sum_{i=1}^{N}\left(-\log(\sigma_l)+\frac{-(x_i-\mu_l)^2}{2\sigma_l^2}\right)p(l|x_i;\Theta^g)
\end{aligned} \tag{A.2}$$

By taking the derivative of Expression (A.2) with respect to μ_l and σ_l and setting it equal to zero, we get:

$$\begin{aligned}
&\sum_{i=1}^{N}\frac{x_i-\mu_l}{\sigma_l^2}p(l|x_i;\Theta^g)=0,\\
&\sum_{i=1}^{N}p(l|x_i;\Theta^g)(\sigma_l^2-(x_i-\mu_l)^2)=0,
\end{aligned}$$

and we can easily solve for μ_l and σ_l to obtain:

$$\begin{aligned}
\mu_l &= \frac{\sum_{i=1}^{N}x_i p(l|x_i;\Theta^g)}{\sum_{i=1}^{N}p(l|x_i;\Theta^g)},\\
\sigma_l^2 &= \frac{\sum_{i=1}^{N}p(l|x_i;\Theta^g)(x_i-\mu_l)^2}{\sum_{i=1}^{N}p(l|x_i;\Theta^g)}.
\end{aligned}$$

The proof is very similar in the d-dimensional case, only it is necessary to recall some results from matrix algebra.

Summarizing, the estimates of the new parameters in terms of the old parameters are as follows:

$$\begin{aligned}
\alpha_l^{(t)} &= \frac{1}{N}\sum_{i=1}^{N}p(l|x_i;\Theta^{(t-1)}),\\
\mu_l^{(t)} &= \frac{\sum_{i=1}^{N}x_i p(l|x_i;\Theta^{(t-1)})}{\sum_{i=1}^{N}p(l|x_i;\Theta^{(t-1)})},\\
\sigma_l^{2(t)} &= \frac{\sum_{i=1}^{N}p(l|x_i;\Theta^{(t-1)})(x_i-\mu_l^{(t)})^2}{\sum_{i=1}^{N}p(l|x_i;\Theta^{(t-1)})}.
\end{aligned}$$

It is not difficult to see that the weights α_l satisfy the initial conditions after applying the above formulae for all likelihoods.

The EM algorithm is known to converge to a locally optimal solution. However, it is not guaranteed to lead us to the best solution, where "best" means the solution yielding maximal likelihood on the data set among all maxima of the likelihood. This limitation is presented in the next section.

A.4 Limitations of EM

The EM algorithm presents some drawbacks:

- It assumes that the number of Gaussian components is known. A solution to this limitation is the implementation of the Minimum Message Lenght (MML) criteria [38] into the EM algorithm. It supposes increasing the complexity and the execution time of the method.
- It is very sensitive to inital conditions. A positive thing is that if the initialization of the parameter vector is close to the global optimum of the parameter space, then it is very likely that by using EM, we obtain the globally optimal solution. We need a more precise method for initializing the parameters. Normally, the k-means algorithm [64] is used to that end.
- The algorithm is of local nature and thus it can be trapped in a local maxima or saddle point of the likelihood function.

Several authors have proposed various nonstochastic and stochastic improvements on the EM algorithm (for example [44, 91, 13]). However, none of these improvements result in a completely satisfactory version of EM.

A.5 Goodness-of-fit Tests

A statistical test in which the validity of one hypothesis is tested without specification of an alternative hypothesis is called a goodness-of-fit test. The general procedure consists in defining a statistical test , which is some function of the data measuring the distance between the hypothesis and the data (in fact, the badness-of-fit). Then the probability of obtaining data which have a still larger value of this statistical test than the value observed is computed, assuming the hypothesis is true. This probability is called the size of the test or confidence level.

The main application of these statistical tests is to evaluate the difference between two given distributions. Such tests answer to the question: Can we disprove, to a certain required level of significance, the null hypothesis that two data sets are drawn from the same population distribution function?

These tests are useful for detecting the number of Gaussian components in a given distribution. The algorithm compares, using the mentioned tests, the EM results assuming different number of Gaussian components for the same distribution. In this section we will present three different goodness-of-fit tests.

Henceforth, we consider two samples, $(X_1, \ldots, X_{n_1})$ and $(Y_1, \ldots, Y_{n_2})$ of two variables X and Y and we want to verify if the two data sets are drawn from the same population distribution function. Then, the null hypothesis is:

$$\mathcal{H}_0 : \ X \text{ and } Y \text{ come from the same distribution,}$$

A.5.1 Kolmogorov-Smirnov test

The most generally accepted test for accepting or disproving a hypothesized density is the Kolmogorov-Smirnov (K-S) test. The idea in this test is as follow: if hypothesis $\mathcal{H}_0$ is true, then the cumulative distribution functions of X and Y must be close. Thus, the Kolmogorov-Smirnov statistic D is defined as the maximum value of the absolute difference between the two cumulative distribution functions. Let F and G be the distribution functions of X and Y, respectively, and F_{n_1} and G_{n_2} two different empirical distribution functions of the respective samples. Then the $D_{n_1 n_2}$ K-S statistic is defined by

$$D_{n_1 n_2} = max_{-\infty<x<\infty}|F_{n_1}(x) - G_{n_2}(x)|$$

The test is based in the Smirnov Theorem, which says that if hypothesis $\mathcal{H}_0$ is true, then, for all $t \geq 0$:

$$\lim_{n_1\to+\infty, n_2\to+\infty} P_{\mathcal{H}_0}\left(\sqrt{\frac{n_1 n_2}{n_1+n_2}} D_{n_1 n_2} \leq t\right) = 1 - 2\sum_{k=1}^{+\infty}(-1)^{k+1} exp(-2k^2t^2).$$

The acceptance or rejection of the null hypothesis involves the computation of the p-value. The p-value is the probability that our sample could have been drawn from the population(s) being tested (or that a more improbable sample could be drawn) given the assumption that the null hypothesis is true. A p-value of 0.05, for example, indicates that we would have only a 5% chance of drawing the sample being tested if the null hypothesis was actually true.

A p-value close to zero signals that our null hypothesis is false, and typically that a difference is very likely to exist. Large p-values closer to 1 imply that there is no detectable difference for the sample size used. A p-value of 0.05 is a typical threshhold used in industry to evaluate the null hypothesis.

In the Kolmogorov-Smirnov test, the p-value is given by

$$p(t) = 2\sum_{k=1}^{+\infty}(-1)^{k+1} exp(-2k^2t^2).$$

Then, the algorithm consists in computing the value of t from the value of the statistic and then computing an approximation of the p-value.

A central feature of the K-S test is that it is invariant under reparametrization of x; in other words, you can locally slide or stretch the x axis and the maximum distance D remains unchanged.

A.5.2 Chi-Square Test

For the two samples Chi-Square test, the data is divided into k bins and the test statistic is defined as:

$$\chi^2 = \frac{\sum_{i=1}^{k}(K_1 R_i - K_2 S_i)^2}{R_i + S_i}$$

where R_i is the observed frequency in the bin i for sample X and S_i is the observed frequency in the bin i for sample Y. K_1 and K_2 are scaling constants that are used to adjust for unequal sample sizes. Specifically,

$$K_1 = \sqrt{\frac{\sum_{i=1}^{k} R_i}{\sum_{i=1}^{k} S_i}}, \qquad K_2 = \sqrt{\frac{\sum_{i=1}^{k} S_i}{\sum_{i=1}^{k} R_i}}.$$

This test is sensitive to the choice of bins. Most reasonable choices should produce similar, but not identical, results.

The statistical test follows, approximately, a chi-square distribution with $k-c$ degrees of freedom where k is the number of non-empty bins and $c = 1$ if the samples sizes are equal and $c = 0$ if they are not equal. Thus, in this case, the p-value is

$$p(t) = 1 - F_{\chi^2(k-c)}(t),$$

and the process of computation is the same as before.

A.5.3 Cramer-Von Mises Test

Let $F_{n_1}(x)$ and $G_{n_2}(x)$ be the empirical distribution functions associated with the X and Y samples respectively. The two-samples Cramer-Von Mises (also called Fisz-Cramer-Von Mises) statistic is defined as

$$W^2 = \frac{n_1 n_2}{(n_1+n_2)^2}\left(\sum_{i=1}^{n_1}\left(F_{n_1}(X_{(i)}) - G_{n_2}(X_{(i)})\right)^2 + \sum_{j=1}^{n_2}\left(F_{n_1}(Y_{(j)}) - G_{n_2}(Y_{(j)})\right)^2\right)$$

where the two sample values are put in increasing order. W^2 is distribution-free under $\mathcal{H}_0$ and the exact and limiting null distributions of W^2 are not standart. Anderson [3] and Burr [11] provide tables for the exact distribution in the case of small sample sizes ($n_1 + n_2 \leq 17$). Otherwise, a table of the asymptotic distribution is available from [4].

The main difference between Cramer-Von Mises and Kolmogorov test is the fact that this test takes better into account the dataset, since in the statistic the sum of variations is considered. The Kolmogorov test is more sensible to the existence of aberrant points in the sample.

Appendix B

Some notes on Gestalt Theory

In his research program, Max Wertheimer ([94]) conjectured the existence of a finite set of grouping laws governing the perceptual synthesis of phenomenal objects: the Gestalt theory. In this program, these laws are classified in two kinds. The first kind are grouping laws which, starting from atomic elements, construct larger groups in the perceived image from the local data recursively. The second kind are principles governing the collaboration and conflicts of gestalt laws. The gestaltists claim that whenever points (or previously formed groups) have one or several characteristics in common, they get grouped and form a new, larger, visual object, *a gestalt*. The list of elementary grouping laws given by Kanizsa in [54] are: vicinity, similarity, continuity of direction, amodal completion, closure, constant width, tendency to convexity, symmetry, common motion, past experience. In Figure B.1 some principles of the gestalt theory are illustrated. In addition, the grouping principle is recursive. For example, if points have been grouped into lines, then these lines may again be grouped according, for example, to parallelism.

The gestalt laws are stated as independent grouping laws. Now, they start from the same building elements. Thus, conflicts between grouping laws can occur and therefore conflicts between different interpretations, that is, different possible groups in a given figure. We can consider three cases:

1. two grouping laws act simultaneously on the same element and give raise to two overlapping groups,

2. two grouping laws compete and one of them wins, the other one being inhibited,

3. conflict: both grouping laws are potentially active, but the groups cannot exist simultaneously. In addition, none of the grouping laws wins clearly.

In Figure B.2 we can see an example of each one of these situations.

Based on this theory, in 1999, Agnès Desolneux, Lionel Moisan and Jean-Michel Morel felt the necessity to pass from qualitative to quantitative arguments. Their initial aim was, given an image, to be able to get the list of all the gestalts it contains. They began with the detection of elementary geometric objects (for example, aligments in [32] or edges and boundaries in [33]), and then, in a second step they grouped these objects according to a recursive and pyramidal algorithm ([34]).

Their method is based on a principle of perception called Helmholtz principle, which attempts to describe when perception decides to group objects according to some quality (color, alignment, etc.). It can be stated in the following way: we have a high probability of grouping two elements if the placement of the two elements has a low likelihood of resulting from an accidental arrangement. This principle, which may also be called a genericity

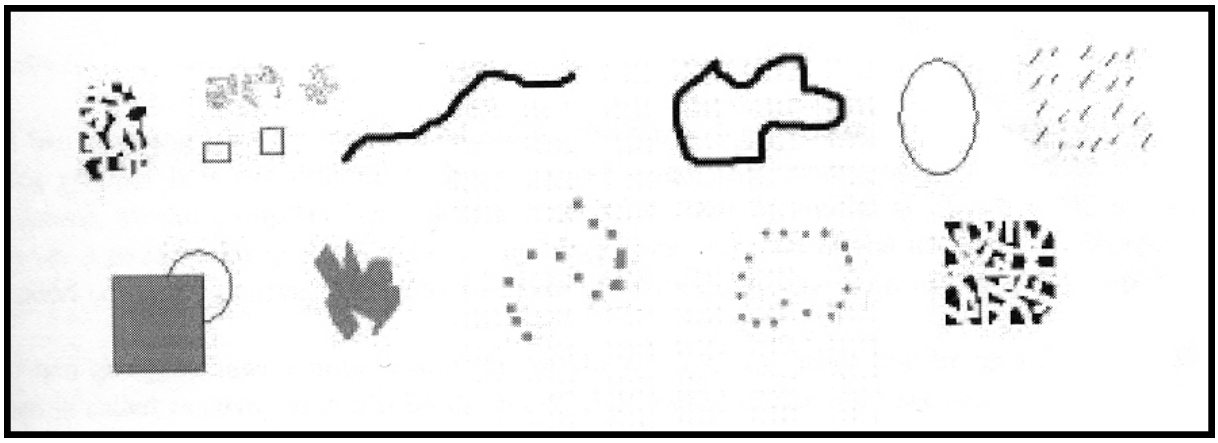


Figure B.1: *Some gestalt principles. From left to right and top to bottom: color constancy and proximity; similarity of shape and similarity of texture; good continuation; closure (of a curve); convexity; parallelism; amodal completion (a disk seen behind the square); color constancy; good continuation again (dots building a curve); closure (of a curve made of dots); modal completion.*

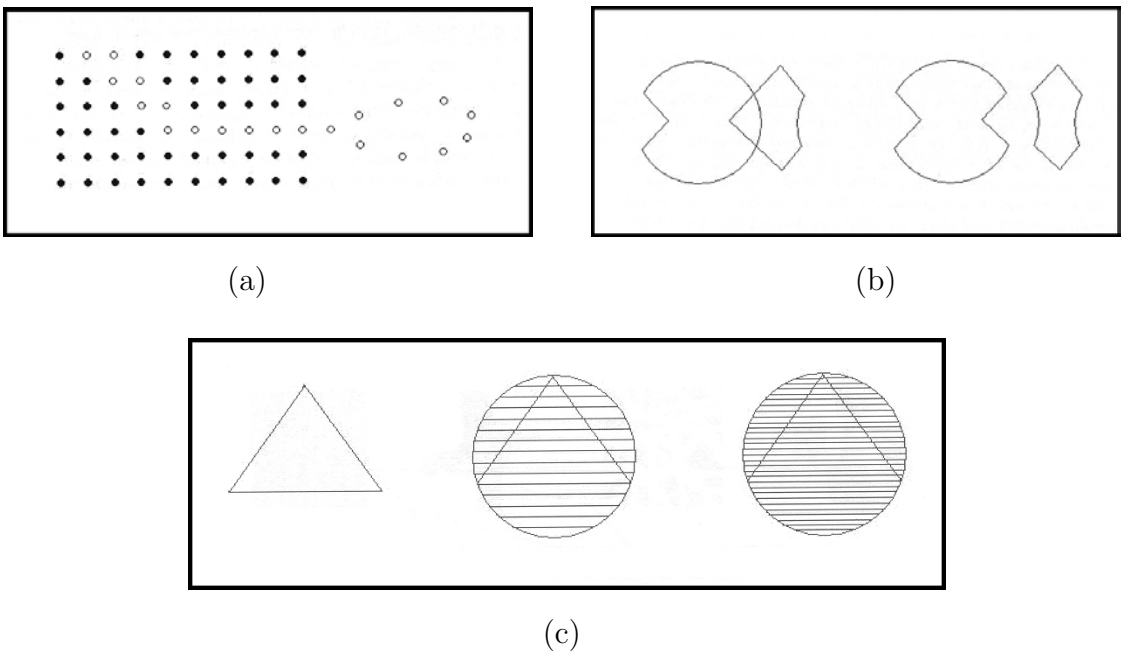


Figure B.2: *Gestalt conflicts: (a) gestalt laws in simultaneous action without conflict: the white dots are elements of the grid and simultaneously participate of a good continuation curve; (b) a conflict of gestalts: two overlapping closed curves or, as suggested on the right, two symmetric curves which touch at two points?; (c) Masking: the basis of the triangle becomes invisible as it is embeded in a group of parallel lines.*

principle, can be generalized as follows. Assume that the atomic objects $O_1, O_2, \ldots, O_n$ are present in an image. Assume that k of them, say $O_1, \ldots, O_k$, have a common feature, say, same color, same orientation, etc. We are then facing the dilemma: is this common feature happening by chance or is it significant? In order to answer this question, we make the following mental experiment: we assume that the considered quality has been randomly and uniformly distributed on all objects, *i.e.* $O_1, \ldots, O_n$. Notice that this quality may be spatial (like position, orientation); then we (mentally) assume that the observed position of objects in the image is a random realization of this uniform process. Then, we may ask the question: is the observed repartition probable or not? The Helmholtz principle states that if the probability in the image of the observed configuration $O_1, \ldots, O_k$ is *very small*, then the grouping of these objects makes sense. Now, what means *very small*? In [32], they define a principle according to which an observed geometric structure is perceptually "meaningful" if the expectation of its occurrences (in other terms, its number of false alarms, NFA) is very small in a random image. The number of false alarms of an event measures the "meaningfulness" of an event: the smaller it is, the more meaningful the event is.

As we have mentioned, this principle can be applied to the detection of different structures in an image, as for example:

- Alignments ([32]), one of the most basic gestalts. In this case, the notion of meaningful segment and maximal meaningful segments are defined.
- Edges and boundaries ([33]), where the gestalt is "strong contrast along a curve". Here, the considered curves are the level lines of the image.

We can apply this same principle to the detection of modes and gaps of a histogram. As in the meaningful alignments theory, the Helmholtz principle can be adopted since there is no a priori knowledge about the histogram model ([30]). For a histogram, the hypothesis can be for example that the values of the data set follow a uniform law. In this sense, the fact that a high percentage of the values concentrate in a certain interval, indicates that this interval is meaningful.

Appendix C

A brief review of binarization techniques

Histogram thresholding is the simplest technique of pixel-based segmentation in image analysis. It is one of the oldest and most popular techniques.

Histogram thresholding can be applied to the intensity histogram of the image and is widely used as a simple but effective tool to separate objects from the background. Examples of its applications are document understanding, character recognition, map processing, x-ray computed tomography analysis, quality material inspection, ... In these applications the intensity histogram normally presents two different modes, one large mode that represents the background and another one smaller, corresponding to the object. In this case the method is called bi-level thresholding or binarization. The output of the thresholding operation is a binary image whose two levels correspond respectively to foreground objects (printed text, a legend, defective part of a material, etc) and to the background.

Many different binarization methods have been proposed in the last years to find the best histogram thresholds for greyscale images. One possible categorization of these methods is according to the information they are exploiting, such as Sezgin and Sankur do in their survey ([86]). In this work the image thresholding techniques are classified into six categories. Next, we detail them with an example of each one with a uniform notation[1]:

1. **Histogram shape-based methods.** This category of methods achieves thresholding based on the shape properties of the histogram. The shape properties analysed are, for example, the curvature of the smoothed histogram or the distance from the convell hull to the histogram.

   Among the reviewed methods in this category, the survey stresses the one proposed by Ramesh, Yoo and Sethi in [80]. These authors propose the selection of the optimum treshold t^* by minimising a functional $E(t)$, that is,

   $$t^* = argmin_{t \in G} E(t).$$

   In a first approach this functional is the sum of square errors,

   $$E(t) = \sum_{i=0}^{t} (i - m_1(t))^2 + \sum_{i=t+1}^{L} (i - m_2(t))^2.$$

[1] In the sequel, we note the histogram and the probability mass function by h_i and p_i, respectively. We suppose that the histogram is divided in $L+1$ grey levels, $G = \{0, \ldots, L\}$.

Although this method is computionally simpler, it can be biased, because the reduction in the square error is achieved by many terms in the sum. This led the authors to come up with a better method based on minimizing the sum of two variances,

$$E(t) = \frac{1}{N_1 - 1}\sum_{i=0}^{t}(i - m_1(t))^2 + \frac{1}{N_2 - 1}\sum_{i=t+1}^{L}(i - m_2(t))^2,$$

where

$$m_1(t) = \frac{\sum_{i=0}^{t} ih_i}{\sum_{i=0}^{t} h_i}, \qquad m_2(t) = \frac{\sum_{i=t+1}^{L} ih_i}{\sum_{i=t+1}^{L} h_i},$$

N_1 is the total number of pixels with grey values less or equal than t and N_2 is the total number of pixels with grey values greater than t.

2. **Clustering-based methods.** In this class of algorithms, the goal is to separate into two clusters the grey-levels. Some methods in this category use mean-square clustering or fuzzy clustering, others model the intensity histogram as a mixture of two Gaussians.

   In this category we find the Otsu method ([71]). It is one of the first developed methods and it still remains one of the most referenced, since it is very simple. Otsu tries to maximize the separability of the resultant classes by using cumulative moments of the histogram. The optimal threshold, t^*, is obtained by minimizing the ratio between the between-class variance and the total variance, that is,

$$t^* = argmin_{t \in G}\frac{\sigma_B^2}{\sigma_T^2}$$

   where

$$\sigma_T^2 = \sum_{i=0}^{L}(i - \mu_L)^2 p_i, \qquad \sigma_B^2 = \omega_0\omega_1(\mu_0\mu_1)^2, \qquad \mu_t = \sum_{i=0}^{t} ip_i$$

$$\omega_0 = \sum_{i=0}^{t} p_i, \qquad \omega_1 = 1 - \omega_0, \qquad \mu_1 = \frac{\mu_L - \mu_t}{1 - \mu_0}, \qquad \mu_0 = \frac{\mu_t}{\omega_0},$$

   Otsu's method is efficient on the basis of uniformity measure between the foreground and the background classes, that is, it gives satisfactory results when the numbers of pixels in each class are close to each other.

3. **Entropy-based methods.** This class of algorithms exploits the entropy of the distribution of the grey levels in a scene. The maximization of the entropy of the thresholded image is interpreted as indicative of maximum information transfer. In this category, some authors try to minimize the cross-entropy between the input grey-level image and the output binary image as indicative of preservation of information.

   The more representative algorithm of this class is the one proposed by Pun ([77]). This method has been superseded by other techniques, but it is important for historical reasons. Assuming that all the elements of the histogram are statistically independent, Pun considers the source entropy as a function of the threshold t,

$$H(t) = -\sum_{i=0}^{t} p_i log(p_i) - \sum_{i=t+1}^{L} p_i log(p_i).$$

After thresholding, the picture has two levels: white (w) and black (b). Then, the *a posteriori* entropy becomes

$$H'(t) = -p'_w log(p'_w) - p'_b log(p'_b).$$

These two entropies have a very different meaning. The first formula is an objective measure of the *a priori* quantity of information associated with black and white points related to the level t of the threshold. The second entropy measures the *a posteriori* information associated with black and white points after the thresholding operation. The goal is, knowing the *a priori* entropy H of the histogram, to try to determine a lower bound for H' as a parametric function of the level t of the threshold. Thus, Pun presents a method for maximizing the *a posteriori* entropy H' using its intrinsic relationships with H.

The method presented in [78] can be included in this category. In this article, the optimal threshold is defined as the value t^* such that

$$\sum_{i=0}^{t^*} p_i = (0.5 + |0.5 - \alpha|),$$

where the anisotropic parameter α measures the histogram asymetry and is defined by:

$$\alpha = \frac{\sum_{i=0}^{na} p_i log(p_i)}{\sum_{i=0}^{n} p_i log(p_i)}.$$

The value na divides the histogram into two parts containing the same number of points or nearly.

4. **Object attribute-based methods.** The algorithms in this category select the threshold value based on some attribute quality or similarity measure between the grey-level original image and the binarized one. These attributes can take the form of edge matching, fuzzy shape similarity, connectivity, etc.

   One of the most recent methods in this category is presented in [49]. In this paper, the authors propose an approach based on minimizing the measures of fuzziness of an input image. Given a certain threshold value, the membership function of a pixel is defined as the absolute difference between the grey level and the average grey level of the region where it belongs (*i.e.*, the object or the background). The larger the absolute difference is, the smaller the membership value becomes. It is expected that the membership value of each pixel in the input image is as large as possible. In addition, two measures of fuzziness are proposed to indicate the fuzziness of an image. In this method, F denotes an image of size $M \times N$ with L levels, $I(i,j)$ is the grey level of the (i,j) pixel in F. The image set is represented as the double array $F = \{I(i,j), \mu_F[I(i,j)]\}$, where the membership function $0 \leq \mu_F[I(i,j)] \leq 1$ represents for each pixel at location (i,j) its fuzzy measure. In this case, the membership function can be defined as:

$$\mu_F[I(i,j)] = \begin{cases} \frac{C}{C+|I(i,j)-\mu_0|} & if\ I(i,j) \leq t \\ \\ \frac{C}{C+|I(i,j)-\mu_1|} & if\ I(i,j) > t, \end{cases}$$

   where C is a constant value such that $\frac{1}{2} \leq \mu_F[I(i,j)] \leq 1$, μ_0 and μ_1 are the average grey levels of the background and the object, respectively. Then, the optimum

threshold is found either by minimizing an entropy function or the Yager's measure. The entropy function, used as a measure of fuzziness, is defined by

$$E(F) = \frac{1}{MNln2} \sum_{i=0}^{M} \sum_{j=0}^{N} S(\mu_F[I(i,j)]),$$

where $S(x) = -xln(x) - (1-x)ln(1-x)$ is the Shannon's function. The Yager's measure of fuzziness, $\eta_p(F)$, is defined as the measure of lack of distinction between a fuzzy set F and its complement $\bar{F}$. Thus, if the distance between a fuzzy image set F and its complement $\bar{F}$ is defined as:

$$D_p(F, \bar{F}) = \left[\sum_i \sum_j |\mu_F[I(i,j)] - \mu_{\bar{F}}[I(i,j)]|^p \right]^{1/p}$$

where $\mu_{\bar{F}}[I(i,j)] = 1 - \mu_F[I(i,j)]$ then the Yager's measure can be defined as:

$$\eta_p(F) = 1 - \frac{D_p(F, \bar{F})}{(MN)^{1/p}}.$$

5. **Spatial methods.** These methods utilize not only information about grey values distributions but also about pixel dependency in a neighborhood, for example, in the form of context probabilities, correlation functions, coocurrence probabilities, 2-D entropy, etc.

   The Abutaleb algorihtm, [1], belongs to this category. Abutaleb considers the joint entropy of two variables: the grey level of each pixel and the average grey-level of its neighborhood centered at that pixel. The author of the article studies the frequency of occurrence of each pair of grey levels. This will draw a surface that, presumably, has two peaks and one valley. The peaks can be separated by choosing the threshold that maximizes the entropy in the two groups. Thus, he considers the 2-D histogram p_{ij}, where the variable $i = 0, \ldots, L$ represents the grey level values and $j = 0, \ldots, L$ represents the average grey-level. Then, the algorithm searches for the values s and t that maximize the entropy-based function

   $$\chi(s,t) = ln[P_{st}(1 - P_{st})] + \frac{H_{st}}{P_{st}} + \frac{(H_{LL} - H_{st})}{(1 - P_{st})},$$

   where

   $$H_{st} = -\sum_{i=0}^{s} \sum_{j=0}^{t} p_{ij} ln(p_{ij}) \quad \text{and} \quad P_{st} = -\sum_{i=0}^{s} \sum_{j=0}^{t} p_{ij}.$$

   The method assumes that the quadrants defined by $i = s+1, \ldots, L$ and $j = 0, \ldots, t$ and by $i = 0, \ldots, s$ and $j = t+1, \ldots, L$ are negligible and contain elements only due to image edges and noise.

6. **Local methods.** In this class of algorithms, a threshold is calculated at each pixel, which depends on some local statistics like range, variance, or surface-fitting parameters of the pixel neighborhood. Some local image characteristics used are local variance, local contrast or center-surround scheme.

   Among the revised methods in this category, the survey stresses the one presented in [84]. In this article, the authors propose a new method for document image

binarization that first perfoms a rapid classification of the local contents of a page to background, pictures and text. Two different approaches are then applied to define a threshold for each pixel: a soft decision method (SDM) for background and pictures, and a specialized text binarization method (TBM) for textual and linedrawing areas. The SDM includes noise filtering and signal tracking capabilities, while the TBM is used to separate text components from background in bad conditions, caused by uneven ilumination or noise. A fast option is to compute first a threshold for every nth pixel, n decided by the user, and, then, to use interpolation for the rest of the pixels.

The first classification of the document is performed by computing two simple features: the average grey value and the transient difference. The transient difference is aimed at extracting the average amount of variations ocurring between the neighbouring pixels in a $n \times n$ area, that is, to follow local surface changes. If we consider $I(i,j)$ the value in the pixel (i,j) of the document, the transient difference (TD) is defined by

$$TD = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} |2I(i,j) - [I(i-1,j) + I(i,j-1)]|}{Ln^2}$$

The TD value is used in soft decision, in order to distinguish between uniform, differential and transient area types. Then, classification of the pixels is perfomed following the next example rules (a, b):

(a) If the average is high and a global histogram peak is in the same quarter of the histogram and transient difference is transient, then the pixels correspond to the background and pictures.

(b) If the average is medium and a global histogram peak is not in the same quarter of the histogram and transient difference is uniform, then the pixels correspond to the textual area.

The second step is the binarization of the different areas. In the binarization of non-textual components process, the author uses nine different rules derived from the feature analysis and membership management. They are based on the weighted bound, feature that is used for characterization of local pixel value profiles by tracking low, medium and high pixels in a small area, and on the transient difference. For text binarization the author uses the formula

$$T(x,y) = m(x,y)\left[1 + k\left(\frac{s(x,y)}{R} - 1\right)\right],$$

where $R = 128$, $k = 0.5$ and $m(x,y)$ and $s(x,y)$ are the local mean and the standard deviation, respectively.

As we can observe, in literature, we can find a vast list of bilevel thresholding methods, it has been and it is still currently being an important field of investigation, since the images of binary information constitute the majority of thresholding applications to date. But many applications where the image has three or more information levels exist. Moreover, the increasing number of color documents and color images becomes a new challenge for binarization and segmentation. Thus, multilevel thresholding, or simply multithresholding, is gaining more relevance in vision applications. A few authors have addressed this issue. Some of them propose an extension of the bilevel thresholding methods to obtain

multithresholding ([49], [80]), while others propose new approaches ([2], [40]). However, the principal drawback of most of these methods is that the number of segments must be *a priori* specified, and the obtained results are not proved to be relevant.

Bibliography

[1] A.S. Abutaleb. Automatic thresholding of gray-level pictures using two-dimensional entropy. *Computer Vision, Graphics and Image Processing*, 47:22–32, 1989.

[2] L Alvarez and J. Esclarín. Image quantization using reaction-diffusion equations. *SIAM-J-Applied Mathematics*, 57(1):153–175, february 1997.

[3] T.W. Anderson. On the distribution of the two-sample cramer-von mises criterion. *Annals of Mathematical Statistics*, 33:1148–1159, 1962.

[4] T.W. Anderson and D.A. Darling. Asymptotic theory of certain goodness of fit criteria based on processes. *Annals of Mathematical Statistics*, 23:193–212, 1952.

[5] M. Ayer, H.D. Brunk, G.M. Ewing, W.T. Reid, and E. Silverman. An empirical distribution function for sampling with incomplete information. *The Annals of Mathematical Statistics*, 26(4):641–647, 1955.

[6] R.E. Barlow, D.J. Bartholomew, J.M. Bremner, and H.D. Brunk. *Statistical Inference Under Order restrictions*. Wiley, New York, 1972.

[7] J. Bilmes. A gentle tutorial on the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models, 1997.

[8] L. Birgé. The Grenander estimator: A nonasymptotic approach. *The Annals of Statistics*, 17(4):1532–1549, 1989.

[9] S.M. Boker. http://kiptron.psyc.virginia.edu/steve_boker/colorvision2/, july 2004.

[10] A. Buades, T. Coll, and J.M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling and Simulation (SIAM interdisciplinary journal)*, 4(2):490–530, 2005.

[11] E.J. Burr. Distribution of the two-sample cramer-von mises criterion for small equal samples. *Annals of Mathematical Statistics*, 34:95–101, 1963.

[12] J.R. Caldas, L.P.C. Bandeira, J.M. da Costa, and P. Pina. Combining fuzzy clustering and morphological methods for old documents recovery. In *IbPRIA Proceedings*, volume 2, pages 387–394, 2005.

[13] G. Celeux and J. Diebolt. The sem algorithm: a probabilistic teacher algorithm derived from the em algorithm for the mixture problem. *Computational Statistics Quaterly*, 2:73–82, 1985.

[14] C. Chang, K. Chen, J. Wang, and M.L.G. Althouse. A relative entropy based approach to image thresholding. *Pattern Recognition*, 27(9):1275–1289, 1994.

[15] C.C. Chang and Y.Y. Su. A dynamic color palette for color images coding. *Lecture Notes in Computer Science*, 2532:369–376, 2002.

[16] H.D. Cheng, X.H. Jiang, Y. Sun, and J. Wang. Color image segmentation: Advances and prospects. *Pattern Recognition*, 34:2259–2281, 2001.

[17] H.D. Cheng, X.H. Jiang, and J. Wang. Color image segmentation based on homogram thresholding and region merging. *Pattern recognition*, 35(2):373–393, 2002.

[18] H.D. Cheng and Y. Sun. A hierarchical approach to color image segmentation using homogeneity. *IEEE Transactions on Image Processing*, 9(12):2071–2082, 2000.

[19] ColorTec. http://www.color-tec.com/1gloss.htm, july 2004.

[20] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[21] T.M. Cover and J.A. Thomas. *Elements of information theory.* Wiley Series in Telecommunications. John Wiley & Sons Inc., New York, 1991.

[22] A. Dawoud and M.S. Kamel. Iterative mutimodel subimage binarization for handwritten character segmentation. *IEEE Image Processing*, 13(9):1223, 2004.

[23] J. Delon, A. Desolneux, J.L. Lisani, and A.B. Petro. Color image segmentation using optimal separators of a histogram. In *VIIP 2004 Proceedings*, September 2004.

[24] J. Delon, A. Desolneux, J.L. Lisani, and A.B. Petro. Histogram analysis and its application to fast camera stabilization. In *IWSSIP 2004 Proceedings*, Pologne, September 2004.

[25] J. Delon, A. Desolneux, J.L. Lisani, and A.B. Petro. Automatic color palette. In *ICIP 2005 Proceedings*, September 2005.

[26] J. Delon, A. Desolneux, J.L. Lisani, and A.B. Petro. Color image segmentation using acceptable histogram segmentation. In *IbPria 2005 Proceedings*, July 2005.

[27] J. Delon, A. Desolneux, J.L. Lisani, and A.B. Petro. A non parametric approach for histogram segmentation. Submitted to Image Processing journal, December 2005.

[28] A. Dembo and O. Zeitouni. *Large deviations techniques and applications.* Springer, New York, 1998.

[29] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[30] A. Desolneux. *Evènements significatifs et applications à l'analyse d'images.* PhD thesis, Ecole Normale Supérieure de Cachan, 2000.

[31] A. Desolneux, S. Ladjal, L. Moisan, and J.M. Morel. Dequantizing image orientation. *IEEE Transactions on Image Processing*, 11(10):1129–1140, october 2002.

[32] A. Desolneux, L.Moisan, and J.M. Morel. Meaningful alignments. *International Journal of Computer Vision*, 40(1):7–23, 2000.

[33] A. Desolneux, L.Moisan, and J.M. Morel. Edge detection by Helmholtz principle. *Journal of Mathematical Imaging and Vision*, 14(3):271–284, 2001.

[34] A. Desolneux, L. Moisan, and J.M. Morel. A grouping principle and four applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4):508–513, 2003.

[35] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification.* John Wiley and Sons, 2000.

[36] EasyRGB. http://www.easyrgb.com, july 2004.

[37] M.D. Fairchild. *Color Appearance Models.* Addison Wesley, 1998.

[38] M.A. Figueiredo and A.K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:381–396, 2002.

[39] R. Floyd and L. Steinberg. An adaptative algorithm for spatial gray scale. *SID 75, Int. Symp. Dig. Tech. Papers*, 36, 1975.

[40] G. Frederix and E.J. Pauwels. A statistically principled approach to histogram segmentation. Technical report, CWI, 2004.

[41] M. Gervautz and W. Purgathofer. *A simple method for color quantization:octree quantization*, pages 287–293. Academic Press Professional, Inc., 1990.

[42] E.B. Goldstein. *Sensación y Percepción.* Editorial Debate, 1995.

[43] Y. Gousseau. *The distribution of shapes in natural images.* PhD thesis, Université Paris Dauphine, 2000.

[44] P.J. Green. On use of the em algorithm for penalized likelihood estimation. *Journal of the Royal Statistical Society B*, 52:443–452, 1990.

[45] U. Grenander. *Abstract Inference.* Wiley, New York, 1980.

[46] R. Grompone and J. Jakubowicz. Forthcoming.

[47] A. Hanbury. Circular statistics applied to colour images. In *8th Computer Vision Winter Workshop*, 2003.

[48] P. Heckbert. Color image quantization for frame buffer display. B.s. thesis, Architecture Machine Group, MIT, Cambridge, Mass., 1980.

[49] L. Huang and M. Wang. Image thresholding by minimizing the measures of fuzziness. *Pattern Recognition*, 28(1):41–51, 1995.

[50] Y.L. Huang and R.F. Chang. A fast finite-state algorithm for generating rgb palettes of color quantized images. *Journal of information science and engineering*, 20:771–782, 2004.

[51] D. Hubel. *Eye, Brain and Vision.* Paperback, 2000.

[52] J.C.Russ. *The Image Processing Handbook*. CRC Press–IEEE Press, third edition, 1998.

[53] P.K. Kaiser. *The Joy of Visual Perception*. Web Book, 1996.

[54] G. Kanisza. *Gramática de la Visión: Percepción y Pensamiento*. Paidos Iberica, Ediciones S. A., 1991.

[55] J.N. Kapur, P.K. Sahoo, and A.K.C. Wong. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing*, 29:273–285, 1985.

[56] K.L. Kelly and D.B. Judd. *Color: Universal Language and Dictionary of Names*. National Bureau of Standarts, 1976.

[57] K.Fu and J.Mui. A survey on image segmentation. *Pattern Recognition*, 13:3–16, 1981.

[58] V. Kravtchenko and J.J. Little. Efficient color object segmentation using the dichromatic reflection model. In *Proceedings of the Pacific Rim Conference on Communications, Computers and Signal Processing*, pages 90–94, 1999.

[59] H. Levkowitz. *Color Theory and Modeling for Computer Graphics, Visualization, and Multimedia Applications*. Kluwer Academic Publishers, 1997.

[60] Cornell University Library. *Digital Imaging Tutorial*. Cornell University Library, 2003.

[61] J. Lillo. *Psicologia de la Percepción*. Debate, 1993.

[62] Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communications*, 28(1):84–95, 1980.

[63] Y. Liu and N. Srihari. Document image binarization based on texture features. *IEEE PAMI*, 19(5):540, 1997.

[64] J.B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, 1967.

[65] A. Majumder. http://www.cs.unc.edu/ majumder/color/paper.html, july 2004.

[66] C. Marimoto and R. Chellappa. Automatic digital image stabilization. In *IEEE International Conference on Pattern Recognition*, 1996.

[67] J. Maurice and G. Lammens. *A computational Model of Color Perception and Color Naming*. PhD thesis, State University of New York, june 1994.

[68] J. Monserrat. *La percepción visual*. Editorial Biblioteca Nueva, 1998.

[69] E. Munar, J. Rosselló, and A. Sánchez-Cabaco. *Atención y Percepción*. Alianza Editorial, 1999.

[70] David Mundie. http://www.anthus.com/colors/nbs.html, 1995.

[71] N. Otsu. A threshold selection method from grey-level histograms. *IEEE Transactions on Systems*, 9:62–66, 1979.

[72] G. Pajares and J.M. de la Cruz. *Visión por Computador. Imágenes digitales y aplicaciones*. Ra–Ma, 2001.

[73] N. Pal and S. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26:1277–1294, 1993.

[74] Pantone. http://www.pantone.com/products/products.asp?idarticle=111&idarea=16, july 2004.

[75] A.J. Pinho and A.J.R. Neves. A survey on palette reordering methods for improving the compression of color-indexed images. *IEEE Transactions on Image Processing*, 13(11):1411–1418, 2004.

[76] K.N. Plataniotis and A.N. Venetsanopoulos. *Color Image Processing and Applications*. Springer, 2000.

[77] T. Pun. A new method for grey-level picture thresholding using the entropy of the histogram. *Signal Processing*, 2:223–237, 1980.

[78] T. Pun. Entropic thresholding. a new approach. *Computer Graphics and Image Processing*, 16:210–239, 1981.

[79] J. Puzicha, M. Held, J. Ketterer, J.M. Buhmann, and D. Fellner. On spatial quantization of color images. *IEEE Transactions on Image Processing*, 9(4):666–682, 2000.

[80] N. Ramesh, J.H. Yoo, and I.K. Sethi. Thresholding based on histogram approximaton. *IEEE Proc Vision Image Signal Processing*, 142(5):271–279, 1995.

[81] P.K. Saha and K. Udupa. Optimum image thresholding via class uncertainty and region homogeneity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:689–706, 2001.

[82] P.V. Sander. Context-based color image compression. In *Computer Science 276r*, Harvard University, 2000. Spring.

[83] G. Sapiro. Color snakes. *Computer Vision and Image understanding*, 68(2):407–416, 1997.

[84] J. Sauvola and M. Pietaksinen. Adaptative document image binarization. *Pattern Recognition*, 33:225–236, 2000.

[85] R. Schettini, G. Ciocca, and S. Zuffi. A survey of methods of colour image indexing and retrieval in image database. In *Color Imaging Science: Exploiting Digital Media*, pages 183–211. John Wiley & Sons, 2002.

[86] M. Sezgin and B. Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13:146–165, january 2004.

[87] W. Skarbek and A. Koschan. Colour image segmentation — a survey. Technical report, Institute for Technical Informatics, Technical University of Berlin, October 1994.

[88] M. Tico, T. Haverinen, and P. Knosmanen. A method of color histogram creation for image retrieval. In *Nordic Signal Processing Symposium*, pages 157–160, 2000.

[89] D.M. Titterington, A.F.M. Smith, and U.E. Markov. *Statistical Analysis of Finite Mixture Distributions*. John Wiley and Sons, 1985.

[90] A. Trémeau, C. Fernandez-Maloigne, and P. Bonton. *Image numérique couleur, de l'acquisition au traitement*. Dunod, 2004.

[91] N. Vlassis and A. Likas. A kurtosis-based dynamic approach to gaussian mixture modeling. *IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems and Humans*, 29(4):393–399, 1999.

[92] H. von Helmholtz. On the theory of compound colors. *Philosophical Magazine*, 4:519–535, 1852.

[93] H. Wang and D. Suter. False-peaks-avoiding mean shift method for unsupervised peak-valley sliding image segmentation. In *Proc. VIIth Digital Image Computing: Techniques and Applications*, pages 581–590, 2003.

[94] M. Wertheimer. Untersuchungen zur Lehre der Gestalt, II. *Psychologische Forschung*, 4:301–350, 1923. Translation published as Laws of Organization in Perceptual Forms, in Ellis, W. (1938). A source book of Gestalt psychology (pp. 71-88). Routledge & Kegan Paul.

[95] W.Kim and R. Park. Color image palette construction based on the hsi color system for minimizing the reconstruction error. In *IEEE International Conference on Image Processing C*, pages 1041–1044, 1996.

[96] G. Wyszecki and W.S.Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley, 1984.

[97] T. Young. On the theory of light and colors. *Philosophical Transactions of the Royal Society*, 91:12–49, 1802.

[98] Z. Zhu, G. Xu, Y. Yang, and J.S. Jin. Camera stabilization based on 2.5d motion estimation and inertial motion filtering. In *IEEE International Conference on Intelligent Vehicles*, 1998.

[99] D. Zugaj and V. Lattuati. A new approach of color images segmentation based on fusing region and edge segmentation outputs. *Pattern recognition*, 31(2):105–113, 1998.