Combined correlation rules to detect skin on colored images based on dynamic color clustering

Rodrigo Augusto Dias Faria

Thesis submitted
to the
Institute of Mathematics and Statistics
of the
University of São Paulo
to
obtain the title
of
Master in Science

Program: Computer Science

Advisor: Prof. Dr. Roberto Hirata Jr

São Paulo, February 2018

Combined correlation rules to detect skin on colored images based on dynamic color clustering

This is the original version of the thesis prepared by the candidate Rodrigo Augusto Dias Faria, such as submitted to the Examining Committee.

Acknowledgements

Texto texto. Texto opcional.

Resumo

FARIA, R. A. D. **Regras de correlação combinadas para detectar pele em imagens coloridas baseadas em agrupamento dinâmico de cores**. Dissertação (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2018.

A detecção de pele desempenha um papel importante em uma ampla gama de aplicações em processamento de imagens e visão computacional. Em suma, existem três abordagens principais para detecção de pele: baseadas em regras, aprendizado de máquina e híbridos. Elas diferem em termos de precisão e eficiência computacional. Geralmente, as abordagens com aprendizado de máquina e as híbridas superam os métodos baseados em regras, mas exigem um conjunto de dados de treinamento grande e representativo, bem como um tempo de classificação custoso, que pode ser um fator decisivo para aplicações em tempo real. Neste trabalho, propomos uma melhoria de um novo método de detecção de pele baseado em regras que funciona no espaço de cores YCbCr. Nossa motivação baseia-se na hipótese de que: (1) a regra original pode ser revertida e, (2) pixels de pele humana não aparecem isolados, ou seja, as operações de vizinhança são levadas em consideração. O método é uma combinação de algumas regras de correlação baseadas nessas hipóteses. Essas regras avaliam as combinações de valores de crominância Cb, Cr para identificar os pixels de pele, dependendo da forma e tamanho dos agrupamentos de cores de pele gerados dinamicamente. O método é muito eficiente em termos de esforço computacional, bem como robusto em cenas de imagens muito complexas.

Palavras-chave: detecção de pele, segmentação de pele humana, modelo de cores YCbCr, regras de correlação, agrupamento dinâmico de cores.

Abstract

FARIA, R. A. D. **Combined correlation rules to detect skin on colored images based on dynamic color clustering**. Thesis (Masters Degree) - Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2018.

Skin detection plays an important role in a wide range of image processing and computer vision applications. In short, there are three major approaches for skin detection: rule-based, machine learning and hybrid. They differ in terms of accuracy and computational efficiency. Generally, machine learning and hybrid approaches outperform the rule-based methods, but require a large and representative training dataset as well as costly classification time, which can be a deal breaker for real time applications. In this work, we propose an improvement of a novel method on rule-based skin detection that works in the YCbCr color space. Our motivation is based on the hypothesis that: (1) the original rule can be reversed and, (2) human skin pixels do not appear isolated, i.e. neighborhood operations are taken in consideration. The method is a combination of some correlation rules based on these hypothesis. Such rules evaluate the combinations of chrominance Cb, Cr values to identify the skin pixels depending on the shape and size of dynamically generated skin color clusters. The method is very efficient in terms of computational effort as well as robust in very complex image scenes.

Keywords: skin detection, human skin segmentation, YCbCr color model, correlation rules, dynamic color clustering.

Contents

List of Acronyms ix

List of Symbols xi

List of Figures xiii

List of Tables xv

1 Introduction 1
- 1.1 Motivação . . . 2
- 1.2 Contributions . . . 2
- 1.3 Objetivos . . . 2
- 1.4 Organização do texto . . . 2

2 Related Work 3

3 Theoretical Background 9
- 3.1 Digital image . . . 9
- 3.2 Basic relationship between pixels . . . 10
  - 3.2.1 Neighborhood . . . 10
  - 3.2.2 Connectivity . . . 11
  - 3.2.3 Arithmetic and logic operations . . . 11
  - 3.2.4 Image boundaries . . . 13
- 3.3 Image histogram . . . 13
- 3.4 Image segmentation . . . 14
  - 3.4.1 Thresholding . . . 15
- 3.5 Color models . . . 16
  - 3.5.1 Munsell color model . . . 16
  - 3.5.2 CIE color model . . . 17
  - 3.5.3 RGB color model . . . 18
  - 3.5.4 CMY color model . . . 19
  - 3.5.5 Color models of the YUV family . . . 20
  - 3.5.6 Color models of the HSI family . . . 21

4 Proposed solution 25
- 4.1 Original method . . . 25

4.2 Extended method . . . 28
4.3 Neighborhood extended method . . . 28
4.4 Supplementary neighborhood operations . . . 29
4.5 Trapezoids parameters tuning with a grid search strategy . . . 32

5 Evaluation **35**
5.1 Datasets . . . 35
5.1.1 UCI . . . 35
5.1.2 SFA . . . 36
5.1.3 Pratheepan . . . 37
5.1.4 HGR . . . 38
5.1.5 Compaq . . . 39
5.2 Evaluation measures . . . 39
5.3 Machine learning experiments . . . 40
5.4 Rule-based experiments . . . 42
5.5 Supplementary neighborhood operations experiments . . . 46
5.6 Grid search parameters experiments . . . 47
5.7 Results and discussion . . . 47

6 Plano de Trabalho **49**
6.1 Atividades previstas . . . 49
6.2 Cronograma proposto . . . 50

A Trapezoids Parameters Tuning Results **51**

Bibliography **59**

List of Acronyms

AR	Aleix and Robert Face Database
CIE	Commission Internationale de l'Eclairage
CMY	Cyan, Magenta and Yellow
FERET	Face Recognition Technology database
HGR	Hand Gesture Recognition database
HSI	Hue, Saturation, Intensity
HSL	Hue, Saturation, Lightness
HSV	Hue, Saturation, Value
ID3	Iterative Dichotomiser 3
IHLS	Improved, Hue, Luminance and Saturation
k-NN	k-Nearest Neighbors
LUT	Look-UP Table
NTSC	National Television System Committee
PAL	Phase Alternating Line
RBF	Radial Basis Function
RGB	Red, Green and Blue
SECAM	Sequential Color with Memory
SFA	Skin of FERET and AR Database
SVM	Support Vector Machines
UCS	Uniform Chromaticity Scale
UCI	University of California in Irvine skin/non skin dataset
YIQ	Luma, Hue and Saturation
YUV	Luma and Chrominance

List of Symbols

$f(x, y)$	Intensity function of an image
W	Number of horizontal samples (lines) of an image
H	Number of vertical samples (columns) of an image
L	Number of gray-levels
L_{min}	The minimum gray-level of a range
L_{max}	The maximum gray-level of a range
L_n	The n-th vector channel of a pixel
L_i	The i-th vector value of a channel of a pixel
$N_4(p)$	Four neighbors of a pixel p
$N_D(p)$	Diagonal neighbors of a pixel p
$N_8(p)$	Eight neighbors of a pixel p
AND	AND logic operator
OR	OR logic operator
XOR	XOR logic operator
NOT	NOT logic operator
L^*	Luminance
a^*	Green/red axis on $L^*a^*b^*$ color model
b^*	Blue/yellow axis on $L^*a^*b^*$ color model
u^*	Green/red axis on $L^*u^*v^*$ color model
v^*	Blue/yellow axis on $L^*u^*v^*$ color model
θ	Hue angle on HSI color model
max	Max operator
min	Min operator
$\mathbb{R}$	Set of real numbers
D	Dataset
N	Dataset cardinality
K	Numbers of dataset partitions in cross-validation
C	Regularization parameter
γ	Polynomial/RBF kernel parameter of a nonlinear SVM
$d(x_i, x_j)$	Distance function between x_i and x_j vectors
argmax	Arguments of maxima operator

List of Figures

3.1 Representation of the raster order of an image . . . 10
3.2 The 4-neighbors representation of a pixel p . . . 11
3.3 The 8-neighbors representation of a pixel p . . . 11
3.4 Representation of an 4-*neighbors* window mask going beyond image borders . . . 13
3.5 Grayscale image with its respective histogram . . . 14
3.6 Gray level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds . . . 15
3.7 Munsell color model. . . . 17
3.8 CIE 1931 chromaticity diagram . . . 18
3.9 Unit cube representing the colors of the RGB model . . . 19
3.10 CMY subtractive color model . . . 20
3.11 Graphical representation of the HSI model . . . 21

4.1 Graphical representation of the trapezoids as well as its parameters . . . 25
4.2 Computation of Cr_{max} based on Cr values histogram of a 724 x 526 image . . . 26
4.3 Neighbors evaluation with respect to a pixel P . . . 28
4.4 Flowchart of our proposed neighbors method . . . 29
4.5 Image samples with the diagonal effect after the neighbors method segmentation . . . 31

5.1 3-dimensional view of the RGB channels of the UCI dataset . . . 36
5.2 Structure of the windows that form the SFA samples . . . 36
5.3 Examples of SFA face image database . . . 37
5.4 Examples of Pratheepan skin dataset . . . 38
5.5 Examples of HGR skin dataset . . . 38
5.6 Examples of Compaq skin/non-skin dataset . . . 39
5.7 UCI samples distribution after splitting the dataset in training and testing subsets . . . 41
5.8 Image samples with the results of each method in SFA dataset . . . 43
5.9 Image samples with the results of each method in Pratheepan dataset . . . 44
5.10 Image samples with the results of each method in HGR dataset . . . 44
5.11 Image samples with the results of each method in Compaq dataset . . . 45

List of Tables

3.1 Definition of addition, subtraction, multiplication and division arithmetic operations in two f_1 and f_2 images . . . 12
3.2 Definition of AND, OR, XOR and NOT logic operations in two f_1 and f_2 images . . 12

5.1 Excerpt with samples from the UCI dataset . . . 35
5.2 Confusion matrix table used during experiments . . . 40
5.3 Grid search table of the parameters of the optimal estimator in the SVM . . . 40
5.4 Grid search table of the parameters of the optimal estimator in k-NN . . . 41
5.5 Results of the experiments with k-NN and SVM in the UCI dataset . . . 41
5.6 Results of the experiments with k-NN and SVM in the SFA and Pratheepan images datasets . . . 42
5.7 Quantitative result metrics of the proposed enhancements and Brancati *et al.* (2017) 42
5.8 Quantitative result metrics of the proposed supplementary neighborhood adaptation 46

6.1 Disciplinas cursadas no programa de mestrado em Ciência da Computação no IME-USP. . . . 49
6.2 Cronograma das atividades previstas . . . 50

A.1 Trapezoids parameters tuning results for Pratheepan dataset and combined rules . . 51
A.2 Trapezoids parameters tuning results for SFA dataset and combined rules . . . 54

Chapter 1

Introduction

Skin detection can be defined as the process of identifying skin-colored pixels in an image. It plays an important role in a wide range of image processing and computer vision applications such as face detection, pornographic image filtering, gesture analysis, face tracking, video surveillance systems, medical image analysis, and other human-related image processing applications.

The problem is complex because of the numerous similar materials with human skin tone and texture, and also because of illumination conditions, ethnicity, sensor capturing singularities, geometric variations, etc. Because it is a primary task in image processing, additional requirements as real time processing, robustness and accuracy are also desirable.

Skin color is a strong characteristic and it is used in most algorithms for skin detection. It is normally used along with other features such as shape, texture, and geometry, or even as a preliminary step to classify regions of interest in an image.

The human skin color pixels have a restricted range of hues and are not deeply saturated, since the appearance of skin is formed by a combination of blood (red) and melanin (brown, yellow), which leads the human skin color to be clustered within a small area in the color space (Fleck *et al.*, 1996).

The choice of a color space is also a key point of a feature-based method when using skin color as a detection cue. Due to its sensitivity to illumination, the input image is, in general, first transformed into a color space whose luminance and chrominance components can be separate to mitigate the problem (Vezhnevets *et al.*, 2003).

Color has the ability of functioning as a descriptor that often simplifies the identification and extraction of an object in a scene. Moreover, the ability of humans to discern thousands of tonalities and intensities compared to only a few dozen levels of gray, put the color as a strong candidate feature in computer vision and image processing applications (Gonzalez and Woods, 2002).

Basically, there are three approaches for skin detection: rule-based, machine learning based and hybrid. They differ in terms of classification accuracy and computational efficiency. Machine learning and hybrid methods require a training set, from which the decision rules are learned. Such approaches outperform the rule-based methods but require a large and representative training dataset as well as it takes a long classification time, which can be a deal breaker for real time applications (Kakumanu *et al.*, 2007).

In this work we propose an improvement of a novel method on rule-based skin detection that works in the YCbCr color space (Brancati *et al.*, 2017). Our motivation is based on the hypothesis that the original rule can be complemented with another rule that is a reversal interpretation of the one proposed originally. Besides that, we also take in consideration that a skin pixel does not appear isolated, so we propose another variation based on neighborhood operations. The set of rules evaluate the combinations of chrominance Cb, Cr values to identify the skin pixels depending on the shape and size of dynamically generated skin color clusters (Brancati *et al.*, 2017). The method is very efficient in terms of computational effort as well as robust in very complex image scenes.

1.1 Motivação

Há diversos objetos ou observações na natureza que não podem ser classificados com conjuntos clássicos pelo fato de que a relação de pertinência não é bem definida (Pedrycz and Gomide, 1998). Especificamente no campo da visão computacional, vários problemas reais de classificação só podem ser resolvidos quando da análise do contexto onde o problema está inserido.

Sendo assim, a motivação deste trabalho está em analisar problemas reais de visão computacional cujos conjuntos de dados possuem incerteza e imprecisão, modelando-os por meio de conjuntos *fuzzy*, explorando a capacidade desses conjuntos de expressarem transições graduais de pertinência e não pertinência.

1.2 Contributions

Enter with the contributions here.

1.3 Objetivos

O principal objetivo deste projeto de mestrado está na avaliação de conjuntos *fuzzy* na aplicação em problemas de classificação em visão computacional, estabelecendo mecanismos comparativos, por meio de indicadores quantitativos e qualitativos, para mensurar seu desempenho em relação aos conjuntos clássicos.

Além disso, a escolha do espaço de cores para a modelagem dos dados como conjuntos *fuzzy* também será analisada com o intuito de compreender sua influência nos resultados obtidos pelo classificador.

1.4 Organização do texto

Quanto à organização deste trabalho, no capítulo 3 são explicitados conceitos fundamentais sobre diferentes modelos de cores, teoria *fuzzy* e classificadores utilizados no desenvolvimento deste estudo. No capítulo 5, são expostos os resultados de experimentos preliminares com conjuntos de dados de pele de seres humanos, bem como a avaliação da influência do modelo de cores escolhido para tratar este tipo de problema de classificação. Por fim, o capítulo 6 apresenta o plano de trabalho com as atividades a serem realizadas no âmbito desta pesquisa.

Chapter 2

Related Work

There are a large number of works of skin detection based on color information and there are a couple of them comparing different techniques and classifiers, mainly from the point of view of performance, color models, skin color modeling and different datasets (Kakumanu *et al.*, 2007; Mahmoodi and Sayedi, 2016; Vezhnevets *et al.*, 2003).

On different techniques, statistical models are those which estimate the probability that an observed pixel is associated with skin, based on a huge training dataset. One approach is the single histogram based Look-UP Table (LUT) that is capable to obtain the distribution of skin pixels in a particular color space, by using a set of training pixels (Mahmoodi and Sayedi, 2016). Considering the RGB color model for instance, a histogram with 256 bins per channel - 256^3 in total - can be constructed for further counting the probability (see Eq. 2) of each possible RGB value (Jones and Rehg, 2002).

$$P(rgb) = \frac{\#counts\ of\ rgb}{total\ counts}$$

In Jones and Rehg (2002), the authors applied this technique to figure out the decision boundary of skin pixels distribution by using a 3-dimensional histogram model constructed from approximately 2 billion pixels. Those pixels were collected from 18,696 images over the Internet to perform skin detection. First, visualization techniques were used to examine the shape of these distributions. Then, by examining the 3D histogram from several angles, Jones and Rehg (2002) realized that its overall shape could be inferred.

So, two different histograms for skin and non-skin in the RGB color space were calculated. Using those histograms along with training data, a surprisingly accurate pixel-wise classifier was derived. The best performance at an error rate of 88% was reached for histograms of size 32.

On the basis of the output of the skin detector, Jones and Rehg (2002) also trained a classifier to determine whether a naked person is present or not in the scene. The features used from the output to create the feature vector were:

- Percentage of pixels detected as skin
- Average probability of the skin pixels
- Size in pixels of the largest connected component of skin
- Number of connected components of skin
- Percent of colors with no entries in the skin and nonskin histograms
- Height of the image
- Width of the image

In the last step, 10,679 images were manually classified into 5,453 naked and 5,226 non-naked image sets to train a neural network classifier. The neural network outputs a number between 0 and

1, with 1 indicating a naked person in the image. The detection rate achieved was of 88%, with a false alarm rate of 11.3%.

Skin detection is sometimes used as a primary task for other applications. Hsu *et al.* (2002) have embedded skin segmentation with the purpose to build a face detector application. The algorithm first performs a lighting compensation on the R, G, and B components of the pixels of the image. Then, these color corrected components are non-linearly transformed into YCbCr space. The skin pixels are detected using an elliptical skin model with Mahalanobis distance, assuming a Gaussian distribution of skin tone color in the color space.

The experiments have shown good skin detection results, around 96% detection rate on HH1 MPEG7 video and 80% on Champion datasets, which contain images with frontal, near-frontal, half-profile and profile faces under varying backgrounds and illumination conditions. On the other hand, the results also showed a high false positive rate. This drawback is reduced further with facial feature detection procedure based on the spatial arrangement of the detected skin patches.

Another typical method explicitly defines, through a number of rules, the boundaries that delimit the grouping of skin pixels in some color space (Vezhnevets *et al.*, 2003). This was the approach adopted by Kovac *et al.* (2003) in the RGB color space, obtaining a true positive rate of 90.66%. Basically, to find out the skin cluster in the RGB color space, the skin color is determined with the following rules (Kovac *et al.*, 2003):

Skin color at uniform daylight illumination

$$R > 95, G > 40, B > 20$$
$$max(R, G, B) - min(R, G, B) > 15$$
$$|R - G| > 15$$
$$R > G$$
$$R > B$$

Skin color at light daylight illumination

$$R > 220, G > 210, B > 170$$
$$|R - G| \leq 15$$
$$R > B$$
$$G > B$$

They (Kovac *et al.*, 2003) also performed experiments in comparison with Hsu *et al.* (2002), where only the chromaticity channels Cb and Cr from the YCbCr color space are used. The results showed that the performance of the classifier is inferior in relation to the approach using all the three channels. Hsu *et al.* (2002) method indeed diminished the influence of noise in dark images, but in images that are captured under standard daylight illumination they label too many pixels as skin, decreasing the performance of the face detector by increasing the number of false positive pixels (Kovac *et al.*, 2003).

Chai and Ngan (1999) proposed a similar method in YCbCr color space, where a skin color map was designed using a histogram approach based on a given set of training images. Chai and Ngan (1999) observed that the Cb and Cr distributions of skin color fall in the ranges [77, 127] and [133, 173], respectively, regardless the skin color variation in different races.

However, after an exhaustive image histogram analysis, Basilio *et al.* (2011) found that the thresholds given by Chai and Ngan (1999) was robust only against images with Caucasian people. Once their (Basilio *et al.*, 2011) purpose was to find human skin from different people races, from any place of the world, a new threshold for each chromaticity (Cb, Cr) channel has been set up, regardless of skin color:

$$80 \leq Cb \leq 120$$
$$133 \leq Cr \leq 173$$

The key advantage of this method is the simplicity of skin detection rules that leads to the construction of a very fast classifier. On the other hand, achieving high recognition rates with this method is difficult because it is necessary to find a good color space and empirically appropriate decision rules (Vezhnevets *et al.*, 2003).

Differently from Kovac *et al.* (2003), the authors of Yogarajah *et al.* (2011) developed a technique where the thresholds defined in the rules are dynamically adapted. The method consists of detecting the region of the eye and extracting an elliptical region to delimit the corresponding face. A Sobel filter is applied to detect the edges of the resulting region which is subjected to a dilation in order to get the optimal non-smooth regions, (i.e. eyes and mouth). The resulting image is subtracted from the elliptical image. As a result, there is a more uniform skin region where the thresholds are calculated.

Every single pixel in the color image is classified as skin and non-skin, based on the calculated dynamic threshold values. When the algorithm detects multiple possible face regions in the image, a dynamic threshold is constructed for each of them and, subsequently submitted to perform skin segmentation on the whole image. Finally, a logical OR operation is applied in all of the segmented regions obtained as of each dynamic threshold (Yogarajah *et al.*, 2011).

The technique was used as part of a preprocessing step in Tan *et al.* (2012) in a strategy combining a 2-dimensional density histogram and a Gaussian model for skin color detection. First, human eyes are located and, then, an elliptical mask model is used to generate the elliptical face region in the image. Due computational simplicity, a Sobel filter is employed to remove non-smooth (i.e., eyes, eye brown, mouth, etc.) regions. Then, the detected edge pixels are further submitted to a dilation operation to get the optimal non-smooth regions. Finally, a new image, that only consists of face regions, is obtained.

It is worth mentioning that the dynamic threshold with smoothed 2-D histogram is based in the assumption that the face and body of a person always share the same colors (Tan *et al.*, 2012).

Thereafter, a 2-D histogram with smoothed densities and a Gaussian model are used as features to represent the skin and non-skin distributions, respectively. They (Tan *et al.*, 2012) also applied a fusion technique that uses the product rule on the two features to obtain better skin detection results.

Experiments were carried out using three public databases: Pratheepan, ETHZ PASCAL, and Stottinger. A comparison between different color spaces as well as proposed fusion and non-fusion approaches were also established. Quantitative results are only available for Stottinger dataset with F-measure = 0.6490, Precision = 0.6403 and Recall = 0.6580 measures.

Naji *et al.* (2012) constructed an explicit classifier in the HSV color space for 4 different – standard skin, shadow/blackish skin, light skin and red concentrated skin and lips – skin ethnic groups in parallel. After primitive segmentation, a rule-based region growth algorithm is applied, in which the output of the first layer is used as a seed, and then the final mask in other layers is constructed iteratively by neighboring skin pixels. The problem of shadow regions have been also addressed by lightening these regions with a skin color correction approach.

A number of 125 images were collected from two different datasets for experiments. The rate of true positive pixels reported was of 96.5% with a very low false positive rate of 0.76%.

Kawulok *et al.* (2013) combined global and local image information to construct a probability map that is used to generate the initial seed for spatial analysis of skin pixels. Seeds extracted using a local model are highly adapted to the image, which greatly improves the spatial analysis result.

In their (Kawulok *et al.*, 2013) proposal, first, the faces are detected in the input image, from where a local skin color model is learned. Skin detection is performed in the input image based on the local model. Next, using a high-probability threshold applied in the local skin probability map,

the seeds are obtained. Finally, based on the seeds gathered on previous step, the spatial analysis is carried out in the global probability map, created on the basis of the global skin color model.

Although color is not used directly in some skin detection approaches, it is one of the most decisive tools that affect the performance of algorithms (Mahmoodi and Sayedi, 2016). Despite the performance of most skin detectors is directly related to the choice of color space, Albiol *et al.* (2001) proved that the optimum performance of the skin classifiers is independent of the color space.

RGB is the most commonly used color space for storing and representing digital images, since the cameras are enabled to provide the images in such model. To reduce the influence of illumination, the RGB channels can be normalized and the third component can be removed, since it does not provide significant information (Kakumanu *et al.*, 2007). This characteristic led Bergasa *et al.* (2000) to construct an adaptive and unsupervised Gaussian model to segment skin into the normalized RGB color space, using only the channels r and g.

In Jayaram *et al.* (2004), a comparative study using a Gaussian approach and a histogram in a dataset of 805 color images in 9 different color spaces has been performed. The results revealed that the absence of the luminance component, which means using only two channels of the color space, significantly impacts the performance as well as the selection of the color space. The best results were obtained in the SCT, HSI and CIELab color spaces with histogram approach.

In Chaves-González *et al.* (2010), the authors compared, under the same circumstances, the performance of 10 color spaces based on the k-means clustering algorithm on 15 images of the Aleix and Robert (AR) face image database (Martínez and Benavente, 1998). The ground truth images have been carefully generated by the authors. This means that parts of the photo which do not include skin color, such as hair, beard, lips, eyes, and background, were completely removed.

Experiments have been executed with k-means clustering method over the test images set for each of the 10 different color spaces. For each color space, Chaves-González *et al.* (2010) have done several tests with each channel lonely, using two channels – with all possible combinations –, and with the three channels.

The tests have been carried out comparing, in a detailed quantitative manner, each color space with an accurate and objective criterion. In other words, the output image produced by the k-means was compared, in a pixel-wise fashion, to the ground truth images to get the quantitative measures. According to the results obtained, using two channels combined did not produce good outcome. On the other hand, the results with each channel isolated were surprisingly good in comparison to the three channels fused. Lastly, the most appropriate color spaces for skin color detection are YCgCr, YDbDr and HSV.

A similar study on color spaces and different skin color modeling have been provided in (Khan *et al.*, 2012). A total of six color spaces (IHLS, HSI, RGB, normalized RGB, YCbCr and CIELab) and nine skin color modeling approaches (AdaBoost, Bayesian network, J48, Multilayer Perceptron, Naive Bayesian, Random Forest, RBF network, SVM and Histogram based) were evaluated on 8991 manually pixel-wise annotated images by means of F-measure. Khan *et al.* (2012) concluded that color space transformation does affect the overall skin performance as well as the removal of the luminance degrades the performance. In addition, classification results can be improved with the usage of lighting correction algorithms.

In Kaur and Kranthi (2012), an algorithm similar to that proposed by Kovac *et al.* (2003) have been implemented, where the boundaries that delimit the grouping of skin pixels are defined by explicit rules. After segmenting the image with the explicit rules, the algorithm also performs morphological and filtering operations to improve the accuracy of the method. The authors applied the algorithm in the YCbCr and CIELab color spaces, ignoring the Y and L luminance components, respectively. The results were more satisfactory when the algorithm was applied on CIELab. A similar technique was implemented in Shaik *et al.* (2015) and Kumar and Malhotra (2015) in the HSV and YCbCr color spaces, the latter providing the best results in both.

Finally, in Brancati *et al.* (2017), a novel rule-based skin detection method that works in the YCbCr color space based on correlation rules that evaluate the combinations of chrominance Cb, Cr values to identify the skin pixels depending on the shape and size of dynamically generated

skin color clusters was proposed. Geometrically, the clusters create trapezoids in the YCb and YCr subspaces that reflect in the inversely proportional behavior of the chrominance components. The method was compared with six well known rule-based methods in literature outperforming them in terms of quantitative performance evaluation parameters. Moreover, the qualitative analysis shows that the method is very robust in critical scenarios.

Chapter 3

Theoretical Background

The image analysis is one of the most important tasks in a computer vision system. Its goal is to create a suitable description with enough information to differentiate the objects in the scene. In general, this description is typically based on shapes, textures, gray levels or color of those objects in the image. With this description, useful interpretation can be extracted from the image by means of an automatic computer system that facilitates human perception.

There is no general agreement among authors regarding where image processing stops and computer vision starts. The first, as the title says, processes the image by applying some transformations on it such that smoothing, sharpening, noise reduction, lightening enhancement, contrasting, stretching, and compression. These will result on a more enhanced and readable image. In addition, the input and output of the process are always images. On the other hand, computer vision has the ultimate goal to use computers to emulate human vision, including learning and being able to make inferences and take actions based on visual inputs (Gonzalez and Woods, 2002). In general, computer vision systems benefit from image processing techniques as pre-processing steps to build better applications. Thus, we can see that they definitely are not different fields, but there is an overlapping between them.

Once this work is intended to explore new methods on human skin detection, we will use techniques from both fields. Color space transformation from image processing, for example, as well as human skin segmentation and understanding as part of computer vision. This is a tentative to imitate the human visual system and its capability to recognize others from the same specie – of course, humans use other characteristics to identify other humans like shape, high, gender, and others, but skin is also part of this recognition system.

Therefore, in this chapter, the theoretical concepts that apply to this research are stated. First, we define and explain the concept of a digital image in section 3.1 as well as basic relationship between pixels in section 3.2. In section 3.3 we will see how image histograms are fundamental for the methods described in chapter 4 of the proposed solution. Also, the background of image segmentation, mainly for the thresholding technique is shown in section 3.4. Thereafter, a brief introduction to color models is provided in section 3.5 in order to give an overview of the main characteristics of some of the most used in the computer vision and image processing area, on which this research is based.

3.1 Digital image

By definition, an image is a two-dimensional function $f(x, y)$, where x and y are spatial coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the *intensity* or *gray level* of the image at that point. The image is said digital when the function $f(x, y)$ is converted to a discrete form. This is made by a process called *digitalization*, which consists of two steps: *sampling* and *quantization* (Gonzalez and Woods, 2002).

Each element of the discrete function $f(x, y)$ is called *pixel* (picture element), where $0 \leq x \leq W - 1$ and $0 \leq y \leq H - 1$. This means that the image can be represented in a matrix form (see

Eq. 3.1), where W is the number of lines and H the number of columns of the matrix. Therefore, W and H defines the size or resolution of the image (Pedrini and Schwartz, 2008).

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,H-1) \\ f(1,0) & f(1,1) & \cdots & f(1,H-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(W-1,0) & f(W-1,1) & \cdots & f(W-1,H-1) \end{bmatrix}$$

Usually, pixels are stored, and therefore read, in this matrix in an order known as *raster*. This information is important so that capture and display devices can be able to establish a common interface, and make necessary transformations in the coordinates, when needed.

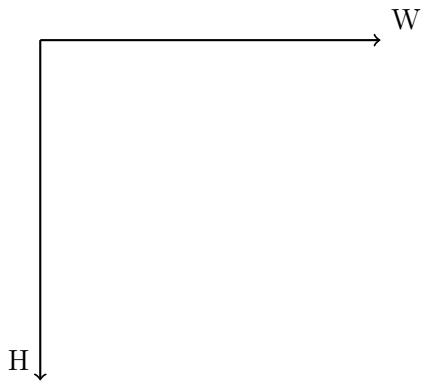

Figure 3.1: *Representation of the raster order of an image. The origin coordinates* $(0,0)$ *starts in the top left corner, where both axes rise. Source: proposed by the author.*

In a monochromatic digital image, the value of a pixel is a scalar in the range $[L_{min}, L_{max}]$, where L is the (integer) number of gray levels (Pedrini and Schwartz, 2008).

In a multispectral image, each pixel has a vector value such that $f(x,y) = (L_1, L_2, \ldots L_n)$ where $L_{min} \leq L_i \leq L_{max}$ and $n = 1, 2, 3, \ldots, n$. In general, L_i can represent different measures for each of (x,y) coordinate as well as different intervals (Pedrini and Schwartz, 2008).

A colored image is a multispectral image, where the color in each (x,y) point is given by three variables: brightness, hue and saturation (Pedrini and Schwartz, 2008). The brightness gives the notion of chromatic intensity. Hue represents the dominant color perceived by an observer. Saturation refers to the relative purity or amount of white light applied to the hue. Combined, hue and saturation are known as chromaticity and, therefore, a color must be characterized by its brightness and chromaticity (Gonzalez and Woods, 2002).

3.2 Basic relationship between pixels

There is a number of applications in image processing and computer vision that uses information of relationship among pixels to create knowledge. Some of these important relationships will be described in the following sections once we will apply it in further chapter 4. It's worth mentioning that we defined an image as a function $f(x,y)$. In this section, when referring to a particular pixel we will denote it in lowercase letters such as p.

3.2.1 Neighborhood

A pixel p with coordinates (x,y) has four horizontal and vertical neighbors whose coordinates are given by:

$$(x+1,y), (x-1,y), (x,y+1), (x,y-1)$$

This set of pixels, called the 4-*neighbors* of p, is denoted by $N_4(p)$ (Gonzalez and Woods, 2002). See figure 3.2 for a reference on how this neighborhood looks like.

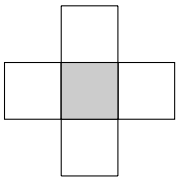

Figure 3.2: *The 4-neighbors representation of a pixel p. The pixel p is centered on the grid with gray background. Source: adapted from Pedrini and Schwartz (2008).*

Each pixel of the image is a unite distance from (x, y). Some neighbors of p might lie outside of the image boundaries if (x, y) is on the border of the image (Gonzalez and Woods, 2002). Those who will use neighborhood operations in the image might take this in consideration to avoid index out of range in those areas.

The four coordinates of the diagonals of p are given by:

$$(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)$$

This set of pixels are denoted by $N_D(p)$. When combined, the 4-*neighbors* and $N_D(p)$ will generate the 8-*neighbors* of p, known as $N_8(p)$ (Gonzalez and Woods, 2002). Formally, we have:

$$N_8(p) = N_4(p) \cup N_D(p)$$

See figure 3.3 for a reference on how the $N_8(p)$ neighborhood looks like.

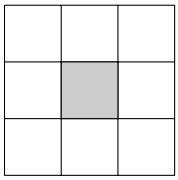

Figure 3.3: *The 8-neighbors representation of a pixel p. The pixel p is centered on the grid with gray background. Source: adapted from Pedrini and Schwartz (2008).*

Despite these are the most common neighbors used in applications, other different distances from p as well as connectivity can be applied. The idea of neighborhood can also be extended to 3-dimensional images where, instead of pixels, the voxels are the coordinates considered.

3.2.2 Connectivity

Connectivity between pixels is a very important concept used to establish the boundaries of objects and regions in an image. To figure out if two pixels are connected, it must be determined if they are neighbors and if their gray levels satisfy some similarity criteria, such as gray levels, color or texture equality. For instance, in a binary image, where the pixels values vary in the range $[0, 1]$, two pixels may be 4-*neighbors*, but they are connected if and only if they have the same value (Gonzalez and Woods, 2002).

3.2.3 Arithmetic and logic operations

Image arithmetic applies one of the standard arithmetic operations or a logical operator to two or more images. The operators are applied in a pixel-wise manner. In other words, the value of a pixel in the output image depends only on the values of the corresponding pixels in the input images. Hence, the images – or the subsets, if is the case – must be of the same size. Although image arithmetic is the most simple form of image processing, they are used extensively in a wide

range of applications and its use can potentially produce very interesting practical results. A main advantage of arithmetic operators is that the process is very simple and therefore fast.

The most common arithmetic operations between two pixels, say $f_1(x, y)$ and $f_2(x, y)$ of two different images f_1 and f_2, are the addition, subtraction, multiplication and division as shown in table 3.1 (Pedrini and Schwartz, 2008).

Name	Operation
Addition	$f_1(x, y) + f_2(x, y)$
Subtraction	$f_1(x, y) - f_2(x, y)$
Multiplication	$f_1(x, y) * f_2(x, y)$
Division	$f_1(x, y)/f_2(x, y)$

Table 3.1: *Definition of addition, subtraction, multiplication and division arithmetic operations in two f_1 and f_2 images. Source: adapted from Pedrini and Schwartz (2008).*

Addition is used often for image averaging to reduce noise. Subtraction is used frequently for static background removal. Multiplication as well as division is applied to correct gray level shading (Gonzalez and Woods, 2002).

Once the arithmetic operations can potentially produce images with values out of the gray levels given in the original images, additional effort is frequently needed to work around this situation. For instance, when adding two images, some pixels of the resulting image may be greater than 255. Similarly, when subtracting two images, some pixels me be with negative values. One way to solve this issue is, after the arithmetic operation, perform a transformation in the gray levels of the resulting image to keep them within a suitable range (Pedrini and Schwartz, 2008).

Logic operations are also useful in computer vision and image processing applications. They are applied only in binary images, while arithmetic operations can be used with higher gray levels. The terminology adopted among authors and application developers is that pixels with zero value (black color) belongs to the objects while one value (white color) corresponds to the background. Table 3.2 shows how the logic operations can be computed (Pedrini and Schwartz, 2008).

Name	Operation
AND	$f_1(x, y)$ AND $f_2(x, y)$
OR	$f_1(x, y)$ OR $f_2(x, y)$
XOR	$f_1(x, y)$ XOR $f_2(x, y)$
NOT	NOT$(f_1(x, y))$

Table 3.2: *Definition of AND, OR, XOR and NOT logic operations in two f_1 and f_2 images. Source: adapted from Pedrini and Schwartz (2008).*

The AND operation outputs 1 in the resulting image when both pixels, at same coordinates in the input images, are equal 1. XOR operation outputs 1 when only one of the pixels – but not both – has value 1; 0 otherwise. The result of OR operation is 1 when at least one of the pixels has 1 value. The NOT operation reverse the value of the pixel in the image (Pedrini and Schwartz, 2008).

All the logic operators can be combined to create other more complex and robust operators. They can be used to combine information between images or to extract information of regions of interest from them (Pedrini and Schwartz, 2008).

In addition to the pixel-wise processing, logic and arithmetic operations can be used for neighborhood processing. Typically, this kind of processing uses masks, where terms such as windows and filters are often used as synonym of masks. The idea of the masks is to turn the value of the

pixel a function of its own value and its neighbors. It is note mentioning that masks application are made under high computational cost. Therefore, they must be used carefully (Pedrini and Schwartz, 2008).

3.2.4 Image boundaries

When working with neighborhood operations, the mask being used, independently of its size, can fall beyond the image boundaries. In other words, this means that, for an image f, with size $W \times H$, some part of the mask operator will be located in a nonexistent pixel from an index out of the range given by $W \times H$ matrix. Figure 3.4 shows an example of this phenomena in a 10×8 image being used as input for an 4-*neighbors* operator.

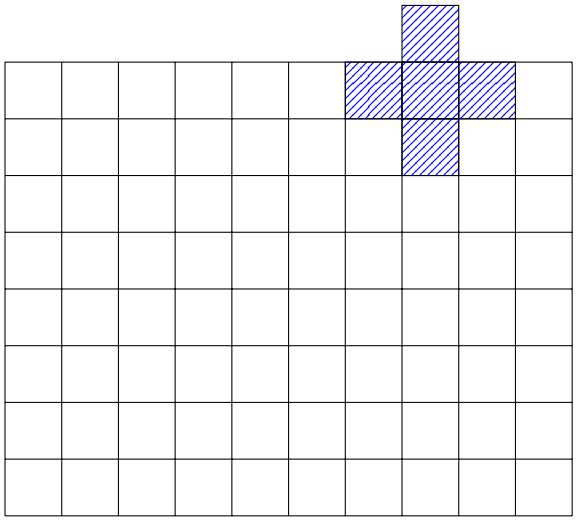

Figure 3.4: *Representation of an 4-neighbors window mask going beyond image borders. Here we can solve this problem, for instance ignoring the first and last lines and columns of the whole image during the operation. Source: adapted from Pedrini and Schwartz (2008).*

There are several ways to work around this problem. One simple mechanism is to simply ignore the pixels on the border where the mask go beyond. Despite this will avoid index out of bound errors, border pixels of the image will not be looked at. Another approach is to copy the corresponding pixels from the input image. Once again, the resulting image will have some not processed pixels in the border. Another strategy is to apply a different mask for the borders, which may consider the difference in the corners as well, to perform the operation, which can turn the operation more complex and computational costly (Pedrini and Schwartz, 2008).

3.3 Image histogram

In an image processing context, the histogram of an image normally refers to a histogram of the pixel intensity values. The most common representation is a graph or plot, which gives an overall idea about the intensity distribution of an image. The foundation of a histogram can be seen as a set of bins, where each bin is representing a certain intensity value from a given range. A simple algorithm to compute a histogram of an image may examines all pixels in the image and assigns each to a bin depending on the pixel intensity. In the end, each bin will have the number of pixels of its own intensity value (Gonzalez and Woods, 2002).

For instance, in an 8-bit grayscale image there are 256 different possible intensities, and so the histogram will graphically display 256 numbers showing the counting of occurrences of those grayscale values. It is also possible to compute histograms of color images. In this particular case, the channels of the color space in use are split individually from where a separate histogram is calculated.

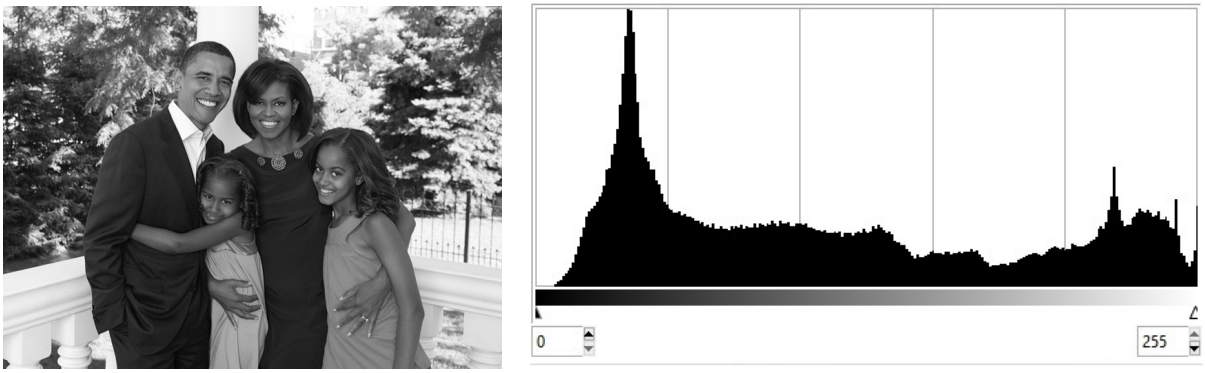

Figure 3.5: *Grayscale image with its respective histogram. Left image is a sample from Pratheepan dataset, transformed to grayscale. On the right we have its histogram of pixel intensity values in the range [0, 255].*

A histogram can be seen as a probability distribution, once the number of pixels for a given intensity level gives an estimate of the probability of occurrence of this intensity level (Gonzalez and Woods, 2002). Several statistical measures can be obtained from a histogram such that minimum and maximum values, mean, median, standard deviation, and percentiles (Pedrini and Schwartz, 2008). Those measures as well as the histogram itself are fundamental for the methods described in chapter 4 of the proposed solution.

3.4 Image segmentation

Image segmentation refers to partitioning an image into different regions that are homogeneous with respect to some image feature (Gonzalez and Woods, 2002). Those regions, or objects, are the fundamental parts of an image. With respect to the human visual perception, humans use their visual sense to effortlessly partition their surrounding environment into different objects to help themselves to recognize them, guide their movements, and for almost every other task in their lives (Plataniotis and Venetsanopoulos, 2000).

Segmentation of complex images is one of the most difficult tasks in the field of image processing, where the accuracy determines whether a successful operation or not (Gonzalez and Woods, 2002). It is a heavy process that includes many interacting components that are involved with the analysis of color, shape, motion, and texture of objects in images. Despite the segmentation of images is a natural activity for the human visual system, it is definitely not easy to create algorithms whose performance is comparable to that of the human visual system (Plataniotis and Venetsanopoulos, 2000).

Image segmentation is usually the first task of any image analysis process. All subsequent tasks, such as feature extraction and object recognition rely heavily on the quality of the segmentation (Plataniotis and Venetsanopoulos, 2000). The algorithms created for image segmentation generally are based on one of two basic properties of intensity values: *discontinuity*, where abrupt changes in intensity, such as points, lines, and edges are detected, and *similarity*, whose approaches are based on regions partitioning, according to a set of predefined criteria (Gonzalez and Woods, 2002).

One of the *similarity* methods it is the thresholding: a popular and clever method used to segment regions of an image, especially in applications where speed is an important factor – that is the case for skin detection, once it is used, in general, for face detection, gesture analysis, face tracking, video surveillance systems, medical image analysis, and other human-related image processing applications. Due the simplicity of implementation, several authors apply this technique on skin detection task (Basilio *et al.*, 2011; Chai and Ngan, 1999; Kaur and Kranthi, 2012; Kovac *et al.*, 2003; Kumar and Malhotra, 2015; Shaik *et al.*, 2015). Others extend this application by using adaptive thresholding (Tan *et al.*, 2012; Yogarajah *et al.*, 2011).

The method proposed by Brancati *et al.* (2017), where a set of dynamic correlation rules are

computed to determine skin pixels can be classified as the kind of *similarity* method. Subsequently, our proposed enhancements described in chapter 4, in a pixel-based manner, is also of this kind. Other methods detailed in chapter 2, aforementioned, and used in Brancati *et al.* (2017) as well here for comparison, are thresholding based. For that reason, a brief introduction on thresholding will be given in section 3.4.1.

3.4.1 Thresholding

Histogram thresholding is one of the simplest pixel-based techniques for image segmentation due its intuitive properties and uncomplicated implementation (Gonzalez and Woods, 2002). Roughly speaking, if an image is composed of distinct regions – which means each region in the histogram are represented by a peak, similar to a Gaussian distribution –, adjacent regions could be split into separated groups, once adjacent peaks are likely separated by a valley (Plataniotis and Venetsanopoulos, 2000). Those valleys can be determined by means of one or more threshold values. However, to find out the threshold value that is the lower bound of a valley is definitely not a trivial task (Plataniotis and Venetsanopoulos, 2000).

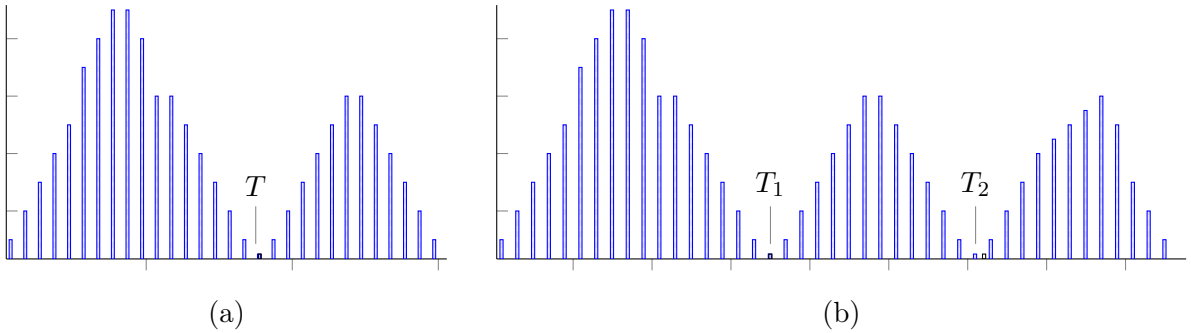

Figure 3.6: *Gray level histograms that can be partitioned by (a) a single threshold, and (b) multiple thresholds. Source: adapted from (Gonzalez and Woods, 2002).*

Let Figure 3.6(a) the gray levels histogram of an image $f(x, y)$. We can clearly see two dominant groups of pixels. One obvious way to separate the objects in this image is to select a threshold T that splits these groups. In other words, for every single point (x, y), if $f(x, y) \leq T$, then (x, y) belongs to the first object (or group), otherwise it belongs to the second object. In general, one of these objects are seen as the background of the image and the other the portion of interest in the image (Gonzalez and Woods, 2002).

The resulting image $g(x, y)$ after applying the thresholding is given by:

$$g(x,y) = \begin{cases} 0, & \text{if } f(x,y) \leq T \\ 1, & \text{otherwise} \end{cases} \tag{3.1}$$

Therefore, pixels labeled 1 corresponds to the object left of T in histogram, and pixels labeled 0 corresponds to the object right of T in histogram – or background as commonly used in literature. The *single-level thresholding* procedure that produces $g(x, y)$ image is known as *binarization* (Gonzalez and Woods, 2002).

Figure 3.6(b) shows a slightly more general case of thresholding, where three distinct groups of the histogram are split by T_1 and T_2 threshold values. Here, *multi-level thresholding* classifies a point (x, y) as belonging to a specific object class as follows (Gonzalez and Woods, 2002):

$$g(x,y) = \begin{cases} l_1, & \text{if } f(x,y) \leq T_1 \\ l_2, & \text{if } T_1 < f(x,y) \leq T_2 \\ l_3, & \text{if } f(x,y) > T_2 \end{cases} \tag{3.2}$$

where l_1, l_2, l_3 are different gray levels used to represent each interval given by the thresholding values.

When T depends only on $f(x, y)$, that is, only on gray level values, the threshold is called *global*. When another property – e.g. the average gray level of a neighborhood centered on (x, y) – is used to define T, the threshold is called *local*. If, in addition, T depends on spatial coordinates x and y, the threshold is called *dynamic* or *adaptive* (Gonzalez and Woods, 2002). This latest is fundamental to either the method developed by Brancati *et al.* (2017) and those enhancements made by us in chapter 4. We can look for them as a particular case of adaptive thresholding to fit the skin model in trapezoidal distributions.

3.5 Color models

The use of color images in computer vision or image processing can be motivated by two main factors. The first refers to the powerful characteristic of color to function as a descriptor that often simplifies the identification and extraction of an object in a scene. The second is related to the ability of humans to discern thousands of tonalities and intensities compared to only a few dozen levels of gray (Gonzalez and Woods, 2002).

The visual perception of color by the human eye should not vary according to the spectral distribution of the natural light incident upon an object. In other words, the color appearance of objects remains stable under different lighting conditions. This phenomenon is known as color constancy (Gevers *et al.*, 2012).

As an example, the grass of a soccer stadium remains green throughout the day, even at dusk when, from a physical point of view, sunlight has a more reddish appearance.

The human perception of colors occurs by the activation of nerve cells that send signals to the brain about brightness, hue and saturation, which are usually the features used to distinguish one color from another (Gonzalez and Woods, 2002).

The brightness gives the notion of chromatic intensity. Hue represents the dominant color perceived by an observer. Saturation refers to the relative purity or amount of white light applied to the hue. Combined, hue and saturation are known as chromaticity and, therefore, a color must be characterized by its brightness and chromaticity (Gonzalez and Woods, 2002).

Colors can be specified by mathematical models in tuples of numbers in a coordinate system and a subspace within that system where each color is represented by a single point. Such models are known as the color models (Gonzalez and Woods, 2002).

These models can be classified as of two types: the additive models in which the primary color intensities are added to produce other colors and subtractive, where colors are generated by subtracting the length of the dominant wave from the white light.

The following sections briefly describe some of the major color models, as well as their variants and main areas of application.

3.5.1 Munsell color model

Pioneer in an attempt to organize the perception of color in a color space, Albert H. Munsell was able to combine the art and science of colors in a single theory (Plataniotis and Venetsanopoulos, 2000).

The principle of equality of visual spacing between the components of the model is the essential idea of the Munsell color model. These components are hue, value, corresponding to luminance, and chroma, corresponding to saturation (Plataniotis and Venetsanopoulos, 2000).

The model is represented by a cylindrical shape and it can be seen in the figure 3.7. The hue is arranged in the circular axis consisting of five base as well as five secondary colors, the saturation in the radial axis and the luminance in the vertical axis in a range varying from 0 to 10.

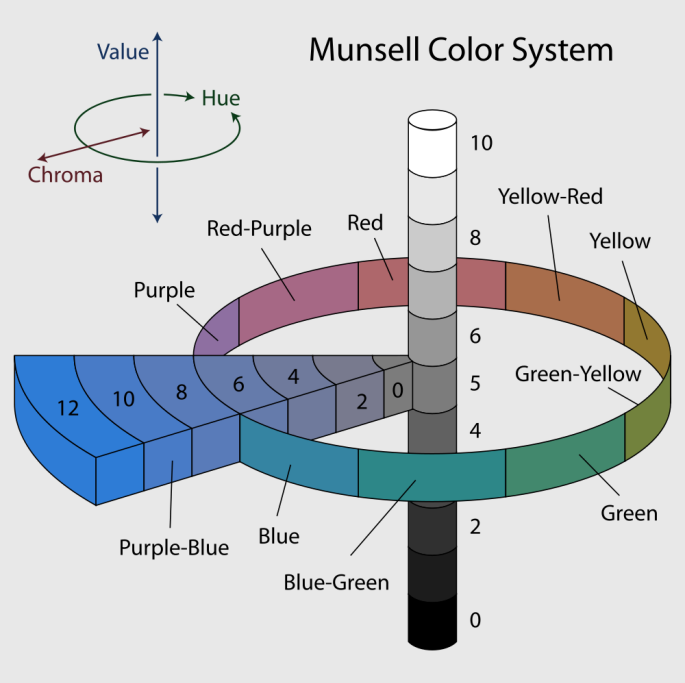

Figure 3.7: *Munsell color model represented by a cylindrical shape. The hue is arranged on the circular axis consisting of five base and five secondary colors, the saturation on the radial axis and the luminance on the vertical axis in a range varying from 0 to 10. Source: Rus (2007).*

3.5.2 CIE color model

In 1931, the CIE established the first mathematical model of color numerical specification, whose objective was to analyze the relationship between the physical aspects of colors in the electromagnetic spectrum and their perception by the human visual system to determine how an ordinary person perceives the color. A review of this specification was published in 1964 (Gonzalez and Woods, 2002).

The experiment that originated the standard consisted in detecting the colors perceived by an observer from a mixture of three primary colors X, Y and Z called tristimulus values. These coordinates gave rise to the CIE XYZ color space which encompasses all the colors that can be perceived by an ordinary human being. For this reason, it is considered an device independent representation (Plataniotis and Venetsanopoulos, 2000).

The system proposed by the CIE XYZ to describe a color is based on a luminance component Y, and two additional components X and Z, that bring the chromaticity information. This system is formed by imaginary colors that can be expressed as combinations of the normalized measures shown in the equations 3.3, 3.4 and 3.5.

$$x = \frac{X}{X+Y+Z} \tag{3.3}$$

$$y = \frac{Y}{X+Y+Z} \tag{3.4}$$

$$z = \frac{Z}{X+Y+Z} \tag{3.5}$$

where $x + y + z = 1$.

Combinations of negative values and other problems related to selecting a set of real primaries are eliminated. The chromaticity coordinates x and y allow to represent all colors in a two-dimensional plane, also known as a chromaticity diagram, which can be seen in the figure 3.8.

The coordinates $(x = 1/3, y = 1/3)$ correspond to the location of white light, also known as

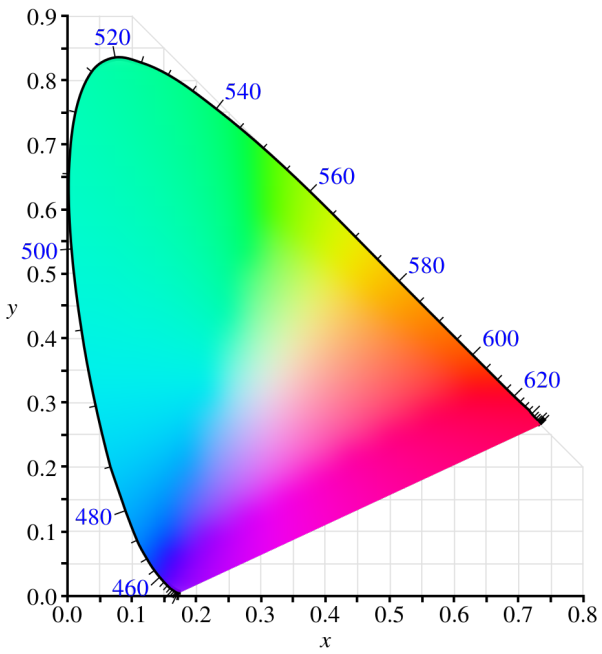

Figure 3.8: *CIE 1931 chromaticity diagram. The points representing pure colors in the electromagnetic spectrum are labeled according to their wavelengths and are located along the curve from the right end of the x-axis, corresponding to the red color, to the left end of the same axis, corresponding to the violet color, forming a polygon similar to a horseshoe. The internal points correspond to all possible combinations of visible colors. Source: Ben (2009).*

white point, and serve as reference in the process of image capture, coding, or reproduction.

CIE also derived and standardized two other color models based on CIE XYZ specification and, likewise, are device independent. Both are perceptually uniform, which means that equal perceptual distances separate all colors in the system (Vezhnevets *et al.*, 2003). As an example, the gray scale of the space should allow for a smooth transition between black and white.

The first one was designed to reduce the problem of perceptual non-uniformity. Some Uniform Chromaticity Scale (UCS) diagrams were proposed based on mathematical equations to transform the values XYZ or the coordinates x, y into a new set of values (u, v), which gave rise to the 1960 CIE uv chromaticity diagram (Gevers *et al.*, 2012).

Still with unsatisfactory results, the CIE made a new change by multiplying the v component by a factor of 1.5. In addition, the brightness scale given by the Y component has been replaced by $L^* = [0, 100]$ to better represent the differences in luminosity that are equivalent. This revision originated the CIE 1976 $L^*u^*v^*$ color model, commonly known by the acronym CIELuv (Gevers *et al.*, 2012).

In 1976 the CIE adopted a new color model, based on the L, a, b model, proposed by Richard Hunter in 1948, which best represented the uniform spacing of colors. Named CIE $L^*a^*b^*$ and known by the acronym CIELab, it is a space based on opponent colors [1] in which the color stimuli of retina is converted to distinctions between light and dark, red and green, and blue and yellow, represented by the axes L^*, a^*, and b^*, respectively (Gevers *et al.*, 2012).

3.5.3 RGB color model

The RGB model, an acronym for Red, Green, and Blue, is an additive color model in which the three primary colors red, green and blue are added to produce the others (Gonzalez and Woods, 2002).

[1] Theory started around 1500 when Leonardo da Vinci concluded that colors are produced by mixing yellow and blue, green and red, and white and black. In 1950, this theory was confirmed when optically-colored signals were detected at the optical connection between the retina and the brain (Gevers *et al.*, 2012).

This system was based on the trichromatic theory of Thomas Young and Hermann Helmholtz in the mid-19th century and can be represented graphically through the unit cube defined on the axes R, G and B, as illustrated in the figure 3.9 (Plataniotis and Venetsanopoulos, 2000).

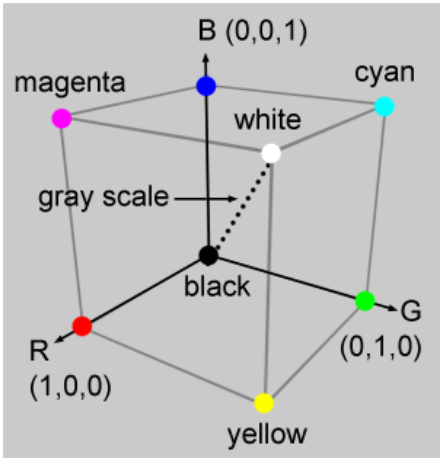

Figure 3.9: *Unit cube representing the colors of the RGB model. The origin, given by the vertex* $(0, 0, 0)$*, represents the black color. The vertex* $(1, 1, 1)$*, opposite the origin, represents the white color. The highlighted vertices on the axes represent the primary colors and the others are the complement of each. Each point inside the cube corresponds to a color that can be represented by the triple* (r, g, b)*, where* $r, g, b \in [0, 1]$*. The shades of gray are represented along the main diagonal of the cube, with each point along this diagonal being formed by equal contributions of each primary color. Source: adapted from Gonzalez and Woods (2002).*

It is noteworthy that there are two ways of representing the RGB space: linear and non-linear. The above-mentioned system shows the non-linear model, whose abbreviation is $R'G'B'$, and is most used by devices and applications because of their similarity to the human visual system. In the literature, this system is frequently cited with the acronym RGB, which makes the nomenclature dubious, since the linear model is also called RGB and, therefore, the conversion between color spaces must be done with some caution. It is also important to note that linear RGB values are rarely used to represent an image since they are perceptually highly non-uniform (Plataniotis and Venetsanopoulos, 2000).

3.5.4 CMY color model

The CMY model is based on the complementary primary colors Cyan, Magenta, and Yellow and, unlike RGB, is a subtractive color model in which colors are generated by subtracting the length of the dominant wave from the white light and, therefore, the resulting color corresponds to the light that is reflected (Gonzalez and Woods, 2002).

One way to get the CMY system is:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} B \\ R \\ R \end{bmatrix} + \begin{bmatrix} G \\ B \\ G \end{bmatrix} \tag{3.6}$$

or by making a change of coordinates by subtracting the primary colors R, G and B of the white color $W = (1, 1, 1)$ (Gonzalez and Woods, 2002):

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.7}$$

Likely RGB, CMY is device dependent. The model is widely used in equipment that deposits colored pigments on paper, such as color printers or photocopiers. The figure 3.10 shows how the model components are combined to generate the other colors.

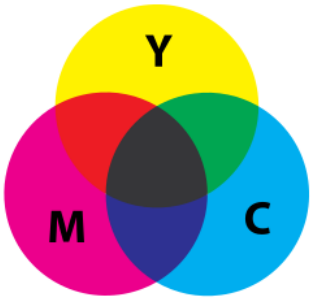

Figure 3.10: *CMY subtractive color model. It is interesting to note that the intersection of yellow with magenta generates the red color, magenta with cyan generates the blue color and cyan with yellow generates the green color. Source: Rus (2008).*

Overlapping the CMY primary colors in equal amounts to generate the black color typically creates a tint that is close to brown or dark green. To avoid this undesired effect, the black component is usually added to the system, represented by the letter K. This operation gives rise to a new model known as **CMYK** (Gonzalez and Woods, 2002).

3.5.5 Color models of the YUV family

Color models of this family is also known as orthogonal color spaces. They are able to reduce the redundancy present in RGB color channels and represent the color with statistically independent components - as independent as possible (Kakumanu *et al.*, 2007).

The acronym YUV stands to a set of color spaces of which the luminance information, represented by the Y component, is coded separately from the chrominance, given by the components U and V. The components U and V are representations of signals of the difference of the blue subtracted from luminance (B-Y) and red subtracted from luminance (R-Y). It is used to represent colors in analogue television transmission systems in the Phase Alternating Line (PAL) and Sequential Color with Memory (SECAM) (Pedrini and Schwartz, 2008).

The transformation of the RGB space to YUV is given by:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.8}$$

where $0 \leq R, G, B \leq 1$.

Analogous to the YUV, the YIQ model was adopted in 1950 by the National Television System Committee (NTSC), an American standard for color television signal transmission. In this model, the Y component corresponds to luminance and the components I (hue) and Q (saturation) encode the chrominance information (Pedrini and Schwartz, 2008).

The transformation of the RGB space to YIQ is given by:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & -0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.9}$$

where $0 \leq R, G, B \leq 1$.

Another color model of the YUV family is the YCbCr, mathematically defined by a coordinate transformation with respect to some RGB space (Pedrini and Schwartz, 2008).

The YCbCr model is widely used in digital videos. In this system, the Y component represents luminance, computed as a weighted sum of RGB values. Cb component gives the measurement of the difference between the blue color and a reference value, similar to the Cr component which is the measurement of the difference between the red color and a reference value (Pedrini and Schwartz, 2008).

The transformation of the RGB space to YCbCr is given by:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.5 \\ 0.5 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.10}$$

3.5.6 Color models of the HSI family

Hue, Saturation, and Intensity (HSI) models are best suited for image processing applications from the user's point of view, due the correlation with human perception of the color(Plataniotis and Venetsanopoulo 2000).

In this model, as in YIQ, the intensity given by I component is decomposed from the chrominance information, represented by the hue (H) and saturation (S) (Plataniotis and Venetsanopoulos, 2000). The combination of these components results in a pyramidal structure which can be seen in figure 3.11.

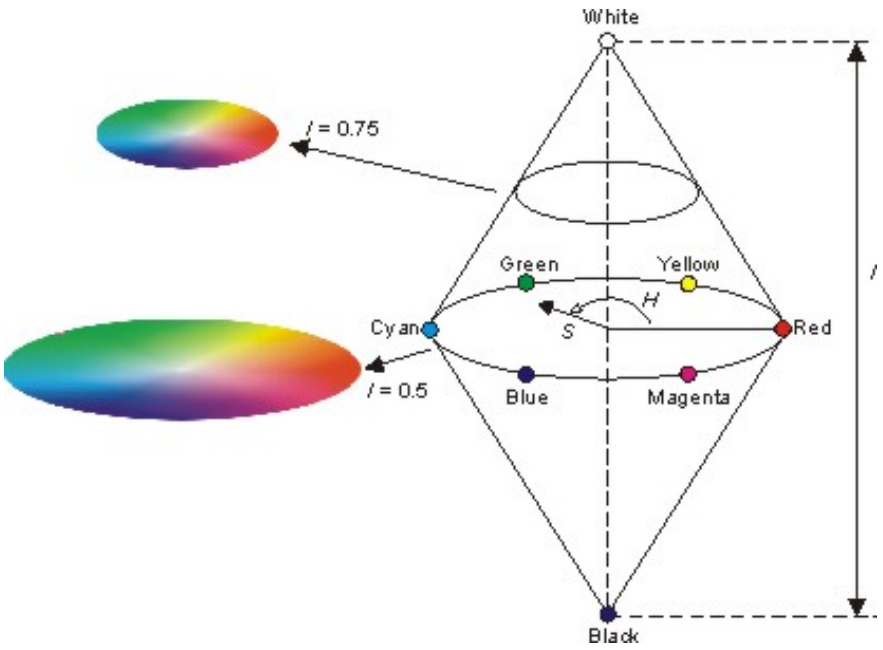

Figure 3.11: *Graphical representation of the HSI model. The hue describes the color itself, in the form of an angle θ, where $\theta \in [0, 360]$. Red is at 0 degree, yellow at 60, green at 120, and so on. The saturation component, which varies between 0 and 1, indicates how much color is polluted with white color. The intensity scale is between $[0, 1]$, where 0 means black and 1, white. Source: Ice (2016).*

The transformation of the components of the RGB space to HSI is given by the equations:

$$\begin{aligned} \theta &= cos^{-1}\left(\frac{(R-G)+(R-B)}{2\sqrt{(R-G)^2+(R-B)(G-B)}}\right) \\ H &= \begin{cases} \theta, & \text{if } B \leq G \\ 360-\theta, & \text{otherwise} \end{cases} \\ S &= 1 - \frac{3min(R,G,B)}{R+G+B} \\ I &= \frac{R+G+B}{3} \end{aligned} \tag{3.11}$$

It is important to note that the values R, G and B must be normalized in the interval between 0 and 1. The intensity I and the saturation S are also normalized between 0 and 1.

Another model of this family is formed by the components Hue, Saturation and Value (HSV) and its three-dimensional graphical representation is a hexagonal pyramid derived from the RGB cube (Pedrini and Schwartz, 2008). Value, in this context, is the luminance component.

The various hue shades are represented at the top of the pyramid, the saturation is measured along the horizontal axis and value is measured along the vertical axis, which passes through the center of the pyramid. The hue, which corresponds to the edges around the vertical axis, varies from 0 (red) to 360 degrees and the angle between the vertices is 60 degrees. The saturation varies from 0 to 1 and is represented as the ratio of the purity of a given hue to its maximum purity, that is, when $S = 1$. Value varies from 0, at the peak of the pyramid representing the black color, to 1 at the base, where the intensities of the colors are maximum (Pedrini and Schwartz, 2008).

The transformation of the components of the RGB space to HSV is given by the equations:

$$\begin{aligned} H &= \begin{cases} 60\dfrac{(G-B)}{M-m}, & \text{if } M = R \\ 60\dfrac{(B-R)}{M-m} + 120, & \text{if } M = G \\ 60\dfrac{(R-G)}{M-m} + 240, & \text{if } M = B \end{cases} \\ S &= \begin{cases} \dfrac{(M-m)}{M}, & \text{if } M \neq 0 \\ 0, & \text{otherwise} \end{cases} \\ V &= M \end{aligned} \tag{3.12}$$

where $m = min(R, G, B)$ and $M = max(R, G, B)$. The luminance V and saturation S are normalized between 0 and 1. The H hue ranges from 0 to 360 degrees.

Similarly to HSV, the Hue, Saturation and Lightness (HSL) model is a three-dimensional representation and is formed by two cones of height 1, whose bases are coincident (Pedrini and Schwartz, 2008).

The hue is determined by the points in the circle of the common bases to the cones. The saturation varies from 0 to 1, depending on the distance to the axis of the cone. The lightness is along the vertical axis common to the two cones and varies in the scale $[0, 1]$, where 0 means black and 1, white (Pedrini and Schwartz, 2008).

The conversion of the RGB space to HSL is given by the equations:

$$\begin{aligned} H &= \begin{cases} 60\dfrac{(G-B)}{M-m}, & \text{if } M = R \\ 60\dfrac{(B-R)}{M-m} + 120, & \text{if } M = G \\ 60\dfrac{(R-G)}{M-m} + 240, & \text{if } M = B \end{cases} \\ S &= \begin{cases} \dfrac{(M-m)}{M+m}, & \text{if } 0 < L \leq 0,5 \\ \dfrac{(M-m)}{2-(M+m)}, & \text{if } L > 0,5 \\ 0, & \text{if } M = m \end{cases} \\ L &= \frac{M+m}{2} \end{aligned} \tag{3.13}$$

where $m = min(R, G, B)$ and $M = max(R, G, B)$. The lightness V and saturation S are normalized between 0 and 1. Note that the transformation of the H component is the same as that used in the conversion of the RGB to HSV space in 3.12 and varies between 0 and 360 degrees.

All the color models of this family have the property of thinking of lighter colors, obtained by

increasing the brightness or lightness, and darker colors, by the diminution of the same values. The intermediate colors are produced by decreasing the saturation (Pedrini and Schwartz, 2008).

Chapter 4

Proposed solution

A state of the art skin detection method has been recently developed by Brancati *et al.* (2017). Here, we review the method and extend it adding more rules to enforce the constraints and seeking for a better performance in terms of false positive rate without hurting the performance of the original method.

4.1 Original method

In order to describe the proposed extensions, we will first transcribe the original method that is based on the definition of image-specific trapezoids, named T_{YCb} and T_{YCr}, in the YCb and YCr subspaces, respectively. The trapezoids are essential to verify a relation between the chrominance components Cb and Cr in these subspaces (Brancati *et al.*, 2017).

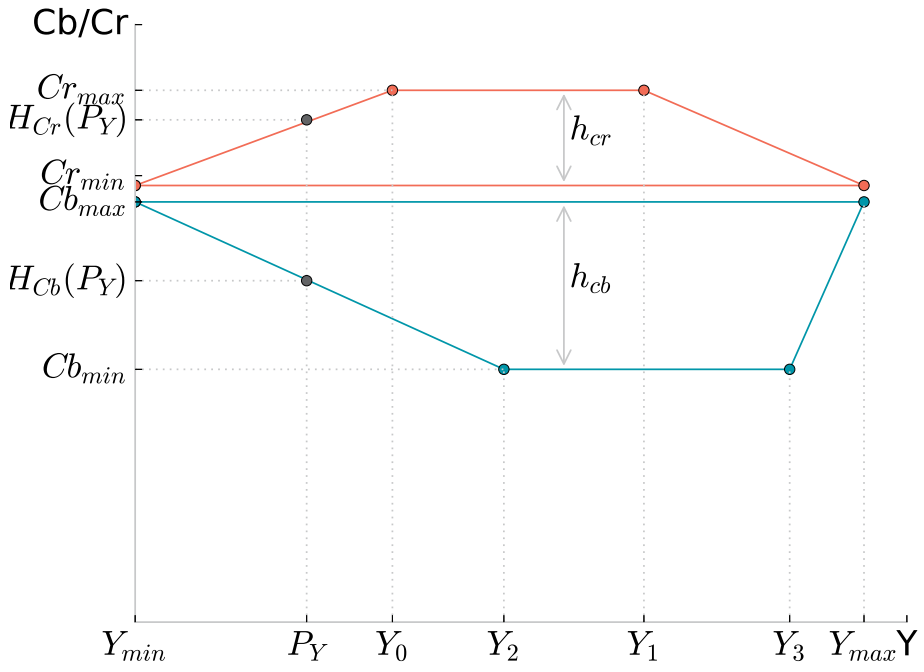

Figure 4.1: *Graphical representation of the trapezoids as well as the parameters* $Y_{min} = 0$, $Y_{max} = 255$, Y_0, Y_1, Y_2, Y_3, Cr_{min}, Cr_{max}, Cb_{min}, Cb_{max}, h_{Cr}, h_{Cb}, $H_{Cr}(P_Y)$, $H_{Cb}(P_Y)$. *Source: adapted from (Brancati et al., 2017).*

The base of the trapezoids T_{YCr} and T_{YCb} (Fig. 4.1) are given by (Y_{min}, Cr_{min}) and (Y_{min}, Cb_{max}) in the YCr and YCb subspaces, respectively. The values $Cr_{min} = 133$, $Cb_{max} = 128$ were selected

according to Chai and Ngan (1999) where a skin color map was designed using a histogram approach based on a given set of training images. Chai and Ngan observed that the Cr and Cb distributions of skin color falls in the ranges [133, 173] and [77, 127], respectively, regardless the skin color variation in different races.

The Cr_{max} parameter is calculated dynamically, taking into account the histogram of the pixels with Cr values in the range $[Cr_{min}, 183]$, looking for the maximum value of Cr associated with at least 0.1%[1] of pixels in the image. The same applies to Cb_{min}, taking the histogram with Cb values in the range $[77, Cb_{max}]$. Y_0 and Y_1 (shorter base of the upper trapezoid) are, respectively, the $5th$ and $95th$ percentile of the luminance values associated with the pixels of the image with $Cr = Cr_{max}$. A similar procedure is used to find the values of the shorter base of the other trapezoid, Y_2 and Y_3 (see Fig. 4.2 for an example).

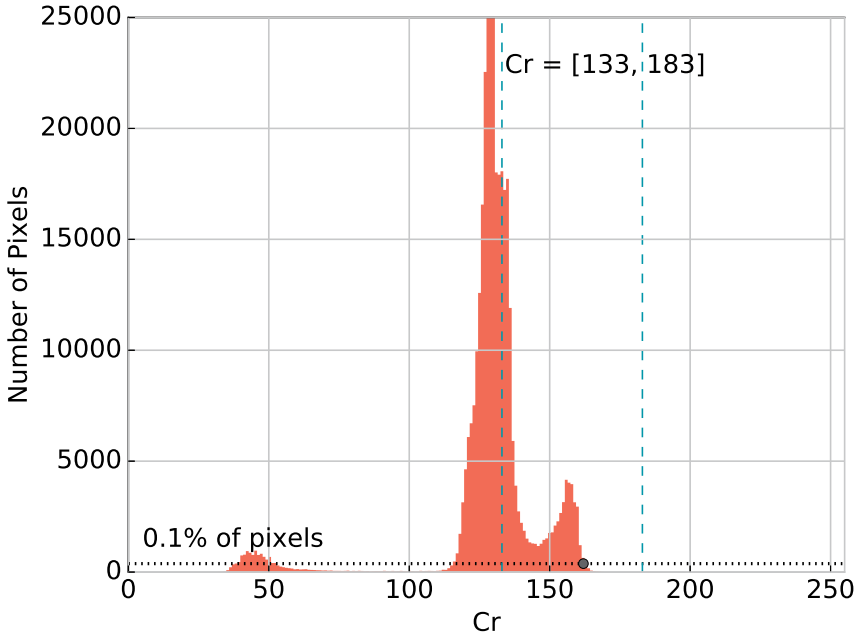

Figure 4.2: *Computation of $Cr_{max} = 162$ based on Cr values histogram of a 724 x 526 image.*

The correlation rules between the chrominance components P_{Cr} and P_{Cb} of a pixel P are defined as:

- the minimum difference between the values P_{Cr} and P_{Cb}, denoted I_P;
- an estimated value of P_{Cb}, namely P_{Cb_s};
- the maximum distance between the points (P_Y, P_{Cb}) and (P_Y, P_{Cb_s}), denoted J_P.

Therefore, to determine if P is skin, the following equations must hold:

$$P_{Cr} - P_{Cb} \geq I_P \tag{4.1}$$

$$|P_{Cb} - P_{Cb_s}| \leq J_P \tag{4.2}$$

The estimated value P_{Cb_s} is given by [2]:

$$P_{Cb_s} = Cb_{max} - dP_{Cb_s} \tag{4.3}$$

[1] In Brancati *et al.* (2017) this rate is reported to be equal to 10%. However, in the distributed source code we found the value 0.1%, that we are using in the experiments.

[2] dP_{Cb_s} is the distance between the points (P_Y, P_{Cb_s}) and (P_Y, Cb_{max}) in the YCb subspace, calculated on the basis of dP_{Cr}, observing the inversely proportional behavior of the components. α is the rate between the normalized heights of the trapezoids in relation to the P_Y value.

where [3]:

$$dP_{Cb_s} = \alpha \cdot dP_{Cr} \tag{4.4}$$

$$dP_{Cr} = P_{Cr} - Cr_{min} \tag{4.5}$$

The coordinates of the other sides of the trapezoids are given by $[P_Y, H_{Cr}(P_Y)]$ and $[P_Y, H_{Cb}(P_Y)]$, such that:

$$H_{Cr}(Y) = \begin{cases} Cr_{min} + h_{Cr}\left(\frac{Y - Y_{min}}{Y_0 - Y_{min}}\right) & Y \in [Y_{min},\ Y_0] \\ Cr_{max} & Y \in [Y_0,\ Y_1] \\ Cr_{min} + h_{Cr}\left(\frac{Y - Y_{max}}{Y_1 - Y_{max}}\right) & Y \in [Y_1,\ Y_{max}] \end{cases} \tag{4.6}$$

$$H_{Cb}(Y) = \begin{cases} Cb_{min} + h_{Cb}\left(\frac{Y - Y_2}{Y_{min} - Y_2}\right) & Y \in [Y_{min},\ Y_2] \\ Cb_{min} & Y \in [Y_2,\ Y_3] \\ Cb_{min} + h_{Cb}\left(\frac{Y - Y_3}{Y_{max} - Y_3}\right) & Y \in [Y_3,\ Y_{max}] \end{cases} \tag{4.7}$$

where $h_{Cr} = Cr_{max} - Cr_{min}$ and $h_{Cb} = Cb_{max} - Cb_{min}$, which are the heights of T_{YCr} and T_{YCb}, respectively.

The computation of those points are useful for the calculation of α. We first compute the distances $\Delta_{Cr}(P_Y)$ and $\Delta_{Cb}(P_Y)$ between the points $(P_Y, H_{Cr}(P_Y))$, $(P_Y, H_{Cb}(P_Y))$ and the base of the trapezoids:

$$\Delta_{Cr}(P_Y) = H_{Cr}(P_Y) - Cr_{min} \tag{4.8}$$

$$\Delta_{Cb}(P_Y) = Cb_{max} - H_{Cb}(P_Y) \tag{4.9}$$

Next, the distances are normalized with respect to the difference in size of the trapezoids:

$$\Delta'_{Cr}(P_Y) = \begin{cases} \Delta_{Cr}(P_Y) \cdot \frac{A_{T_{YCb}}}{A_{T_{YCr}}} & \text{if } A_{T_{YCr}} \geq A_{T_{YCb}} \\ \Delta_{Cr}(P_Y) & \text{otherwise} \end{cases} \tag{4.10}$$

$$\Delta'_{Cb}(P_Y) = \begin{cases} \Delta_{Cb}(P_Y) & \text{if } A_{T_{YCr}} \geq A_{T_{YCb}} \\ \Delta_{Cb}(P_Y) \cdot \frac{A_{T_{YCr}}}{A_{T_{YCb}}} & \text{otherwise} \end{cases} \tag{4.11}$$

where $A_{T_{YCr}}$ and $A_{T_{YCb}}$ are the areas of trapezoid T_{YCr} and T_{YCb}, respectively.

Then, the value of α is given by:

$$\alpha = \frac{\Delta'_{Cb}(P_Y)}{\Delta'_{Cr}(P_Y)} \tag{4.12}$$

Finally, I_P and J_P are given by:

$$I_P = sf \cdot [(\Delta'_{Cr}(P_Y) - dP_{Cr}) + (\Delta'_{Cb}(P_Y) - dP_{Cb_s})] \tag{4.13}$$

$$J_P = dP_{Cb_s} \cdot \frac{dP_{Cb_s} + dP_{Cr}}{\Delta'_{Cb}(P_Y) + \Delta'_{Cr}(P_Y)} \tag{4.14}$$

where:

$$sf = \frac{min((Y_1 - Y_0), (Y_3 - Y_2))}{max((Y_1 - Y_0), (Y_3 - Y_2))} \tag{4.15}$$

[3] dP_{Cr} is the distance between (P_Y, P_{Cr}) and (P_Y, Cr_{min}) points in the YCr subspace.

4.2 Extended method

The hypothesis defined in the original method is based on rules that an estimated value of the point P_{Cb}, namely P_{Cb_s}, must hold in order for the correlation to be valid. On the basis of the inversely proportional behavior of the chrominance components, we will rewrite the correlation rules with respect to the P_{Cr} point.

Thus, we have to refactor the correlation rules to put them in terms of the estimated value of P_{Cr}, that we denote as P_{Cr_s} [4]:

$$P_{Cr_s} = dP_{Cr_s} + Cr_{min} \tag{4.16}$$

where [5]:

$$dP_{Cr_s} = \alpha \cdot dP_{Cb} \tag{4.17}$$

$$dP_{Cb} = Cb_{max} - P_{Cb} \tag{4.18}$$

Next, the constraints given by I_P and J_P in the Eq. 4.13 and 4.14 respectively, can be redefined as:

$$I_P^{'} = sf \cdot [(\Delta_{Cr}^{'}(P_Y) - dP_{Cr_s}) + (\Delta_{Cb}^{'}(P_Y) - dP_{Cb})] \tag{4.19}$$

$$J_P^{'} = dP_{Cr_s} \cdot \frac{dP_{Cb} + dP_{Cr_s}}{\Delta_{Cb}^{'}(P_Y) + \Delta_{Cr}^{'}(P_Y)} \tag{4.20}$$

Therefore, to determine if the pixel P is skin, we have to modify the conditions given by Eq. 4.1 and 4.2:

$$P_{Cr} - P_{Cb} \geq I_P^{'} \tag{4.21}$$

$$|P_{Cr} - P_{Cr_s}| \leq J_P^{'} \tag{4.22}$$

Doing this simple extension, we are now able to apply the method to the same sets of images to evaluate, in fact, the inversely proportional behavior of the chrominance components. More than that, we can combine all these constraints, given by the pair equations 4.1 and 4.2, 4.21 and 4.22, to reinforce the firstly defined hypothesis.

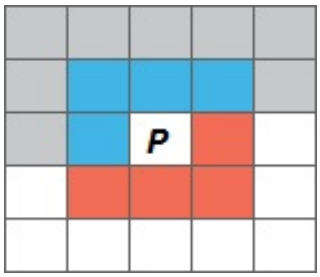

Figure 4.3: *Neighbors evaluation with respect to P. If the image is scanned in raster order, $N_8^-(P)$ is the set of points that can be reached before P in an 8-neighbors window.*

4.3 Neighborhood extended method

Both methods presented in sections 4.1 and 4.2 can be applied to detect skin pixels, either separated or in a conjunction rule. However, skin pixels do not usually appear isolated and we can improve the method using neighbor pixels information, when evaluating a pixel P, in order to decide if P

[4] dP_{Cr_s} is the distance between the points (P_Y, P_{Cr_s}) and (P_Y, Cr_{min}) in the YCr subspace, calculated on the basis of dP_{Cb}, observing the inversely proportional behavior of the components. α is the rate between the normalized heights of the trapezoids in relation to the P_Y value.

[5] dP_{Cb} is the distance between (P_Y, P_{Cb}) and (P_Y, Cb_{max}) points in the YCb subspace.

represents human skin, or not. Let $N_8^-(P)$ the 8-*neighbors* of P that can be reached before P when scanning the image in raster order (blue points in Fig. 4.3).

Thus, we classify P as skin in the following manner: if the constraints given by the pair of equations 4.1 and 4.2, as well as 4.21 and 4.22 hold, then P is classified as skin. When only one of conditions is satisfied, then we check the decision in $N_8^-(P)$. If three or more pixels are skin, then P will also be classified as a skin pixel. Figure 4.4 shows a flowchart of the aforementioned procedure described.

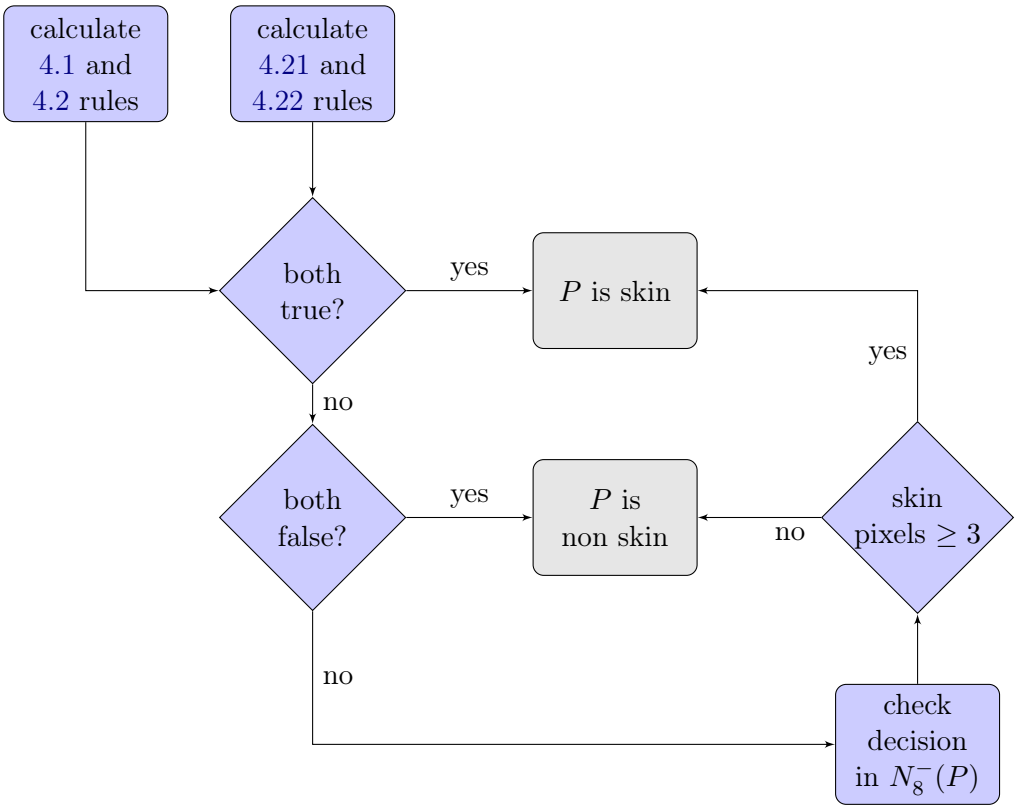

Figure 4.4: *Flowchart of our proposed neighbors method. In* ***both false*** *decision, the* ***no*** *path means that one of the rules is true and we are in doubt if P is skin or not – here is where the neighbors are used to find out the label of P. Source: proposed by the author.*

4.4 Supplementary neighborhood operations

The given neighborhood method presented in section 4.3 will end up with an undesired behavior on the output images that we called *diagonal effect* (see Fig. 4.5). This is caused due the shape of the window being used. Once we look only for the four already seen pixels of the 8-*neighbors* window, the operation is so based in a non-symmetry mask. Ideally, we should look for all the eight neighbors of the pixel P being evaluated.

Therefore, we created an adaptation of the neighborhood method shown in 4.3. In this version, we basically scan the image, with a size of $W \times H$, in the raster order, and apply the original and reverse rules for every single pixel. We keep the result in a matrix of the same size $(W \times H)$ of the input image. For each coordinate of this output matrix, we will have a two positions vector with the result of the original and reverse rules answer for this pixel. Next, we read each position of this output matrix and apply the 8-*neighbors* operations in four different ways:

(1) we look into the rules answer performing an AND. In other words, if both original and reverse rules are saying this pixel is skin, then we classify it as skin;

(2) we look into the rules answer performing an OR. In other words, if one of the rules (original or reverse) is saying this pixel is skin, then we classify it as skin;

(3) we look into the neighbors only querying the original (P_{Cb_s}) rules;

(4) we look into the neighbors only querying the reversed (P_{Cr_s}) rules.

Of course, this variation will add some additional computational cost once we will scan the image one more time. This implementation can be enhanced, but the idea here is to only explore better the connectivity of the 8-*neighbors* window and check, on the basis of a symmetric mask window, if the *diagonal effect* is gone as well as the measures are improved. Some experiments can be seen further in section 5.5.

Figure 4.5: *Image samples with the diagonal effect after the neighbors method segmentation. Each image is from (top-down) SFA, Pratheepan, HGR, and Compaq datasets, respectively, where: (a) original image (b) ground truth (c) combined method (f) neighbors method. Independently of the classification accuracy, we can clearly see the diagonal effect present in the output of the neighbors method segmentation in comparison with combined. Besides being a visually undesirable effect, this phenomenon causes us to have an increase in the false positive rate.*

4.5 Trapezoids parameters tuning with a grid search strategy

As we could see on previous sections, this model is based on the definition of trapezoids to fit the skin color pixels distribution from a given image. According to Brancati *et al.* (2017), the coordinates of each trapezoid within the YCr and YCb sub-spaces are calculated based on the Y luminance component values. Y_0 and Y_1 are those used to define the shorter base of the upper trapezoid, and Y_2 and Y_3 the points used to define the shorter base of the lower trapezoid.

Y_0 and Y_1 are, respectively, the $5th$ and $95th$ percentile of the luminance values associated with the pixels of the image with $Cr = Cr_{max}$ (see Fig. 4.2 for an example). Similarly, Y_2 and Y_3 are, respectively, the $5th$ and $95th$ percentile of the luminance values associated with the pixels of the image with $Cb = Cb_{min}$.

To the best of our knowledge, there is any justification to choose the $5th$ and $95th$ percentile to be the right parameters to define the trapezoids coordinates. For this reason, we decided to shot different combination of these parameters to figure out which pair better works for the model fitting. In addition, we would like to answer the question: why $5th$ and $95th$ percentile have been used?

Thus, we used a well-known technique called grid search to find the best range combination from a hyper-parameter space. By constructing the model in this manner, we can leverage the classification results by finding the optimized parameters' combination, if other different from the ones defined earlier by Brancati *et al.* (2017). Despite we do not exhaustively consider all parameter combinations, we used an efficient search strategy by sampling a given number of candidates. For each chosen parameters candidate, we dynamically used them in the combined method[6] to test every single image of each dataset described in section 5.1. Lastly, we sort the results table using respectively, $F-measure$, $Precision$, and $Recall$ metrics, as detailed in section 5.2, and established a comparison to get optimized parameters. The algorithm TRAPEZOIDS-PARAMETERS-GRID-SEARCH($dataset$) shows how the grid search have been performed.

In short, TRAPEZOIDS-PARAMETERS-GRID-SEARCH($dataset$) will trigger combinations of $Ymin$ and $Ymax$ in the range $[5, 95]$ with a step of 5. In line 6, the parameters are changed in the combined method and applied further in 7 to classify each image within the dataset. Next, the metrics are computed on every single image in line 8 by comparing the output with the ground truth. The resulting metrics are pushed into $Results$ matrix in line 9 and sorted according the criteria aforementioned in the end of the procedure.

```
TRAPEZOIDS-PARAMETERS-GRID-SEARCH(dataset)
1  Ymin = 5
2  Results = []
3  while Ymin ≤ 95
4      Ymax = Ymin
5      while Ymax ≤ 95
6          SET-COMBINED-METHOD-Y-PARAMETERS(Ymin, Ymax)
7          SEGMENT-DATASET-IMAGES(dataset)
8          Precision, Recall, Fmeasure = COMPUTE-METRICS(dataset)
9          PUSH(Results, Ymin, Ymax, Precision, Recall, Fmeasure)
10
11         Ymax = Ymax + 5
12     Ymin = Ymin + 5
13
14 SORT-BY(Results, Fmeasure, Precision, Recall)
15 return Results
```

Detailed output of this experiments and analysis can be seen in chapter 5, specifically in section 5.6, where an examination of the behavior of the most performing parameters is given as well

[6] In fact, we could apply this approach in any of the described methods, but once the trapezoids definition do not change among them, we think that employing it only in combined method is sufficient for the parameters optimization.

as others which did not succeed.

Chapter 5

Evaluation

In this section we present some experimental evaluations of the proposed extensions along with the original method in three widely known datasets: SFA, Pratheepan and HGR. In addition, a brief definition of the evaluation metrics used is shown for the sake of clarity.

5.1 Datasets

5.1.1 UCI

Named UCI in this work, this dataset was proposed by Bhatt and Dhall (2012) and obtained from the machine learning repository of the University of California in Irvine (Lichman, 2013). The dataset consists pixel samples of images of various skin and non-skin textures obtained from thousands of arbitrary faces images of different ages, sex and races (Minear and Park, 2004; Phillips *et al.*, 1996).

The UCI contains 245,057 samples, composed of 3 attributes that constitute the input vector $x = [x_1, x_2, x_3]$, $x \in \mathbb{R}^d$, where d is the space dimension which represents, respectively, blue (B), green (G) and red (R) channels of the RGB color model. In addition, a fourth column determines the class to which the sample x belongs, denoted by y, where $y \in Y$ and $Y = \{+1, -1\}$. In other words, each sample is an RGB pixel with a given label.

The table 5.1 exemplifies a short excerpt from the UCI database. It is worth mentioning that 194,198 out of the 245,057 are non-skin pixels and 50,859 pixels with different skin tones. In addition, the images that were used to extract the dataset were not made available by the authors.

B	G	R	Label
74	85	123	1
207	215	255	1
74	82	122	1
202	211	255	1
54	72	125	1
...	...	...	...
166	164	116	-1
148	150	91	-1
29	26	5	-1
167	166	115	-1
180	177	133	-1

Table 5.1: *Excerpt with samples from the UCI dataset. Each of the first three columns represents a pixel channel of the RGB color space ranging from 0 to 255. The fourth column is the label assigned to the sample, which can assume +1 if it is skin and -1, otherwise. Originally, the class representing a non-skin pixel had value 2, replaced by -1 for compliance with the experiments.*

Since the data are points in the RGB space, it is possible to plot it for a better interpretation of them, as shown in the figure 5.1.

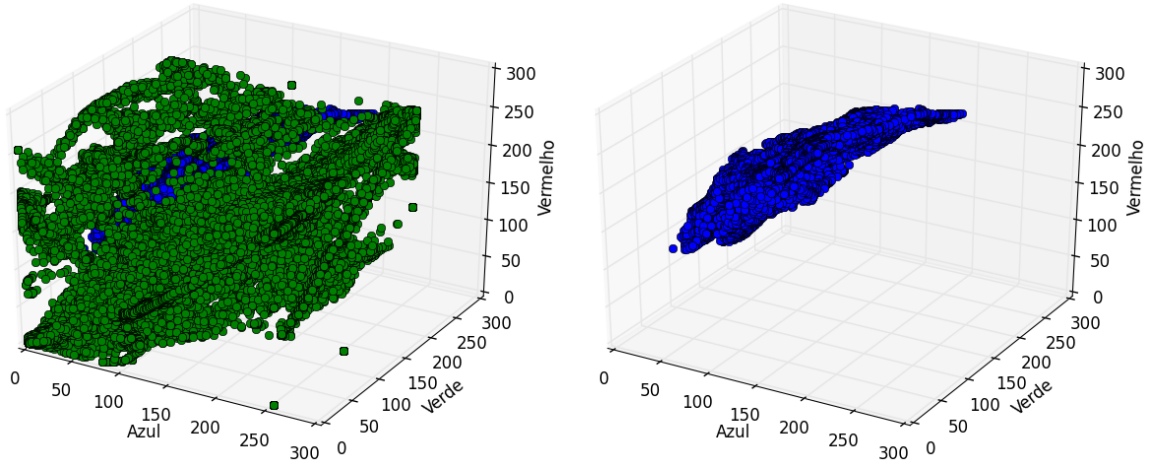

Figure 5.1: *3-dimensional view of the RGB channels of the UCI dataset. The blue points are skin samples and the green ones are non-skin. On the left are all samples of the dataset; to the right only the skin samples. Source: proposed by the author.*

5.1.2 SFA

SFA, the name of the dataset proposed by Casati *et al.* (2013), stands for Skin of FERET and AR Database. The SFA is a set of images of frontal faces obtained from two other color image databases: the FERET, created by Phillips *et al.* (1996), and the AR proposed by Martínez and Benavente (1998), which provided 876 and 242 images each, respectively. It is important to notice that AR images have a white background and small variations of skin color. In other words, the environment is more controlled than the images in FERET Casati *et al.* (2013). Figure 5.3 shows some of the 1,118 samples available.

Casati *et al.* (2013) also extracted different window patches of each skin and non-skin samples to facilitate future research. The samples were randomly generated considering the ground truth mask [1] of each image, being three samples of skin and five of non-skin. Each sample is a window of size $n \times n$, where n is odd, with a central pixel, from which other sample sizes have been created, ranging from 1×1 to 35×35, as can be seen in figure 5.2.

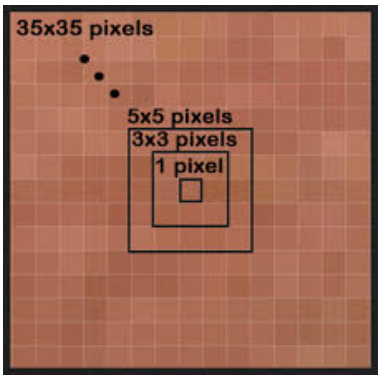

Figure 5.2: *Structure of the windows that form the SFA samples. In total, there are 3,354 skin samples and 5,590 non-skin samples for each window size. Source: Casati et al. (2013).*

[1] Ground truth is the term used to denote an image whose point of interest is properly segmented and highlighted, discarding the remaining pixels giving them uniform colors.

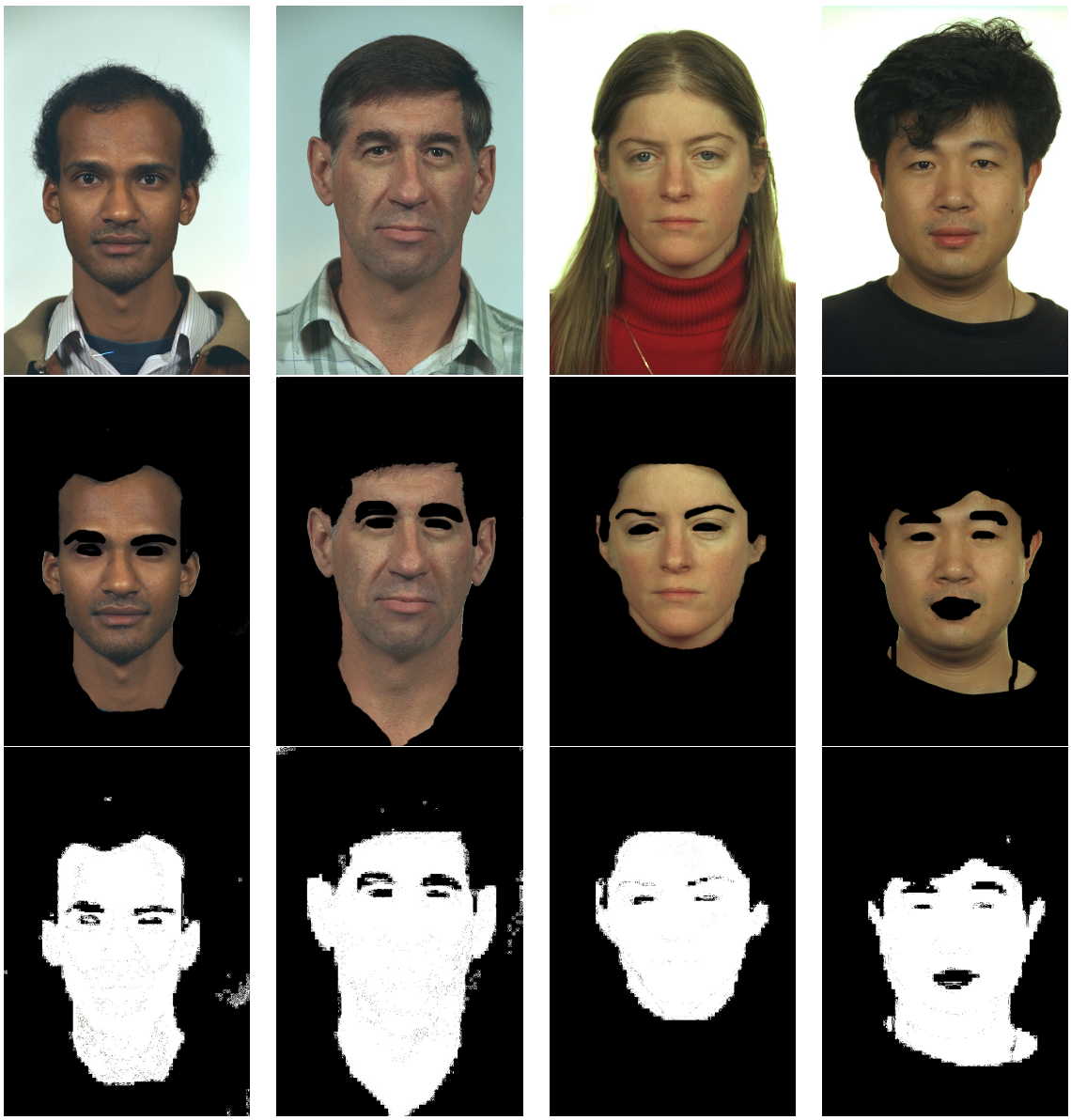

Figure 5.3: *Examples of SFA face image database. First row are the original images and the second contain the colored ground truth with the skin color pixels annotated manually. The black color RGB = (0, 0, 0) was assigned to all pixels in the background. In the third row we have the binary ground truth images. We generated these samples based on the colored ground truth, by creating a mask, assigning (255, 255, 255) for the pixels which were not background (0, 0, 0). One can see some noise in the results, but the samples were enough for further experiments. In addition, the original images were not perfectly annotated. Therefore, some salt noise can be seen in non-skin regions. Source: Casati et al. (2013).*

5.1.3 Pratheepan

The images in the Pratheepan dataset were downloaded randomly from Google for human skin detection search. There are 78 images of family and face captured with a range of distinct cameras using different color enhancement and under different illumination conditions Tan *et al.* (2012). Figure 5.4 shows some of the 78 samples available.

Figure 5.4: *Examples of Pratheepan skin dataset. At the top is the original image and at the bottom the ground truth with the skin color pixels annotated. Here, the ground truth are binary images, where the black color RGB = (0, 0, 0) was assigned to all pixels in the background. Source: Tan et al. (2012).*

5.1.4 HGR

The database for Hand Gesture Recognition (HGR) contains the gestures from Polish and American Sign Language. There are 1,558 images acquired in different conditions of background, dimensions and lightening. In addition to original and ground truth binary skin mask images, it includes hand feature points location in separate files. Figure 5.5 shows some of the 1,558 samples available (Grzejszczak *et al.*, 2016; Kawulok *et al.*, 2014; Nalepa and Kawulok, 2014).

Figure 5.5: *Examples of HGR skin dataset. At the top is the original image and at the bottom the ground truth with the skin color pixels annotated. Differently from Pratheepan and SFA, the ground truth is also binary images, but the black color RGB = (0, 0, 0) was assigned to all pixels when they represent skin patches. Source: Grzejszczak et al. (2016); Kawulok et al. (2014); Nalepa and Kawulok (2014).*

The images within were acquired in three different series. A set of 899 was captured in uncontrolled background and lighting. A small set of 85 was obtained in gray (44) and (41) uncontrolled background; the lighting was uniform. The third group contains 574 images in controlled background (green tone), using uniform lighting conditions (Grzejszczak *et al.*, 2016; Kawulok *et al.*, 2014; Nalepa and Kawulok, 2014).

5.1.5 Compaq

Compaq can be considered as the first large skin dataset and, probably is the most used for skin detection classifiers. It consists of 13,635 images crawled from the Internet, which 4,670 contain skin regions and another subset of 8,965 images not containing any skin. The ground truth images are poorly annotated on the basis of an automatic software tool (Mahmoodi and Sayedi, 2016).

It is worth mentioning that this database is no longer available and we had obtained a copy of it by contacting the authors. We also had to fix some few images due lack of ground truth or files corrupted. The final amount of images with skin used in the experiments is 4,669. Figure 5.6 shows some of the 4,669 samples available used in the experiments (Jones and Rehg, 2002).

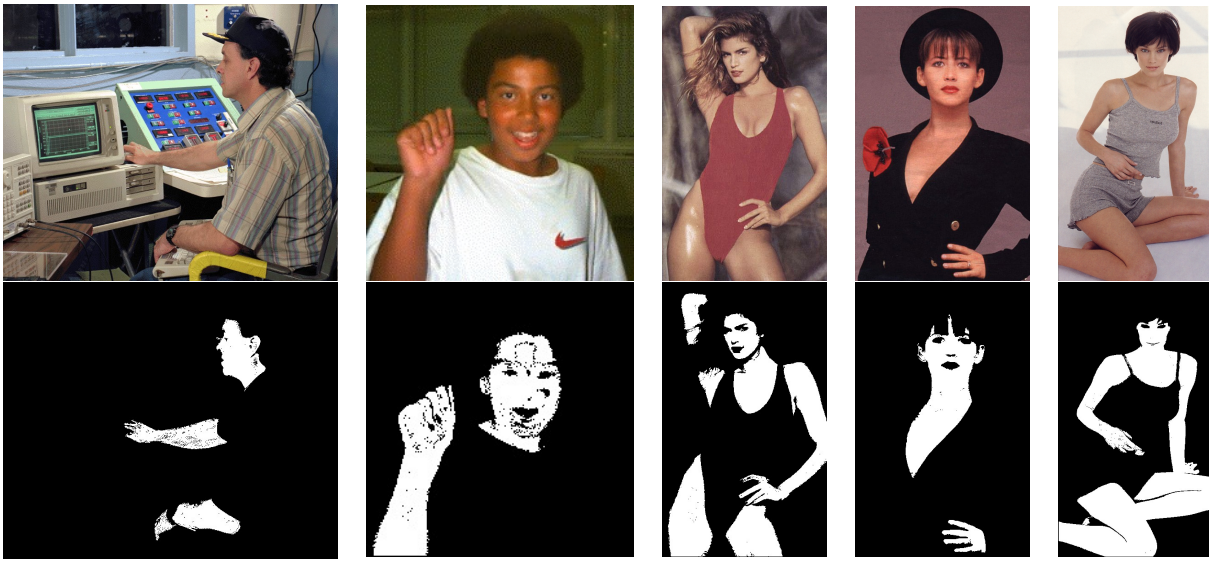

Figure 5.6: *Examples of Compaq skin/non-skin dataset. At the top is the original image and at the bottom the ground truth with the skin color pixels annotated. Here, the ground truth are binary images, where the black color RGB = (0, 0, 0) was assigned to all pixels in the background. Source: Jones and Rehg (2002).*

5.2 Evaluation measures

Precision, Recall, Specificity and *F-measure* have been used as evaluation metrics. They are the same used in Brancati *et al.* (2017) to compare the performance with state-of-the-art methods. They are also widely used by the scientific community. These metrics are given by the following formulas:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$F - measure = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

where TP, TN, FP, FN are, respectively, the number of true positive, true negative, false positive, and false negative pixels counted in the image, which are obtained from the confusion matrix (see table 5.2).

ground truth \ prediction outcome	skin	non-skin
skin	True Positive	False Negative
non-skin	False Positive	True Negative

Table 5.2: *Confusion matrix table used to count the number of true positive, true negative, false positive, and false negative pixels in the image during experiments. These numbers are fundamental input for evaluation measures.*

5.3 Machine learning experiments

Preliminary experiments were carried out using machine learning techniques in order to evaluate the UCI dataset, described in section 5.1.1, once we do not have the original images used to create it. The idea is to establish a bottom line to understand if this dataset can be useful for further experiments.

The first experiment was carried out with k-NN and SVM, available in the scikit-learn package (Pedregosa *et al.*, 2011). The color space originally used was RGB, as cited in the description of datasets in 5.1. In both cases, the chosen cross-validation strategy was 10-fold, which is a common choice of this approach in practice (Abu-Mostafa *et al.*, 2012). In addition, the technique of grid search of scikit-learn was also used with the objective of finding the most suitable parameters for each classifier.

The grid search is used in scikit-learn to find the optimal parameters of a classifier when they can not be learned by the estimator, such as the *kernel* and *gamma* in the SVM or number of neighbors of k-NN (Pedregosa *et al.*, 2011). The parameter tables used in the training of SVM and k-NN can be seen in 5.3 and 5.4. The parameters of each line are combined in an attempt to find the optimal estimator. For example, a choice in SVM training would be *kernel* = rbf, C = 100, *gamma* = 1e-4. All possible combinations are exploited, with the best of them being returned (Pedregosa *et al.*, 2011).

kernel	***C***			***gamma***			***degree***	
rbf	1	10	100	1e-3	1e-4	1e-5		
poly	1	10	100		1e-4	1e-5	3	4
linear	1	10	100					

Table 5.3: *Grid search table of the parameters of the optimal estimator in the SVM. The kernel column refers to the kernels used in training that are Gaussian, polynomial and linear, respectively. C is a regularization parameter that tells SVM the amount of error allowed during the training. Gamma is a parameter used only in Gaussian and polynomial kernels. Degree is the degree of the polynomial; used only in the polynomial kernel (Pedregosa et al., 2011).*

The results of this experiment can be seen in the table 5.5. The dataset used was UCI. It is noteworthy that the training was performed splitting the data with 30% randomly separated for test in both classifiers.

As can be seen in the table 5.5, both classifiers had very high quality measures in the UCI dataset, close to 100%, which is probably an over-fitting situation. One possible root cause is the splitting of training and test data subsets. We observed that there are many replicates among the

samples, so it is possible that samples already seen by the classifier during the training are used in the test step.

In fact, once we used a seed to generate the subsets, we applied the same strategy to split the data and count how many samples of the test subset were seen in the training subset as shown in figure 5.7. Based on the distribution of the splitting and the results of the measures given in table 5.5, we can say that UCI dataset is not suitable for this application.

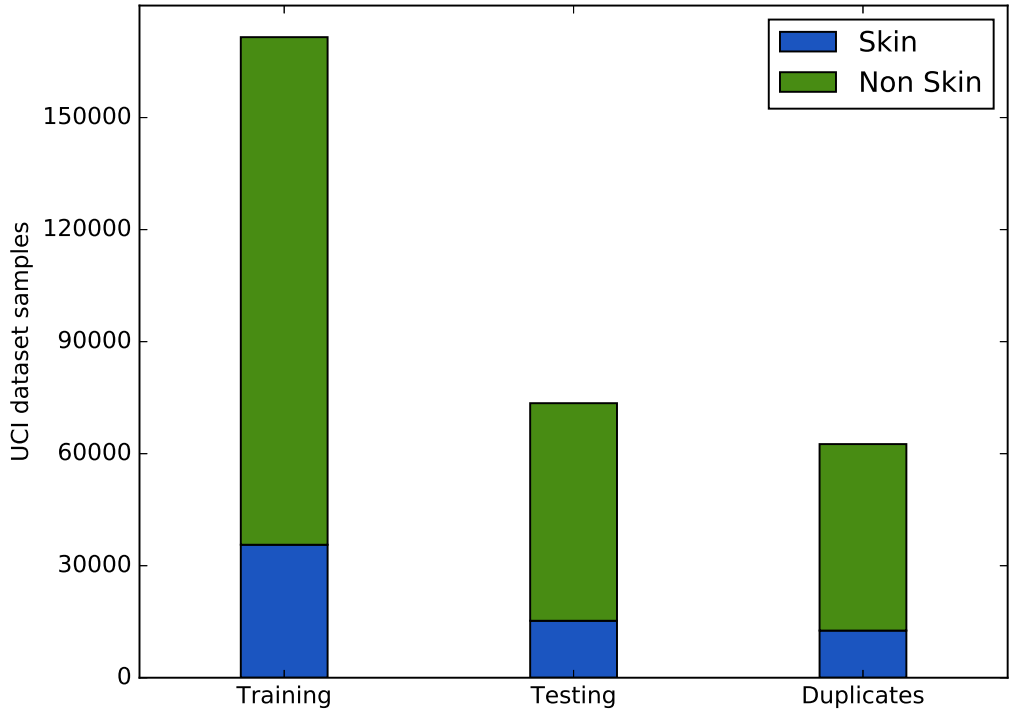

Figure 5.7: *UCI samples distribution after splitting the dataset in training and testing subsets. We used the scikit-learn package function for splitting, where 30% is the size of test set. The right-most bar, entitled as duplicates, shows how many samples of the test set was seen in training set. In the order of 83% for skin samples and 86% for non skin samples. Source: proposed by the author.*

For the benefit of the doubt, we submitted the learned function given by those classifiers in real images of Pratheepan and SFA datasets. This experiment can help us to understand the ability of

n_ neighbors							***weights***		***algorithm***
3	5	9	15	25	31	35	distance	uniform	auto

Table 5.4: *Grid search table of the parameters of the optimal estimator in k-NN. The n_ neighbors column refers to the number of neighbors considered in the training. Weights is the weight function used in the prediction, where uniform indicates that the points have equal weights and distance indicates that the inverse of the distance is applied in the classification. The third column indicates which algorithm should be used; auto means that the algorithm will be decided based on the data (Pedregosa et al., 2011).*

Classifier	**Color model**	***Precision***	***Recall***	***F-measure***
k-NN	RGB	0,9995	0,9995	0,9995
SVM	RGB	0,9995	0,9995	0,9995

Table 5.5: *Results of the experiments with k-NN and SVM in the UCI dataset. The optimal k-NN parameters found during the training in UCI were* $n\_neighbors = 3$, $weights = uniform$. *In the case of SVM, the optimal parameters found in UCI training were* $kernel = rbf$, $C = 100$ *and* $gamma = 1e-3$.

the learned model to generalize for upcoming non seen samples.

Dataset	Classifier	Precision	*Recall*	*Specificity*	*F-measure*
SFA	k-NN	0.9322	0.5895	0.9835	0.7223
	SVM	0.9079	0.4839	0.9838	0.6314
Pratheepan	k-NN	0.5930	0.7719	0.8568	0.6707
	SVM	0.6179	0.7259	0.8841	0.6675

Table 5.6: *Results of the experiments with k-NN and SVM in the SFA and Pratheepan images datasets. Despite some of the measures are quite good, we can see that they are far way from the ones given during the training in table 5.5, which says that the learned models do not have a good generalization.*

5.4 Rule-based experiments

In this section we present some experimental evaluations of the proposed extensions described in sections 4.2 and 4.3, as well as the original method in four widely known datasets: SFA, Pratheepan, HGR and Compaq. The latest three of them have also been used in Brancati *et al.* (2017).

Table 5.7 shows quantitative result metrics of the experiments. Column 1 refers to the dataset used. Column 2 refers to the method being experimented: Original for the original hypothesis; Reverse refers to the reverse hypothesis with respect to P_{Cr_s} parameter; Combined refers to the combination of both of the former methods (see Sec. 4.2); Neighbors refers to the extension of the method using the neighborhood approach (see Sec. 4.3).

Dataset	Hypothesis	Precision	Recall	Specificity	F-measure
Compaq	Original	0.4354	**0.8046**	0.8046	0.5650
	Reverse	0.3971	0.7232	0.7921	0.5127
	Combined	**0.4906**	0.6251	**0.8856**	0.5498
	Neighbors	0.4708	0.7421	0.8463	**0.5761**
Pratheepan	Original	0.5513	**0.8199**	0.8230	0.6592
	Reverse	0.5249	0.7326	0.8188	0.6116
	Combined	**0.6681**	0.6683	**0.9164**	0.6682
	Neighbors	0.6280	0.7515	0.8871	**0.6843**
HGR	Original	0.8938	0.7664	0.9274	0.8252
	Reverse	0.7929	**0.8429**	0.8337	0.8171
	Combined	**0.8994**	0.6952	**0.9390**	0.7843
	Neighbors	0.8818	0.7935	0.9211	**0.8353**
SFA	Original	0.8636	0.4214	0.9692	0.5664
	Reverse	0.8563	**0.7730**	0.9381	**0.8125**
	Combined	**0.9288**	0.3958	**0.9894**	0.5551
	Neighbors	0.9176	0.5111	0.9826	0.6565

Table 5.7: *Quantitative result metrics of the proposed enhancements and Brancati et al. (2017). For each dataset, we have four different applications: the original hypothesis with respect to P_{Cb_s}, the reverse hypothesis with respect to P_{Cr_s}, the one which combines both, and the extension using the neighborhood approach.*

Figures 5.8, 5.9, 5.10 and 5.11, present some qualitative results with image samples in column (a) along with the results for each method tested. Column (b) presents the respective ground truth for each image in column (a), column (c) presents the original method Brancati *et al.* (2017) results, column (d) presents the respective reverse method results, column (e), the combined method results and column (f) the extended neighborhood method.

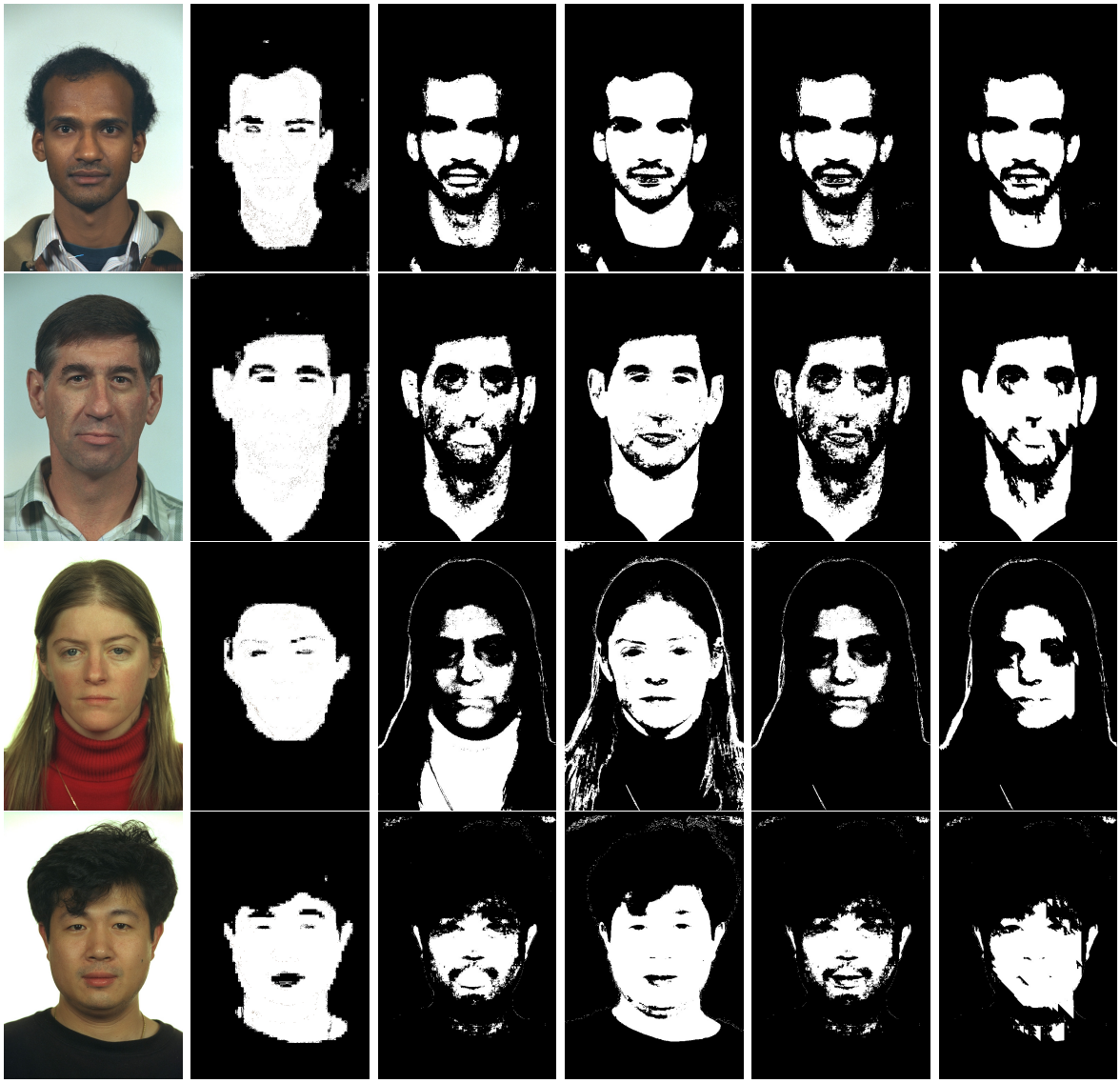

Figure 5.8: *Image samples with the results of each method in SFA dataset: (a) original image (b) ground truth (c) original method Brancati et al. (2017) (d) reverse method (e) combined method (f) neighbors method.*

Figure 5.9: *Image samples with the results of each method in Pratheepan dataset: (a) original image (b) ground truth (c) original method Brancati et al. (2017) (d) reverse method (e) combined method (f) neighbors method.*

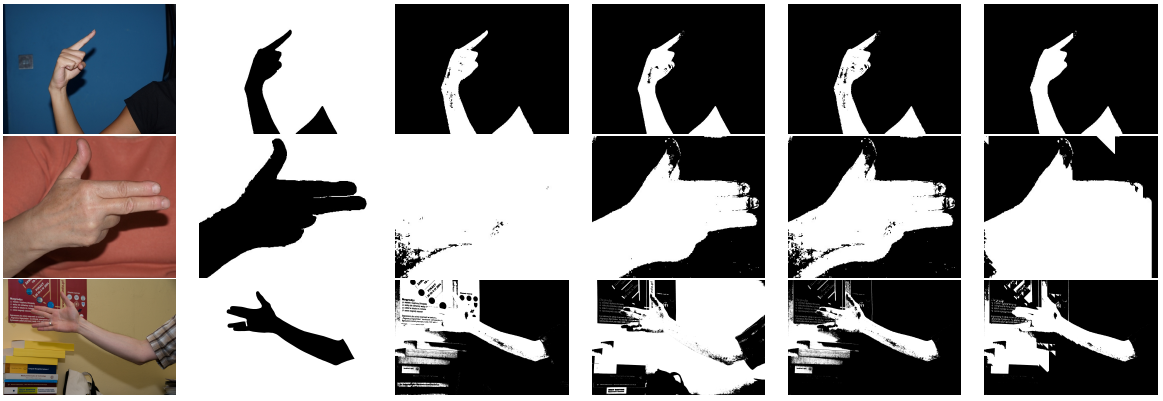

Figure 5.10: *Image samples with the results of each method in HGR dataset: (a) original image (b) ground truth (c) original method Brancati et al. (2017) (d) reverse method (e) combined method (f) neighbors method.*

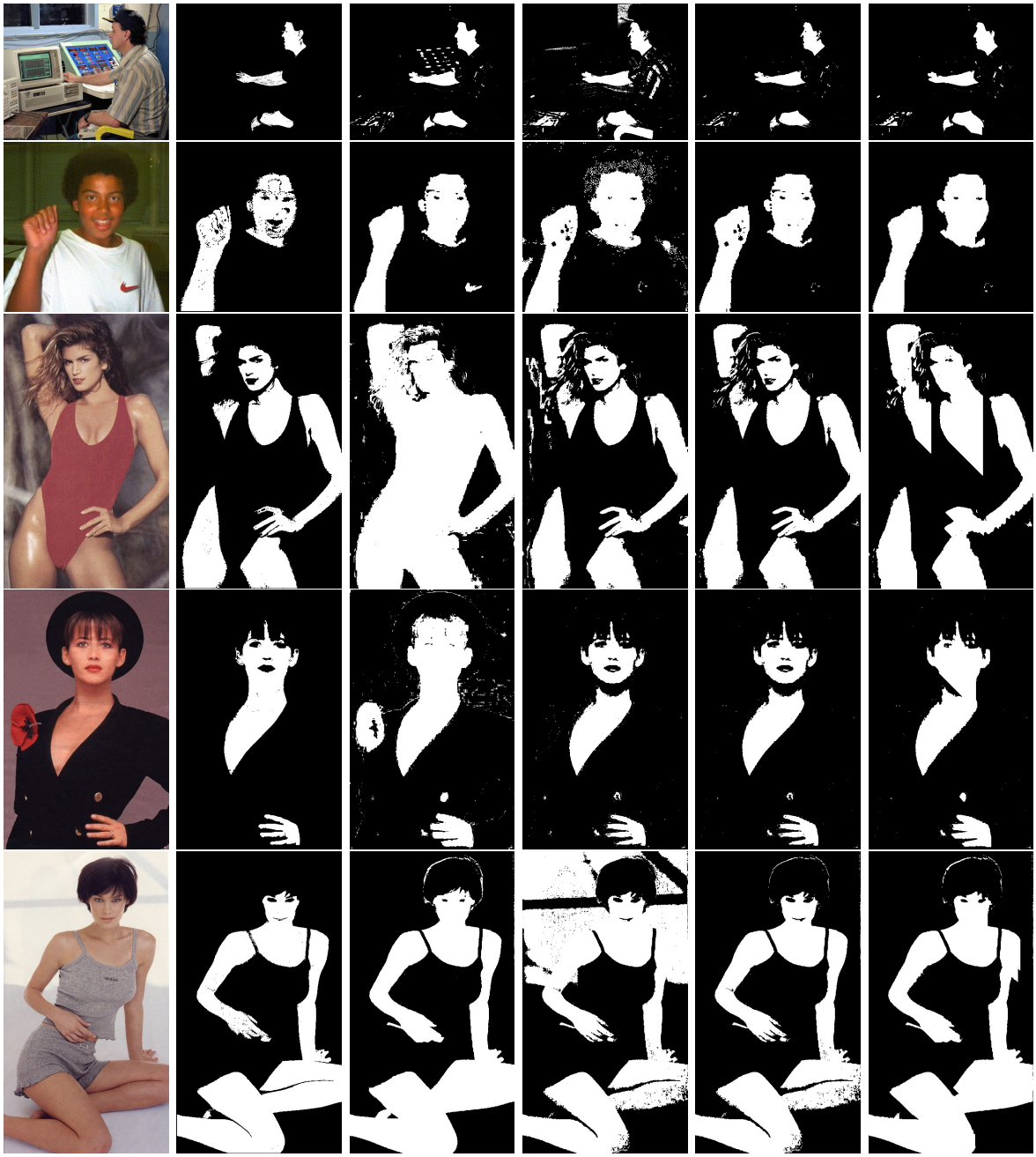

Figure 5.11: *Image samples with the results of each method in Compaq dataset: (a) original image (b) ground truth (c) original method Brancati et al. (2017) (d) reverse method (e) combined method (f) neighbors method.*

5.5 Supplementary neighborhood operations experiments

In this section we will show some experimental results of the supplementary neighborhood adaptation described in section 4.4. In short, we basically scan the image, with a size of $W \times H$, in the raster order, and apply the original and reverse rules for every single pixel. We keep the result in a matrix of the same size ($W \times H$) of the input image. For each coordinate of this output matrix, we will have a two positions vector with the result of the original and reverse rules answer for this pixel. Finally, we count those answers in four different strategies. The results can be seen in table 5.8.

Dataset	Hypothesis (Neighbors)	Precision	Recall	Specificity	F-measure
Compaq	AND	**0.5121**	0.6252	**0.8941**	0.5630
	OR	0.3786	**0.9037**	0.7203	0.5336
	P_{Cr_s} only	0.4132	0.7254	0.8032	0.5265
	P_{Cb_s} only	0.4478	0.8053	0.8120	**0.5755**
Pratheepan	AND	**0.6731**	0.6789	**0.9127**	**0.6760**
	OR	0.4624	**0.8837**	0.7321	0.6072
	P_{Cr_s} only	0.5285	0.7414	0.8163	0.6171
	P_{Cb_s} only	0.5630	0.8218	0.8292	0.6682
HGR	AND	**0.9007**	0.7203	**0.9378**	0.8005
	OR	0.7978	**0.9084**	0.8238	**0.8495**
	P_{Cr_s} only	0.7937	0.8600	0.8331	0.8256
	P_{Cb_s} only	0.8818	0.7935	0.9211	0.8353
SFA	AND	**0.9345**	0.3947	**0.9899**	0.5549
	OR	0.8345	**0.8181**	0.9176	0.8262
	P_{Cr_s} only	0.8612	0.7922	0.9375	0.8252
	P_{Cb_s} only	0.8953	0.7690	0.9286	**0.8273**

Table 5.8: *Quantitative result metrics of the proposed supplementary neighborhood adaptation. For each dataset, we have four different applications of the neighbors operations, respectively: applying an AND between the original and reverse rules, applying an OR between the original and reverse rules, considering the P_{Cr_s} (reverse) only, and considering the P_{Cb_s} (original) only.*

It is worth mentioning that the goal here is to explore better the connectivity of the 8-*neighbors* window and check, on the basis of a symmetric mask window, if the *diagonal effect* is gone as well as the measures are improved. Therefore, we are not worried about the additional computational cost taken to scan the image one more time. Even though, this implementation can be enhanced by, for instance, keeping the latest three lines of the image scanned in memory to be verified by the 8-*neighbors* window backward to have a final decision when evaluating a pixel.

From the table 5.8 we can see that the application of the neighborhood approach using AND gives us the best precision and specificity in all datasets. On the other hand, we lost performance in the recall in relation to the others approaches. Speaking of recall, the loss is in the order of 20% up to 30% in the Compaq, Pratheepan and HGR datasets, and approximately 50% in the SFA.

Clearly there is a trade-off between increasing precision and decreasing recall and vice versa, just as in the experiments done in section 5.4 with all methods. When we use a more relaxed rule, as in the case of the OR or the isolated rules, we get a better recall, but there is also an abrupt drop in precision and specificity. Another side effect of this phenomenon is to increase the false positive rate.

The AND conjunction approach is most similar to that implemented in neighborhood extended method (see Sec. 4.3). The results obtained by that approach show low variability in relation to the adaptations shown in the experiments in this section. Therefore, for the sake of implementation, the approach given in the neighborhood implementation may be a good starting point. Nevertheless,

we can apply some other complementary technique, such as weights in the pixels, to measure those more or less relevant to the decision making in the neighborhood. This can potentially diminish the effect of this trade-off and obtain more harmonic measures while removing the undesired *diagonal effect*.

5.6 Grid search parameters experiments

5.7 Results and discussion

The original method was compared with six well known rule-based methods in literature using four different datasets, three of them, HGR, Pratheepan and Compaq, have also been used here.

Because the method had the best *F-measure* in the HGR and Pratheepan datasets in comparison with the other six methods and, in addition, because it performed the top first *Precision* in HGR and second in Pratheepan, we decided to compare the proposed extensions only to the original method.

Table 5.7 shows quantitative result metrics of the experiments. Column 1 refers to the dataset used. Column 2 refers to the method being experimented: Original for the original hypothesis; Reversed refers to the reverse hypothesis with respect to P_{Cr_s} parameter; Combined refers to the combination of both of the former methods (see Sec. 4.2); Neighbors refers to the extension of the method using the neighborhood approach.

As one can see, the reverse hypothesis performed better than the original method and achieved the best *Recall* in HGR and SFA. It also achieved the best *F-measure* in SFA with a 0.8125 rate, which gave almost 0.25 in gain compared to the original.

In general, the reverse method increased the *Recall* but did not perform well in *Precision* and *Specificity* measures. When we combined both methods, the best *Precision* and *Specificity* were achieved for all datasets but it loses some performance in *Recall*. However, it has very high *F-measure* rates.

The combined method along with the neighborhood approach achieved the best *F-measure* in HGR and Pratheepan. Moreover, the other metrics still are in a very high rate for all datasets, being in the top second in almost cases.

Therefore, the combined and extended approaches are very competitive compared to the original method. Furthermore, all the variations of the original method are still computed in quadratic time, maintaining the desired computational efficiency that are useful in different application domains, mainly near real time systems (processing time of about 10ms for a typical image of dimensions 300x400).

Chapter 6

Plano de Trabalho

Os créditos em disciplinas necessários para o programa de mestrado em Ciência da Computação no IME-USP foram cumpridos de fevereiro de 2015 até junho de 2016, conforme a tabela 6.1. Em meados de 2015, iniciou-se a pesquisa bibliográfica para o desenvolvimento deste projeto. Durante esse período, até o presente momento, foram realizados os experimentos preliminares que estão incorporados neste texto para a etapa de qualificação. Após apresentação, as recomendações da comissão julgadora serão ponderadas e devem ser refletidas nas atividades subsequentes listadas na seção 6.1.

Código	**Disciplina**	**Término**
MAC5832	Aprendizagem Computacional: Modelos, Algoritmos e Aplicações	jun/2015
MAC5744	Introdução à Computação Gráfica	jun/2015
MAC5768	Visão e Processamento de Imagens - Parte I	jun/2015
MAC5711	Análise de Algoritmos	nov/2015
MAC6914	Métodos de Aprendizagem em Visão Computacional	nov/2015
MAC4722	Linguagens, Autômatos e Computabilidade	jun/2016
MAC5714	Programação Orientada a Objetos	jun/2016

Table 6.1: *Disciplinas cursadas no programa de mestrado em Ciência da Computação no IME-USP.*

6.1 Atividades previstas

1. **Revisar leituras adicionais:** outras referências deverão ser adicionadas ao presente trabalho, principalmente, conteúdo relacionado a métodos de *kernel*, que servirão de embasamento para compreender e estudar a técnica de *kernels* em conjuntos *fuzzy* proposta por Guevara *et al.* (2014).

2. **Incorporar novos conjuntos de dados:** essa atividade consiste em analisar e integrar novos conjuntos de dados ao projeto para utilização nos experimentos, tanto de bancos de imagens de cor de pele, como outros conjuntos de imagens coloridas onde a classificação de dados intervalares possa ser aplicada.

3. **Investigar características passíveis de uso em classificadores de dados intervalares:** os classificadores treinados durante a etapa de experimentos preliminares consistiam apenas da informação de cor da imagem, independente do espaço de cores escolhido. Portanto, essa atividade visa compreender o papel de outros atributos, tais como textura e características locais, durante o treinamento e se a inclusão de tais atributos, de fato, pode acarretar em melhoria de performance.

4. **Desenvolver novas ferramentas:** novas ferramentas devem ser desenvolvidas para dar suporte aos experimentos subsequentes, tais como a tabela de busca implementada para otimiza-

ção de parâmetros do *FuzzyDT*. Todo software criado por este projeto de pesquisa será disponibilizado para toda a comunidade científica em formato de código livre.

5. **Elaborar novos experimentos:** com base nas ferramentas desenvolvidas e conjuntos de dados estabelecidos, elaborar e executar novos experimentos.

6. **Analisar os resultados:** os resultados obtidos deverão ser avaliados e reportados no projeto de pesquisa e em publicações a serem realizadas. Eventualmente, correções nas ferramentas desenvolvidas e/ou nos experimentos também serão agregadas nesta etapa.

7. **Publicar resultados:** artigos científicos com os resultados obtidos devem ser publicados em revistas ou conferências sobre processamento de imagens, classificação ou outras áreas relacionadas com o contexto do projeto de pesquisa.

8. **Escrever a dissertação:** a dissertação deverá ser escrita assim que as demais atividades tiverem sido concluídas, principalmente em relação aos experimentos e análise de resultados, o que fornecerá dados consolidados para a conclusão do projeto de pesquisa.

6.2 Cronograma proposto

As atividades listadas na seção 6.1 serão realizadas de acordo com o cronograma disposto na tabela 6.2. As tarefas foram divididas dentro de um período de onze meses de execução e a previsão de defesa da dissertação é outubro de 2017.

Atividade	Meses 2016/2017										
	dez	jan	fev	mar	abr	mai	jun	jul	ago	set	out
1	X	X	X								
2	X	X									
3		X	X								
4			X	X	X	X					
5					X	X	X				
6						X	X	X			
7							X	X			
8								X	X	X	X

Table 6.2: *Cronograma das atividades previstas*

Appendix A

Trapezoids Parameters Tuning Results

In this appendix we present the results of the grid search algorithm applied on each of the four datasets described in section 5.1. We only applied the combined rules method, once the trapezoids parameters definition do not change among the different methods, as described in chapter 4. Every table is sorted by *F-measure*, *Precision* and *Recall*, respectively. The results are presented in Tables A.1, A.2 for Pratheepan, SFA, HGR and Compaq, respectively.

Table A.1: *Trapezoids parameters tuning results for Pratheepan dataset and combined rules.*

Y_min	Y_max	Precision	Recall	Specificity	F-measure
10	15	0.6531	0.7001	0.9066	0.6758
5	15	0.6639	0.6857	0.9133	0.6746
10	55	0.6660	0.6829	0.9137	0.6744
10	50	0.6657	0.6832	0.9134	0.6743
10	80	0.6659	0.6824	0.9138	0.6741
10	85	0.6659	0.6822	0.9138	0.6740
10	95	0.6659	0.6822	0.9138	0.6740
10	90	0.6659	0.6822	0.9138	0.6739
10	75	0.6663	0.6809	0.9139	0.6735
10	60	0.6665	0.6804	0.9138	0.6733
10	45	0.6639	0.6829	0.9131	0.6733
10	65	0.6665	0.6796	0.9139	0.6730
10	30	0.6609	0.6855	0.9129	0.6730
10	25	0.6600	0.6865	0.9128	0.6730
10	70	0.6664	0.6794	0.9139	0.6728
10	35	0.6615	0.6845	0.9129	0.6728
10	40	0.6631	0.6820	0.9132	0.6724
5	50	0.6701	0.6733	0.9165	0.6717
5	55	0.6698	0.6728	0.9166	0.6713
5	45	0.6686	0.6731	0.9163	0.6709
10	20	0.6517	0.6911	0.9053	0.6708
5	60	0.6704	0.6704	0.9168	0.6704
5	40	0.6679	0.6727	0.9164	0.6703
5	20	0.6631	0.6768	0.9151	0.6699
5	65	0.6705	0.6690	0.9168	0.6698
5	35	0.6660	0.6736	0.9161	0.6698
5	70	0.6704	0.6690	0.9168	0.6697
5	30	0.6654	0.6739	0.9161	0.6696
5	95	0.6701	0.6686	0.9169	0.6693
5	90	0.6700	0.6686	0.9169	0.6693
5	80	0.6700	0.6685	0.9169	0.6693
5	25	0.6643	0.6744	0.9159	0.6693
5	85	0.6700	0.6684	0.9169	0.6692
15	50	0.6598	0.6774	0.9114	0.6685
15	55	0.6596	0.6775	0.9114	0.6685

Continued on next page

Table A.1 – continued from previous page

Y_min	Y_max	Precision	Recall	Specificity	F-measure
15	65	0.6605	0.6754	0.9117	0.6679
15	45	0.6572	0.6790	0.9107	0.6679
20	45	0.6487	0.6882	0.8992	0.6679
5	75	0.6701	0.6656	0.9169	0.6678
15	80	0.6591	0.6764	0.9116	0.6677
15	60	0.6604	0.6748	0.9117	0.6676
15	75	0.6596	0.6758	0.9117	0.6676
15	90	0.6591	0.6764	0.9116	0.6676
15	95	0.6591	0.6763	0.9116	0.6676
15	85	0.6591	0.6762	0.9116	0.6675
15	70	0.6597	0.6747	0.9117	0.6671
15	35	0.6546	0.6800	0.9090	0.6671
15	30	0.6542	0.6805	0.9091	0.6671
20	30	0.6432	0.6927	0.9008	0.6670
15	40	0.6567	0.6774	0.9109	0.6669
20	40	0.6479	0.6868	0.8991	0.6668
20	55	0.6506	0.6832	0.9002	0.6665
20	35	0.6466	0.6876	0.9017	0.6665
20	25	0.6352	0.7010	0.8913	0.6665
20	50	0.6507	0.6830	0.9001	0.6664
20	65	0.6514	0.6804	0.9005	0.6656
20	60	0.6512	0.6799	0.9005	0.6652
20	75	0.6506	0.6800	0.9006	0.6650
20	80	0.6503	0.6802	0.9006	0.6649
20	70	0.6507	0.6796	0.9006	0.6648
20	90	0.6503	0.6800	0.9006	0.6648
20	95	0.6503	0.6800	0.9006	0.6648
20	85	0.6502	0.6799	0.9006	0.6647
15	25	0.6486	0.6814	0.9043	0.6646
35	40	0.6285	0.7051	0.8886	0.6646
35	45	0.6292	0.7015	0.8830	0.6634
5	5	0.5603	0.8123	0.8339	0.6632
10	10	0.5603	0.8123	0.8339	0.6632
15	15	0.5603	0.8123	0.8339	0.6632
20	20	0.5603	0.8123	0.8339	0.6632
25	25	0.5603	0.8123	0.8339	0.6632
30	30	0.5603	0.8123	0.8339	0.6632
35	35	0.5603	0.8123	0.8339	0.6632
40	40	0.5603	0.8123	0.8339	0.6632
45	45	0.5603	0.8123	0.8339	0.6632
50	50	0.5603	0.8123	0.8339	0.6632
55	55	0.5603	0.8123	0.8339	0.6632
60	60	0.5603	0.8123	0.8339	0.6632
65	65	0.5603	0.8123	0.8339	0.6632
70	70	0.5603	0.8123	0.8339	0.6632
75	75	0.5603	0.8123	0.8339	0.6632
80	80	0.5603	0.8123	0.8339	0.6632
85	85	0.5603	0.8123	0.8339	0.6632
90	90	0.5603	0.8123	0.8339	0.6632
95	95	0.5603	0.8123	0.8339	0.6632
30	50	0.6390	0.6868	0.8917	0.6620
35	50	0.6330	0.6937	0.8855	0.6620
30	45	0.6354	0.6908	0.8902	0.6619
35	55	0.6324	0.6932	0.8858	0.6614
30	55	0.6380	0.6863	0.8916	0.6613
25	45	0.6351	0.6896	0.8903	0.6612
25	50	0.6380	0.6844	0.8917	0.6604
25	55	0.6381	0.6839	0.8923	0.6602
30	65	0.6384	0.6821	0.8921	0.6595

Continued on next page

Table A.1 – continued from previous page

Y_min	Y_max	Precision	Recall	Specificity	F-measure
30	60	0.6382	0.6823	0.8921	0.6595
25	65	0.6388	0.6808	0.8926	0.6591
25	60	0.6387	0.6805	0.8926	0.6589
35	65	0.6326	0.6874	0.8864	0.6589
35	60	0.6323	0.6879	0.8863	0.6589
30	40	0.6314	0.6885	0.8892	0.6587
25	75	0.6382	0.6802	0.8927	0.6585
25	70	0.6382	0.6801	0.8927	0.6585
25	95	0.6379	0.6803	0.8927	0.6585
25	80	0.6379	0.6803	0.8927	0.6584
25	90	0.6379	0.6803	0.8927	0.6584
25	85	0.6379	0.6802	0.8927	0.6584
25	40	0.6316	0.6874	0.8892	0.6583
40	45	0.6155	0.7074	0.8717	0.6583
65	90	0.6167	0.7057	0.8771	0.6582
65	95	0.6165	0.7057	0.8771	0.6581
35	90	0.6334	0.6842	0.8892	0.6578
35	95	0.6334	0.6842	0.8891	0.6578
35	85	0.6334	0.6840	0.8892	0.6578
30	70	0.6367	0.6798	0.8921	0.6576
30	75	0.6367	0.6796	0.8921	0.6575
65	85	0.6147	0.7065	0.8772	0.6574
35	70	0.6324	0.6843	0.8886	0.6573
30	80	0.6364	0.6794	0.8921	0.6572
30	95	0.6364	0.6793	0.8921	0.6572
30	90	0.6363	0.6793	0.8921	0.6571
30	85	0.6363	0.6792	0.8921	0.6571
35	75	0.6323	0.6836	0.8887	0.6570
35	80	0.6316	0.6833	0.8885	0.6564
45	65	0.6234	0.6915	0.8812	0.6557
25	35	0.6261	0.6864	0.8907	0.6549
40	50	0.6203	0.6933	0.8798	0.6547
50	65	0.6201	0.6925	0.8793	0.6543
60	90	0.6185	0.6944	0.8791	0.6543
40	55	0.6212	0.6909	0.8806	0.6542
5	10	0.6325	0.6772	0.8971	0.6541
70	95	0.6097	0.7056	0.8744	0.6541
60	95	0.6183	0.6942	0.8791	0.6540
40	65	0.6249	0.6857	0.8829	0.6539
65	80	0.6108	0.7035	0.8754	0.6539
70	90	0.6096	0.7053	0.8741	0.6539
45	95	0.6243	0.6859	0.8845	0.6536
45	90	0.6242	0.6858	0.8844	0.6535
50	95	0.6216	0.6889	0.8826	0.6535
50	90	0.6216	0.6888	0.8826	0.6535
60	85	0.6167	0.6949	0.8791	0.6535
70	85	0.6078	0.7064	0.8742	0.6534
45	85	0.6239	0.6856	0.8844	0.6533
40	90	0.6253	0.6836	0.8853	0.6532
40	95	0.6253	0.6836	0.8853	0.6532
50	85	0.6213	0.6886	0.8826	0.6532
40	85	0.6253	0.6833	0.8854	0.6530
40	60	0.6235	0.6854	0.8827	0.6530
45	70	0.6232	0.6857	0.8841	0.6529
40	70	0.6250	0.6833	0.8851	0.6528
40	75	0.6247	0.6826	0.8850	0.6524
45	75	0.6228	0.6845	0.8841	0.6522
50	70	0.6197	0.6883	0.8817	0.6522
45	80	0.6225	0.6847	0.8840	0.6521

Continued on next page

Table A.1 – continued from previous page

Y_min	Y_max	Precision	Recall	Specificity	F-measure
55	90	0.6185	0.6896	0.8810	0.6521
55	95	0.6184	0.6897	0.8809	0.6521
60	70	0.6135	0.6960	0.8771	0.6521
50	80	0.6198	0.6876	0.8821	0.6520
50	75	0.6199	0.6873	0.8820	0.6518
40	80	0.6236	0.6821	0.8848	0.6516
60	80	0.6139	0.6937	0.8779	0.6514
55	85	0.6167	0.6898	0.8810	0.6512
60	75	0.6142	0.6912	0.8775	0.6504
15	20	0.6239	0.6787	0.8888	0.6502
65	75	0.6071	0.6980	0.8749	0.6494
55	75	0.6149	0.6873	0.8804	0.6491
55	80	0.6142	0.6883	0.8803	0.6491
55	70	0.6128	0.6890	0.8795	0.6487
45	60	0.6147	0.6818	0.8813	0.6466
45	55	0.6109	0.6866	0.8787	0.6465
30	35	0.6107	0.6845	0.8833	0.6455
50	55	0.6047	0.6913	0.8755	0.6451
50	60	0.6103	0.6836	0.8787	0.6449
80	95	0.5963	0.7015	0.8735	0.6446
80	85	0.5911	0.7082	0.8652	0.6443
75	85	0.5944	0.7028	0.8739	0.6441
80	90	0.5956	0.7008	0.8729	0.6439
70	80	0.5972	0.6977	0.8744	0.6436
75	95	0.5954	0.6994	0.8734	0.6433
75	90	0.5954	0.6982	0.8733	0.6427
25	30	0.6070	0.6769	0.8884	0.6401
70	75	0.5876	0.6867	0.8810	0.6333
60	65	0.5901	0.6819	0.8761	0.6327
65	70	0.5834	0.6873	0.8742	0.6311
45	50	0.5889	0.6787	0.8739	0.6306
85	95	0.5807	0.6883	0.8829	0.6300
55	65	0.5915	0.6719	0.8777	0.6291
85	90	0.5743	0.6847	0.8797	0.6247
90	95	0.5640	0.6905	0.8703	0.6209
75	80	0.5646	0.6887	0.8621	0.6205
55	60	0.5786	0.6688	0.8751	0.6204

Table A.2: *Trapezoids parameters tuning results for SFA dataset and combined rules.*

Y_min	Y_max	Precision	Recall	Specificity	F-measure
5	5	0.9319	0.6567	0.9789	0.7705
10	10	0.9319	0.6567	0.9789	0.7705
15	15	0.9319	0.6567	0.9789	0.7705
20	20	0.9319	0.6567	0.9789	0.7705
25	25	0.9319	0.6567	0.9789	0.7705
30	30	0.9319	0.6567	0.9789	0.7705
35	35	0.9319	0.6567	0.9789	0.7705
40	40	0.9319	0.6567	0.9789	0.7705
45	45	0.9319	0.6567	0.9789	0.7705
50	50	0.9319	0.6567	0.9789	0.7705
55	55	0.9319	0.6567	0.9789	0.7705
60	60	0.9319	0.6567	0.9789	0.7705
65	65	0.9319	0.6567	0.9789	0.7705
70	70	0.9319	0.6567	0.9789	0.7705
75	75	0.9319	0.6567	0.9789	0.7705

Continued on next page

Table A.2 – continued from previous page

Y_min	Y_max	Precision	Recall	Specificity	F-measure
80	80	0.9319	0.6567	0.9789	0.7705
85	85	0.9319	0.6567	0.9789	0.7705
90	90	0.9319	0.6567	0.9789	0.7705
95	95	0.9319	0.6567	0.9789	0.7705
85	90	0.8302	0.4357	0.9532	0.5714
70	75	0.8558	0.4256	0.9664	0.5685
80	90	0.8703	0.4198	0.9747	0.5664
85	95	0.8593	0.4223	0.9699	0.5663
60	65	0.8617	0.4216	0.9696	0.5662
50	55	0.8708	0.4192	0.973	0.566
80	95	0.8723	0.4186	0.9756	0.5657
75	80	0.8363	0.4271	0.9612	0.5654
65	70	0.8497	0.4233	0.9677	0.5651
75	90	0.8779	0.4166	0.9793	0.565
75	95	0.8786	0.4162	0.9794	0.5648
70	90	0.8864	0.4143	0.9815	0.5647
70	95	0.8868	0.4139	0.9815	0.5644
70	85	0.8843	0.4144	0.9811	0.5644
70	80	0.8793	0.4156	0.9798	0.5644
60	70	0.8853	0.4141	0.9822	0.5643
55	60	0.8643	0.4189	0.9739	0.5643
65	80	0.8871	0.4134	0.9819	0.564
65	75	0.8826	0.4143	0.9809	0.5639
60	75	0.891	0.4123	0.9836	0.5638
60	80	0.8935	0.4116	0.9839	0.5636
75	85	0.8721	0.4162	0.9783	0.5635
55	65	0.8865	0.4127	0.9824	0.5633
45	55	0.895	0.4108	0.9838	0.5631
65	90	0.8909	0.4115	0.9831	0.563
55	70	0.8906	0.4116	0.9831	0.563
50	60	0.8882	0.412	0.9819	0.5629
65	85	0.89	0.4115	0.9829	0.5628
55	80	0.8968	0.41	0.9843	0.5627
55	75	0.8948	0.4104	0.9841	0.5627
65	95	0.891	0.4112	0.9832	0.5627
45	60	0.8988	0.4094	0.9847	0.5626
60	90	0.895	0.4102	0.9841	0.5625
60	95	0.8949	0.41	0.9841	0.5624
60	85	0.8946	0.4102	0.984	0.5624
55	90	0.898	0.4092	0.9846	0.5622
35	40	0.8894	0.411	0.981	0.5622
40	45	0.8878	0.4112	0.9814	0.5621
45	65	0.8996	0.4087	0.9849	0.562
55	95	0.8979	0.409	0.9846	0.562
55	85	0.8976	0.4091	0.9845	0.562
45	50	0.8796	0.4129	0.9797	0.562
45	75	0.9035	0.4077	0.9857	0.5619
45	70	0.9008	0.4083	0.9851	0.5619
50	80	0.8993	0.4086	0.985	0.5619
30	35	0.8984	0.4086	0.9833	0.5618
50	75	0.8982	0.4087	0.9849	0.5618
45	80	0.9039	0.4075	0.9857	0.5617
50	70	0.8948	0.4094	0.984	0.5617
50	65	0.8911	0.4099	0.9835	0.5615
35	45	0.9007	0.4078	0.9849	0.5614
50	90	0.9003	0.4078	0.9852	0.5614
40	55	0.8995	0.4081	0.985	0.5614
40	50	0.8978	0.4083	0.9848	0.5614
45	90	0.9045	0.4068	0.9859	0.5612

Continued on next page

Table A.2 – continued from previous page

Y_min	Y_max	Precision	Recall	Specificity	F-measure
50	95	0.9002	0.4077	0.9852	0.5612
50	85	0.8999	0.4078	0.9851	0.5612
45	85	0.9044	0.4067	0.9859	0.5611
45	95	0.9045	0.4066	0.9859	0.561
35	50	0.9038	0.4068	0.9857	0.561
40	60	0.9013	0.4071	0.9854	0.5609
30	40	0.9056	0.4059	0.9854	0.5606
35	55	0.905	0.4061	0.9859	0.5606
40	70	0.9038	0.4063	0.9858	0.5606
40	65	0.9025	0.4065	0.9856	0.5606
40	75	0.9051	0.4059	0.9861	0.5605
35	60	0.9065	0.4054	0.9861	0.5603
40	80	0.9053	0.4057	0.9862	0.5603
35	65	0.9072	0.405	0.9863	0.56
30	45	0.907	0.405	0.986	0.56
40	90	0.9061	0.4051	0.9863	0.5599
25	30	0.9051	0.4053	0.9855	0.5599
35	75	0.9085	0.4045	0.9865	0.5598
35	70	0.9079	0.4047	0.9864	0.5598
30	50	0.9077	0.4046	0.9864	0.5597
40	95	0.906	0.4049	0.9864	0.5597
40	85	0.9057	0.405	0.9863	0.5597
35	80	0.9086	0.4043	0.9866	0.5596
25	35	0.9104	0.4039	0.9867	0.5595
30	55	0.9087	0.4041	0.9867	0.5594
30	60	0.9099	0.4037	0.9869	0.5592
35	90	0.9093	0.4038	0.9867	0.5592
35	95	0.9093	0.4036	0.9867	0.5591
35	85	0.909	0.4037	0.9866	0.5591
30	65	0.9102	0.4033	0.987	0.559
80	85	0.8353	0.42	0.9621	0.5589
25	40	0.9114	0.403	0.9869	0.5588
30	75	0.9107	0.403	0.9871	0.5588
30	70	0.9103	0.4031	0.987	0.5588
30	80	0.9108	0.4029	0.9871	0.5587
30	90	0.9111	0.4023	0.9872	0.5582
25	45	0.9116	0.4021	0.9872	0.5581
30	95	0.9111	0.4022	0.9872	0.5581
30	85	0.9108	0.4023	0.9872	0.5581
20	25	0.9108	0.4021	0.9865	0.5579
25	50	0.9121	0.4017	0.9873	0.5578
20	30	0.9154	0.4009	0.9876	0.5576
25	55	0.9132	0.4013	0.9877	0.5576
25	60	0.9136	0.4009	0.9877	0.5573
25	65	0.9139	0.4007	0.9878	0.5571
25	70	0.9139	0.4005	0.9878	0.557
25	75	0.9143	0.4004	0.9879	0.5569
25	80	0.9143	0.4003	0.9879	0.5568
20	35	0.917	0.3994	0.988	0.5565
25	90	0.9146	0.3998	0.988	0.5563
20	40	0.9173	0.3991	0.9881	0.5562
25	95	0.9146	0.3996	0.988	0.5562
25	85	0.9143	0.3997	0.988	0.5562
20	45	0.9169	0.3986	0.9881	0.5557
20	50	0.9168	0.3984	0.9882	0.5554
20	55	0.9171	0.398	0.9883	0.5551
20	60	0.9173	0.3977	0.9883	0.5548
20	65	0.9177	0.3974	0.9884	0.5546
20	75	0.9178	0.3971	0.9885	0.5544

Continued on next page

Table A.2 – continued from previous page

Y_min	Y_max	Precision	Recall	Specificity	F-measure
20	70	0.9177	0.3972	0.9885	0.5544
20	80	0.9178	0.397	0.9885	0.5543
20	90	0.9178	0.3966	0.9885	0.5538
20	85	0.9178	0.3965	0.9885	0.5538
20	95	0.9178	0.3965	0.9885	0.5537
15	20	0.9164	0.3957	0.9871	0.5528
15	25	0.9184	0.3946	0.9878	0.552
15	30	0.9205	0.3939	0.9883	0.5517
15	40	0.9212	0.393	0.9887	0.551
15	35	0.9211	0.3931	0.9887	0.551
15	50	0.921	0.3926	0.9888	0.5505
15	45	0.9209	0.3926	0.9887	0.5505
15	55	0.9211	0.3922	0.9889	0.5502
15	60	0.921	0.3919	0.9889	0.5499
15	65	0.9212	0.3916	0.989	0.5496
15	75	0.9217	0.3913	0.9891	0.5494
15	70	0.9213	0.3914	0.989	0.5494
15	80	0.9216	0.3913	0.9891	0.5493
15	90	0.9218	0.3909	0.9891	0.549
15	85	0.9218	0.3908	0.9891	0.5489
15	95	0.9218	0.3908	0.9891	0.5489
90	95	0.8085	0.4137	0.9555	0.5474
10	15	0.9199	0.3885	0.9882	0.5463
10	20	0.922	0.3862	0.9884	0.5444
10	25	0.9224	0.3858	0.9886	0.5441
10	30	0.9238	0.3854	0.9889	0.5439
10	35	0.9244	0.3847	0.9893	0.5433
10	40	0.9244	0.3847	0.9894	0.5433
10	45	0.9241	0.3845	0.9894	0.543
10	50	0.9241	0.3844	0.9894	0.543
10	55	0.9242	0.3841	0.9895	0.5426
10	60	0.9241	0.3838	0.9895	0.5423
10	65	0.9244	0.3836	0.9896	0.5422
10	70	0.9245	0.3835	0.9896	0.5421
10	75	0.9245	0.3834	0.9896	0.542
10	80	0.9245	0.3834	0.9896	0.542
10	90	0.925	0.383	0.9897	0.5417
10	85	0.925	0.3829	0.9897	0.5416
10	95	0.925	0.3829	0.9897	0.5416
5	10	0.9247	0.3793	0.9884	0.538
5	15	0.9269	0.3768	0.9889	0.5358
5	20	0.9277	0.3759	0.9892	0.535
5	25	0.9275	0.3756	0.9893	0.5347
5	90	0.9292	0.375	0.9901	0.5344
5	50	0.9291	0.375	0.99	0.5344
5	85	0.9293	0.375	0.9901	0.5343
5	55	0.9293	0.3749	0.99	0.5343
5	60	0.9293	0.3749	0.99	0.5343
5	95	0.9292	0.375	0.9901	0.5343
5	65	0.9292	0.3749	0.9901	0.5343
5	80	0.9292	0.3749	0.9901	0.5343
5	45	0.9289	0.375	0.99	0.5343
5	40	0.9285	0.3751	0.9898	0.5343
5	30	0.9278	0.3752	0.9895	0.5343
5	70	0.9292	0.3749	0.9901	0.5342
5	75	0.9292	0.3749	0.9901	0.5342
5	35	0.9283	0.3748	0.9897	0.534

Bibliography

Abu-Mostafa *et al.***(2012)** Yaser S. Abu-Mostafa, Malik Magdon-Ismail and Hsuan-Tien Lin. *Learning from data : a short course*. AMLBook.com. 40

Albiol *et al.***(2001)** Alberto Albiol, Luis Torres and Edward J. Delp. Optimum color spaces for skin detection. In *International Conference on Image Processing*, pages 122–124. 6

Basilio *et al.***(2011)** Jorge Alberto Marcial Basilio, Gualberto Aguilar Torres, Gabriel Sánchez Pérez, L Karina Toscano Medina and Hector M Perez Meana. Explicit image detection using ycbcr space color model as skin detection. *Applications of Mathematics and Computer Engineering*, pages 123–128. 4, 14

Ben(2009) R. G. Ben. CIE 1931 xy color space diagram. https://en.wikipedia.org/wiki/File:CIE1931xy_blank.svg, 2009. Accessed on 12/10/2016. 18

Bergasa *et al.***(2000)** Luis Miguel Bergasa, Manuel Mazo, Alfredo Gardel, Miguel A. Sotelo and Luciano Boquete. Unsupervised and adaptive gaussian skin-color model. *Image and Vision Computing*, 18(12):987–1003. 6

Bhatt and Dhall(2012) Rajen B. Bhatt and Abhinav Dhall. Skin segmentation dataset. https://archive.ics.uci.edu/ml/datasets/Skin+Segmentation, 2012. Accessed on 05/06/2016. 35

Brancati *et al.***(2017)** Nadia Brancati, Giuseppe De Pietro, Maria Frucci and Luigi Gallo. Human skin detection through correlation rules between the YCb and YCr subspaces based on dynamic color clustering. *Computer Vision and Image Understanding*, 155:33–42. xv, 1, 6, 14, 15, 16, 25, 26, 32, 39, 42, 43, 44, 45

Casati *et al.***(2013)** João Paulo Brognoni Casati, Diego Rafael Moraes and Evandro Luis Linhari Rodrigues. SFA: A human skin image database based on FERET and AR facial images. In *IX workshop de Visão Computacional*. 36, 37

Chai and Ngan(1999) Douglas Chai and King N. Ngan. Face segmentation using skin-color map in videophone applications. *IEEE Trans. on Circ. and Sys. for Video Tech.*, 9(4):551–564. 4, 14, 26

Chaves-González *et al.***(2010)** Jose M. Chaves-González, Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido and Juan M. Sánchez-Pérez. Detecting skin in face recognition systems: A colour spaces study. *Digital Signal Processing*, 20(3):806–823. 6

Fleck *et al.***(1996)** Margaret M Fleck, David A Forsyth and Chris Bregler. Finding naked people. In *European conference on computer vision*, pages 593–602. Springer. 1

Gevers *et al.***(2012)** Theo Gevers, Arjan Gijsenij, Joost van de Weijer and Jan-Mark Geusebroek. *Color in Computer Vision: Fundamentals and Applications*. Wiley. 16, 18

Gonzalez and Woods(2002) Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Prentice Hall, 2nd edition. 1, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20

Grzejszczak *et al.***(2016)** Tomasz Grzejszczak, Michal Kawulok and Adam Galuszka. Hand landmarks detection and localization in color images. *Multimedia Tools and Applications*, 75(23): 16363–16387. ISSN 1573-7721. doi: 10.1007/s11042-015-2934-5. 38

Guevara *et al.***(2014)** Jorge Guevara, Roberto Hirata Jr and Stephane Canu. Positive definite kernel functions on fuzzy sets. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 439–446. doi: 10.1109/FUZZ-IEEE.2014.6891628. Accessed on 14/09/2016. 49

Hsu *et al.***(2002)** Rein-Lien Hsu, M. Abdel-Mottaleb and A. K. Jain. Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):696–706. ISSN 0162-8828. doi: 10.1109/34.1000242. 4

Ice(2016) Black Ice. The HSI color space. http://www.blackice.com/images/HSIColorModel.jpg, 2016. Accessed on 15/10/2016. 21

Jayaram *et al.***(2004)** Sriram Jayaram, Stephen Schmugge, Min C. Shin and Leonid V. Tsap. Effect of colorspace transformation, the illuminance component, and color modeling on skin detection. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 813–818. IEEE. 6

Jones and Rehg(2002) Michael J. Jones and James M. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision*, 46(1):81–96. 3, 39

Kakumanu *et al.***(2007)** Praveen Kakumanu, Sokratis Makrogiannis and Nikolaos Bourbakis. A survey of skin-color modeling and detection methods. *Pattern recognition*, 40(3):1106–1122. 1, 3, 6, 20

Kaur and Kranthi(2012) Amanpreet Kaur and B. V. Kranthi. Comparison between YCbCr color space and CIELab color space for skin color segmentation. *International Journal of Applied Information Systems*, 3(4):30–33. 6, 14

Kawulok *et al.***(2013)** Michal Kawulok, Jolanta Kawulok, Jakub Nalepa and Maciej Papiez. Skin detection using spatial analysis with adaptive seed. In *2013 IEEE International Conference on Image Processing*, pages 3720–3724. IEEE. 5

Kawulok *et al.***(2014)** Michal Kawulok, Jolanta Kawulok, Jakub Nalepa and Bogdan Smolka. Self-adaptive algorithm for segmenting skin regions. *EURASIP Journal on Advances in Signal Processing*, 2014(170):1–22. ISSN 1687-6180. doi: 10.1186/1687-6180-2014-170. URL http://asp.eurasipjournals.com/content/2014/1/170. 38

Khan *et al.***(2012)** Rehanullah Khan, Allan Hanbury, Julian Stöttinger and Abdul Bais. Color based skin classification. *Pattern Recognition Letters*, 33(2):157–163. 6

Kovac *et al.***(2003)** Jure Kovac, Peter Peer and Franc Solina. *Human skin color clustering for face detection*, volume 2. IEEE. 4, 5, 6, 14

Kumar and Malhotra(2015) Amit Kumar and Shivani Malhotra. Performance analysis of color space for optimum skin color detection. In *2015 Fifth International Conference on Communication Systems and Network Technologies*, pages 554–558. IEEE. 6, 14

Lichman(2013) M. Lichman. UCI machine learning repository. http://archive.ics.uci.edu/ml, 2013. 35

Mahmoodi and Sayedi(2016) Mohammad Reza Mahmoodi and Sayed Masoud Sayedi. A comprehensive survey on human skin detection. *International Journal of Image, Graphics and Signal Processing*, 8(5):1–35. Accessed on 23/04/2016. 3, 6, 39

Martínez and Benavente(1998) Aleix Martínez and Robert Benavente. The AR face database. Technical report, Purdue University. 6, 36

Minear and Park(2004) Meredith Minear and Denise Park. Productive aging lab face database. https://pal.utdallas.edu/facedb/, 2004. 35

Naji *et al.***(2012)** Sinan A. Naji, Roziati Zainuddin and Hamid A. Jalab. Skin segmentation based on multi pixel color clustering models. *Digital Signal Processing*, 22(6):933–940. 5

Nalepa and Kawulok(2014) Jakub Nalepa and Michal Kawulok. Fast and accurate hand shape classification. In Stanislaw Kozielski, Dariusz Mrozek, Pawel Kasprowski, Bozena Malysiak-Mrozek and Daniel Kostrzewa, editors, *Beyond Databases, Architectures, and Structures*, volume 424 of *Communications in Computer and Information Science*, pages 364–373. Springer. ISBN 978-3-319-06931-9. doi: 10.1007/978-3-319-06932-6_35. URL http://dx.doi.org/10.1007/978-3-319-06932-6_35. 38

Pedregosa *et al.***(2011)** F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830. 40, 41

Pedrini and Schwartz(2008) Hélio Pedrini and William Robson Schwartz. *Análise de imagens digitais: princípios, algoritmos e aplicações*. Thomson Learning, São Paulo. 10, 11, 12, 13, 14, 20, 22, 23

Pedrycz and Gomide(1998) Witold Pedrycz and Fernando Gomide. *An Introduction to Fuzzy Sets: Analysis and Design*. A Bradford Book. 2

Phillips *et al.***(1996)** P. Jonathon Phillips, Harry Wechsler, Jeffrey Huang and Patrick J. Rauss. The facial recognition technology (FERET) database. https://www.nist.gov/programs-projects/face-recognition-technology-feret, 1996. Accessed on 23/06/2016. 35, 36

Plataniotis and Venetsanopoulos(2000) Konstantinos N. Plataniotis and Anastasios N. Venetsanopoulos. *Color Image Processing and Applications*. Springer, 1st edition. 14, 15, 16, 17, 19, 21

Rus(2007) Jacob Rus. The Munsell color system. https://commons.wikimedia.org/wiki/File:Munsell-system.svg, 2007. Accessed on 12/10/2016. 17

Rus(2008) Jacob Rus. The "primary" and "secondary" colors in a four-color print process. https://en.wikipedia.org/wiki/File:SubtractiveColor.svg, 2008. Accessed on 12/10/2016. 20

Shaik *et al.***(2015)** Khamar Basha Shaik, Ganesan P., V. Kalist, B. S. Sathish and J. Merlin Mary Jenitha. Comparative study of skin color detection and segmentation in HSV and YCbCr color space. *Procedia Computer Science*, 57:41–48. 6, 14

Tan *et al.***(2012)** Wei Ren Tan, Chee Seng Chan, Pratheepan Yogarajah and Joan Condell. A fusion approach for efficient human skin detection. *IEEE Transactions on Industrial Informatics*, 8(1):138–147. 5, 14, 37, 38

Vezhnevets *et al.***(2003)** Vladimir Vezhnevets, Vassili Sazonov and Alla Andreeva. A survey on pixel-based skin color detection techniques. In *IN PROC. GRAPHICON-2003*, pages 85–92. 1, 3, 4, 5, 18

Yogarajah *et al.***(2011)** Pratheepan Yogarajah, Joan Condell, Kevin Curran, Paul McKevitt and Abbas Cheddad. A dynamic threshold approach for skin tone detection in colour images. *International Journal of Biometrics*, 4(1):38–55. 5, 14