Face Detection and Recognition in Indoor Environment

H A D I E S T E K I

KTH Computer Science
and Communication

Master of Science Thesis
Stockholm, Sweden 2007

Face Detection and Recognition in Indoor Environment

H A D I E S T E K I

Master's Thesis in Numerical Analysis (30 ECTS credits)
at the Scientific Computing International Master Program
Royal Institute of Technology year 2007
Supervisors at CSC were Alireza Tavakoli Targhi and Babak Rasolzadeh
Examiner was Axel Ruhe

TRITA-CSC-E 2007:139
ISRN-KTH/CSC/E--07/139--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication

KTH CSC
SE-100 44 Stockholm, Sweden

URL: www.csc.kth.se

Abstract

This thesis examines and implements some state-of-the-art methods for human face recognition. In order to examine all aspects of the face recognition task at a generic level, we have divided the task into three consecutive steps: 1) Skin color detection (for identifying regions with skin-colored), 2) Face detection (no identification) and 3) Face recognition (for identifying *which* face is detected). Using a statistical model for the color region we can narrow down the possible regions in which faces could be found. Furthermore, trying to do an identification on a region that does not include a face is inefficient, since identification is a computationally complex process. Therefore, using a faster and less complex algorithm to do the general face detection before face recognition is a faster way to identify faces. In this work, we use a machine learning approach for boosting weaker hypotheses into a stronger one in order to detect faces. For face recognition, we look deeper into the appearance of the face and define the identity as a specific texture that we try to recognize under different appearances. Finally, we merge all the different steps into an integrated system for full-frontal face detection and recognition. We evaluate the system based on accuracy and performance.

Acknowledgement

This research would not have been started at all without the great response in the early days from Alireza Tavakoli. Without the encouraging support from my supervisors, Alireza Tavakoli and Babak Rasolzadeh at the Royal Institute of Technology (KTH-CVAP), there would have been very little to write.
To this end, the great interest shown by my examiner Axel Ruhe at the Royal Institute of Technology (KTH-NA) has been encouraging.

Contents

Chapter 1

Introduction

1.1 Face Detection and Recognition

Finding faces in an arbitrary scene and successfully recognizing them have been an active topics in Computer Vision for decades. A general statement of the face recognition problem (in computer vision) can be formulated as follows: Given still or video images of a scene, identify or verify one or more persons in the scene using a stored database of faces. Although face detection and recognition is still an unsolved problem meaning there is no 100% accurate face detection and recognition system, however during the past decade, many methods and techniques have been gradually developed and applied to solve the problem.

Basically, there are three types of methods in automatic face recognition: verification, identification and watch-list. In the verification method, a comparison of only two images is considered. The comparison is positive if the two images are matched. In the identification method, more than one comparison should be done to return the closest match of the input image. The watch-list method works similar to the identification method with a difference that the input face can also be rejected (no match).

The method presented in this thesis consists of three steps: skin detection, face detection, and face recognition. The novelty of the proposed method is using a skin detection filter as a pre-processing step for face detection. A scheme of main tasks is shown in Figure 1.1.

- Skin Detection: This first step of the system consists of detecting the skin color region in our input data which generally consists of still-images or video sequences taken from some source such as a camera, a scanner or a file. The experience suggests that human skin has a characteristic color, which is easily recognized by humans. The aim here is to employ skin color modeling for face detection.

- Face Detection: The second step is detecting the presence of face(s) and determining their locations from the result of step 1. Thus, the input of step 2

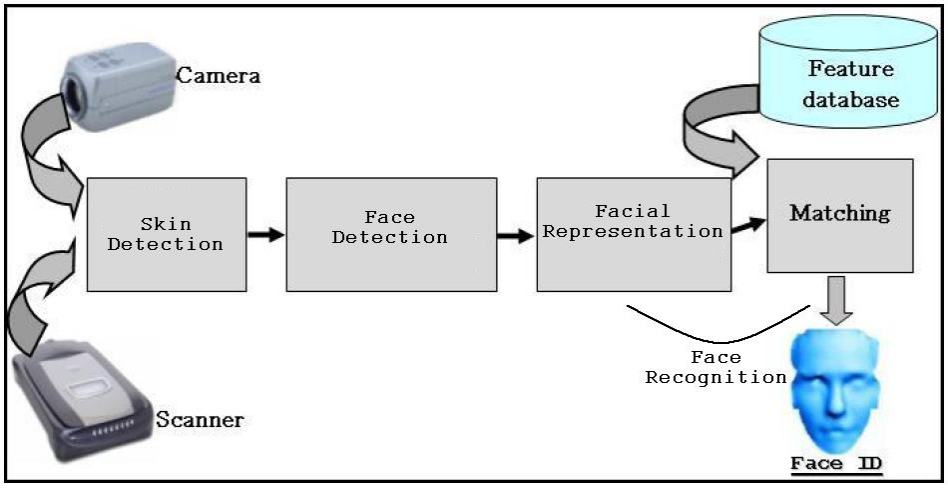

Figure 1.1. General scheme for our system

is the output of step 1. In the case of no skin detecting, the entire original image will be used as input for face detection.

- Facial Representation and Matching: Facial representation of the characteristics of the face is computed in the third step. A comparison for matching will then be done. If the match is close enough to one of the faces in the database, the identity of the face is sorted. Otherwise, the face is rejected in the watch-list mode and the closest face is returned in the identification mode.

1.2 Review of Recent Work

A primitive face detection method can be finding faces in images with controlled background by using images with a plain monocolor background, or using them with a predefined static background. The drawback of such methods is that removing the background will always yield face boundaries.

When color exists, another method can be finding faces with the help of color. In case of having access to color images, one might use the typical skin color to find face segments. The process is carried out in two main steps. The first step is skin filtering by detecting regions which are likely to contain human skin in the color image. The result of this step followed by thresholding is a binary skin map which shows us the skin regions. The second step is face detection by extracting information from regions which might indicate the location of a face in the image by taking the marked skin regions (from first step) and removing the darkest and brightest regions from the map. The removed regions have been shown through empirical tests to correspond to those regions in faces which are usually the eyes and eyebrows, nostrils, and mouth. Thus, the [1] skin detection is performed using a skin filter which relies on color and texture information. The face detection is performed on a

greyscale image containing only the detected skin areas. A combination of thresholding and mathematical morphology is used to extract object features that would indicate the presence of a face. The face detection process works predictably and fairly reliably. The test results show very good performance when a face occupies a large portion of the image, and reasonable performance on those images depicting people as part of a larger scene. The main drawbacks are:

1. Risk of detecting non-face objects when the face objects do not occupy a significant area in the image.

2. Large skin map (for example, a naked person as the image).

Finally, the method does not work with all kinds of skin colors, and is not very robust under varying lighting conditions.

Another method is face detection in color images using PCA, Principal Component Analysis. Principal Components Analysis can be used for the localization of face region. An image pattern is classified as a face if its distance to the face model in the face space is smaller than a certain threshold. The disadvantage of this method is that it leads to a significant number of false classifications if the face region is relatively small. A classification based on shape may fail if only parts of the face are detected or the face region is merged with skin-colored background. In [2], the color information and the face detection scheme based on PCA are incorporated in such a way that instead of performing a pixel-based color segmentation, a new image which indicates the probability of each image pixel belonging to a skin region (skin probability image) is created. Using the fact that the original luminance image and the probability image have similar gray level distributions in facial regions, Principal Components Analysis is used to detect facial regions in the probability image. The utilization of color information in a PCA framework results in a robust face detection even in the presence of complex and skin colored background[2].

Hausdorff Distance (HD) is another method used for face detection. HD is a metric between two point sets. The HD is used in image processing as a similarity measure between a general face model and possible instances of the object within the image. According to the definition of HD in $2D$, if $A = a_1, \ldots, a_n$ and $B = b_1, \ldots, b_m$ denote two finite point sets, then

$$H(A, B) = Max(h(A, B), h(B, A)),$$

when

$$h(A, B) = Max_{a \in A} Min_{b \in B} \|a - b\| .$$

In [3], $h(A, B)$is called the directed Hausdorff Distance from set A to B. A modification of the above definition is useful for image processing applications, which is the so called MHD. It is defined as

$$h_{mod}(A,B) = \frac{1}{|A|}\sum_{a\in A} Min_{b\in B}\left\|a-b\right\|.$$

By taking the average of the single point distances, this version decreases the impact of outliers, making it more suitable for pattern recognition purposes. Now let A and B be the image and the object respectively, the goal is to find the transformation parameter such that HD between the transformed model and A is minimized. The detection optimization problem can be formulated as:

$$d_{p^-} = Min_{p\in P} H(A, T_p(B))$$

When $T_p(B)$ is the transformed model, $h(T_p(B), A)$ and $h(A, T_p(B))$ are the forward and reverse distance, respectively. The value of d_{p^-} is the distance value of the best matching position and scale. The implemented face detection system consists of a coarse detection and a refinement phase, containing segmentation and localization step. Coarse Detection: An AOI (Area Of Interest) with preset width/height rate is defined for an incoming image. This AOI will then be resampled to a fixed size which is independent of the dimension of the image.

- Segmentation: An edge intensity image will be calculated from the resized AOI with the Sobel operator. Then, local thresholding will give us the edge points.
- Localization: The modified forward distance, $h(T_p(B), A)$, is sufficient to give an initial guess for the best position. The d_{p^-} minimizes $h(T_p(B), A)$ will be the input for the next step (refinement).

Refinement phase: given a d_{p^-}, a second AOI is defined covering the expected area of the face. This AOI is resampled from the original image resulting in a greyscale image of the face area. Then segmentation and localization are like previous phase with modified box reverse distance $h_{box}(A'^-, T_{p'}(B'))$. Validation is based on the distance between the expected and the estimated eye positions: the so called (normalized) relative error with definition,

$$d_{box} = \frac{Max(d_l, d_r)}{\|C_l - C_r\|}$$

,

where d_l and d_r are the distance between the true eye centers and the estimated positions. In [3], a face is found if $d_{eye} < 0.25$. Two different databases are utilized. The first one contains 1180 color images of 295 test persons (360×288). The

second one contains 1521 images of 23 persons with larger variety of illumination, background and face size (384 x 288). The value of 98.4% on the first test set and 91.8% on the second one is obtained as the robustness of the method. The average processing time per frame on a PIII $850MHz$ system is 23.5 ms for the coarse detection step and an additional 7.0 ms for the refinement step, which allows the use in real time video applications ($> 30fps$) [3].

A major problem of the Hausdorff Distance method is the actual creation of a proper face model ($T_p(B)$). While a simple "hand-drawn" model will be sufficient for the detection of simple objects, a general face model must cover the broad variety of different faces. In order to optimize the method, finding of a well-suited model for HD based face localization can be formulated as a discrete global optimization problem is interested. For this issue The General Algorithm (GA) is employed as a standard approach for multi-dimensional global optimization problems, namely the simple Genetic Algorithm (SGA) described by Goldberg [4].

A Genetic Algorithm (GA) approach is presented for obtaining a binary edge model that allows localization of a wide variety of faces with the HD method. The GA performs better when starting from scratch than from a hand-drawn model. Three different initializations of the population are tested in [3]: blank model, average edge model, and hand-drawn model. An improvement from 60 % to 90% is achieved for localization performance. Therefore, GA is a powerful tool that can help in finding an appropriate model for face localization. Face localization can be improved by a multi-step detection approach that uses more than one model in different grades of details. Each of these models can then be optimized separately. This does not only speed up the localization procedure but also produces more exact face coordinates [6].

1.3 Thesis Outline

The rest of the thesis consists of three main parts, namely color, face detection and face recognition. Each single part has been described in a seperate chapter. In Chapter 2, the color has been discussed. Chapter 3 and 4 explain basic principles of face detection and recognition respectively. In Chapter 5, the utilizing method in this work has been discussed including thee main parts. The experimental evaluation is then presented in Chapter 6, and finally the conclusions and proposed future works in Chapter 7.

Chapter 2

Color: An Overview

Skin color has proved to be a useful and robust cue for face detection. Image content filtering and image color balancing applications can also benefit from automatic detection of skin regions in images. Numerous techniques for skin color modeling and recognition have been proposed in the past years. The face detection methods, that use skin color as a detection cue have gained strong popularity among other techniques. Color allows fast processing and is highly robust to geometric variations of the skin pattern. The experience suggests that human skin has a characteristic color, which is easily recognized by humans. So trying to employ skin color modeling for face detection was an idea suggested both by task properties and common sense. In this paper, we discuss pixel-based skin detection methods, which classify each pixel as skin or non-skin individually. Our goal in this work is to evaluate two most important color spaces and try to find out and summarize the advantages.

2.1 Skin modeling

The major goal of skin modeling is to discriminate between skin and non-skin pixels. This is usually accomplished by introducing a metric, which measures distances (in general sense) of pixel color to skin tone. The type of this metric is defined by the skin color modeling methods. A classification of skin-color modeling is accomplished by [11]. In this work, the Gaussian model will be discussed.

2.1.1 Gaussian Model

The Gaussian model is the most popular parametric skin model. The model performance directly depends on the representativeness of the training set, which is going to be more compact for certain applications in skin model representation.

2.1.2 Single Gaussian

Skin color distribution can be modeled by an elliptical Gaussian joint probability density function (pdf), defined as:

$$P(c|skin) = \frac{1}{2\pi|\sum_s|^{\frac{1}{2}}}.e^{\frac{-1}{2}(c-\mu_s)^T\sum_s^{-1}(c-\mu_s)}.$$

Here, c is a color vector and μ_s and Σ_s are the distribution parameters (mean vector and covariance matrix respectively). The model parameters are estimated from the training data by:

$$\mu_s = \frac{1}{n}\sum_{j=1}^{n} c_j \; and \; \sum_s = \frac{1}{n-1}\sum_{j=1}^{n}(c_j-\mu_s)(c_j-\mu_s)^T,$$

when $j = 1, \ldots, n$, and n is the total number of skin color samples c_j. The $P(c|skin)$ probability can be used directly as the measure of how "skin-like" the c color is [16], or alternatively, the Mahalanobis distance from the c color vector to mean vector μ_s, given the covariance matrix σ_s can serve for the same purpose [17]:

$$\lambda_s(c) = (c-\mu_s)^T\sum_s^{-1}(c-\mu_s)$$

2.1.3 Mixture of Gaussians

A more sophisticated model, capable of describing complex shaped distributions is the Gaussian mixture model. It is the generalization of the single Gaussian, the pdf in this case is:

$$P(c|skin) = \sum_{i=1}^{k}\pi_i.P_i(c|skin),$$

where k is the number of mixture components, P_i are the mixing parameters, obeying the normalization constraint $\sum_{i=1}^{k}\pi_i = 1$, and $P_i(c|skin)$ are Gaussian pdfs, each with their own mean and covariance matrix. Model training is performed with a well-known iterative technique called the Expectation Maximization (EM) algorithm, which assumes the number of components k to be known beforehand. The details of training Gaussian mixture model with EM can be found, see for example in [18]. The classification with a Gaussian mixture model is done by comparing the $p(c|skin)$ value to some threshold. The choice of the component number k is important here. The model needs to explain the training data reasonably well with the given model on one hand, and avoid data over-fitting on the other. The number of components used by different researchers varies significantly: from 2 in [18] to 16 in [19]. A bootstrap test for justification of $k = 2$ hypothesis was performed in [20]. In [17], $k = 8$ was chosen as a "good compromise between the accuracy of of estimation of the true distributions and the computational load for thresholding".

2.2 Skin-Color Space

Skin color detection and modelling have been frequently used for face detection. A rapid survey of common color spaces will be given. Then, we will try to figure out which color space would be more appropriate for our purposes.

2.2.1 RGB

RGB is a color space originated from CRT display applications (or similar applications), when it is convenient to describe color as a combination of three colored rays (red, green and blue). It is one of the most widely used color spaces for processing and storing of digital image data. However, high correlation between channels, significant perceptual non-uniformity, mixing of chrominance and luminance data make RGB not a very favorable choice for color analysis and colorbased recognition algorithms [8].

2.2.2 Normalized RGB

Normalized RGB is a representation that is easily obtained from the RGB values by a simple normalization procedure:

$$r = \frac{R}{R+G+B};\ g = \frac{G}{R+G+B};\ b = \frac{B}{R+G+B}.$$

As the sum of the three normalized components is known ($r + g + b = 1$), the third component does not hold any significant information and can be omitted, reducing the space dimensionality. The remaining components are often called "pure colors", for the dependance of r and g on the brightness of the source RGB color is diminished by the normalization. A remarkable property of this representation is for matte surfaces: while ignoring ambient light, normalized RGB is invariant (under certain assumptions) to changes of surface orientation relatively to the light source [21]. This, together with the simplicity of the transformation helped this color space to gain popularity among researchers.

2.2.3 HSI, HSV, HSL - Hue Saturation Intensity (Value, Lightness)

Hue-saturation based color spaces are introduced when there is a need for the user to specify color properties numerically. They describe color with intuitive values, based on the artist's idea of tint, saturation and tone. Hue defines the dominant color (such as red, green, purple and yellow) of an area, saturation measures the colorfulness of an area in proportion to its brightness [8]. The "intensity", "lightness" or "value" is related to the color luminance. The intuitiveness of the color space components and explicit discrimination between luminance and chrominance properties made these color spaces popular in the works on skin color segmentation. However, [22] points out several undesirable features of these color spaces, including hue discontinuities

and the computation of "brightness" (lightness, value), which conflicts badly with the properties of color vision.

$$\begin{cases} H = \arccos \frac{\frac{1}{2}((R-G)+(R-B))}{\sqrt{((R-G)^2+(R-B)(G-B))}} \\ S = 1 - 3\frac{(R,G,B)}{R+G+B} \\ V = \frac{1}{3}(R+G+B) \end{cases}$$

An alternative way of Hue-Saturation computation using log opponent values was introducing additional logarithmic transformation of RGB values aimed to reduce the dependance of chrominance on the illumination level. The polar coordinate system of Hue-Saturation spaces, resulting in cyclic nature of the color space makes it inconvenient for parametric skin color models that need tight cluster of skin colors for best performance. Here, different representations of Hue-Saturation using Cartesian coordinates can be used [8]:

$$X = S\cos H; Y = S\sin H.$$

2.2.4 YCrCb

$YCrCb$ is an encoded nonlinear RGB signal, commonly used by European television studios and for image compression work. Color is represented by luma (which is luminance, computed from nonlinear RGB [22]), constructed as a weighted sum of the RGB values, and two color difference values Cr and Cb that are formed by subtracting luma from the red and blue components in RGB [9].

$$\begin{cases} Y = 0.299R + 0.587G + 0.114B \\ Cr = R - Y \\ Cb = B - Y \end{cases}$$

The simplicity of the transformation and explicit separation of luminance and chrominance components makes this color space attractive for skin color modelling.

2.2.5 Which skin color space

One of the major questions in using skin color in skin detection is how to choose a suitable color space. A wide variety of different color spaces has been applied to the problem of skin color modelling. From a recent research [8], a briefly review of the most popular color spaces and their properties is presented. For real world applications and dynamic scenes, color spaces that separate the chrominance and luminance components of color are typically preferable. The main reason for this is that chrominance-dependent components of color are considered, and increased

robustness to illumination changes can be achieved. Since for example, HSV seems to be a good alternative, but HSV family presents lower reliability when the scenes are complex and they contain similar colors such as wood textures [10]. Moreover, in order to transform a frame, it would be necessary to change each pixel to the new color space which can be avoided if the camera provides RGB images directly, as most of them do. Therefore, for the purpose of this thesis, a choice between RGB and $YCrCb$ color spaces is considered.

Chapter 3

Face Detection

Face detection is a useful task in many applications such as video conferencing, human-machine interfaces, Content Based Image Retrieval (CBIR), surveillance systems etc. It is also often used in the first step of automatic face recognition by determining the presence of faces (if any) in the input image (or video sequence). The face region including its location and size is the output of a face detection step. In general, the face recognition problem (in computer vision) can be formulated as follows: Given still or video images of a scene, determine the presence of faces and then identify or verify one or more faces in the scene using a stored database of faces. Thus, the accuracy of a face recognition system is depended on the accuracy of the face detection system. But, the variability of the appearance in the face patterns makes it a difficult task. A robust face detector should be able to find the faces regardless of their number, color, positions, occlusions, orientations, an facial expressions, etc. Although this issue is still an unsolved problem, many methods have been proposed for detecting faces. Additionally, color and motion, when available, may be characteristics in face detection. Even if the disadvantages of color based methods like sensitivity on varying lighting conditions make them not as robust methods, they can still be easily used as a pre-processing step in face detection.

Most of the robust face detecting methods can be classified into two main categories: feature based and image based techniques. The feature based techniques make explicit use of face knowledge. They start by deriving low-level features and then apply knowledge based analysis. The image based techniques rely on a face in 2D. By using training schemes and learning algorithms the data can be classified into face or non-face groups. Here, a brief summary of feature and image based techniques will be presented.

3.1 Feature Based Techniques

As the title declared the focus in this class is on extracting facial features. The foundation of face detection task in feature based methods is the facial feature searching problem. Even these techniques are quite old and had been active up to the middle 90's. However, some feature extraction is still being utilized e.g. facial features using Gabor filters. The advantages of the feature based methods are their relative insensitivity to illumination conditions, occlusions and viewpoint whereas complex analysis (because computationally heavy) and the difficulties with low-quality images are the main drawbacks of these methods.

3.2 Image Based Techniques

Basically, these methods scan an input image at all possible locations and scale and then classify the sub-windows either as face or non-face. In fact, the techniques rely on training sets to capture the large variability in facial appearances instead of extracting the visual facial features (i.e. previous techniques).

Since the face detection step will be strictly affected on the performance of the whole system, a robust face detector should be employed. The accuracy and speed up of the face detectors have been studied in previous works. In this thesis, the chosen face detector is an efficient detector scheme presented by Viola and Jones (2001) using Haar-like features and Adaboost as training algorithm. In the next section, a brief description of the chosen scheme is given.

3.3 Face Detection based on Haar-like features and AdaBoost algorithm

This technique relies on the use of simple Haar-like features with a new image representation (integral image). Then AdaBoost is used to select the most prominent features among a large number of extracted features. Finally, a strong classifier from boosting a set of weak classifiers would be extracted. This approach has proven to be an effective algorithm to visual object detection and also one of the first real-time frontal-view face detectors. The effectiveness of this approach is based on four particular facts[12].

1. Using a set of simple masks similar to Haar-filters.

2. Using integral image representation which speeds up the feature extraction.

3. Using a learning algorithm, AdaBoost, yielding an effective classifier, which decreases the number of features.

4. Using the Attentional Cascade structure which allows background region of an image to be quickly discarded while spending more computation on promising object-like regions.

A discussion of each particular fact is presented below.

3.3.1 Haar-like features: Feature extraction

Working with only image intensities (i.e. the greylevel pixel values at each and every pixel of image) generally makes the task computationally expensive. An alternate feature set to the usual image intensities can be much faster. This feature set considers rectangular regions of the image and sums up the pixels in this region. Additionally, features carry better domain knowledge than pixels.
The Viola-Jones features can be thought of as pixel intensity set evaluations in its simplest form. The feature value is defined as the difference value between the sum of the luminance of some region(s) pixels and the sum of the luminance of other region(s) pixels. The position and the size of the features depend on the detection box. For instance, features of type 3 in Figure 3.1 will have 4 parameters: the position (x, y) in the detection box, the size of the white region (or positive region) (w), the size of black region(or negative region) (b), and the height (h) of the feature.

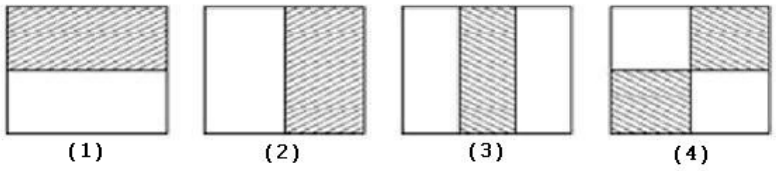

Figure 3.1. Four examples of the type of feature normally used in the Viola-Jones system.

3.3.2 Integral Image: Speed up the feature extraction

In order to have a reliable detection algorithm we need to develop two main issues, namely accuracy and speed. There is generally a trade-off between them. To improve the speed of the feature extraction one efficient way is to use the integral image representation. The integral image representation of an image (1) is defined as:

$$Int(x, y) = \sum_{x'=0}^{x} \sum_{y'=0}^{y} I(x', y').$$

Hence, the integral image at location (x, y) is the summation of all pixel values above and left of (x, y) inclusive. The computational tasks will be easier by using

the integral image representation which yields a speed up in the feature extracting process. This is done in such a manner that any rectangle in an image can be calculated from the corresponding integral image, by indexing the integral image only four times.

In Figure 3.2, there is an evaluation of a rectangle as an example. The rectangle is specified as four coordinates (x_1, y_1) upper left and (x_4, y_4) lower right.

$$A(x_1, y_1, x_4, y_4) = Int(x_1, y_1) + Int(x_4, y_4) - Int(x_1, y_4) - Int(x_4, y_1).$$

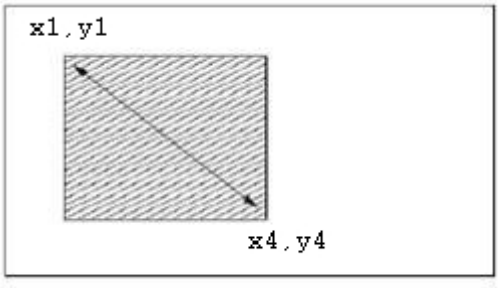

Figure 3.2. An example of integral image application.

Chapter 4

Face Recognition

A simple definition of face recognition is determining the identity of a given face. Thus, the facial representation and characteristics must be first extracted and then matched to database. The database consists of facial representations of known individuals. The given face can be recognized as a known person or rejected as a new unknown person after matching. The main issue is extracting facial representation. Several methods have been proposed based on sex different approaches.

1. Feature based methods: The feature based method is the earliest approach in face recognition, based on geometrical relations in poses, image conditions and rotation, e.g. distances between facial features like eyes. The recognition is usually done by using the Euclidian distance.

2. Model based methods: The model based methods basically consist of three steps, (a) Defining a model of the face structure, (b) Fitting the model to the given face image, and (c) Using the parameters of the fitted model as the feature vector to calculate similarity between the input face and those in the database.

3. Appearance based methods: The appearance based methods are similar to model based methods. The aim here is to achieve higher accuracy through larger training data set by finding some transformation for mapping the faces from the input space into a low-dimensional feature space (e.g. Principal Component Analysis).

4. 3D based methods: As the name is suggesting, 3D based methods rely on three dimensional poses. The time consuming nature of 3D based poses makes theses methods complicated to use.

5. Video based methods: Most of these methods are based on facial special structure. The best frames of video are chosen to feed into face recognition system.

6. Hybrid & Multimodal based methods: theses methods may be a combination of some of the above presented methods in order to have better performance, e.g. a combination of appearance and feature based methods.

In this work, a new feature space for representing face images is used. The facial representation is based on the local binary pattern, presented by Ojala 1996. A proper description of Local Binary Patterns (LBP) is presented in this work.

4.1 Local Binary Patterns

In short, a Local Binary Pattern is a texture descriptor. Since 2D surface texture is a valuable issue in machine vision, having a sufficient description of texture is useful in various applications. The Local Binary Patterns operator is one of the best performing texture descriptors. The LBP operator labels pixels of an image by thresholding the NxN neighborhood of each pixel with the value of the center pixel and considers the result as a binary mask. Figure 4.1 shows an example of an LPB operator utilizing 3x3 neighborhoods. The operator assigns a binary code of 0 and 1 to each neighbor of the mask. The binary code of each pixel in the case of 3x3 masks would be a binary code of 8 bits and by a single scan through the image for each pixel the LBP codes of the entire image can be calculated. An easy manner to show the final LBP codes over the image is the histogram of the labels where a 256-bin histogram represents the texture description of the image and each bin can be regarded as a micro-pattern, see Figure 4.2 for more details, [13].

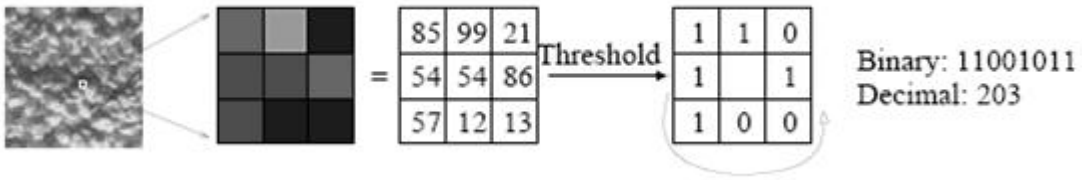

Figure 4.1. Example of an LBP calculation.

Since each bin represents a micro-pattern, the curved edges can be easily detected by LBP. Local primitives which are coded by these bins include different types of curved edges, spots, flat areas, etc. An example of texture primitives which can be detected by LBP comes in the Figure 4.3.

Ideally, a good texture descriptor should be easy to compute and has high extra-class variance (i.e., between different persons in the case of face recognition) and low intra-class variance, which means that the descriptor should be robust with respect to aging of the subjects, alternating illumination and other factors.

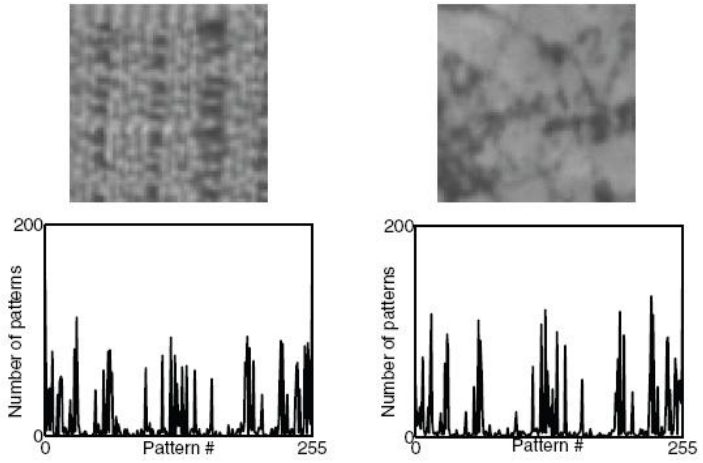

Figure 4.2. 256 bins LBP histograms of two samples .

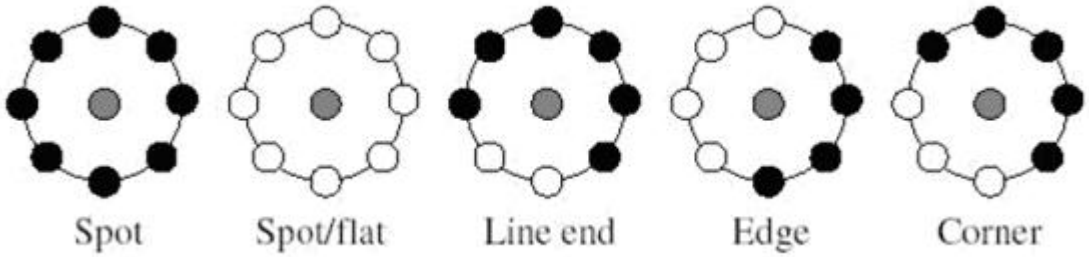

Figure 4.3. Example of texture primitives detected by LBP ,(white circles represent ones and black cirlces zeros).

The original LPB operator has been extended to consider different neighborhood sizes (Ojala et al. 2002*b*). For example, the operator $LBP(4,1)$ uses only 4 neighbors while $LBP(16,2)$ considers the 16 neighbors on a circle of radius 2. In general, the operator $LBP(P,R)$ refers to a neighborhood size of P of equally spaced pixels on a circle of radius R that form a circularly symmetric neighbor set. Figure 4.4 shows some examples of neighborhood sets. $LBP(P,R)$ produces 2^p different output values, corresponding to the 2^P different binary patterns that can be formed by the P pixels in the neighbor set. It has been shown that certain bins contain more information than others [15]. Therefore, it is possible to use only a subset of the 2^P local binary patterns to describe the textured images. Ojala et al. (2002b) defined these fundamental patterns (also called "uniform" patterns) as those with a small number of bitwise transitions from 0 to 1 and vice versa. For example, 00000000 and 11111111 contain 0 transition while 00000110 and 01111000 contain 2 transitions and so on. Thus a pattern is uniform if the number of bitwise transitions is

equal or less than 2. Accumulating the patterns which have more than 2 transitions into a single bin yields an LBP descriptor, denoted $LBP_{u2}(P, R)$, with less than 2^P bins. For example, the number of labels for a neighborhood of 8 pixels is $2^8 = 256$ for standard LBP and 59 for $LBP_{u2}(8, 1)$, where 58 bins of uniform patterns and 1 bin for all non-uniform patterns. For the 16-neighborhood the numbers are 65536 and 243, respectively. Since the LBP descriptor of most important information in a facial image such as edges, lines, spots etc (see Figure 4.3) contain equal or less than 2 transitions, one image can be described by its uniform pattern. For instance, in the case of using the $(8, 1)$ neighborhood or $(16, 2)$ amount of uniform patterns is around 90% or 70%, respectively. Therefore, the uniform pattern is preferred for face recognition.

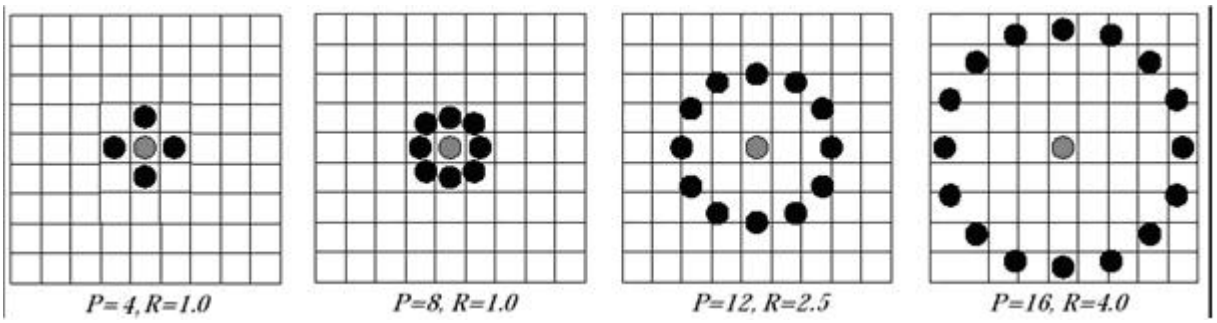

Figure 4.4. neighborhood set for different (P, R). The pixel values are bilinearly interpolated whenever the sampling point is not in the center of a pixel.

Chapter 5

The Face Recognition System

In this section, the model used in this thesis is described. The model is based on three main steps. At first pre-processing with Gaussian Mixture Model, then Face Detection with Adaboost, and finaly Face Recognition with Local Binary Pattern.

5.1 Pre-processing with Gaussian Mixture Model

The goal here is to detect regions in the color image which are likely to contain human skin. The result of this step is then used to filter the original image and then thresholding it, yielding a binary skin map which shows us the skin regions in the original image. These skin regions will be later sent to Face Detection process. The skin detection is performed using a Gaussian Mixture Model. To estimate the parameters of the Gaussian mixture the EM algorithm is employed when the parameters are evaluated by an E-step (expectation) and M-step (maximization), respectively. The algorithm begins by making some initial guess for the parameters of the Gaussian mixture model. The initial value here is obtained using k-means clustering. The samples are initially labelled using the k-means clustering where k is the number of components in the mixture model. We can then evaluate the new parameters using the following equation:

$$\Delta^{t+1} = E^{t+1} - E^t = -\sum_{j}^{n} \ln(\frac{p^{t+1}(x_j)}{p^t(x_j)}),$$

where $P^{t+1}(X)$ denotes the probability density evaluated using new values for the parameters, while $p^t(X)$ represents the density evaluated using the old parameter values. By setting the derivates of Δ^{t+1} to zero [23], the following equations are obtained:

$$\mu_i^{t+1} = \frac{\sum_{j=1}^{n} p^t(i|x_j)x_j}{\sum_{j=1}^{n} p^t(i|x_j)},$$

$$\Sigma_i^{t+1} = \frac{\sum_{j=1}^{n} p^t(i|x_j)||x_j - \mu_i^t||}{\sum_{j=1}^{n} p^t(i|x_j)}, \ and$$

$$\pi_i^{t+1} = \frac{1}{n}\sum_{j=1}^{n} p^t(i|x_j),$$

where $p^t(i|x_j) = \frac{p^t(x_j|i)\pi^t(i)}{p^t(x_j)}$.
Before we find the parameters we need choose a certain number of mixture components (k) which gives us the best performance in our application. A recent work [7], shows that a single Gaussian distribution is not sufficient to model human skin color and nor effective in a general application. Thus, k must be equal to or greater than 2. As said before, the number of components used by different researchers varies significantly. Furthermore, the size of skin samples seems to be a crucial factor in this process. This sample is initially labelled using the k-means clustering and then the final parameter values are obtained by using EM algorithm. Finally we will sum up the discussion with finding a reasonable threshold and skin color space in this work. Two different skin color spaces, namely RGB and YCrCb will be discussed. Since we focus on the face detection procedure, we need to have a threshold that gives us a region where we are not loosing information and also raises the speed up of the system.

5.2 Face Detection

5.2.1 Feature extraction with AdaBoost

AdaBoost, short for Adaptive Boosting, calls a weak classifier repeatedly and then for each call a distribution of weights is updated that indicates the importance of examples in the data set for the classification. The algorithm of the method is given in Algorithm 1. First of all we consider the weak classifiers and the way of building them. Given a single feature vector evaluated at x, denoted by $f_i(x)$m and the given threshold θ_i, the weak classification $h_i(x)$ is defined as follow:

$$\begin{cases} h_i(x) = 1 & if\ p_i f_i(x) < p_i\theta_i, \\ 0 & otherwise, \end{cases}$$

where p_i is the parity and x is the image-box to be classified. The output is a binary number 0 or 1 and depends on whether the feature value is less than the given threshold. The value of parity p_i and the threshold θ_i can be estimated

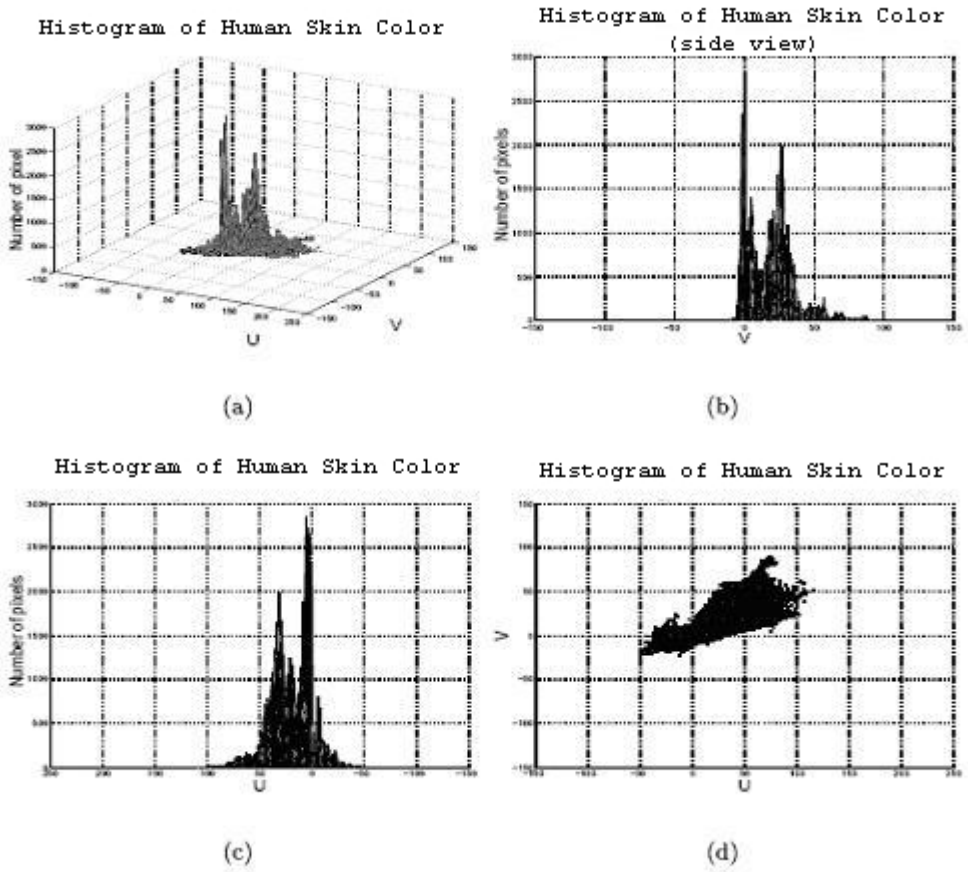

Figure 5.1. Histogram of skin color viewed from different angels

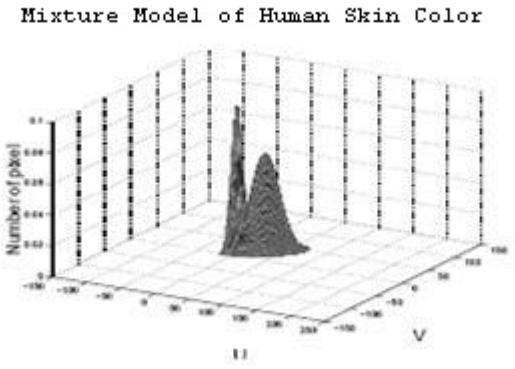

Figure 5.2. Estimated Density Function

from the evaluation of each feature type. There are two basic methods to carry out the estimation of p_i and θ_i. Both methods rely on estimating two probability distributions when applied for feature values to the positive samples (face data) and the negative samples (non-face data), respectively. The threshold can then be determined either by taking the average of their means or by finding the cross-over point. The cross-over point corresponds to

$$f_i s.t. p(f_i|non - face) = p(f_i|face).$$

In this work, the cross-over point is used. The Figure 5.3 shows how the probability distributions looks like

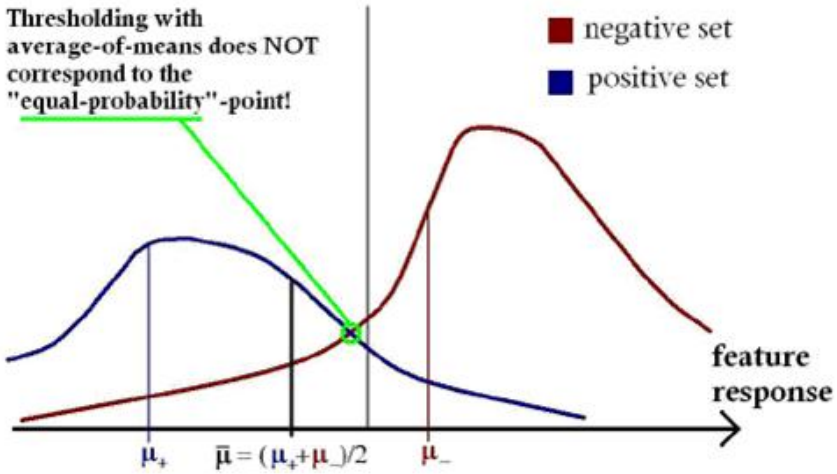

Figure 5.3. An example of how the distribution of feature values for a specific feature may look like over the set of all training samples.

As mentioned before a set of the features defines a set of weak classifiers. Now the set of weak classifiers (a binary set) is ready and it is time to employ AdaBoost to find a forming of a strong classifier through weak ones. The idea of combining weak classifiers to strong ones is a logical step that we as humans use during the decision making process. For example, to determine that someone is who they say they are we may ask them a series of questions, each one possibly no stronger than the prior, but when the person has answered all the questions we make a stronger decision about the validity of the persons identity [7].
The main factor in AdaBoost is the application of weight distributions. The process is started by an initial weight distribution, at each iteration after calling the weak classifiers the weight distribution will be updated if the classification is correct. Therefore, weak classifiers that manage to classify difficult sample images are given higher weighting in the final strong classifier.

In Figure 5.4 some of the initial rectangular features are shown selected by AdaBoost.

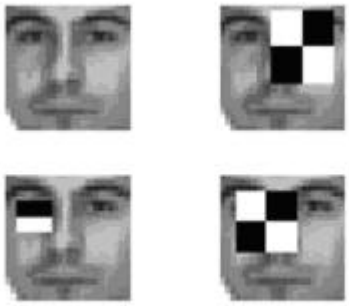

Figure 5.4. xample of some features selected by AdaBoost.

The algorithm of AdaBoost and how to define a strong classifier from weak ones is given below.

AdaBoost Algorithm

Input: Example images $(x_1, \ldots, x_n)$, and associated labels $(y_1, \ldots, y_n)$, where $y_i \in \{0, 1.$ Where $y_i = 0$ denotes a negative example and $y_i = 1$ a positive one. And, m is the number of negative examples and $l = n - m$ the number of positive examples.

Initialize: Set the n weights to:

$$\begin{cases} (2m)^{-1} & if\ y_i = 0, \\ (2l)^{-1} & if\ y_i = 1, \end{cases}$$

For $t = 1, \ldots, T$ do

1. Normalize the weights

$$\omega_{t,i} \leftarrow \frac{\omega_{t,i}}{\sum_{j=1}^{n} \omega_{t,j}}.$$

   So that is a probability distribution.

2. For each feature j train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to the $\omega_{t,i}$'s as $\epsilon_j = \sum_i \omega_{t,i} |h_j(x_i) - y_i|$.

3. Then, choose the classifier h_t as the h_j that gives the lowest error. Set e_t to that e_j.

4. Update the weights:

$$\omega_{t+1,i} = \omega_{t,i} \beta_t^{1-e_i},$$

   where $e_i = (1)0$ if example x_i is classified (in)correctly, and $\beta_i = \frac{i}{1-\epsilon_i}$.

End for

Output: A strong classifier defined by:

$$h_x = \begin{cases} 1 & if\ \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t, \\ 0 & otherwise, \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$.

5.2.2 A fast decision structure (The Casecade):

Since the time for classification is usually proportional to the number of evaluations or weak classifiers, increasing the speed of a classification task generally implies increasing the classification error which yields decreasing the accuracy. Viola & Jones proposed a method for both reducing the classification time and maintaining classifier robustness and accuracy through the use of a classifier cascade. The elegant key behind the cascade is based on a tree structure of classifiers. In the manner that in the early stages of the tree, classifiers are largely naive. As a positive sample progress through the cascade, assuming that the sample is indeed positively classified, then the process of classification will become finer, and the number of features that are evaluated will increase.

Indeed, in detection task in a large image a large majority of the sub-windows observed by the scanning classifier will be rejected and just a small regional area(s) in the image might be the target(s). Therefore, the generality of the first number of stages must be sufficiently high to reduce the number of these false positive sub-windows from processing into the higher stages of the cascade. The aim is to provide inference with the lowest possible false positives, and highest possible detection rate. The Figur 5.5 shows a cascade process.

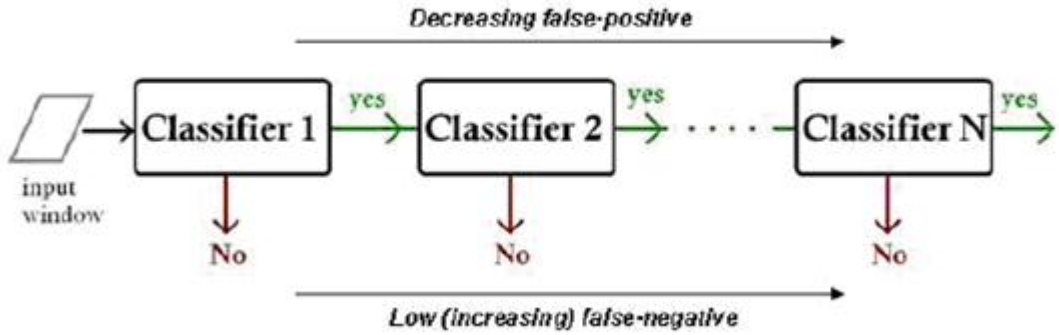

Figure 5.5. The attentional cascade using increasingly specialized ckassifiers

Viola & Jones show that given a trained classifier cascade, the false positive rate is the product of the all false positive rates in the chain. Based on this fact and deeper reasoning for the motivation of the cascade, they provide a general algorithm for the training process that the cascade must undertake in order to build its stages. The algorithm is shown in algorithm of Table 2. The minimum acceptable detection rate (d) and the maximum false positive rate (f) are required in the algorithm.

Casecade Algorithm:

Input: Allowed false positive rates f, detection rates d and the target false positive rate F_{target}. A set P of positive (faces) and of N negative examples (non-faces).

Initialize: Set $F_0 = 1.0, D_0 = 1.0, i = 0$.
While $F_i > F_{target}$ and $n_i < N$ do

- $i = i + 1$
- $n_i = 0; F_i = F_{i-1}$

  While $F_t > f_{i-1}$ do

  - $n_i = n_i + 1$
  - Use P and N to train with n_i features using AdaBoost.
  - Evaluate the current cascade classifier on the validation set to determine F_i and D_i.
  - Decrease the threshold of the ith classifier until the current cascade classifier has a detection rate of at least $d \times D_{i-1}$ (this also affects F_i).

  End while

- $N = 0$
- If $F_i > F_{target}$ then evaluate the current cascaded detector on the set of the non-face images and put any false detections into the set N.

End while

5.3 Faced Recognition based LBP

The result of operating an LBP operator over a whole face image is shown in the Figure 5.6. The LBP operator encodes only the occurrences of the micro-patterns without any indication about their locations. The uniform LBP with $P = 8$ and $R = 2$ is employed here.

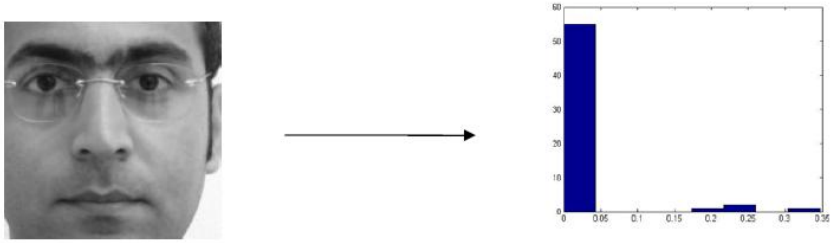

Figure 5.6. An uniform LBP(8,2) operator over a devided facial image.

For efficient face representation, the spatial information must be returned. For this purpose, the face image is divided into several regions (or blocks) from which the local binary pattern histograms are computed and concatenated into a single histogram (see Figure 5.7). In such a representation, the texture of facial regions is encoded by the LBP while the shape of the face is recovered by the concatenation of different local histograms.

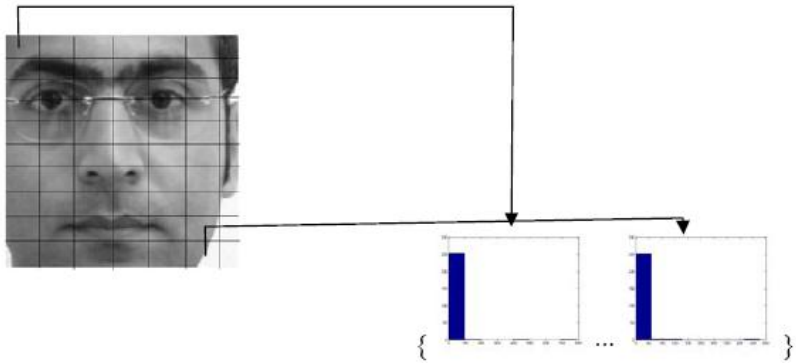

Figure 5.7. An uniform LBP(8,2) operator over a divided facial image.

To compare a target face histogram S to a model histogram M one can use the nearest neighbor classification scheme. One of the best dissimilarity metrics for histograms is the χ^2 (Chi-square):

$$\chi^2(S, M) = \sum_{i=0}^{l} \frac{(S_i - M_i)^2}{S_i + M_i},$$

where l is the length of the feature vector used to represent the face image. Note that the choice of the χ^2 measure is motivated by the experimental findings (Ahonen et al. 2004a) which show that χ^2 gives better results than other dissimilarity measures such as the histogram intersection $D(S, M)$ and Log-likelihood statistic $L(S, M)$.

$$D(S, M) = \sum_{i=0}^{l} \min(S_i, M_i)$$

$$L(S, M) = -\sum_{i=0}^{l} S_i \log M_i$$

There are other parameters that should be chosen to optimize the performance of the proposed representation. The first one is choosing the LBP operator. Choosing an operator that produces a large amount of different labels makes the histogram long and thus calculating the distances gets slow. Using a small number of labels makes the feature vector shorter, but also means losing more information. As a result, ideally, the aim is to find the LBP operator which gives a short feature histogram and captures all the information needed to discriminate between the faces. Another parameter is the division of the images into regions. A large number of small regions produces long feature vectors causing high memory consumption and slow classification, whereas using large regions causes more spatial information to be lost. The experimental research by (A. Hadid) is done with different LBP operators and window sizes. The mean recognition rates for the $LBP_{u2}(16, 2), LBP_{u2}(8, 2)$ and $LBP_{u2}(8, 1)$ as a function of the window size are plotted in Figure 5.8. The original 130×150 pixel image was divided into $k \times k$ windows, $k = 4, 5, \cdots, 11, 13, 16$ resulting in window sizes from 32×37 to 8×9. As expected, a larger window size causes a decreased recognition rate because of the loss of spatial information. The $LBP_{u2}(8, 2)$ operator in 18×21 pixel windows was selected since it is a good trade-off between recognition performance and feature vector length.

Furtheremore, according to psychological findings some facial features such as eyes play more important roles in face recognition than other features such as nose. Hence, when a facial image is divided into several regions we can expect that some of the regions carry more useful information that others in term of distinguishing between faces. Therefore, using different weights for the regions may improve the recognition rate. Choosing weights should be depended on the importance of the corresponding regions in recognition. For instance, since the eye regions are important for recognition, a high weight can be attributed to the corresponding regions. In this work the recognition method is based on non-weighted LBP approach.

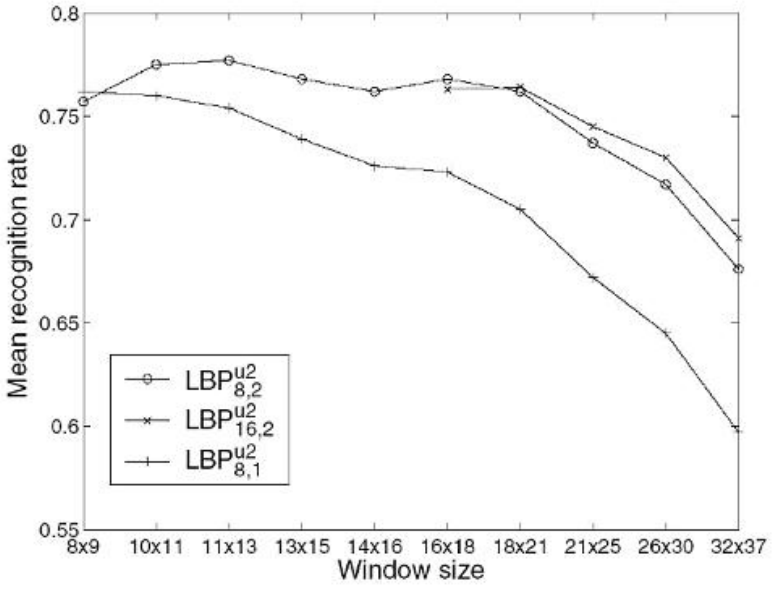

Figure 5.8. The mean recognition rate for three LBP operators as a function of the window. The five smallest windows were not tested using the $LBP_{u2}(16, 2)$ operator because of the high dimension of the feature vector that would have been produced.

Chapter 6

The Experiment Evaluation

6.1 Gaussian Mixture Model

In this section, some results are given from experimental evaluation. For fair performance evaluation of different skin color modelling methods, identical testing conditions are preferred. Unfortunately, many skin detection methods provide results on their own, publicly unavailable, databases. In this work, for evaluating the Gaussian mixture model we try to make an identical testing condition.

6.1.1 Skin color space

In this work, two very popular skin color spaces namely RGB and $YCrCb$ are considered. The system needs initial skin samples. This step is done using manually segmentation of our current face image database. For each skin color space, two identical skin samples with different sizes are employed, skin sample A and B with a size of 384×512 and 620×962, respectively. One way to define training and test databases of these two samples is to gather a random set of skin samples in which manner that 50 % of each skin sample is allocated randomly as the train database and the rest of the skin sample which has the same size is our test database. In fact, we do know that all the points in test database lie in the skin color region. We then challenge the Gaussian mixture model to determine the probability density of test database to lie into the skin region. The skin region in the color space obviously has been defined by estimating the parameters of the Gaussian mixture, using EM algorithm described before. Finally, by thresholding the probability density functions we will obtain a binary image containing 0 and 1. The corresponding threshold will be discussed and will be obtained experimentally.

6.1.2 Number of components

Choosing the correct number of components in the Mixture of Gaussians is very important. As mentioned before, the number of components must be greater than

2. It is obvious that a high number of components will yield to a time-consuming process or equal. Since we were trying to reduce the process time to approach a fast method, four different numbers of components have been chosen here. We studied the behavior of the system when the numbers of components are $2, 3, 4$ and finally 5.

6.1.3 Results

The results are given in the table below. Each experiment consists of the evaluation of each single number of components on each skin sample databases with two different space colors. Indeed, to have a fair performance evaluation the code ran five times for each evaluation and the average result has been considered. The best empirical threshold here is $10^{-}6$ which gives us the best performance of evaluation.

Sample A			Sample B		
Nr_Comp	RGB	YCrCb	Nr_Comp	RGB	YCrCb
2	78.4327	86.2628	2	96.5473	97.6038
3	85.8193	91.4176	3	96.3046	97.5287
4	88.2212	94.0479	4	96.4809	98.0265
5	90.1113	96.3597	5	97.3823	98.1599

Figure 6.1. The result of evaluation. Sample A and Sample B are the skin samples of size 384x512 and 620x962 respectively. Nr_Comp is the number of components in Gaussian Mixture Model.

In the case of choosing a color space concerning the Figure 6.1, the best choice might be $YCrCb$ when the number of components in Gaussian mixture is 5. But, as discussed in Chapter 2, color transformation would be required. Each image of size 640×480 pixels requires $(640x480 =)307200$ transformations. In order to avoid such a huge amount of heavy calculations, and since most cameras provide RGB images directly and with respect to our result a choice of 2 for number of components in Gaussian mixture model on RGB format is the best alternative in this thesis. The result shows a reliability of 96.5473 % which is convenient.

Furthermore, since skin color detection works as a pre-process step for our face detection, we do not want to loose any information of skin area in an image, when we minimize the input image and prepare a new input image consisting of skin areas for the face detection step.
Normally, in an image given by a camera in surveillance application the skin area might take only 15% of the whole image. In order to have better performance and faster method, we can speed up the system by using the result of skin detection step.

The Figure 6.2 shows the skin detection region using RGB color space with Gaussian Mixture model (number of components = 2).

Figure 6.2. Skin Detection using RGB color and Gaussian Mixture model with two components.

The original image (top left) has a size of 800×600 pixels. After implementing Gaussian Mixture with two components the binary image (top right) appears. The skin region will then be indicated from binary image. A rectangular cut of the skin region shown in bottom left. The output image from this step is the last image consisting of skin regions (bottom right) which has a size of 356×223 whereas almost 15% of the original image.

The Figure 6.3 shows skin detection using the same condition as previous example, on an image of two persons with a different background.

The original image is an image of $3MB$, (2048×1536 pixels). The rectangular cut of the skin region has a size of 817×1281 whereas almost 33% of the original image (for 2 persons).

6.2 Face Detection

The major part here is implementing AdaBoost and Casecade algorithms. As database 2166 face and 4032 non-faces are used to make the features. After running the *AdaBoost algorithm* a strong classifier has been obtained. The question here is how we can be sure that the obtained strong classifier is working well. For this issue we test it on the portion of test data. Once again the threshold is taken by using cross-over point. By challenging the algorithm we find the values for both

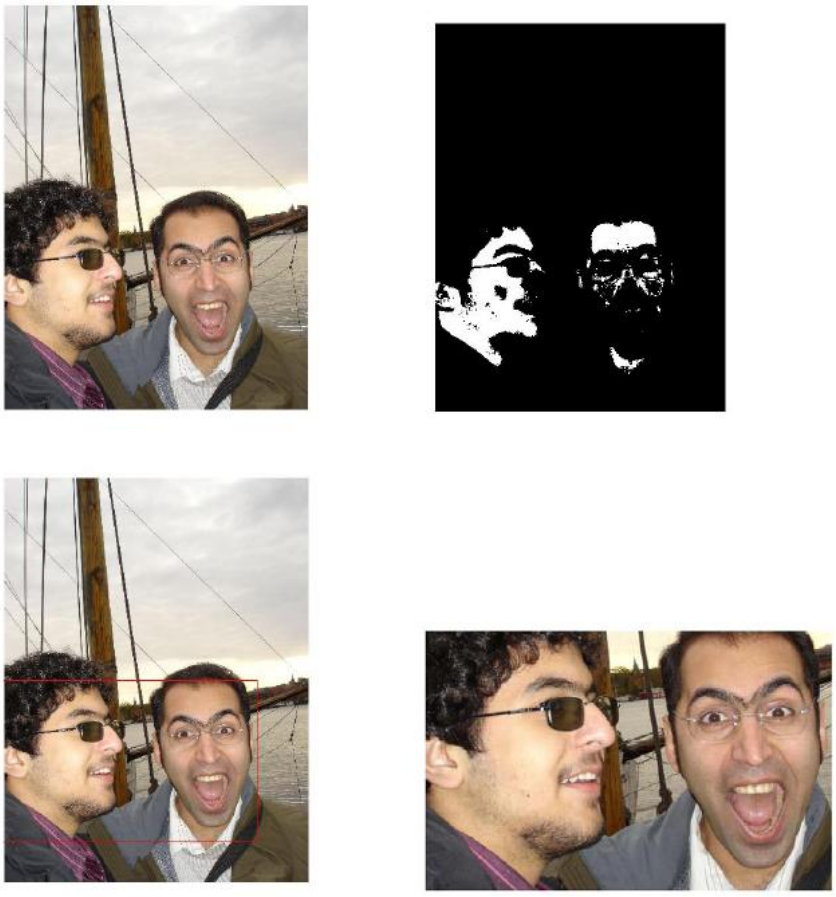

Figure 6.3. Skin Detection using RGB color and Gaussian Mixture model with two components on different background.

true-positive (TP) and true-negative (FN). For simplifying the definition of true-positive and true-negative the table below shows the relation between these two.

	Predicted class	
True class	Yes	No
Yes	True-Positive (TP)	False-Negative (FN)
No	False-Positive (FP)	True-Negative (TN)

Figure 6.4. The relationships between predicted class vs. true class.

Note that, the relationships between two classes can also be defined as $\frac{TP+FN}{|P|} = \frac{FP+TN}{|N|} = 1$, where $|p|$ and $|N|$ are the number of positive and neg-

ative samples, respectively. In other words, the "rates" for each parameter is $tp = \frac{TP}{|P|}, fn = \frac{FN}{|P|}, fp = \frac{FP}{|N|}$ and $tn = \frac{TN}{|N|}$. The number of true-positive and false-positive will depend on the threshold applied to the final strong classifier. In order to summarize this variation we introduce here *ROC-curve* (Receiver Operating Characteristic). The *ROC-curve* plots the fraction of true-positives against the fraction of false-positives as the threshold varies from $-\infty$ to ∞ . From this curve you can ascertain what loss in classifier specificity (false-positive rate) you will have to endure for a required accuracy (true-positive rate). In the Figure 6.5 the ROC-curve of our strong classifier for face detection is shown.

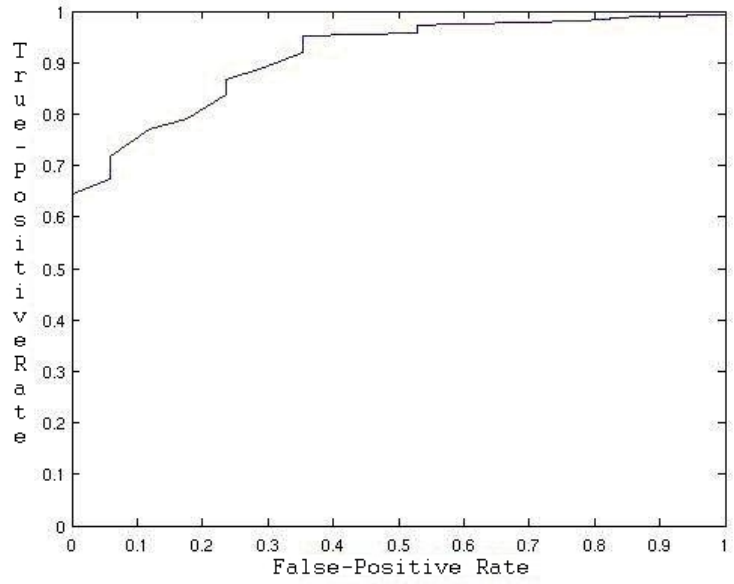

Figure 6.5. ROC-curve for the strong classifier in Face Detection.

Now when the ROC-curve is ready, the performance of the classifier can be evaluated.

The next step is implementing the *casecade algorithm* to obtain cascade, or in other words, the best features to have faster classification. For this issue, a Matlab function is employed to visualize the feature. The interesting part is to know which features were selected for the strong classifier. We let the code run for a limited number of features and the result is shown in Figure 6.6.

As mentioned before, in an input image a large majority of the sub-windows observed by the scanning classifier will be rejected and just a small regional area(s)

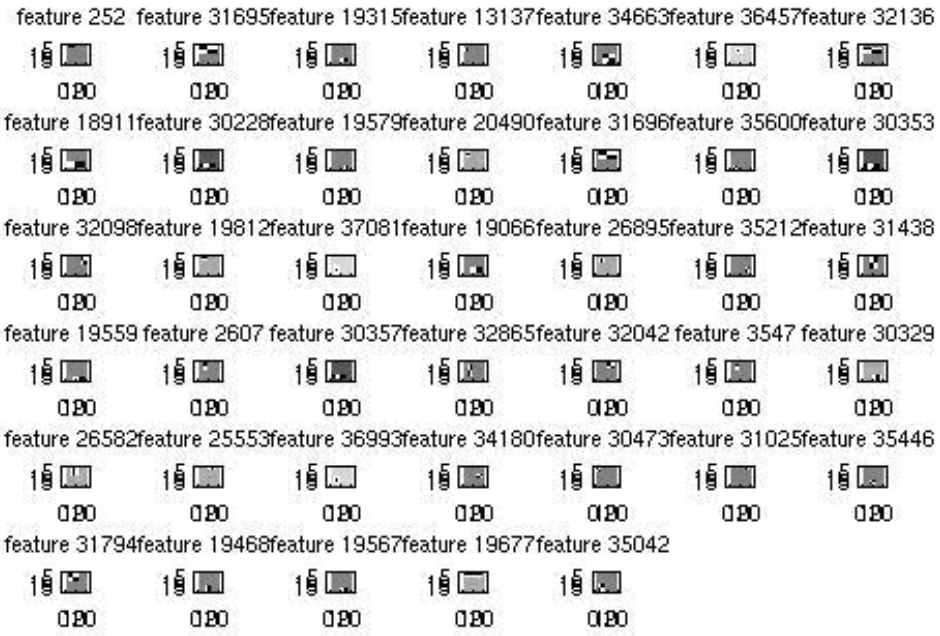

Figure 6.6. The chosen features in cascade system.

in the image might be the target(s). Therefore, we decided to use a cascade system to provide inference with the lowest possible false positives, and highest possible detection rate. In Figure 6.6, we see ranking of the selected features for the strong classifier. From the previous part in skin detection, the skin regions of an input image would be our input in this step (Face Detection step). Otherwise, in the case of no skin detection the entire image will be scanned in order to have a robust system. The skin detection evaluation was sufficiently convenient. We can now assume the output of the previous part is the input for our challenge in this step. As discussed before, about 85% of scanning the entire original image is saved here and we scan only 15% of the original image. The next figure shows how the face detection works.

The face detector system works very well for frontal facial images with respect to predictable achievement 0.99 and 0.01 for true-positive rate and false-negative rate, respectively.

Figure 6.7. In rectangular the skin region (left), the detected face (right).

6.3 Face Regocnition

In order to carry out an experiment design, we first need to have a dataset. First of all, we pick three various datasets with the following properties:

Collection I: This collection contains of 20 different persons and each person has 4 different poses. All poses have the same size which is 100x100. Figure 6.8 shows an example of a person in *collection I*.

Figure 6.8. person 1 in *Collection I*.

As it is shown in Figure 6.8, the face patch consists of the whole face with a small background region which might vary in each image. This collection is the smallest dataset in our test. The first test has been implemented on this dataset. The illumination and background vary. The facial image can be non-symmetric, e.g. the third image in Figure. 6.8 where the facial image is rotated.

Collection II: This collection contains 15 different persons and each person has 11 different poses. Similar to *collection I* all poses have the same size of 100x100. Figure 6.9 shows an example of a person in *collection II*.
This collection has a similar face region as *collection I*, and also same size of image. The distinction of this collection is that each person has a fair enough different poses. This collection is employed for the second test of LBP code. The illumination is challenging in this set. Various backgrounds, color and texture are considered

Figure 6.9. person 1 in *Colelction II.*

,e.g. the sixth image in Figure 6.9 has a completely different background texture. The face expression varies, sadness and happiness in each pose is clear.

Collection III: This collection contains 40 different persons and each person has 10 different poses. The point is some people have resemble blance to one another and this can be challenging in our test. Thus, the most test procedure is focus on this collection. All poses have the same size of 48×48. Figure 6.10 shows an example of a person in *collection III.*

Figure 6.10. person 1 in *Collection III.*

One of the major properties of this collection is the fair enough number of different poses; further the face region is larger than the two previous sets. Hence, the background challenge would be poor even if we still can see small various background color. This collection is employed for the third and final test of LBP code. The illumination is challenging in this set. Each face can have various expressions, just like *collection II.*

The LBP method is now employed to exploit the final histogram whereas a subset of dataset creates the train sample and the rest of dataset will be the ratio of test set (denoted by RT). Now we are able to calculate the LBP histogram of each test set and compare it with the final histogram which is a collection of LBP histogram in train set. This comparison is carried out by calculating Chi-square distance between the histogram of test image and each element of the final histograms of train set. The minimum value of all the Chi-square distances would be the dedicated distance for our recognition which means that a certain image in the training set has the closest distance and is the similar one to our test set. But, there is still that risk

to have a false recognition. In order to work out this issue, we decide a threshold and compare the minimum value with our threshold which is obtained by empirical experiment. Our experience shows that the threshold is dependent to the size of the image and even more on the region area of data set. Thus, on the following tests the value of the threshold is found experimentally.

The best performance is achieved by using sub-blocks of 18x21 pixels. In this work, in order to have a comparison, a size of 20x20 for sub-blocks is used. Hence, with respect to the image and sub-block sizes of the *Collection I* and *II* 's images will be divided to 25(5x5) sub-blocks, and also 4(2x2) sub-blocks for *Colletion III*'s images. Since in LBP calculation, there is no need to survey the whole image, and at *Collection III* the images of size 48x48 must be studied, therefore a sub image of each image is considered namely a sub-image with size of 40x40 which can be divided onto 4 sub-blocks. It is clear that the same manner should be used both for train and test images of the same size.

6.3.1 Results

A convincing result is shown in previous work [15] for the recognition rates of the weighted and non-weighted approaches. A comparison between the LBP based method and PCA, EBGM and BIC methods is shown in Figure 6.11. The databases used in this experiment is FERET database which contains five train sets namely fa, fb, fc, $dupI$ and $dupII$.

- The fa set, used as a gallery set, contains frontal images of 1196 people.
- The fb set (1195 images), in which the subjects were asked for an alternative facial expression than in the f a photograph.
- The fc set (194 images), in which the photos were taken under different lighting onditions.
- The $dupI$ set (722 images), in which the photos were taken later in time.
- The $dupII$ set (234 images), which is a subset of the $dupI$ set containing those images that were taken at least a year after the corresponding gallery image.

In [15], the author claims that the LBP based method outperforms other approaches on all probe sets. For instance, the method achieved a recognition rate of 97% in the case of recognizing faces under different facial expressions (fb set), while the best performance among the tested methods did not exceed 90%. Under different lighting conditions (fc set), the LBP based approach has also achieved the best performance with a recognition rate of 79% against 65%, 37% and 42% for PCA, BIC and $EBGM$, respectively. The relatively poor results on the fc set confirm that illumination change is still a challenge to face recognition. Additionally, recognizing duplicate faces (when the photos are taken later in time) is another

fig:AbdResult[htbp]

	fb	*fc*	*dupI*	*dupII*
Weighted LBP	0.97	0.79	0.66	0.64
Nonweighted LBP	0.93	0.51	0.61	0.54
PCA, MahCosine	0.85	0.65	0.44	0.22
BIC, BayesianMAP	0.82	0.37	0.52	0.32
EBGM	0.90	0.42	0.46	0.24

Figure 6.11. The recognition rates of the LBP and comparison algorithms for the FERET probe sets[15].

challenge, although the proposed method performed better than the others.

An evaluation experiment is done in this work. The LBP based approach on our datasets has also achieved the best performance with a recognition rate of 95%. Each experiment consists of the following parameters:

- *Database*: The database using here is *collections I* to *III*, where described before.
- *Threshold*: In order to avoid false recognition a proper threshold will be used. The value of threshold has been obtained experimentally.
- *RT*: The ratio of test set.

The results show the reliability of recognition process. The results consists of:

- *TR*: The True Recognition rate. (The proportional of correct recognition to the number of test data.)
- *FR*: The False Recognition rate.
- *NR*: The Non Recognition rate.

It is worth mentioning that finding a proper threshold plays an important role in recognition rate, choosing a threshold between 0.03 and 0.05 (in *collection III*) may yield less *NR* and more *FR*. The issue is the similarities of different persons in the database.

6.4 Overall performance

In this chapter, the overall performance of our face detection is described. The system can be used as an access control system, a surveillance system and even for

RT	DataBase collection	Threshold	Nr. sub-blocks	T R	F R	N R
25%	I	0.8	5x5	74%	3%	22%
10%	II	0.9	5x5	91%	2.5%	6.5%
10%	III	0.1	2x2	95%	0	4%
20%	III	0.1	2x2	93%	0	7%
30%	III	0.1	2x2	90%	0	10%
40%	III	0.1	2x2	85%	0	15%
50%	III	0.1	2x2	82%	0	18%
80%	III	0.1	2x2	69%	0	31%

Figure 6.12. The result of The LBP based approach on our datasets.

understanding human actions. For this issue, indoor environment is considered with limited illumination. First, a set of permitted persons to the system is collected, training face images in then extracted from the set and stored as our training data in face recognition. A camera is then set (indoor environment) to capture our test data. Each frame is analysed, according to the process proposed in Section 5.1, the skin region is selected by using Gaussian mixture model with a reliability of 96.5473%. The segmented skin region is the output of the next step, namely face detection step, otherwise the whole image have to been sent to the next step. According to our experience a successful segmented skin region normally includes 15% of a common image for our purpose. Thus, we can hopefully optimize 85% of the image size. The face detection method described in Section 5.2 is now ready to be utilized for the skin region. The resulting face regions must have different sizes, depending on the distance of camera. A closer face to camera has obviously larger face region. Thus, we should be using different face region sizes to have better performance. For this issue, a comparison of system performance is done by using a simple webcam with 640x480 sensor resolution. Two sizes are finally chosen, even more than two makes the code slower. Each single region is then processed by Local Binary Patterns to obtain a sample histogram. A comparison of sample histogram and feature database gives a matching of the test image. It is clear that a proper feature database will yield to a better performance in system. Figure 6.13 shows the different parts of the system.

In this work, to build the system, five persons are collected. Then, 10 different facial poses of each person have been stored as feature databases. Figure 6.14 shows some examples of images from the face database.

To build the result first, we extracted a face model from each training sequence and used it as our test image. Figure 6.15 shows the recognition results.

Referring back to Figure 6.2, where the color skin detection has been challenged by an image with different background and two faces, one is a frontal facial image and another one is a profile facial image. The final system is tested on the skin region result from Figure 6.2. The system detects both facial and profile faces and

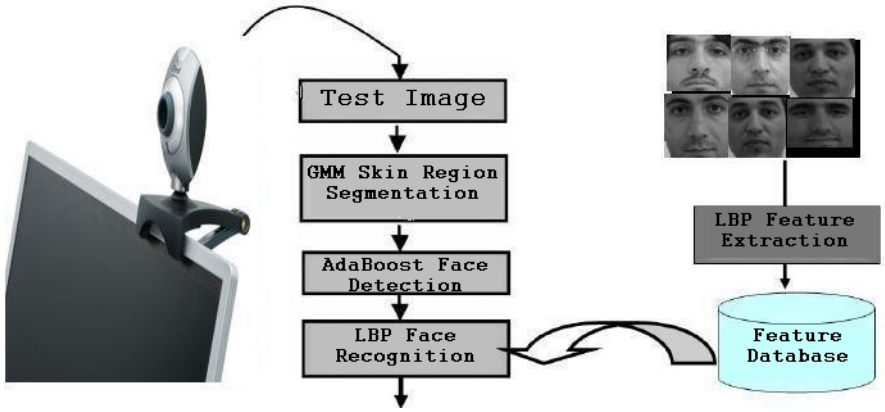

Figure 6.13. The different part of the system

Figure 6.14. Examples of images from the face database considered in the access control based on Face Recognition.

recognizes only the face in database namely the frontal facial image. It is worthy of mentionning that the profile facial image is not in the database, thus the system does not recognize the person. Figure 6.16 clears the result.

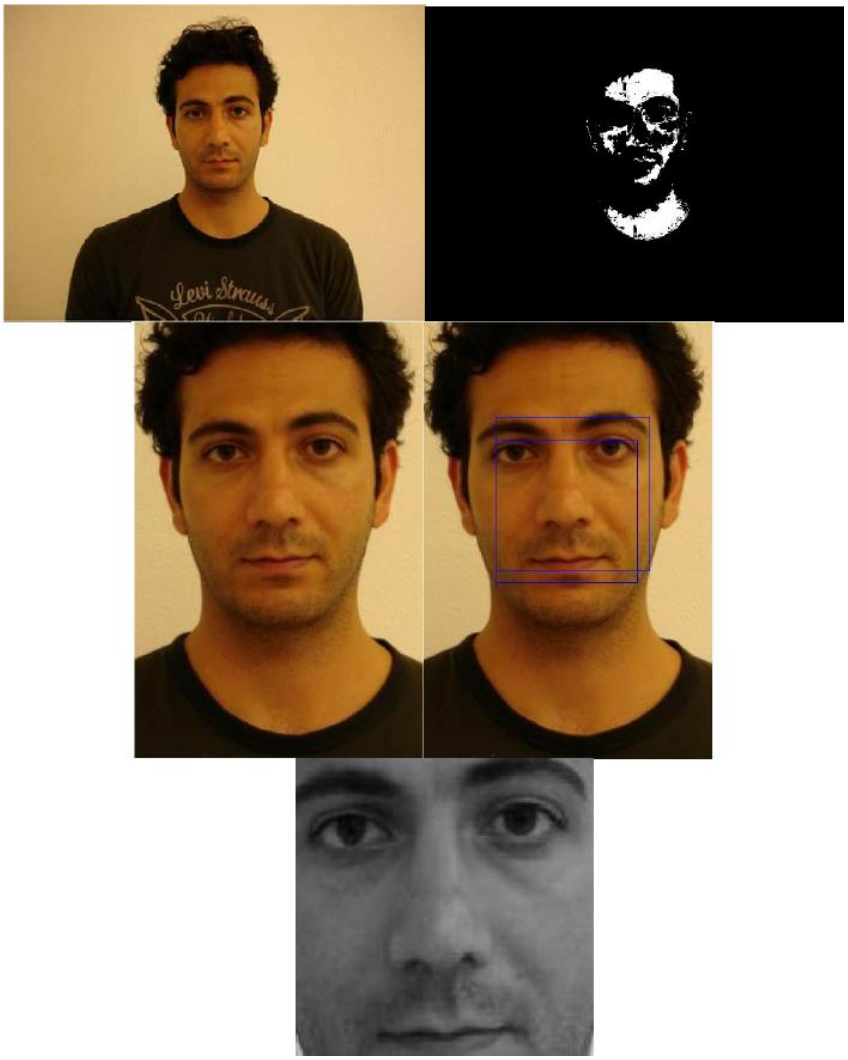

Figure 6.15. An example of system performance , original image (top-left), skin detection region (top-right), skin segmentation (middle-left), face detection from skin segmentation region (middle-right), face recognition (bottom)

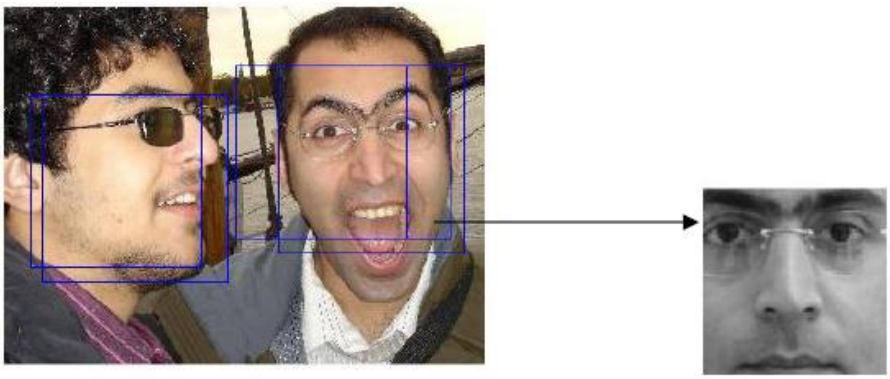

Figure 6.16. The result of Face Detection(left) and recognition(right) in an image of more than one facial images

Chapter 7

Conclusions and Future Works

7.1 Conclusions

We presented a system using face recognition for different applications. We introduced a pre-processing scheme, namely skin detection, to speed up the systemt. The proposed method is more adequate for indoor scenarios where the illumination and backgrounds might be under control. A demo is made for real-time processing, based on Matlab code.

The system recognizes some faces much easier than others. Indeed, some parameters of the system have been set by default and thus are not optimal. The effect of illumination on skin detection step is considerable. For instance, when using more than one camera with different sensor resolution the system must be adjusted with new cameras. The other difficulty is obtaining parameters by empirical experiments.

The implemented algorithms seem to work well in indoor environment relatively reliable with proper speed. Although the implementation is based on Matlab code, the system can run on real-time.

7.2 Future work

Since face detection and recognition is still an unsolved issue, approaching new conceivable methods have been always interesting. In this work, three different steps of proposed face recognition system is discussed. In the skin detection step, a fix threshold is used and the system should be calibrated with using of new camera by modifying the threshold in corresponding part. Furthermore the skin sample should be fed manually. One may suppose an adaptive method by updating Gaussian mixture parameter at each iteration for each test image. New parameters can adjust the system for each test image.
In face detection, the features are extracted from only frontal facial faces. The system can also detect non frontal facial partially, one may approve a robust face detector by using also non-facial faces. The focus in this work is to recognize facial images, thus non-facial faces have not been considered.

In face recognition, the system can be modified by having an online learning system. The learning system in this work is based on extracting face recognition features from manually captured frontal face images (see Figure 6.14). One may let a camera capture a video scene and then carry the learning process out from it.
Additionally, Matlab program is not an optimized software for having a real-time performance. Hence, the best performance of such system is expected from another language program such as C++. The aim here has been implementing the proposed algorithms in order to have a face recognition system. Thus, a proper improvement of the system can be recoding the algorithm in a powerful programming language such as C++.

List of Figures

Bibliography

[1] Kapur, Jay P. Face Detection in Color Images. University of Washington Department of Electrical Engineering 1997.

[2] Menser, B.; Muller, F. Image Processing and Its Applications, 1999. Seventh International Conference on (Conf. Publ. No. 465) Volume 2, Issue , 1999 Page(s):620 - 624 vol.2

[3] Jesorsky Oliver, Kirchberg J, and Frischholz Robert W. In Proc. Third International Conference on Audio- and Video-based Biometric Person Authentication, Springer, Lecture Notes in Computer Science, LNCS-2091, pp. 9095, Halmstad, Sweden, 68 June 2001.

[4] Goldberg David E. Genetic Algorithms in Search, Optimization and Machine Learning. 1989, isbn:0201157675, Addison-Wesley Longman Publishing Co., Inc.Boston, MA, USA.

[5] International ECCV 2002 Workshop on Biometric Authentication, Springer, Lecture Notes in Computer Science, LNCS-2359, pp. 103111, Copenhagen, Denmark, June 2002.

[6] M.H. Yang, N. Ahuja, Gaussian Mixture Model for Human Skin Color and Its Application in Image and Video Databases, Proc. of the SPIE, vol. 3656: Conf. on Storage and Retrieval for Image and Video Databases (SPIE 99), pp. 458-466, San Jose, Jan., 1999

[7] Vezhnevets V., Sazonov V., Andreeva A., "A Survey on Pixel-Based Skin Color Detection Techniques". Proc. Graphicon-2003, pp. 85-92, Moscow, Russia, September 2003.

[8] Jesus Ignacio Buendo Monge " Hand Gesture Recognition for Human-Robot Interaction". Master of Science thesis KTH. Sweden 2006.

[9] M.Störring " Computer vision and human skin color" Ph.D. dissertation, Aalborg University, Denmark, 2004.

[10] Jay P. Kapur,EE499 Capstone Design Project Spring 1997,University of Washington Department of Electrical Engineering. http://www.geocities.com/jaykapur/face.html

[11] Babak Rasolzadeh. Image Based Recognition and Classification.KTH. Sweden 2006.

[12] Timo Ahonen. Face Description with Local Binary Patterns: Application to Face Recognition. Oulu Finland 2005.

[13] B. Rasolzadeh, L. Petersson and N. Pettersson, Response Binning: Improved Weak Classifiers for Boosting. IEEE Intelligent Vehicles Symposium (IV2006), Tokyo, Japan, June 2006.

[14] Abdenour Hadid, Face Description with Local Binary Patterns: Application to Face RecognitionLearning and Recognizing faces: From still images to video sequences, Oulu Finland 2005.

[15] MENSER, B., AND WIEN, M. 2000. Segmentation and tracking of facial regions in color image sequences. In Proc. SPIE Visual Communications and Image Processing 2000, 731740.

[16] TERRILLON, J.-C., SHIRAZI, M. N., FUKAMACHI, H., AND AKAMATSU, S. 2000. Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. In Proc. of the International Conference on Face and Gesture Recognition, 5461.

[17] YANG, M.-H., AND AHUJA, N. 1998. Detecting human faces in color images. In International Conference on Image Processing (ICIP), vol. 1, 127130.

[18] JONES, M. J., AND REHG, J. M. 1999. Statistical color models with application to skin detection. In Proc. of the CVPR 99, vol. 1, 274280.

[19] YANG, M., AND AHUJA, N. 1999. Gaussian mixture model for human skin color and its application in image and video databases. In Proc. of the SPIE: Conf. on Storage and Retrieval for Image and Video Databases (SPIE 99), vol. 3656, 458466.

[20] SKARBEK, W., AND KOSCHAN, A. 1994. Colour image segmentation a survey . Tech. rep., Institute for Technical Informatics, Technical University of Berlin, October.

[21] POYNTON, C. A. 1995. Frequently asked questions about colour. In ftp://www.inforamp.net/pub/users/poynton/doc/colour/ColorFAQ.ps.gz.

[22] R.A. Render and H.F. Walker, " Mixture densities, maximum likelihood and the EM algorithm," SIAM review Wiley, New York, 1985.

TRITA-CSC-E 2007:139
ISRN-KTH/CSC/E--07/139--SE
ISSN-1653-5715

www.kth.se