1-1-2003

Automatic human face detection in color images

Son Lam Phung
Edith Cowan University

AUTOMATIC HUMAN FACE DETECTION IN COLOR IMAGES

by

Son Lam Phung
B.Eng. (First Class Honours)

Thesis submitted for the degree of
Doctor of Philosophy (Engineering)

School of Engineering and Mathematics
Faculty of Computing, Health and Science
Edith Cowan University
Perth
WESTERN AUSTRALIA

Supervisor: Associate Professor Abdesselam Bouzerdoum
Associate Supervisor: Dr. Douglas Chai

USE OF THESIS

The Use of Thesis statement is not included in this version of the thesis.

ABSTRACT

Automatic human face detection in digital image has been an active area of research over the past decade. Amor ; its numerous applications, face detection plays a key role in face recognition system for biometric personal identification, face tracking for intelligent human computer interface (HCI), and face segmentation for object-based video coding. Despite significant progress in the field in recent years, detecting human faces in unconstrained and complex images remains a challenging problem in computer vision. An automatic system that possesses a similar capability as the human vision system in detecting faces is still a far-reaching goal. This thesis focuses on the problem of detecting human faces in color images. Although many early face detection algorithms were designed to work on gray-scale images, strong evidence exists to suggest that face detection can be done more efficiently by taking into account color characteristics of the human face.

In this thesis, we present a complete and systematic face detection algorithm that combines the strengths of both analytic and holistic approaches to face detection. The algorithm is developed to detect quasi-frontal faces in complex color images. This face class, which represents typical detection scenarios in most practical applications of face detection, covers a wide range of face poses including all in-plane rotations and some out-of-plane rotations. The algorithm is organized into a number of cascading stages including skin region segmentation, face candidate selection, and face verification. In each of these stages, various visual cues are utilized to narrow the search space for faces. In this thesis, we present a comprehensive analysis of skin detection using color pixel classification, and the effects of factors such as the color space, color classification algorithm on segmentation performance. We also propose a novel and efficient face candidate selection technique that is based on color-based eye region detection and a geometric face model. This candidate selection technique eliminates the computation-intensive step of window scanning often employed in holistic face detection, and simplifies the task of detecting rotated faces.

Besides various heuristic techniques for face candidate verification, we develop face/nonface classifiers based on the naive Bayesian model, and investigate three feature extraction schemes, namely intensity, projection on face subspace and edge-based. Techniques for improving face/nonface classification are also proposed,

including bootstrapping, classifier combination and using contextual information. On a test set of face and nonface patterns, the combination of three Bayesian classifiers has a correct detection rate of 98.6% at a false positive rate of 10%.

Extensive testing results have shown that the proposed face detector achieves good performance in terms of both detection rate and alignment between the detected faces and the true faces. On a test set of 200 images containing 231 faces taken from the ECU face detection database, the proposed face detector has a correct detection rate of 90.04% and makes 10 false detections. We have found that the proposed face detector is more robust in detecting in-plane rotated faces, compared to existing face detectors.

DECLARATION

I certify that this thesis does not, to the best of my knowledge and belief:

(i) incorporate without acknowledgement any material previously submitted for a degree or diploma in any institution of higher education;

(ii) contain any material previously published or written by another person except where due reference is made in the text; or

(iii) contain any defamatory material.

Signature Date 03 Oct 2003

ACKNOWLEDGMENTS

I wish to express my deepest appreciation to Associate Professor Abdesselam Bouzerdoum. I was fortunate to have him as my mentor, and benefited enormously from his academic expertise, his inspiring guidance and his constant support. The many discussions with him helped me greatly in shaping the research ideas presented in this thesis.

I'm also deeply grateful to Dr. Douglas Chai, my associate supervisor, for many valuable advices. I learnt a great deal from his insights and experience.

I'm glad to be assisted by many people during the course of this project. Thanks go to Mr. Tin Yuen Ke for sharing with me the painstaking months of creating the ECU face detection database; Mr. Quoc Huy Nguyen for creating artistic illustrations; my colleagues in the Visual Information Processing research group for participating in some of the experiments.

This academic endeavor was also a personal journey, along which I felt fortunate to have met many friends. Special thanks go to Slavko Nikolic for his enormous helps, and Dr. Farid Boussaid for the motivating conversations. I'm also very grateful to Dr. Amine Bermak for his support, Steve Tivive for sharing memorable time as students at ECU, and Neil Nguyen for being so brotherly.

My gratitude also goes to my family for their understanding and encouragements.

And finally, thank you, Widya, for your love and for continuingly asking if I had finished "the chapter".

Son Lam Phung

July 2003

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AdaBoost	Adaptive Boosting
ANN	Artificial Neural Network
cdf	cumulative density function
CDR	Correct Detection Rate
CIE	Commission Internationale de l'Eclairage (International Commission on Illumination)
CNN	Convolutional Neural Network
CR	Classification Rate
EM algorithm	Expectation-Maximization algorithm
FDR	False Detection Rate/ False Alarm Rate
FLD	Fisher Linear Discriminant
HCI	Human-Computer Interface
HSV	Hue-Saturation-Value color space
ICA	Independent Component Analysis
JPEG	Joint Photographic Experts Group
MAP	Maximum *a posteriori* classifier
MLP	Multilayer Perceptron
MPEG	Motion Picture Experts Group
MRC	Maximal Rejection Classifier
NTSC	National Television System Committee
PAL	Phase Alternation Line
PCA	Principal Component Analysis
pdf	probability density function
PFS	Projection onto Face Subspace
ROC curve	Receiver Operating Characteristic curve
RGB	Red-Green-Blue color space
SECAM	Sequentiel Couleur Avec Mémoire (Sequential Color with Memory)
SOM	Self-Organizing Map
SVM	Support Vector Machine
UCS	Uniform Color Space

Chapter 1

Introduction

The human face, a vital means of conveying a person's identity, always receives a special attention in our social interactions. Through some complex underlying mechanisms, which are transparent to us, we are capable of learning a myriad of useful information based on the visual cues on a person's face. The list includes the identity, and, to a large extent, emotional state, gender, ethnic origin and age of the person. For decades, understanding this capability of ours has drawn research interests from diverse fields including psychophysics, neurosciences, computer science and engineering. It is not hard to imagine the vast applications that are possible if computer systems can be constructed to reliably infer a person's identity, emotional state, and gender from the person's facial image. However, such systems need, in the first place, an algorithm to detect and locate the face in a visual scene.

This thesis focuses on the task of automatic human face detection in digital images. Automatic face detection has dual purposes: it asserts if human faces are present in a digital image, and if so it pinpoints exactly their positions and spatial extents. We present in this thesis a complete algorithm for face detection in color images. This introductory chapter is organized into three main parts. The motivation and significance of automatic human face detection, its major challenges and practical applications are presented in Section 1.1. The research objectives are defined in Section 1.2. Finally, the organization of the thesis is outlined in Section 1.3.

1.1 Motivation and Significance

Face detection is an important task in the broader research field of automated facial image understanding, which aims to mimic in computers some of our cognitive capability in interpreting facial images. The list of common research topics in automated understanding of facial images includes:

- **face detection**: detecting and locating faces in unconstrained images;
- **face recognition**: differentiating the face images of a person from the face images of other people, see [17] for a review;
- **facial expression analysis**: identifying the emotional state of a person by analyzing the person's facial expressions, see [76] for a review of the state-of-the-art;
- **lip reading**: inferring a spoken message by a visual means of tracking the lip movement;
- **face tracking**: locating and following the face as it moves around in a visual scene.

Research in face detection is motivated both by its scientific challenges and by its practical applications. In the two subsections below, we will discuss in turn these two motivations.

1.1.1 Challenges in Automatic Face Detection

Automatic face detection inevitably involves two important concepts: "face" and "nonface". The concept of "face" encompasses all human faces in their fullest variations, and the concept of "nonface" representing all other objects. Automatic face detection essentially means the identification of visual attributes that distinguish the face from all other objects. Therefore, face detection is very much different from face recognition, which requires the identification of visual attributes that differentiate face images of a person from face images of all other people. In face detection, we search for common attributes among the face images of all people, whereas in face recognition, we seek attributes that are ideally different for different people, yet similar for faces of the same person regardless of the facial expression or other intra-personal variations.

In finding visual attributes that are common for all human faces, yet distinct from all other objects, we need to realize that there exist a vast number of factors that can influence the appearance of the human face in a 2-D image. Because of such factors, there are wide variations in the face pattern. These variations can be divided into two groups: (i) intrinsic variations that are due to intrinsic properties of the human face; (ii) extrinsic variations that are caused by external factors such as the imaging condition.

A. Intrinsic Face Variations

Intra-personal variations refer to the differences in the face appearance of the same person. The human face is a non-rigid object that possesses a formidable degree of shape variability caused by facial muscles. Many facial features such as the eyes, eyebrows, lips and cheeks usually change from their neutral shapes as a result of different facial actions. Besides facial expression, aging and the presence/absence of some facial features such as beard and moustache can also cause intra-personal variations. *Inter-personal* variations refer to the differences in the face appearance between different people. People of different ethnic origins often differ in skin colors or general facial features. Even within the same ethnic group, very few people have faces that are exactly alike (apart from identical twins). Furthermore, there are facial differences between males and females.

B. Extrinsic Face Variations

The external factors that cause extrinsic variations in the face patterns include face pose, face size, and lighting condition. Because the face is a 3-D object, its 2-D projection image is highly dependent on the orientation between the face and the camera, i.e. the face pose. The face contour varies a great deal under different views (e.g. frontal upright view or profile view). Some facial features are not visible under certain views. The face size in an arbitrary input image is usually not known, and this forces some face detection algorithms to search the input image at multiple scales. Lighting conditions on the scene coupled with the optical properties of the capturing devices add another dimension of complexity to face detection. For example, the colors of the facial region

in an image vary depending on the spectrum of the illuminating light and the lighting compensation scheme implemented in the camera hardware.

Although the image background has no effect on the face appearance, its content and complexity can influence significantly the outcome of face detection. This is understandable because face detection aims to locate faces among a plethora of other objects in the image background whose appearance can make face/nonface discrimination easy or difficult.

1.1.2 Applications of Automatic Face Detection

A. Face Recognition

Face recognition, which is the identification of humans from unique characteristics of their faces, is traditionally an important application of face detection. There are two main different aspects of face recognition: face verification and face identification. In *face verification*, a user claims an identity and presents a live face; the system determines if the live face matches with the one stored in the system for the claimed identity. In *face identification*, a live face is presented and the system locates the identity that matches the live face most closely. Face verification involves one-to-one comparison, whereas face identification needs one-to-many comparison. Nevertheless, a face recognition system typically compares a probe face with face(s) stored in a database. The system accepts a facial image as the input, which contains a face in a controlled and relatively simple background scene. However, the precise position of the face in the image is usually not known, and there are certain variations in the face pose. Therefore, a fully automatic face recognition system in practice requires a face detection stage, in which the face is localized in the input facial image, and then normalized to a canonical form accepted by the recognition algorithm. The canonical form is usually a fully frontal upright face where only the part relevant to face comparison is kept. The face detection stage is also essential in non-invasive video surveillance applications where less can be assumed about the face size and location in the captured images.

B. Face Segmentation for Region-Of-Interest Coding

In a region-of-interest image/video coding scheme, higher bitrates are allocated for image regions, which are deemed as important to the viewers. It has been found that in videophone or teleconferencing applications, the viewers tend to pay more attention to the face of the speaker. Therefore, facial regions in the video frames can be extracted by a face segmentation algorithm, and then coded at higher fidelity compared to the image background [14]. This way, even if the overall bitrate is the same, viewers can perceive significant improvements in the image quality.

C. Perceptual Human-Computer Interface

If the face can be detected in a scene and its movement can be tracked in real-time, many applications are possible, especially in the area of perceptual human-computer interface. A perceptual human computer interface (HCI) employs communication channels that humans routinely use to interact with each other. One of such channels is vision where the computer actively tracks, understands, and responds to the pose, gesture or facial expression of the user. For example, Xu and Sugimoto [129] developed a real-time face tracking system, in which the face of the user is kept at the center of the captured video through software manipulation of a pan-tilt-zoom controllable camera. Such a system is useful in videoconferencing or video surveillance applications where the person can move freely in front of the camera. Bradski [6] presented an experimental computer interface based on face tracking that allows the user to play a 3D graphics game (Quake 2) by moving his/her head. Using this interface, the leftward, rightward, backward, and forward head movements trigger the virtual player in the game to slide leftwards, rightwards, backwards, and forwards, respectively; roll leftward and rightward head movements trigger the virtual player to look left and right, respectively; up and down head movements signal the virtual player to shoot. Silva *et al.* [102] built a vision-based interface in which the computer mouse is moved according to the movement of the user's nose tip, and a mouse click can be entered by a mouth opening action.

D. Multimedia Content Management

Face detection has many applications in the management of multimedia contents that have become ubiquitous on the Web and in digital archives. Text-based descriptions are usually insufficient for such media-rich contents, which integrate text, images and video. Satoh *et al.* [98] developed the Name-It system, which aims to associate automatically faces in a video clip (TV news) with the correct person names. In their system, faces are located using the neural-network face detector developed by Rowley [90], names are extracted by analyzing both the audio component and the text caption of the video clip; association between faces and names are based on their temporal coincidence. With such a system, it is possible to enter a person's name and retrieve video clips relating to the person. Lienhart *et al.* [62] presented a video abstracting tool for automatic creation of movie trailers for feature films, in which face detection and face recognition are utilized to find video frames containing close-up shots of the main actors/actresses. These video frames are grouped to form part of the movie trailer.

1.2 Research Objectives

In this thesis, we present a complete and systematic face detection algorithm that combines the strengths of both analytic and holistic approaches to face detection. The algorithm is developed to detect quasi-frontal faces in complex color images. This face class, which represents typical detection scenarios in most practical applications of face detection, covers a wide range of face poses including all in-plane rotations and some out-of-plane rotations. In our approach, color characteristics of the human face are extensively used to make its detection more efficient. The proposed algorithm is organized into a number of cascading stages including skin segmentation, face candidate selection, and face verification. At each stage along the cascade, the search space for face is narrowed down, and this is a central theme of the work presented in this thesis.

The main objectives of this thesis are to:

1. understand the problem of automatic face detection, its definition and significance, its challenges and practical applications;

2. provide a comprehensive review of the state-of-the-art in face detection.
3. formulate an efficient and robust strategy to detect in-plane rotated faces in color images;
4. investigate and compare techniques for modeling object colors and the approach of object detection using color pixel classification;
5. develop image segmentation techniques that can be used to enhance the pixel-wise approach to skin segmentation;
6. study face candidate selection schemes that are suitable for scale-invariant and rotation-invariant face detection;
7. investigate pattern classification techniques for differentiating between face and nonface patterns;
8. create a comprehensive database that supports the construction and evaluation of color-based as well as gray-scale-based approaches to face detection;
9. develop a complete and systematic face detector that integrates the above system components, and perform detailed analysis of the face detector; and
10. discuss possible applications of the face detector.

1.3 Thesis Organization

The research objectives that are outlined in the previous section are systematically addressed in eight chapters and the appendix. An overview of the thesis chapters is given below:

1. The current chapter is a general introduction of the thesis. The chapter states the research topic, its motivation and significance. In the current chapter, research objectives are defined; the outline of the thesis is given; main challenges and practical applications of face detection are discussed. [*Objective 1*]

2. Chapter 2 presents a comprehensive review of the state-of-the-art of face detection as evidenced in published literature. We systematically investigate two major existing approaches to face detection, namely analytic and holistic. Face detection is a very rapidly changing field with new and novel approaches being published nearly every six

months. In this review, we focus on new developments in the field within the last few years. [*Objective 2*]

3. Chapter 3 addresses the approach of skin detection using color pixel classification. In this chapter, we attempt to answer two important questions: (i) which classification algorithm and which color space give the best skin detection performance in terms of accuracy, speed and memory load? (ii) how effective is the approach of skin detection using pixel color for general images? Although skin detection is routinely used in face detection algorithms, these two questions have not been adequately addressed in the existing literature. We report in this chapter several important findings regarding color-based skin detection. [*Objective 4*]

4. Chapter 4 presents our proposed techniques to enhance the result of pixel-wise skin detection. These techniques take into account texture characteristics of the human skin to remove false detections, and prepare segmented skin regions for the next stage of face detection. [*Objective 5*]

5. Chapter 5 focuses on the problem of face candidate selection. The motivation for the work described in this chapter is twofold: (i) to eliminate the computational-intensive and brute-force approach of window scanning commonly adopted in holistic face detection; (ii) to address a deficiency in existing skin-color-based approaches to face detection, which treat each segmented skin region in its entirety as a face candidate. The face candidate selection technique presented in this chapter attempts to solve both the scale and in-plane rotation problems in face detection. [*Objective 6*]

6. Chapter 6 addresses the task of classifying face/nonface patterns, which is a difficult problem in its own right. It is fair to say that face/nonface classification has been widely used as the "test-bed" for machine learning techniques, such as new neural network architectures. Here, we present a statistical pattern classification technique based on the naive Bayesian model. Face/nonface classifiers developed in this chapter are used to perform the final verification of the face candidates formed in the previous stage of face detection. [*Objective 7*]

7. Chapter 7 presents a complete face detection algorithm that integrates the components described in Chapters 3 to 6 of the thesis. We report a comprehensive analysis of the face detector. Comparisons with existing face detectors are also presented. The last part of the chapter discusses possible applications of the proposed face detector. [*Objectives 3, 9, 10*]

8. Chapter 8 provides concluding remarks and summarizes the major contributions of this thesis. It also includes suggestions on how the works presented in this thesis may be extended. [*Objective 3*]

9. Appendix A describes the ECU face detection database, its datasets and usage. It also presents a technique for performance evaluation of face detection algorithms using the database. [*Objective 8*]

Chapter 2

Face Detection: A Review

2.1 Introduction

In this chapter, we present a comprehensive review of existing approaches to face detection. Background materials that are essential for understanding subsequent chapters are also given. The problem of automatic human face detection can be described as follows. Given an arbitrary 2-D input image, determine whether there are any human faces in the image, and if there are, return the location of each human face in the image. Research on face detection dated back to the early 1970s with the works published by Sakai, Nagao and Kanade [93, 94]. However, interests in face detection remained stagnant until the mid 1990s, and even then, studies in face detection were motivated mainly by its application in face recognition for biometric personal identification [17]. In recent years, research in face detection has gained pace with more emphasis on detecting faces in color images for applications in multimedia content management [1, 98], region-of-interest video coding [14, 71], and advanced human-computer interfaces [6, 102, 115].

There are a huge number of face detection approaches in literature. A systematic investigation of these approaches requires the categorization of face detection approaches into logical and meaningful groups. Different categorization schemes are

possible. For example, Yang *et al.* [132] divided face detection approaches into four major categories: knowledge-based top-down, feature invariant, integration of multiple features, and appearance-based face detection, whilst Hjelmås and Low [42] used two categories: feature-based and image-based face detection. In this review, existing face detection approaches are divided into two broad categories: *analytic* and *holistic*. Analytic approaches detect faces by using explicit features, which typically have close resemblance to the visual features that humans perceive from a face. Heuristic rules about relationships between these features are used to distinguish between face and nonface patterns. For example, we can exploit the fact that the face has an approximately elliptical shape with two eye regions appearing darker than the rest of the face. Holistic approaches, on the other hand, treat the whole face as a pattern and employ machine learning techniques to extract automatically the decision rules that differentiate between face and nonface patterns. The categorization of face detection approaches is illustrated in Fig. 2.1.

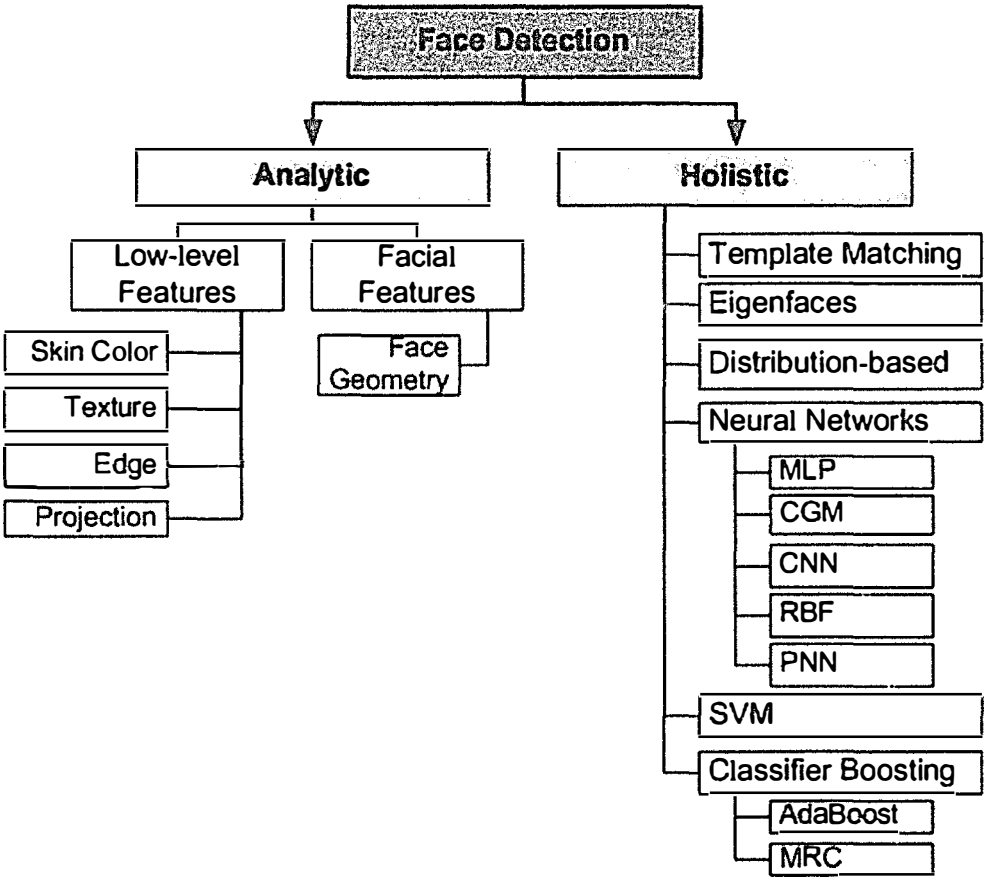


Figure 2.1: Categorization of face detection approaches.

The current chapter is organized as follows. Analytic and holistic approaches to face detection are presented in Section 2.2 and Section 2.3, respectively. Summaries in tabular form of existing face detection approaches, face databases, and face detection software are presented in Section 2.4. The chapter summary is given in Section 2.5.

2.2 Analytic Face Detection

Analytic face detection approaches can be divided into two groups: those that use low-level image features such as skin color, texture, edge, and projection profiles, and those that use facial features such as eyes, mouth and nose.

2.2.1 Low-level Features

A. Skin Color

Skin color is one of the most commonly used features for face detection. In a skin color-based approach, faces are located by first searching for skin-colored regions in the image. This approach, which is fast and invariant to the face pose, requires a color pixel classifier to differentiate skin pixels from nonskin pixels. There are many existing color pixel classifiers; they are based on linear decision boundaries [14, 33, 105], Gaussian densities [16, 44, 72, 98, 131], self-organizing maps [8], or Bayesian decision theory [121]. Color spaces that have been used for skin detection include RGB [129], HSV [4, 105], YCbCr [15], normalized RGB [98, 130], CIE Luv [131], and Farnsworth UCS [127].

Face detection based on skin color is susceptible to the color constancy problem. This problem arises because the color of an object in a captured image also depends on the lighting condition and the capturing device characteristics. A number of techniques have been proposed to cope with this problem. First, the dependency on the lighting intensity can be reduced if only chrominance information is used in skin color detection [4, 72, 121]. Second, we can use skin color models that are adaptive to the lighting condition. For example, Bojic and Yang [4] constructed an adaptive skin color model for each

specific video sequence. However, their method requires skin colors in the first video frame to be labeled either manually or using other automatic extraction schemes. Third, colors in the input image can be corrected. For example, Störring *et al.* [109] proposed an image correction approach, in which the illuminant color is first estimated, and the image captured is then rendered to an equivalent image under a canonical lighting condition. Recently, Hsu *et al.* [43] introduced a lighting compensation approach, in which image pixels are adjusted so that the average gray value of "reference white" colors are linearly scaled to 255. Hsu *et al.* considered pixels with the top 5 percent of the luminance values as the reference white. The image is not corrected if the number of reference white colors is small.

Apart from the color constancy problem, most existing skin color-based approaches to face detection have another limitation in that they use relatively simple post-processing techniques. These post-processing techniques work well when the face can be segmented neatly using skin colors but fail when detected skin regions consist of not only the face but also other exposed skin regions such as neck, chest, and arms or even background regions that have skin color. This short-coming of existing skin color-based face detection approaches will be addressed in this research.

B. Face Texture

The three face detection approaches described below rely on the texture feature of the human face. In these approaches, pattern classification techniques are used to characterize the texture feature. Rikert *et al.* [89] argued that the recognition of an object is not based purely on geometric reasoning, and must rely on texture recognition under difficult conditions such as deformation or changes in pose. In their approach, the face texture is described using a wavelet-based texture model. Each pixel location in the input image is represented by a vector of wavelet coefficients that are generated by a set of multi-scale and multi-orientation filters. The classification of this vector into face/nonface classes is done through a Gaussian mixture model of its distribution, and the Bayes' rule. On a test set of 266 frontal faces and 2,500 nonfaces, the proposed approach achieved a detection rate of 96% for a false detection rate of 5%.

Wavelets were also applied by Garcia and Tziritas [33] to analyze the face texture for the purpose of face/nonface classification. In their approach, a rectangular face candidate region, which is found by skin color detection and region merging, is processed using the wavelet packet decomposition. The region (intensity face candidate) is divided into four equal rectangular parts; each part is decomposed, through filtering-and-subsampling steps, into an *approximation* image and a series of *detail* images. Garcia and Tziritas used the following pair of conjugate quadrature filters:

$$H(z) = 0.853 + 0.377(z + z^{-1}) - 0.111(z^2+z^{-2}) - 0.024(z^3 + z^{-3}) - 0.038(z^4 + z^{-4}), \quad (2.1)$$

$$G(z) = -z^{-1}H(-z^{-1}). \quad (2.2)$$

The number of decomposition levels is chosen according to the region size: three levels for large regions (height ≥ 128 pixels), and two levels for medium regions (height < 128 pixels). The region is represented by a feature vector that consists of the standard deviations of the approximation and detail images. The feature vector is classified into the face class if the Bhattacharrya distance to the average face is below a threshold. In Garcia and Tziritas' approach, the features are assumed to be uncorrelated. Under this assumption, the Bhattacharrya distance between two feature vectors is computed as the sum of distances between feature pairs. In addition, the standard deviation of an approximation image is found to have a Gaussian distribution, whereas the standard deviation of a detail image is found to follow a Laplacian distribution. The Bhattacharrya distance between two feature vectors $\upsilon_k = (\sigma_{0k}, \sigma_{1k}, \ldots, \sigma_{Nk})$ and $\upsilon_l = (\sigma_{0l}, \sigma_{1l}, \ldots, \sigma_{Nl})$ is computed as follows:

$$D(\upsilon_k, \upsilon_l) = \frac{1}{2}\sum_{i=0}^{3} \ln\left(\frac{\sigma_{ik}^2 + \sigma_{il}^2}{2\sigma_{ik}\sigma_{il}}\right) + \sum_{i=4}^{N} \ln\left(\frac{\sigma_{ik} + \sigma_{il}}{2\sqrt{\sigma_{ik}\sigma_{il}}}\right), \quad (2.3)$$

where N is the number of features. Features $\sigma_{0\bullet}$, $\sigma_{1\bullet}$, $\sigma_{2\bullet}$, and $\sigma_{3\bullet}$ in the first summation term correspond to the four approximation images, and features $\sigma_{4\bullet}$ to $\sigma_{N\bullet}$ in the second summation term correspond to the detail images. Different distance thresholds are used for large and medium regions. On a test set of 100 images containing 104 faces, this face detection approach had a correct detection rate of 94.23% and a false detection rate of 19.23%.

Luo and Eleftheriadis [67] proposed a texture-based approach to detecting faces in JPEG images and MPEG videos[1]. Their approach is novel in that face detection is done directly on the DCT domain[2]. Each region of $M \times M$ DCT blocks (a DCT block has a size of 8×8) is represented as a feature vector of length d by selecting in each DCT block k coefficients that correspond to the lowest frequencies, where $k = d/(M \times M)$. This feature vector is classified as face/nonface using a distribution-based technique similar to [111] (see Subsection 2.3.3A). There are two major issues with detecting faces in the DCT domain: (i) the misalignment between the DCT block boundary and the face boundary; and (ii) detecting faces of variable sizes. To address these issues, Luo and Eleftheriadis created six face models for face sizes from 40×40 to 80×80. For each face size, the model is developed for all 64 possible alignment positions between the face and the DCT block boundary. On a test set of 586 frames containing 36 frontal upright faces, the face detection algorithm (with skin color filtering) had a correct detection rate of 94.4% and a false detection rate of 25%.

C. Edge

Sirohey [104] used a Canny edge detector to generate an edge map of the input image. Edge segments that have common break points and similar directions are joined. An ellipse-fitting technique is applied to group edge segments that approximate an ellipse. A bound on the aspect ratio of the best-fit ellipse is used to verify a face candidate. This approach was tested on 48 frontal images with moderately cluttered background and an accuracy of 80% was achieved.

Govindaraju [35] used the Marr-Hildreth operator for edge detection. Post-processing steps such as thinning, spur removal, filtering and corner detection are then performed. The detected edges are labeled as belonging to the left side, right side or the hairline of a face, and then combined to form a face candidate. The aspect ratio of the face candidate is tested if it matches the golden aspect ratio of $(1+\sqrt{5})/2$. The algorithm has a detection rate of 70% on a test set of 150 images.

---

[1] JPEG stands for Joint Photographic Experts Group, MPEG stands for Motion Picture Experts Group.

[2] Discrete Cosine Transform

D. Projection Profiles

In the face detection approach by Campos *et al.* [11], a horizontal edge map of the input image is computed and represented as a 1-D signal. The signal is characterized by a set of Fourier descriptors computed from its Fourier transform. Campos *et al.* discussed three strategies for constructing the descriptor set: selecting the d-lowest-frequency descriptors, selecting the *d*-largest descriptors and adaptive descriptor selection. Descriptor sets are classified by thresholding the distances to the prototype descriptor set for faces.

2.2.2 Facial Features

Face detection approaches in this category consider the human face as a constellation of facial features, such as eyes, eyebrows, mouth, and nose, that are arranged spatially in a non-rigid, yet characteristic configuration. The face, therefore, can be detected by searching for these facial features and verifying a number of constraints about the configuration of facial features or the face geometry. These approaches have a major advantage in that both the feature search and face verification steps, in most cases, are not computation-intensive. A drawback is that for the facial feature search to be accurate, input images must have good resolution. Furthermore, it is difficult to extract heuristics that are valid under a wide range of conditions.

A. Hsu, Abdel-Mottaleb, and Jain's Approach

In the face detection approach by Hsu *et al.* [43], potential face regions are first identified through skin color detection. In each detected skin region, eye regions are detected by observing that the eye has high Cb and low Cr, and contains both bright and dark pixels. Therefore, two eye maps are defined, *EyeMapC* based on the chrominance images, and *EyeMapL* based on the luminance image:

$$EyeMapC = \frac{1}{3}\left\{Cb^2 + (255 - Cr)^2 + (Cb / Cr)\right\}, \tag{2.4}$$

$$EyeMapL = \frac{Y \oplus g}{Y \ominus g + 1}. \tag{2.5}$$

Each of the three terms in (2.4) is normalized to the range [0, 255]. In (2.5), $\oplus$ and Θ, respectively, are gray-scale morphological dilation and erosion by structuring element g. The chrominance eye map is histogram-equalized and multiplied with the luminance eye map, and the resulting eye map is dilated, masked and normalized to brighten the eyes and suppress other facial areas.

Mouth regions are detected by observing that they have stronger red component Cr and weaker blue component Cb compared to other facial regions. The mouth map is defined as follows:

$$MouthMap = Cr^2(Cr^2 - \eta(Cr/Cb))^2, \tag{2.6}$$

where the parameter η is estimated as the ratio of the average Cr^2 to the average of Cr/Cb for the mouth. Again, the term Cr^2 and Cr/Cb in (2.6) are normalized to the range [0, 255].

The centers of the detected eye and mouth regions are grouped into eye-eye-mouth triangles; these triangles are considered as face candidates. Each face candidate is verified by checking (i) the luminance variations; (ii) the average gradient orientations of the eye and mouth regions; (iii) the geometry and the orientation of the triangle; and (iv) the presence of an ellipse-like boundary around the triangle. The Hough transform is used to find the best-fit ellipse of the face boundary. On a test set of 332 images containing 683 faces, the proposed algorithm had a correct detection rate of 80.35%, and a false detection rate of 10.41%.

B. Xu and Sugimoto's Approach

In a real-time face tracking system developed by Xu and Sugimoto [129], skin regions are first detected using a Gaussian model of the skin color in the RGB color space. A color vector $\mathbf{x} = (R, G, B)^T$ is considered as a skin color if the (squared) Mahalanobis distance to the skin color cluster is below a predetermined threshold. The Mahalanobis distance from a vector $\mathbf{x}$ to a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ is defined as follows:

$$M(x) = \left[(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]^{\frac{1}{2}}. \tag{2.7}$$

thresholding is compared with rotated versions of a binary face template using the Hausdorff distance. This comparison not only eliminates some nonfaces but also provides an estimate of the face rotation angle. Face candidates are finally verified using a holistic technique based on the reduced Coulomb energy classifier [21].

E. Yow's Approach

Yow [134] proposed a feature-based face detection approach in which facial points such as eyes, lips, nostrils are detected using elongated Gaussian-derivative filters. The detected facial features are grouped to form face candidates using affine invariants in the face geometric structure. The face candidates are then processed by belief networks, and finally verified against constraints about the face boundary and face motion. In Yow's approach, the active contour technique is used to extract the face boundary. Yow showed that the feature-based approach is robust under illumination and viewpoint changes. However, it has a disadvantage in that the feature detector based on Gaussian derivative filters generates many false alarms, even for images of moderate complexity.

F. Burl, Leung and Perona's Approach

In the face detection approach by Burl *et al.* [9], facial points, namely left eye, right eye, left nostril, right nostril, and middle point between the nose and the lip are detected by a Gaussian derivative filter. Each face constellation constructed from the pool of detected facial points is represented by a feature vector that consists of all distances between facial points in the constellation. This feature vector is normalized to achieve rotation, translation and scale invariance, and the distribution of the normalized feature vector for face is modeled as a Gaussian. Burl *et al.*'s approach is interesting in that it handles the case of incomplete constellation caused by missing facial points.

G. Face Geometry Heuristics

Following, heuristics about the face geometry that have been used for face verification are described. These heuristics are useful in removing quickly obvious nonfaces.

- **Aspect ratio**: The aspect ratio of the facial region must be in a valid range. In [82], the range is [0.8, 2.2], whereas in [121], the range is [1, 1.7].

- **Circularity**: The human face resembles an oval shape even under difference viewing angles. The most commonly used shape circularity measure is the ratio of area to squared perimeter[3] [16, 72]:

$$Cicularity = \frac{Area}{Perimeter^2} \tag{2.8}$$

- **Compactness**: The following measure, which is similar to the above circularity measure, was used by Menser and Wien [72]:

$$Compactness = \frac{Area}{Width \times Height} \tag{2.9}$$

- **Ellipse fitting**: Sobottka and Pitas [105] computed the best-fit ellipse E for each face candidate region C. A measure V of how the region fits in the ellipse is used to verify the face candidate. In the following definition, the numerator is the number of distinct pixels between region C and the interior of E, $|C \setminus E|$ denotes the number of pixels in C that are not in E:

$$V = \frac{|C \setminus E| + |E \setminus C|}{|E|} \tag{2.10}$$

2.3 Holistic Face Detection

Holistic approaches to face detection typically involve a scanning process, in which fixed-size rectangular regions or windows of the input images are processed sequentially to determine if they contain face patterns. To detect faces of varying sizes, the input images are repeatedly scaled down, and then scanned. The key element of these approaches is the classification of each window into face and nonface. Almost all face/nonface classification algorithms in holistic face detection share a common point in that the decision rule is automatically learned from a set of examples, rather than being hand-engineered through designer's knowledge as in feature-based approaches. Hence, holistic approaches provide a better mechanism to cope with face variations.

[3] This measure has a maximum value of $0.25/\pi$ when the shape is a circle.

2.3.1 Template Matching

Template matching is a well-known technique in visual object detection and recognition. In template matching, a template or prototype pattern of the object of interest is created, and an input pattern is matched against the template through a similarity measure [48]. An ideal template should be generic to represent all patterns belonging to the same object, yet specific to differentiate with patterns of all other objects. Templates can be divided into two groups: rigid and non-rigid.

A. Rigid Templates

Rigid templates have no adjustable parameters and are created entirely from a training set. Because a rigid template has a fixed format, the degree of match between an input pattern and the template can be measured directly using some form of correlation metrics. Let **t** be the template and **x** be an input pattern. The following correlation metrics are commonly used in template matching (these metrics are almost equivalent after some normalization):

$$dot\,product = \mathbf{x}^{\mathrm{T}}\mathbf{t} = \sum_i x_i t_i \,, \tag{2.11}$$

$$Euclidean\ distance = |\mathbf{x} - \mathbf{t}| = \sqrt{(x_i - t_i)^2} \,, \tag{2.12}$$

$$cosine\ vector\ angle = \frac{\mathbf{x}^T\mathbf{t}}{|\mathbf{x}||\mathbf{t}|} = \frac{\sum_i x_i t_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i t_i^2}} \tag{2.13}$$

$$correlation\ coefficient = \frac{\sum_i (x_i - \mu_x)(t_i - \mu_t)}{\sigma_x \sigma_t} \,. \tag{2.14}$$

Except for its simplicity, correlation-based template matching is very limited in face/nonface classification because the matching scores are susceptible to image variations caused by geometric transformations (e.g. rotation, shifting, and scaling).

Cai *et al.* [10] used templates in Fig. 2.4a and Fig. 2.5a for profile and frontal faces, and the correlation coefficient metric for comparison with the templates. Wong *et al.* [126]

intrinsic properties of the snake and controls how much it can deform. External energy which depends on the image features, such as image gradient and peak, forces the snake contour to move to new locations. Once released in the vicinity of an object candidate, the snake will lock onto nearby edges and eventually approximate the object shape. Snakes were used by Lam and Yan [59] and Gunn and Nixon [37] to detect head boundaries, and by Yullie *et al.* [136] to detect eyes.

2.3.2 Principal Component Analysis - Eigenfaces

Turk and Pentland [116] applied principal component analysis (PCA) to face recognition and detection. Their approach, which is widely known as *eigenfaces*, was motivated by an earlier work of Kirby and Sirovich [54] on the application of the Karhunen-Loève transform to facial image representation. In the eigenfaces approach, each face image of D pixels is treated as a vector in D-dimensional space. PCA is performed on a set of training face vectors in order to find a set of k most significant eigenvectors, where $k \ll D$. These eigenvectors, termed as eigenfaces, account for most variations in the training vectors.

Eigenfaces can be constructed as follows. Suppose $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_N$ are N face column-vectors in $\mathbf{R}^D$. First, we compute the average face vector:

$$\mathbf{F}_m = \frac{1}{N}\sum_{i=1}^{N}\mathbf{F}_i, \tag{2.15}$$

and the covariance matrix:

$$\mathbf{C} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{F}_i - \mathbf{F}_m)(\mathbf{F}_i - \mathbf{F}_m)^{\mathrm{T}}. \tag{2.16}$$

Next, the set of all eigenvectors of $\mathbf{C}$ is computed. A vector $\mathbf{v}$ in $\mathbf{R}^D$ is an *eigenvector* of $\mathbf{C}$ if there exists a scalar λ such that:

$$\mathbf{C}\mathbf{v} = \lambda\mathbf{v}. \tag{2.17}$$

The scalar λ is called an *eigenvalue* of $\mathbf{C}$.

The set of k-most significant eigenvectors that correspond to the k-largest eigenvalues of $\mathbf{C}$ is selected. This set $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k)$ forms an orthogonal basis of a face subspace. Each new pattern $\mathbf{x} \in \mathbf{R}^D$ is represented by its projection onto the face subspace:

$$\mathbf{w} = \mathbf{V}^T(\mathbf{x} - \mathbf{F}_m). \tag{2.18}$$

The pattern $\mathbf{x}$ can be reconstructed from its feature vector $\mathbf{w}$ as follows:

$$\mathbf{x}_p = \mathbf{V}\mathbf{w}^T + \mathbf{F}_m. \tag{2.19}$$

The distance to the face subspace (DTFS) is defined as its reconstruction error:

$$\text{DTFS} = \|\mathbf{x} - \mathbf{x}_p\| \tag{2.20}$$

The feature vector $\mathbf{w}$ is used in face recognition to discriminate different face classes (i.e. the faces of a person is said to belong to one face class). For face detection purpose, an input pattern is considered as a face if its distance to the face subspace is small.

Principal component analysis was later used for face candidate verification by Menser and Wien [72], and Satoh *et al.* [98]. In Menser and Wien's approach, PCA is applied to the skin probability image generated from a skin color detection step; in Satoh *et al.*'s approach, the Euclidean distance between two feature vectors in the face subspace is used to measure the similarity to a face pattern.

2.3.3 Distribution-based Approaches

A. Gaussian Distribution Model

Sung and Poggio [111] proposed an example-based learning approach for locating frontal upright faces. Input windows of size 19×19 are treated as vectors in 381-dimensional space. The distributions of face and nonface vectors in this space are modeled using 12 Gaussian clusters: six clusters for face and six clusters for nonface. These clusters are found from a training set of faces and nonfaces using the elliptical *k-means* clustering algorithm. Subsequently, an input pattern is represented by a feature vector that consists of 24 values: 12 Euclidean distances and 12 modified Mahalanobis distances. Each distance is measured from the projections of the pattern onto the PCA subspace (constructed specifically for the cluster) to the cluster center. The feature

vector is classified by a multilayer perceptron (MLP). The algorithm works well for images consisting near-frontal upright faces. On the MIT test set A, which contains 301 faces of 71 people, the algorithm had a correct detection rate of 96.3% with 3 false detections. On the MIT test set B, which contains 23 low quality images with 149 faces, the correct detection rate was 79.9% with five false detections. However, the algorithm can not detect faces with large rotation.

Sung and Poggio suggested a bootstrap strategy for updating the face/nonface classifier. This strategy has been incorporated in several holistic approaches to face detection [25, 75, 91]. The classifier, after training, is applied to a validation set; false detections and false rejections are collected to be used as further training examples for the classifier. This strategy is instrumental in coping with the problem of finding a representative set of nonobject patterns. Sung and Poggio also described a number of preprocessing techniques: excluding background pixels from classification (i.e. background masking), and subtracting the best-fit intensity plane (i.e. illumination gradient correction).

B. Bayesian Decision Theory

If the *a posteriori* probability functions of face and nonface are known, the classification of face/nonface can be done by assigning an input pattern to the class with the highest *a posterior* probability. This classification scheme is known as the Bayesian decision rule or the *maximum-a-posteriori* (MAP) rule [21]:

$$\mathbf{x} \text{ is face if } P(face|\mathbf{x}) > P(nonface|\mathbf{x}), \tag{2.21}$$

where $P(face|\mathbf{x})$ and $P(nonface|\mathbf{x})$ are the respective *a posteriori* probability functions of face and nonface classes. Applying the Bayesian theorem, we can express this decision rule in terms of the class-conditional probability density functions (pdfs) as follows:

$$\mathbf{x} \text{ is face if } \frac{p(\mathbf{x}\,|\,face)}{p(\mathbf{x}\,|\,nonface)} \geq \frac{P(nonface)}{P(face)}, \tag{2.22}$$

where $p(\mathbf{x}|face)$ and $p(\mathbf{x}|nonface)$ are the respective class-conditional pdfs, and $P(face)$ and $P(nonface)$ are the respective *a priori* probabilities. A number of different approaches have been proposed for estimating the class-conditional pdfs $p(\mathbf{x}|face)$ and $p(\mathbf{x}|nonface)$. For example, Moghaddam and Pentland [73] used PCA and Gaussian

mixtures; Colmenarez and Huang [20] used Markov processes; Schneiderman and Kanade [100] used a naive Bayesian model. Yang *et al.* [133] proposed two multimodal density models for pdf estimation: one using a mixture of factor analyzers; the other using a mixture of Gaussians. In Yang *et al.*'s models, mixture parameters are estimated using the Expectation/Maximization algorithm. Lui [65] used features based on the 1D Harr representation, and vertical and horizontal projections, and modeled each class-conditional pdf as a Gaussian distribution. In Lui's approach, the nonface patterns, which are near the face class, are selected and used for pdf estimation.

2.3.4 Neural Networks

A. Multi-layer Perceptrons

Rowley *et al.* [91] presented a neural network-based approach to face detection. Their approach is perhaps one of the best known and most referenced works in the field of face detection. In their approach, the input window size is 20×20, the down-sampling scale factor is 1.2, and the network output is between -1 and 1 indicating whether the window contains a face. Rowley *et al.* used a feed-forward neural net with retinal-like input connections. The input window of size 20×20 is divided into several overlapped subregions: 4 sub-regions each of size 10×10, 16 sub-regions each of size 5×5, and 6 sub-regions each of size 20×5. Each subregion is connected to a neuron in the hidden layer.

Rowley *et al.* suggested two post-processing techniques: (i) merging overlapping detections from a single network; and (ii) arbitrating detection results of multiple networks. A location in the input image is classified as belonging to a face if there are sufficient detections in its spatial neighborhood (detections at nearby scale are also counted). Outputs of individual network classifiers are fused using arbitration schemes that include the logical arithmetic operators, and a neural network-based arbitrator. However, arbitration takes place between only two neural networks, and this limits its effectiveness. Like other holistic approaches involving window scanning, Rowley *et al.*'s approach is quite computation-intensive because a large number of overlapping windows need to be processed. Different versions of the face detection

approach had detection rates ranging from 77.9% to 90.3% on the CMU database. An optimized version of the algorithm, running on 200MHz R4400 SGI Indigo 2, took between two to four seconds to process a 320×240 image.

B. Constrained Generative Models

The face detection algorithm developed by Féraud *et al.* [25] is organized into a number of cascading stages; each stage aims to eliminate a portion of nonface windows. In the first stage, an input video sequence is processed by a motion filter, which they reported to exclude up to 90% of nonfaces. The motion filter works based on the observation that the face is often under constant movement as a result of speaking, breathing and eye blinking. In the second stage, a color filter is used to eliminate windows that do not contain skin colors. The color filter is based on a lookup table of skin colors that are collected from face images. In the third stage, the remaining windows are histogram-equalized, smoothed, and normalized by subtracting the mean intensity. In the fourth stage, the normalized window is classified into face and nonface using a neural network known as the Constrained Generative Model (CGM). Féraud *et al.* used a window size of 15×20. A window, treated as a vector, is classified as a face if its Euclidean distance to the face subset is below a threshold. The novelty in their approach is that CGM is trained to predict the projection onto the face subset. The technique of combining several neural networks to improve face/nonface classification is also used. The face/nonface classifier, with different configurations, had detection rates ranging from 73% to 95% on the CMU test set, and from 74.7% to 80.1% on another test set of 13,182 images.

C. Convolutional Neural Networks

Garcia and Delakis [32] presented a face detection approach that uses convolutional neural network (CNN) to classify windows of size 32×36 into face and nonface. In the CNN architecture, a network has a cascade of convolution layers that act as feature extractors. A convolution layer (C layer) consists of a number of planes; each plane is a 2-D convolutional filter. The filters can have different sizes and coefficients; each filter aims to detect a visual feature. The outputs of the convolutional layer are sent to a sub-

sampling layer (layer S), which simply reduces the sizes of its input matrices by a factor of 2. In the CNN architecture, C-S pairs are connected in a feed-forward topology. Each output matrix of the last C-S pair is averaged to give one value. The values obtained from all matrices form a feature vector, which is then classified by a cascade of perceptron layers (layers N). CNNs have two important strengths: (i) they utilize a retinal-like connection scheme, which is biologically more plausible than the approach of lexicographically grouping image pixels into a feature vector; (ii) convolution layers can be trained to automatically extract good features for the classification of visual patterns. In Garcia and Delakis' approach, convolutional neural networks are trained with the error back-propagation algorithm. They reported a detection rate of 98% on a test set of 104 images.

Recently, Tivive and Bouzerdoum [114] developed a new class of convolution neural networks called SICoNNets. The difference between a SICoNNet and a CNN is that the convolution layers of a SICoNNet consist of a special type of neurons, called shunting neuron, that model the shunting inhibition mechanism in early vision [5]. The activity of a static shunting neuron can be described by the following non-linear equation:

$$z_i = \frac{I_i + b_i}{a_i + f(\sum_i w_{ij} I_j)}, \tag{2.23}$$

where z_i is the output of the ith neuron, a_i and b_i are its biases, f is the activation, I_j is an input to the neuron, and w_{ij} is a connection weight. Tivive and Bouzerdoum investigated three connection schemes, namely full-connected, semi-connected, and binary-connected. The SICoNNets are trained with a variant of the backpropagation algorithm (the resilient backpropagation with momentum). On a set of 1000 face and nonface patterns, a trained SICoNNet had a correct classification rate of 97.6% at a false alarm rate of 3.4%.

D. Radial Basis Functions Networks

Huang *et al.* [45, 46] presented a face detection approach that uses a radial basis function network for face/nonface classification. They used preprocessing techniques similar to [91]: linear lighting correction, histogram equalization, and window corner masking. In their approach, a 386-dimensional vector is formed from each 20×20 input

window. This vector is further processed by PCA (see Subsection 2.3.2) to reduce its dimensionality to a value between 40 and 120. The PCA feature vector is then classified into face/nonface using a radial basis function network.

A radial basis function (RBF) network has feed-forward architecture. It has a hidden layer, which is made up of radial basis neurons, followed by a perceptron output layer. Radial basis neurons model Gaussian functions; each neuron has a center μ and a covariance σ^2, and generates an output of

$$h(\mathbf{x}) = \exp\left(\frac{\|\mathbf{x}-\boldsymbol{\mu}\|^2}{2\sigma^2}\right). \tag{2.24}$$

The output of the RBF network is computed as follows:

$$y(\mathbf{x}) = g\left(\sum_{j=1}^{N} w_j h_j(\mathbf{x}) + w_0\right), \tag{2.25}$$

where w_0 is a bias factor, and w_j (j = 1, 2, ..., N) are the connection weights, and $g(.)$ is the activation function. The centers of the radial basis neurons are often initialized by the k-means clustering algorithm. After that, the network parameters are updated through a gradient-descent training algorithm in order to minimize an error function, which is usually the mean-square error evaluated on the training set. However, in Huang *et al.*'s approach, the distance from face subspace (see Subsection 2.3.2) is included in the error function to improve classification performance. These authors showed that the RBF face/nonface classifier had similar performance compared to the classifier by Sung and Poggio [111] (described in Subsection 2.3.3A).

RBF networks were also used by Wang *et al.* [122] for face/nonface classification. In their approach, two RBF-based face/nonface classifiers were developed: one operating at low resolution of 3×3, and the other at a higher resolution of 15×15. The AdaBoost algorithm was used to improve the classification rates of individual RBF classifiers. For face/nonface classification, Wang *et al.* reported a correct detection rate of 91% and a false detection rate of less than 3×10^{-5} on the CMU test set.

E. Polynomial Neural Networks

Huang *et al.* [47] developed a face/nonface classification algorithm that uses a polynomial neural network (PNN). A PNN is a generalized linear classifier. Let $\mathbf{x} = (x_1, x_2, \ldots, x_d)^T$ be an input pattern, the output of a PNN is computed as follows:

$$y(\mathbf{x}) = g\left(\sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d}\sum_{j=1}^{d} w_{ij} x_i x_j + w_0\right), \tag{2.26}$$

where w_i and w_{ij} are adjustable weights, and $g(.)$ is a sigmoid activation function. Image normalization steps and PCA are performed on the 20×20 input window, and a feature vector is extracted. The PNN is trained with the stochastic gradient descent algorithm to classify this feature vector into face or nonface. The training algorithms for PNN are much simpler than those for multilayer perceptrons. However, the number of connection weights in a PNN, which is proportional to the square of the input vector's dimension, can be prohibitively large. The performance of this classifier on the CMU test set was slightly lower than that of the system by Rowley *et al.* [91].

2.3.5 Support Vector Machines

Support vector machines [117] are an emerging paradigm in pattern recognition. The main difference between neural networks and support vector machines is that the former are trained to minimize training errors (i.e. empirical errors) whereas the latter is trained to minimize the upper bound on generalization errors (i.e. structural errors). A SVM classifier maximizes the margin between classes by selecting a minimum number of support vectors. Support vector machines (SVMs) have been used to detect frontal faces by Osuna *et al.* [75], Bassiou *et al.* [3], Heisele *et al.* [41], and Tasi *et al.* [112]. In Osuna *et al.*'s approach, the window size is 19×19, and pre-processing techniques such as masking, illumination gradient correction, and histogram equalization are used. Their face detector was reported to run 30 times faster than the detector by Sung and Poggio [111]. Osuna *et al.* reported a detection rate of 97.1% on the MIT test set A, and 74.2% on the MIT test set B.

2.3.6 Classifier Combination and Boosting

In this category of holistic face detection, face/nonface classification is done by combining a large number of individual classifiers. We discuss below two approaches of classifier combination and boosting: classifiers cascade based on the AdaBoost algorithm, and the maximal rejection classifier. Both approaches are similar in that each new classifier is trained to focus on the examples that the preceding classifiers have difficulty in learning.

A. Classifiers Cascade based on AdaBoost algorithm

Recently, Viola and Jones [120] proposed an object detection approach that is based on the AdaBoost algorithm and the use of simple image features. In their approach, four Harr-like features are computed for each rectangle in the classification window (see Fig. 2.6); each feature is the difference between the sum of pixels in the white parts and the sum of pixels in the gray parts. Viola and Jones proposed the integral image representation: the value of the integral image at a given point is the sum of all the pixels above and to the left of the point. This image representation allows fast computation of the features even across different scales. For a classification window of size 24×24, an exhaustive set of 45,396 such features is generated. However, not all features are used for face/nonface classification. Viola and Jones applied the AdaBoost algorithm as a feature selection mechanism.

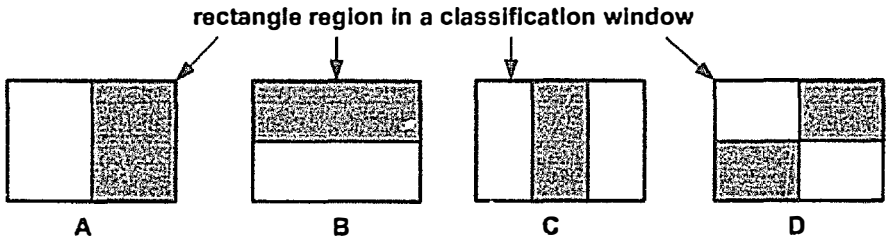


Figure 2.6: Four Harr-like features used by Viola and Jones [120].

The AdaBoost algorithm, introduced by Freund and Schapire [30], is a recent development in machine leaning. The AdaBoost algorithm trains several weak

classifiers through a progressive and error-based adjustment of the distribution of the training examples. These weak classifiers are then combined into a strong classifier. Freund and Schapire show that the classification error on the training set can be reduced exponentially with the number of boosting rounds. The AdaBoost algorithm in its original form is shown in Table 2.1.

Table 2.1: The original AdaBoost algorithm [29].

Input: Set of N labeled examples $\{ (1, c(1)), (2, c(2)), \ldots, (N, c(N)) \}$

Distribution D over the examples

Weak learning algorithm *WeakLearn*

Number of iterations T

Initialize the weight vector $w_i^1 = D(i)$ for $i = 1, 2, \ldots, N$.

Do for $t = 1, 2, \ldots, T$

1. Set

$$\mathbf{p}^t = \frac{\mathbf{w}^t}{\sum_{i=1}^{N} w_i^t}$$

2. Call *WeakLearn*, providing it with the distribution $\mathbf{p}_t$; get back a hypothesis h_t.
3. Calculate the error of h_t: $\varepsilon_t = \sum_{i=1}^{N} p_i^t \, | h_t(i) - c(i) |$.
4. Set $\beta_t = \dfrac{\varepsilon_t}{1-\varepsilon_t}$.
5. Set the new weights vector

$$w_i^{t+1} = w_i^t \beta^{1-|h_t(i)-c(i)|}$$

Output the hypothesis

$$h_f(i) = \begin{cases} 1, & \text{if } \sum_{i=1}^{T} (\log \frac{1}{\beta}) h_t(i) \geq \sum_{i=1}^{T} (\log \frac{1}{\beta}) \\ 0, & \text{otherwise} \end{cases}.$$

We return to the face detection approach by Viola and Jones. For a given training set and a predetermined number n_i, a strong classifier that uses n_i features is constructed through n_i rounds of AdaBoost training. At each round of training, the best feature is selected among the pool of features, and a weak classifier (simple thresholding) that operates solely on the feature is added to the strong classifier. In Viola and Jones'

approach, face/nonface classification is done using a cascade of strong classifiers. The threshold used for each strong classifier is selected so that it produces a specified correct detection rate on a validation set. The number of strong classifiers is increased incrementally until an overall target overall false detection rate is achieved. The final face detector consists of 32 strong classifiers that use a total of 4297 features. On the CMU frontal face test set, the face detector has a correct detection rate of 88.8% for a false detection rate of 9.8%. At the time of this writing, the face detector is one of the fastest detectors: it can detect faces at 15 frames per second (frame size = 384×288) on a 700MHz Pentium III. This speed advantage can be explained by the fast scheme for feature computation, and the use of a boosted cascade for face/nonface classification. Viola and Jones [118] later developed a variant of the AdaBoost algorithm that takes into account the highly asymmetric distribution between faces and nonfaces. They showed that the asymmetric AdaBoost yields significant performance improvements over the conventional AdaBoost.

Recently, Lienhart *et al.* [60, 61] extended the set of Harr-like features used by Viola and Jones. The extended set includes four edge features, eight line features and two center-surround features (Fig. 2.7). Lienhart *et al.* showed that at the same correct detection rate, the extended feature set has a 10% lower false detection rate compared with the basic feature set used by Viola and Jones. However, the face detector runs at a slower speed compared to the face detector by Viola and Jones. Lienhart *et al.* have included an implementation of their face detector in the OpenCV library, which we will use in Chapter 7 for comparison with our face detector.

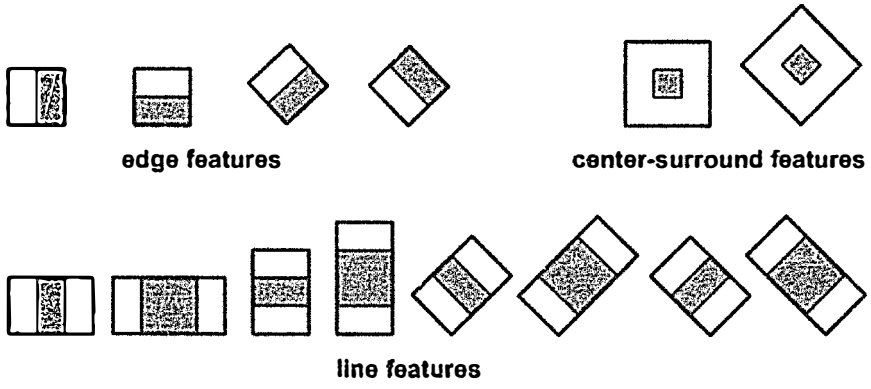


Figure 2.7: Extended set of Harr-like features used by Lienhart *et al.* [60, 61].

The AdaBoost algorithm has been also used by Fröba *et al.* [31] and Wang *et al.* [122] to boost the performance of underlying face/nonface classifiers. In Fröba *et al.*'s approach, the weak classifier is edge orientation matching based on the Bayes rule; in Wang *et al.* approach, the weak classifier is the radial basis function network.

B. Maximal Rejection Classifier (MRC)

Elad *et al.* [22] proposed an object/non-object classification algorithm based on the maximal rejection principle and successive linear projections. They applied this algorithm to the problem of detecting frontal upright faces in gray-scale images. The algorithm relies on the assumption that the object class forms a convex hull. For given two training sets, X of object examples and Y of non-object examples, a projection vector **u** is selected so that there is maximum separability between the projections of the two sets on **u**. Elad *et al.* show that, if the distribution between objects and non-objects is highly unbalanced, this is equivalent to minimizing the following distance function:

$$d(\mathbf{u}) = \frac{\mathbf{u}^{\mathrm{T}}\Sigma_x\mathbf{u}}{\mathbf{u}^{\mathrm{T}}(\Sigma_x + \Sigma_y + (\mu_y - \mu_x)(\mu_y - \mu_x)^{\mathrm{T}})\mathbf{u}}. \tag{2.27}$$

where Σ_x and Σ_y are the covariance matrices; μ_x and μ_y are the means of sets X and Y, respectively. The vector **u** that minimizes d can be found by solving the generalized eigenvalue problem $\mathbf{Au} = \lambda\mathbf{Bu}$, where:

$$\mathbf{A} = \Sigma_x,$$

$$\mathbf{B} = \Sigma_x + \Sigma_y + (\mu_y - \mu_x)(\mu_y - \mu_x)^{\mathrm{T}}. \tag{2.28}$$

Once the projection vector is identified, the examples in **X** and **Y** on **u** are classified by thresholding their projections on **u**. The examples that correspond to the overlaps between the projections of X and Y are considered as *unknown* class. These examples will be used in the next round of training to select a new projection vector. This process continues repeatedly to produce a set of projection vectors. Elad *et al.* used 50 projection vectors for face/nonface classification. The MRC is similar in spirit to the Fisher Linear Discriminant (FLD), which aims to minimize the within-class variances and maximize the between-class variances. The only difference is that MRC deals with non-equally probable classes. Elad *et al.* reported a similarity in detection accuracy

between the MRC face detector and the neural network face detector by Rowley *et al.* [91]. However, the MRC face detector was found to be many times faster than the neural network face detector.

2.4 Summary of Face Detection

The existing face detection approaches are summarized in Table 2.2. The classification rates are as reported by the respective authors; some authors use common databases for testing, whereas others use a custom database to test certain aspect of their algorithms. Therefore, classification rates listed in the table are only indicative of the detection performance, and should not be used for an objective comparison of different face detection approaches. Online face detection demo and software are given in Table 2.3. Online databases for facial image analysis are given in Table 2.4.

Table 2.2: Summaries of face detection approaches.

Authors	Descriptions	Image Type	Performance
Burl *et al.* [9]	Gaussian-derivative facial feature detectors Gaussian model of distances between facial feature	intensity, frontal	(84%, *, 150)
Chai & Ngan [15]	Fixed-range skin color map in Cb-Cr plane Face verification based on shape criteria	color, mainly frontal	(82, *, 60)
Elad *et al.* [22]	Maximal rejection classifier	intensity, frontal upright	*
Féraud *et al.* [25]	Constrained generative model Multi-staged algorithm Window size = 15×20	color, frontal/ side views	(86%, 1.97%, CMU)
Garcia & Delakis [32]	Convolutional neural networks Window size = 32×36	intensity, frontal/ small in-plane rotation	(97.5%, 2.4%, 100)
Garcia & Tziritas [33]	Linear model of skin color in YCbCr and HSV, Wavelets	color, frontal/ small in-plane rotation	(94.2%, 19.23%, 100)
Govindaraju [35]	Edge-based face model Marr-Hildreth edge detector	intensity, frontal upright	(70%,*, 150)
Hsu *et al.* [43]	Color correction, Color-based detection of skin, eye, and mouth Geometric verification of facial feature placement	color, semi-frontal/ side views	(80.35%, 10.41%, 382)
Huang *et al.* [46]	Radial basis function network, PCA Window size = 20×20	intensity, frontal upright	(84.6%, 16.1%, MIT set B)
Huang *et al.* [47]	Polynomial neural network, PCA Window size = 20×20	intensity, frontal upright	(84.73%, *, CMU)
Jeon *et al.* [49]	Eye region through intensity thresholding, Face binary template using Hausdorff distance, First order reduced Coulomb energy classifier Window size = 21×21	intensity, frontal upright	(91%, 0%, CMU)

Authors	Descriptions	Image Type	Performance
Liu [65]	Bayesian classifier with Gaussian density estimation, Modeling of nonfaces that are near the face class, 1D Harr wavelet and projection features, Window size = 21×21	intensity, frontal upright	(97.4%, 0.4%, CMU)
Luo & Eleftheriadis [67]	Gaussian mixture skin color model Gaussian clustering of DCT-coefficient feature vectors	color, frontal upright	(81%, 21%, 586)
Lienhart *et al.* [60, 61]	Extension of Viola & Jones [118-120] Extended Harr-like features	intensity, frontal upright	(82.3%, 4.7%, CMU)
Menser & Wien [72]	Gaussian skin color model in Cb-Cr plane, Connected operators based on shape criteria, PCA of skin probability image	color, mostly frontal upright	*
Osuna [75]	Support vector machines Window size = 19×19	intensity, frontal upright	(97.1%, 1.3%, MIT Set A), (74.2%, 13.4%, MIT set B)
Rikert *et al.* [89]	Wavelet-based texture model of human face	intensity, upright (frontal/ non-frontal)	*
Rowley *et al.* [91]	Neural networks (MLP) Window size = 20×20	intensity, frontal or in-plane rotated	(88.4%, 19.5%, CMU)
Schneiderman & Kanade [100]	Naive Bayesian classifier Window size = 64×64	intensity, frontal or profile	(90.5%,6.8%, CMU)
Sirohey [104]	Edge-based ellipse-fitting of face boundary, Canny edge detector	intensity, mostly frontal	(80%, *, 48)
Sobottka & Pitas [105]	Region-based ellipse-fitting	color, frontal upright	*
Sung and Poggio [111]	Gaussian clustering of faces and nonfaces Window size = 19×19	intensity, frontal upright	(96.3%, 1.0%, MIT set A), (79.9%, 3.5%, MIT set B)
Viola & Jones [118]	Cascade of face/nonface classifiers AdaBoost algorithm for feature detection Harr-like features Window size = 24×24	intensity frontal & upright	(88.8%, 9.8%, CMU)
Yow [134]	Gaussian-derivative facial feature detectors, Belief networks, Active contours for face boundary extraction	intensity, semi-frontal	(66.3%, 14.7%, Manchester Set A), (17.8%, 16.7%, Manchester Set B)

Note:
Performance is given as (*correct detection rate%, false detection rate%, database name* or *database size*)
False detection rate is the ratio of the number of false detections to the number of true faces.
Database size (i.e. the number of images) is given for custom dataset.
* not available entry

Table 2.3: Online face detection demonstration and software.

Face detector	Authors	Descriptions
BuFaLo face detector	Frank Fritze	URL: www.geocities.com/fritzfra2001/ Based on the works of Viola and Jones [120], Lienhart *et al.* [61].
CMU face detector	H. Schneiderman, H. Rowley, *et al.* [91, 99]	URL: www.vas.ri.cmu.edu/cgi-bin/demos/findface.cgi Face must be upright and at least 20 pixels from mouth to eyes.
University of Crete face detector	C. Garcia, M. Delakis [32]	URL: www.csd.uoc.gr/~cgarcia/FaceDetectDemo.html In-plane rotation angle must not exceed 20°.

Table 2.4: Online image databases for facial image analysis.

Database	Descriptions
Face Recognition Databases	
AR face database [70] (Purdue Uni. face database)	Over 4000 images, 126 people, different facial expression, lighting conditions URL: *http://rvl1.ecn.purdue.edu/~aleix/*
AT&T face database [97] (ORL face database)	400 images, 40 distinct people, different lighting conditions, facial expressions, facial details URL: *http://www.uk.research.att.com/facedatabase.html*
UMIST face database [36]	564 images, 20 people, gray-scale, image size 220×220, profile to frontal views URL: *http://images.ee.umist.ac.uk/danny/database.html*
Facial Expression Databases	
Japanese Female Facial Expression database (JAFFE) [68]	213 images, 10 Japanese females, 6 basic facial expressions + 1 neutral URL: *http://www.mis.atr.co.jp/~mlyons/jaffe.html*
Face Detection Databases	
BioID face database [50]	1521 images; 23 people, a manual set of eye positions
CMU face database [91]	130 images, 507 faces, gray-scale URL: *http://vasc.ri.cmu.edu/idb/html/face/index.html*
Manchester face database	*Set A*: 300 images, frontal upright faces *Set B*: 90 images, frontal upright faces, with occlusion.
MIT face database [111]	*Set A*: 301 images, 71 different people, near frontal faces. *Set B*: 23 images, 149 faces, varying quality.
Stirling Psychological Image Collection [39]	Over 800 face images URL: *http://pics.psych.stir.ac.uk/*

2.5 Chapter Summary

This chapter provides a systematic review of existing approaches to automatic human face detection in digital images. These approaches can be divided into two main categories: analytic and holistic. Analytic approaches treat the human face as a composition of facial features, and use heuristic constraints about these features to distinguish face and nonface patterns. Analytic approaches are often not computation-intensive. However, they are highly susceptible to the image quality and variations in the imaging condition. Holistic approaches treat the human face as a whole pattern and employ machine learning techniques to differentiate face from nonface. Holistic approaches involve window scanning, in which windows at different image locations and at multiple scales are classified into face and nonface patterns. As such, these approaches are in general very computation-intensive. The advantage of these approaches is that the classifiers are mostly developed with machine learning algorithms, which are capable of handling large variations in the face pattern and extracting non-obvious decision rules.

We propose a new face detection algorithm that combines the strengths of both analytic and holistic approaches to face detection. Components of the new algorithm are presented in Chapters 3-6. Chapter 3 addresses the skin detection technique that relies on color pixel classification. Chapter 4 addresses various techniques to enhance pixel-wise skin segmentation. Chapter 5 presents a technique for selecting face candidates from the segmented skin regions. Chapter 6 introduces holistic techniques to perform the final verification of face candidates. Chapter 7 presents a complete face detector together with its analysis and applications.

Chapter 3

Skin Detection Using Color Pixel Classification

3.1 Introduction

This chapter and the next focus on the problem of detecting skin regions in a digital image. In recent years, skin detection has attracted considerable research interest. It is a crucial step in several color-based approaches to human face detection [33, 43, 127]. It also plays a key role in applications such as hand segmentation for gesture analysis [138] and objectionable image filtering [27, 52]. In these applications, the search space for the objects of interest, such as faces or hands, is reduced through the detection of skin regions in the input image. This strategy of search space reduction is motivated by the pre-attentive mechanism commonly exhibited in biological vision systems. To this end, skin detection is very effective because it is fast, involves a small amount of computation and can be done regardless of pose.

In this chapter, we use the term "skin detection" to refer to the detection of skin at pixel level, in which image pixels are divided into skin and nonskin pixels. In comparison, in the next chapter, we use the term "skin segmentation" to refer to the detection of skin at region level, in which image pixels are grouped into skin regions and nonskin regions.

The focus of the current chapter is the skin detection approach that differentiates skin pixels from nonskin pixels on the basis of pixel color. This approach is based on the observation that the human skin has very consistent and distinctive colors compared to the colors of most natural and man-made objects. This skin detection approach involves a color classification algorithm whose major requirements are stated below:

- **Very low false rejection at low false detection.** Color pixel classification is the first step in skin detection; therefore it is crucial that almost all skin colors are picked up while keeping false detections to a minimum. False detections can be handled in later stages when more *a priori* knowledge is available.
- **Robust detection of different skin types.** There are many skin color types including whitish, blackish, yellowish and brownish, which must all be classified as skin.
- **Robustness to variations in lighting conditions.** Skin color can appear markedly different under different lighting conditions. It is, therefore, impractical to construct a skin color classification algorithm that works under all possible lighting conditions. However, a good classification algorithm should exhibit some sort of robustness to lighting variations.

Many different skin color classification algorithms have been proposed. These include piece-wise linear decision boundary classifiers [15, 33, 106], classifiers based on parametric [72, 129, 131] and nonparametric density estimation [52, 83, 121], and self-organizing maps [8]. The color spaces used in classification also vary: RGB [98], YCbCr [15, 72, 121], HSV [138], CIE-Luv [131], Farnsworth uniform color space [127], and normalized RGB [130]. A number of comparative studies on skin color classification have been reported. Brand *et al.* [7] compared three different approaches: simple thresholding of the R/G ratio, color space mapping with 1-D indicator, and RGB skin probability map. These authors used the Compaq skin database created by Jones and Rehg [51], which consists of nonskin images (i.e. images that have no skin region), and skin images (i.e. images of skin regions that are coarsely segmented). Terrillon *et al.* [113] compared two classifiers based on the Gaussian and Gaussian mixture density models across 9 chrominance spaces. However, only 110 images of 30 Asian and Caucasian people were used in their experiments. Shin *et al.* [101] compared skin detection in eight color spaces. In their experiments, the skin and nonskin samples were

taken separately from different sources: skin samples from the AR face recognition database and the University of Oulo physics-based face databases, and nonskin samples from the University of Washington content-based image retrieval dataset. There is a clear need to evaluate skin detection algorithms on a comprehensive database that is designed specifically for the testing of skin detection in unconstrained imagery.

In this chapter, we address two important questions related to the approach of skin detection using color pixel classification:

- which classification algorithm and which color space give the best skin detection performance in terms of accuracy, speed and memory load?
- how effective is the approach of skin detection using pixel color for general images?

In order to answer the first question, we analyze several color classification algorithms that have been reported in the literature. These algorithms include piece-wise linear decision boundary, classifiers based on Gaussian densities, Bayesian classifier with nonparametric density estimation, and the self-organizing map. Furthermore, we propose a new classification algorithm that uses multilayer perceptron neural networks. The segmentation performances of different color spaces are compared using the Bayesian classifier with nonparametric density estimation. To answer the second question, we report the skin detection results on the ECU database. This database consists of over 3,300 color images that have been manually segmented for skin regions.

This chapter is organized as follows. The different families of color spaces, namely RGB, YCbCr, HSV, and CIE-Lab are described in Section 3.2. Characteristics of the human skin colors are investigated using visualization tools in Section 3.3. Major classification algorithms, namely the piece-wise linear decision boundary, classifiers based on Gaussian densities, Bayesian classifier based on nonparametric density estimation, and multilayer perceptrons are presented in Section 3.4. A comparative analysis of the classification algorithms is presented in Section 3.5. The skin detection approach using the Bayesian color pixel classifier is discussed in Section 3.6. The chapter summary is given in Section 3.7.

3.2 Color Spaces

Color is a visual sensation formed when an electromagnetic waveform in visible spectrum falls on light-sensitive cones in the retina. Three attributes are associated with this visual sensation:

- *brightness* (or luminance): indicates if an area appears to exhibit more or less light.
- *hue*: indicates if an area appears to be similar to combination of two of the basic colors, red, green, blue. It is the visual attribute that we use to give color names, such as blue, yellow, purple, etc.
- *saturation* (or colorfulness, color purity): indicates if an area appears to exhibits more or less of its hue. For example, pink and red differ in saturation with the red being more saturated.

A *color space* specifies how colors are represented and coded. Almost all color spaces in existence originate from the tri-chromatic theory, which states that any color (visible light) can be described in terms of three reference colors or *primaries*. Typically, a color is expressed as a linear combination of the primaries, and each weight is a *color value*. Color values corresponding to a primary are said to belong to a *color channel*. Clearly, different sets of primaries and color transforms (linear or nonlinear) lead to different color spaces. There exist a large number of color spaces; color spaces that have similar characteristics are grouped into a color space family. Next, we shall describe four representative color spaces:

- the RGB color space;
- the HSV color space, which represents hue-saturation-intensity color spaces;
- the YCbCr color space, which represents class-Y color spaces;
- the CIE-Lab color space, which represents perceptually uniform color spaces (UCS).

3.2.1 The RGB Color Space

This color space, which is widely used in computer graphics, specifies color in terms of three primary colors: red (R), green (G) and blue (B). A common format of the RGB color space uses 8 bits for each color channel (i.e. color values between 0 and 255), this gives 16.8 million different colors. In the RGB color space, brightness information is present in all color channels. For the color image in Fig. 3.1, the R, G, and B color channels are shown in Fig. 3.2. For the rest of this section on color spaces, we continue to use this *carphone* image in visualizing color channels.

Figure 3.1: The *carphone* image.

(a) R channel (b) G channel (c) B channel

Figure 3.2: Color channels of the RGB color space.

3.2.2 The HSV Color Space

In the HSV color space, color is measured in terms of *hue* (H), *saturation* (S) and *intensity value* (V). The HSV color space is intuitive because it is designed to approximate the way humans perceive and interpret color. Other similar color spaces

are HSL (hue-saturation-lightness) and HCI (hue-colorfulness-intensity). Figure 3.3 illustrates the hexcone of the HSV color space. The point at the base of the hexcone is black, where V = 0. The color becomes brighter when V increases from 0 to 1. As H increases from 0 to 360°, the color changes from red to yellow, green, cyan, blue, magenta, and back to red. When S increases from 0 to 1, the color changes from unsaturated (white) to saturated (non-white). From an artist's viewpoint, any color with V = 1, S = 1 is a pure color. Table 3.1 presents the color conversion from RGB to HSV. For convenience of computation, the H value is normalized so that its nominal range is [0, 1]. A visualization of the color channels of the HSV color space for the *carphone* image is shown in Fig. 3.4. In this figure, the H, S, V channels are scaled to be displayed as gray-scale images.

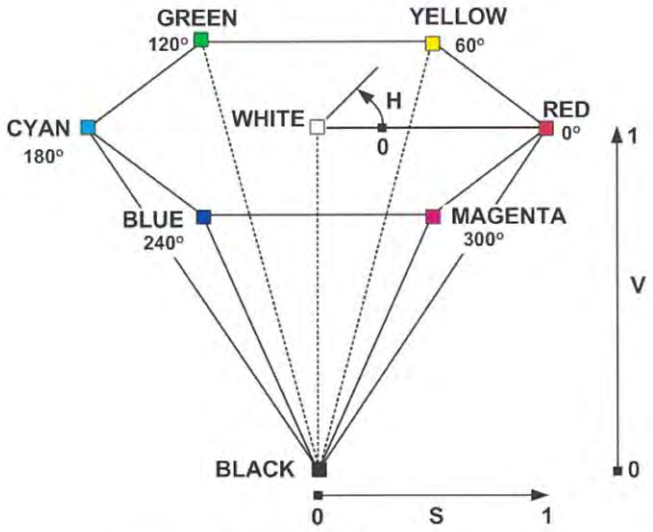


Figure 3.3: The HSV color space.

(a) H channel (b) S channel (c) V channel

Figure 3.4: Color channels of the HSV color space.

Table 3.1: Color conversion from RGB to HSV.

$$V = \max(R, G, B)$$

$$\Delta = V - \min(R, G, B)$$

$$S = \begin{cases} 0 & \text{if } V=0 \\ \Delta / V & \text{otherwise} \end{cases}$$

If $\Delta \neq 0$ then

$$H = \frac{1}{6} \times \begin{cases} (G-B)/\Delta & \text{when } R=V \\ 2+(B-R)/\Delta & \text{when } G=V \\ 4+(R-G)/\Delta & \text{when } B=V \end{cases}$$

$$H = H + 1 \text{ when } H < 0$$

else

$$H = 0$$

End if

3.2.3 The YCbCr Color Space

This color space belongs to a color space family known as "class Y", which includes the YCbCr for digital image and video, the YIQ for NTSC television[4], the YUV for SECAM television[5] and PAL television[6]. In this color space family, color information is stored in one luma and two chroma signals. The luma signal corresponds to the luminance channel Y, and the two chroma signals correspond to the two chrominance channels (i.e. CbCr, IQ, or UV). The separation between luminance and chrominance channels, together with signal modulation schemes, allowed back-and-white receivers to continue displaying black-and-white picture when color television was first introduced. In addition, this separation enables efficient processing and transmission of video signals. For example, because the human vision system responds more strongly to brightness changes than chrominance changes, the separated chrominance signals can be compressed by sub-sampling.

[4] National Television Standards Committee

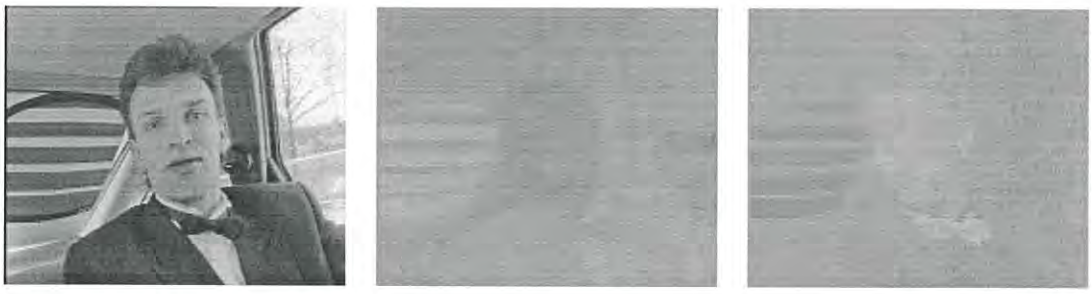
[5] Sequentiel Couleur Avec Mémoire (Sequential Color with Memory)

[6] Phase Alternation Line

The YCbCr color space is specified in the ITU[7] Recommendation BT 601, and it is a scaled and offset version of the YUV color space. Y is defined to have a nominal 8-bit range of [16, 235]; Cb and Cr are defined to have a nominal range of [16, 240]. In the following conversion formula, the range of RGB is assumed to be [0, 255]:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \frac{1}{255} \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112 \\ 112 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{3.1}$$

The Y, Cb, Cr *channels* of the carphone image are shown in Fig. 3.5.



(a) Y channel (b) Cb channel (c) Cr channel

Figure 3.5: Color channels of the YCbCr color space.

3.2.4 The CIE-Lab Color Space

All color spaces described so far are perceptually nonuniform. Comparing colors in a perceptually nonuniform color space is difficult because the same Euclidean distance between color values does not guarantee the same perceived color difference. Perceptually uniform color spaces are developed to alleviate this problem. This color space family includes the CIE-Lab, CIE-Luv and Farnsworth uniform color spaces (UCS). The advantage of using a perceptually uniform color space is that color differences can be measured in terms of the Euclidean distance, for example:

$$diff(L_1a_1b_1, L_2a_2b_2) = \sqrt{(L_1 - L_2)^2 + (a_1 - a_2)^2 + (b_1 - b_2)^2} \tag{3.2}$$

[7] International Telecommunication Union

In the CIE-Lab color space[8] (Fig. 3.6), L defines the lightness of the color, and a and b define the color information along the red/green and blue/yellow axes, respectively. The range of L is [0, 100], and the range of a and b is [-128, 127]. The L, a, and b channels of the *carphone* image are shown in Fig. 3.7.

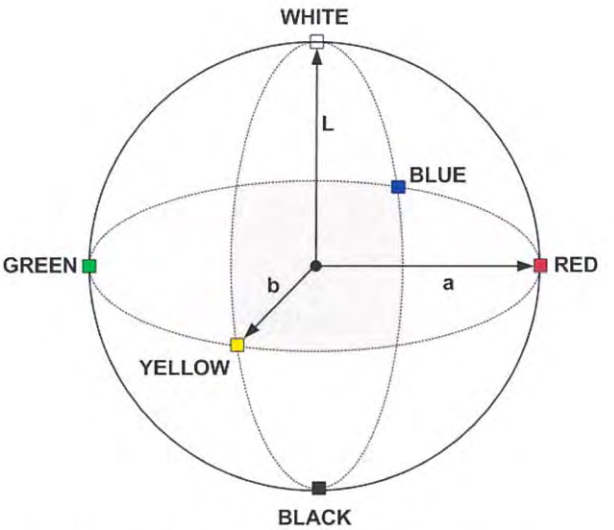


Figure 3.6: The CIE-Lab color space.

(a) L channel (b) a channel (c) b channel

Figure 3.7: Color channels of the CIE-Lab color space.

[8] CIE stands for Commission Internationale de l'Eclairage, or International Commission on Illumination.

Table 3.2: Color conversion from RGB to CIE-Lab.

- Convert RGB to CIE-XYZ as in (3.5)
- $L = \begin{cases} 116\times(Y/Y_n)^3 - 16, & \text{if } Y/Y_n > 0.008856 \\ 903.3\times(Y/Y_n), & \text{otherwise} \end{cases}$
- $a = 500 \times (f(X/X_n) - f(Y/Y_n))$
- $b = 200 \times (f(Y/Y_n) - f(Z/Z_n))$

where

$$f(t) = \begin{cases} t^{1/3} & \text{if } Y/Y_n > 0.008856 \\ 7.787t + 16/116 & \text{otherwise} \end{cases}$$

$X_n = 0.9504$, $Y_n = 1$, and $Z_n = 1.0889$ are the CIE-XYZ coordinates for the white reference point in the CIE standard illuminant D65.

The conversion steps from RGB to CIE-Lab are presented in Table 3.2. The nonlinear conversion involves an intermediate color space known as the CIE-XYZ color space, which is described next.

CIE-XYZ color space. The CIE-XYZ color space was developed by the Commission Internationale de l'Eclairage in 1931, and it is the root of all color spaces. The CIE-XYZ color space has three imaginary primary colors X, Y and Z defined so that all visible colors can be represented using non-negative color values. The CIE-XYZ color space is sometimes represented as Yxy where the pair (x, y) describes the color chromaticity:

$$x = X/(X+Y+Z), \tag{3.3}$$

$$y = Y/(X+Y+Z), \tag{3.4}$$

The plot of all (x,y) pairs for visible colors is called the chromaticity diagram; it is shown in Fig. 3.8. Conversion from RGB to CIE-XYZ is linear and device dependent.

The following conversion formula is specified in the ITU Recommendation BT 709 for the CIE standard illuminant D65 (the range of RGB is [0, 1]):

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3.5}$$

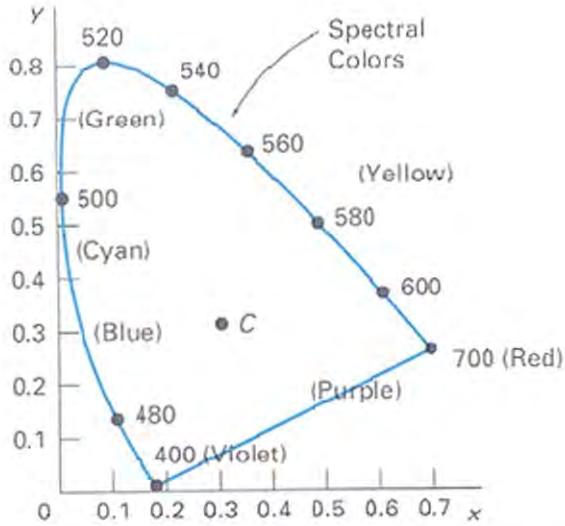


Figure 3.8: The *x-y* chromaticity diagram.

3.2.5 Chrominance Planes

In color spaces such as HSV, YCbCr and CIE-Lab, a color has three channels, of which two are chrominance channels and the other is a luminance channel. The chrominance channels are so called because they determine the chromaticity or colorfulness of the color. The luminance channel, on the other hand, describes the brightness of the color. A chrominance plane is formed by two color channels that contain chrominance information. For example, for the HSV, YCbCr and CIE-Lab color spaces, the chrominance planes are HS, CbCr, and ab, respectively; the luminance channels are V, Y, and L, respectively. Another common chrominance plane is the normalized rg plane of the RGB color space:

$$r = R/(R + G + B), \tag{3.6}$$

$$g = G/(R + G + B), \tag{3.7}$$

$$b = B/(R + G + B). \tag{3.8}$$

A number of skin color classification algorithms operate on a chrominance plane. The rationale for such algorithms is that using only chrominance channels makes classification less sensitive to changes in the lighting intensity.

In this study, we focus on four main color spaces, namely RGB, HSV (representing hue-colorfulness-brightness color spaces), YCbCr (representing class-Y color spaces) and CIE-Lab (representing perceptually uniform color spaces), and four chrominance planes, namely normalized rg, HS, CbCr and ab.

3.3 Skin Color Characteristics

In this section, we use visualization tools to investigate the characteristics of the human skin color, especially its distribution in different color spaces. The section is divided into three main parts. First, we examine the skin color distribution in different color spaces (Subsection 3.4.1). Second, we investigate how the skin color distribution varies for different skin color types (Subsection 3.4.2). Third, we explore the characteristics of the chrominance and luminance components of skin colors (Subsection 3.4.3).

3.3.1 Skin Color Distribution in Different Color Spaces

The distributions of skin colors in the four color spaces RGB, HSV, YCbCr and CIE-Lab are shown in Fig. 3.9. Each dot represents a skin color. To generate this figure, we collected a set of over 14,000 skin colors. These colors were automatically sampled from the skin regions of the first 100 images of the ECU face detection database. Fig. 3.9 shows that the human skin color indeed has a very specific distribution in all four color spaces.

3.3.2 Distributions of Different Skin Color Types

Although there is a large number of skin colors, all skin colors can be divided into a number of major skin types. The major skin types are whitish, blackish and

yellowish/brownish, which correspond respectively to people of Caucasian, African, and Asian descents. The distributions of skin colors for these three skin types are shown in Fig. 3.10. The figure is generated from 30 skin segmented images (10 images per skin type) in the ECU database. The figure illustrates the difference between the skin color clusters for the three skin color types. The difference is not obvious for the RGB color space, but it is quite significant for other color spaces HSV, YCbCr and CIE-Lab. The figure shows that the distinction between these skin color types is mainly due to the luminance (Y in YCbCr, V in HSV, and L in CIE-Lab). The luminance of whitish skin colors is high, the luminance of blackish skin colors is low, and the luminance of yellowish/brownish skin colors is medium.

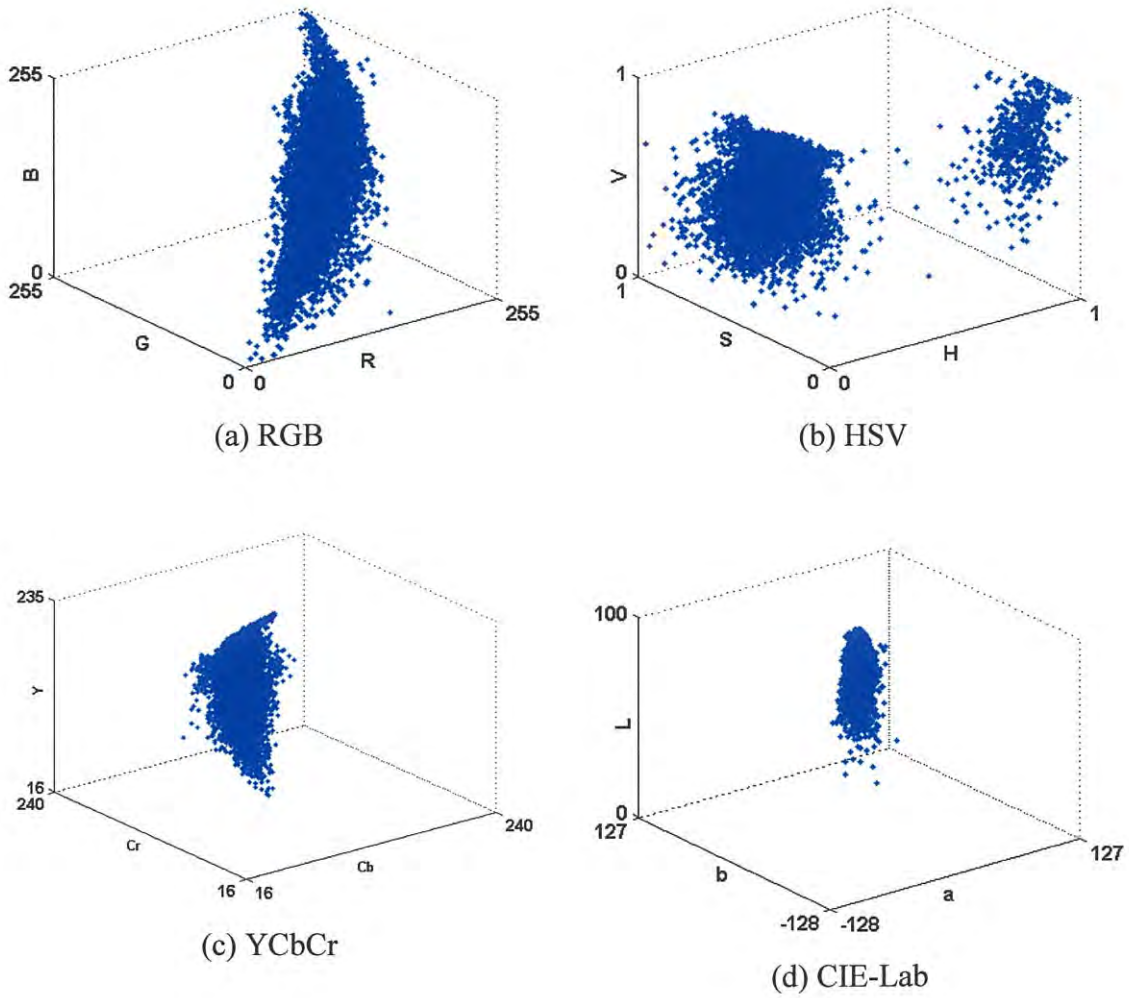


Figure 3.9: Distributions of skin colors in RGB, HSV, YCbCr, and CIE-Lab.

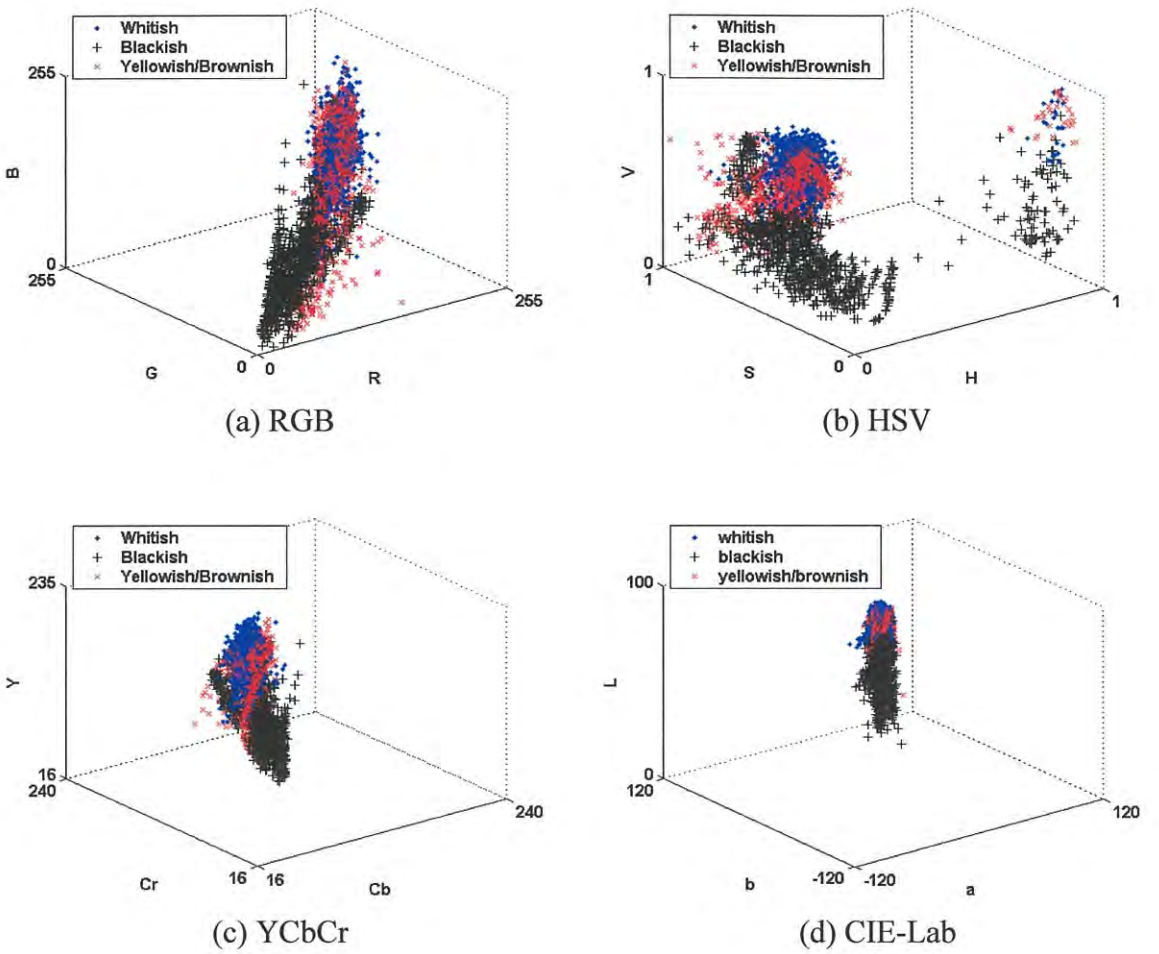
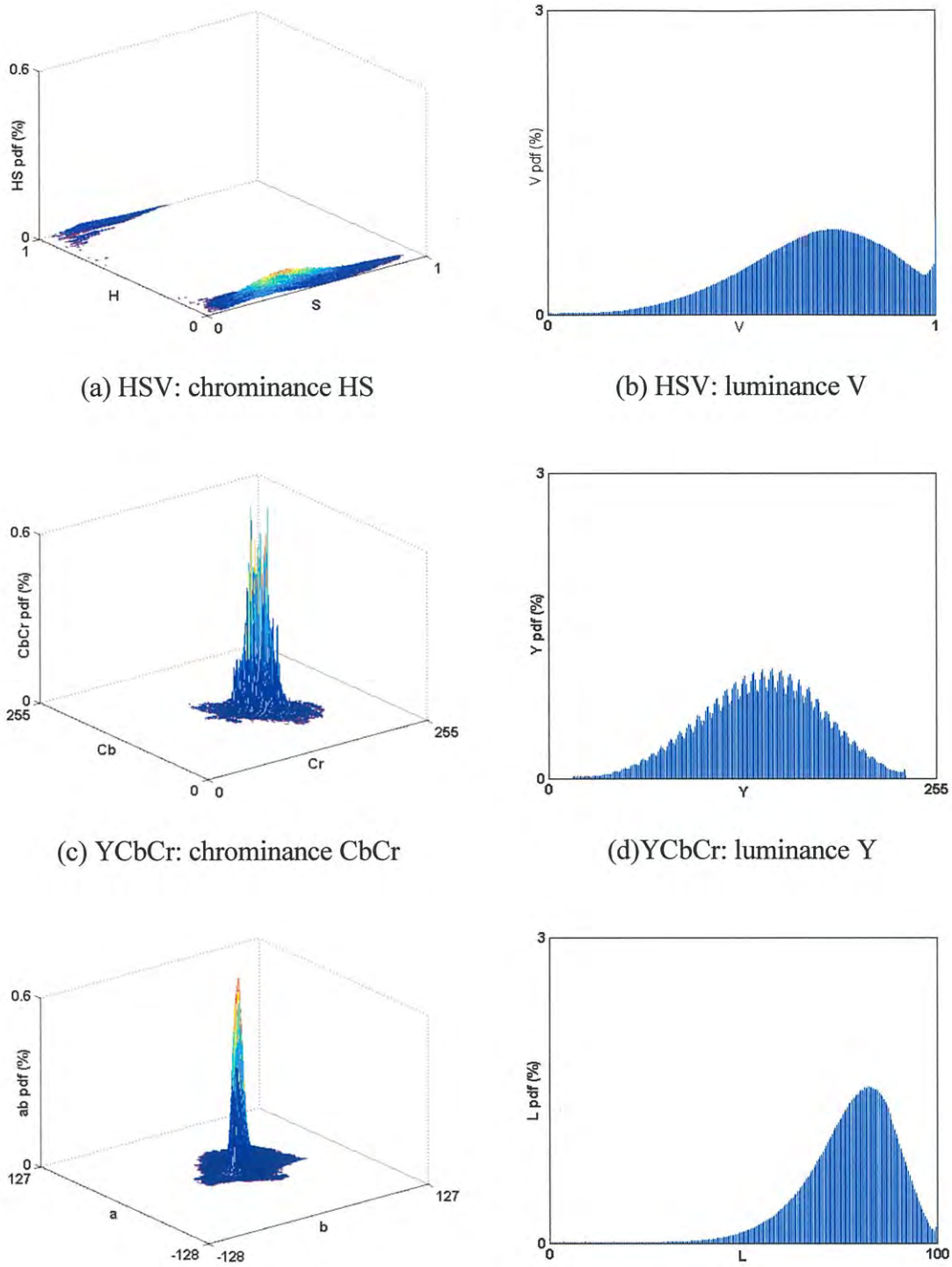


Figure 3.10: Distributions of whitish, blackish and yellowish/brownish skin colors.

3.3.3 Chrominance and Luminance of Skin Colors

The distributions of skin colors in the chrominance plane and in the luminance axis for the HSV, YCbCr and CIE-Lab color spaces are shown in Fig. 3.11. This figure shows that the luminance of skin color has a unimodal distribution. In addition, skin colors form compact clusters in the chrominance planes of the YCbCr and CIE-Lab color spaces. The distribution of skin colors in the HS plane of the HSV color space consists of two clusters in the high and low regions of H. These two regions will merge into one cluster if we use the angular representation, as shown in Fig. 3.3, of the hue value H.



(a) HSV: chrominance HS

(b) HSV: luminance V

(c) YCbCr: chrominance CbCr

(d)YCbCr: luminance Y

(e) CIE-Lab: chrominance ab

(f) CIE-Lab: luminance L

Figure 3.11: Chrominance and luminance distributions of skin colors.

3.4 Color Pixel Classification Algorithms

In this section, we describe five skin detection approaches that use color pixel classifications: piece-wise linear decision boundary, classifiers based on Gaussian densities, self-organizing maps, Bayesian classifier, and multilayer perceptrons.

3.4.1 Piece-wise Linear Decision Boundary

In this approach to color pixel classification, skin and nonskin colors are separated by a piece-wise linear decision boundary that is explicitly defined [15, 33, 106]. A simple technique to create a decision boundary is through thresholding the color channels. Because the luminance of skin colors is found to spread widely, this thresholding technique is only effective for chrominance channels. For example, Chai and Ngan [15] considered a color in the YCbCr space as a skin color if:

$$77 \leq Cb \leq 127 \text{ and } 133 \leq Cr \leq 173. \tag{3.9}$$

Sobottka and Pitas [106] defined the following ranges for skin colors in the HSV color space:

$$0.23 \leq S \leq 0.68 \text{ and } 0 \leq H \leq 50^{o}. \tag{3.10}$$

Garcia and Tziritas [33] proposed another decision boundary that is formed by eight planes in the YCbCr color space. The decision rule is summarized in Table 3.3.

Table 3.3: Garcia and Tziritas' piece-wise linear skin color decision boundary.

$Cb = Cb - 128$
$Cr = Cr - 128$
If $(Y > 128)$
$\theta_1 = -2 + (256-Y)/16;\ \theta_2 = 20 - (256-Y)/16;\ \theta_3 = 6;\ \theta_4 = -8$
Else
$\theta_1 = 6;\ \theta_2 = 12;\ \theta_3 = 2 + Y/32;\ \theta_4 = -16 + Y/16$
A skin color must satisfy the following conditions:
$Cr \geq -2(Cb+24);\quad Cr \geq -(Cb+17);\quad Cr \geq -4(Cb+32);\quad Cr \geq 2.5(Cb+\theta_1)$
$Cr \geq \theta_3;\quad Cr \geq 0.5(\theta_4 - Cb);\quad Cr \leq (220 - Cb)/6;\quad Cr \leq 4(\theta 2 - Cb)/3$

3.4.2 Classifiers based on Gaussian Densities

Classifiers in this category assume that the distribution of skin colors $p(\mathbf{x})$ has a parametric functional form. The most common parametric form is the Gaussian function. A Gaussian function, $g(\mathbf{x}; \boldsymbol{\mu}, \mathbf{C})$, is characterized by a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\mathbf{C}$:

$$p(\mathbf{x}) = g(\mathbf{x}; \boldsymbol{\mu}, \mathbf{C}) = (2\pi)^{-d/2} |\mathbf{C}|^{-1/2} \exp\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})\}, \tag{3.11}$$

where d is the dimension of the feature vector $\mathbf{x}$, and $|\mathbf{C}|$ is the determinant of matrix $\mathbf{C}$. The mean vector $\boldsymbol{\mu}$ and the covariance matrix $\mathbf{C}$ are estimated from a training set of skin color $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$:

$$\boldsymbol{\mu} = \underset{\mathbf{x}\in S}{E}\{\mathbf{x}\}, \tag{3.12}$$

$$\mathbf{C} = \underset{\mathbf{x}\in S}{E}\{(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\}, \tag{3.13}$$

where E is the expectation operator. Once $p(\mathbf{x})$ is computed as in (3.11), a color $\mathbf{x}$ is considered as skin color if:

$$p(\mathbf{x}) \geq \theta_{\mathrm{p}}. \tag{3.14}$$

where θ_{p} is a threshold value. Xu and Sugimoto [129] used a 3-D Gaussian in the RGB color space to model the skin colors of East Asian people. To cope with other skin types and lighting changes, Menser and Wien [72] used a 2-D Gaussian in the CbCr plane. The skin color decision boundary is an ellipse in 2-D, and an ellipsoid in 3-D.

In some chrominance planes, such as CbCr, the distribution of skin color can be modeled using a single Gaussian. However, this is not the case in other chrominance planes, such as uv (of CIE-Luv) or HS (of HSV). In such chrominance planes, a Gaussian mixture model would be more appropriate. A Gaussian mixture is a weighted sum of Gaussian functions:

$$p(\mathbf{x}) = \sum_{i=1}^{K} \pi_i g(\mathbf{x}; \boldsymbol{\mu}_i, \mathbf{C}_i), \tag{3.15}$$

where K is the number of Gaussians, and π_i, $\boldsymbol{\mu}_i$, $\mathbf{C}_i$ are, respectively, the weight, the mean, and the covariance of the ith Gaussian. The weights are positive and must satisfy

the condition: $\sum_{i=1}^{K} \pi_i = 1$. The parameter set $\Phi = \{ \pi_i, \boldsymbol{\mu}_i, \mathbf{C}_i \mid i = 1, 2, ..., K \}$ is usually estimated using the *Expectation/Maximization* (EM) algorithm [21]. The idea is to find iteratively a parameter set that maximizes the joint-probability of occurrence of the training vectors, or the following *log-likelihood* of the training set:

$$\mathcal{L} = \log \prod_{i=1}^{N} p(\mathbf{x}_i \mid \Phi) \tag{3.16}$$

The Gaussian mixture model of skin colors has been applied to the chrominance planes uv by Yang and Ahuja [131], and HS by Zhu *et al.* [138]. We reported in [82] a skin color model in the YCbCr color space which consists of 3 Gaussian clusters for three levels of luminance (low, medium and high).

3.4.3 Self-Organizing Maps

The self-organizing map (SOM) is an artificial neural network developed by Kohonen in the early 1980s [57]. It has found many applications in unsupervised clustering of high-dimensional data. Brown *et al.* [8] proposed two SOM-based schemes for classifying skin and nonskin colors. The first scheme uses only skin samples, and the second scheme uses both skin and non-skin samples. Brown *et al.* found that the second scheme performs better than the first, and both schemes show no preference to any of the four color spaces HSV, Cartesian HSV, TSL and normalized rg.

In Brown *et al.*'s approach, a SOM consisting of several neurons arranged in a grid of fixed topology is used. Each neuron is associated with a codebook vector, which is initialized randomly at the start of training. During training, input vectors are presented sequentially to the SOM. Each time, the winning neuron, which is the closest to the training vector according to some distance measure, is identified. The codebook vectors of the winning neuron and neurons in its neighborhood are updated as follows:

$$\boldsymbol{\mu}_{new} = \boldsymbol{\mu} + \alpha(\mathbf{x} - \boldsymbol{\mu}), \tag{3.17}$$

where $\boldsymbol{\mu}$ is the neuron codebook vector, and $\mathbf{x}$ is a training vector, and α is the learning rate. As training progresses, the learning rate and the neighborhood size are reduced. Once training is completed, each neuron is assigned a label (i.e. skin or nonskin) to

which it responds most frequently. A new vector is assigned with the label of the corresponding winning neuron. This scheme is similar to the *nearest-neighbor* classification algorithm. The only difference is that clustering is done by a SOM.

It should be pointed out that clustering in a SOM is driven by the distances between the input vectors rather than their class memberships. Kohonen [57] suggested that for classification tasks, a SOM-based architecture, known as the leaning vector quantization (LVQ), is more suitable. A LVQ network consists of two layers: a competitive layer that learns to cluster data according to the competitive learning rule, and a linear layer that maps the clusters to the final class. During training, the weight vector of the winning neuron is moved *towards* or *away* from a training vector, depending on whether or not the training vector is correctly classified.

3.4.4 Bayesian Classifier with Nonparametric Density Estimation

This classifier uses the Bayesian decision rule for minimum cost, which is a well-established technique in statistical pattern classification [21]. The Bayesian classifier has been used by a number of authors for skin detection [52, 83, 121]. It can be described as follows. Let ω_1 and ω_2 denote the skin color and nonskin color classes, respectively. Let $P(\omega_i)$, $i = 1, 2$, be the *a priori* probability of class ω_i. Let $p(\mathbf{x}|\omega_i)$ be the class-conditional probability density function (pdf) of class ω_i. Let λ_{ij}, $i, j = 1, 2$, be the cost of classifying a color $\mathbf{x}$ into class ω_i when the color actually belongs to class ω_j. Clearly, λ_{ij} is the cost of a correct decision when $i = j$, and the cost of an incorrect classification when $i \neq j$. A color pixel $\mathbf{x}$ is classified as skin color (class ω_1) if:

$$\frac{p(\mathbf{x}\,|\,\omega_1)}{p(\mathbf{x}\,|\,\omega_2)} \geq \theta_{\mathrm{pr}}, \tag{3.18}$$

where θ_{pr} is a threshold. The theoretical threshold value that minimizes the total classification cost is:

$$\theta_{\mathrm{pr}} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \times \frac{P(\omega_2)}{P(\omega_1)}. \tag{3.19}$$

In practice, this threshold is determined experimentally.

The class-conditional pdfs in (3.18) are computed using a histogram technique (i.e. nonparametric density estimation). From a training set of skin and nonskin colors, two histograms $h_1(\mathbf{x})$ and $h_2(\mathbf{x})$ are generated, where $h_i(\mathbf{x})$ is the count of pixels from class ω_i that have a value of $\mathbf{x}$. The class-conditional pdfs are obtained by normalizing the color histograms:

$$p(\mathbf{x}|\omega_i) = h_i(\mathbf{x}) / \sum_{\mathbf{x}} h_i(\mathbf{x}) . \tag{3.20}$$

The histogram technique for pdf estimation is viable in this classification problem because the feature vector $\mathbf{x}$ has a low dimension (at most 3). Implementation of the Bayesian classifier requires a very large labeled set of skin and nonskin samples. This is one of the reasons we have prepared such a large number (over 3000) of segmented skin images in the ECU database.

It can be argued that the Bayes decision rule optimizes the class separability in the sense of Bayes error. In this study, we use the Bayesian classifier to compare different choices of the feature vector $\mathbf{x}$ (i.e. different color spaces and chrominance planes). We also investigate the effect of the histogram size n (i.e. the number of histogram bins per dimension) on classification performance. Because of storage constraints, there is a need for using a small n. For example, for the RGB space a histogram needs 128MB if $n = 256$, and only 256KB if $n = 32$. The effects of the color space and histogram size are investigated in Section 3.5.

3.4.5 Multilayer Perceptrons

A. A Brief Introduction to Multilayer Perceptrons

Multilayer perceptron (MLP) is a neural network architecture that has been widely used in pattern classification [24]. A multilayer perceptron (Fig. 3.12) has three types of layers: an input layer, an output layer, and one or more hidden layers. Except for the input layer whose sole purpose is to receive an input vector from the environment, each layer in a MLP has a number of basic processing elements called *neurons*. The layers in

a MLP are connected in a feed-forward topology by weighted *synapses*, through which each neuron receives inputs from neurons in the preceding layer, and broadcasts its output to neurons in the succeeding layer.

The processing performed at each neuron can be described by the following equation:

$$o = f(\sum_i w_i p_i + b), \tag{3.21}$$

where o is the output of the neuron, f is its *activation function*, p_i's are the inputs received from the previous layer, w_i's are *connection weights* to the neuron, and b is a scalar factor called the *bias*. That is, the neuron applies its activation function on the weighted sum of its inputs to generate a scalar output. The network output is a vector consisting of the outputs of the neurons in the last layer[9]. This output vector is computed by propagating the network's input through intermediate hidden layers. In the MLP architecture, the number of layers, the number of neurons in each layers, and the type of activation functions used in each layer are determined by the designer. The connection weights and biases are found by means of a training algorithm.

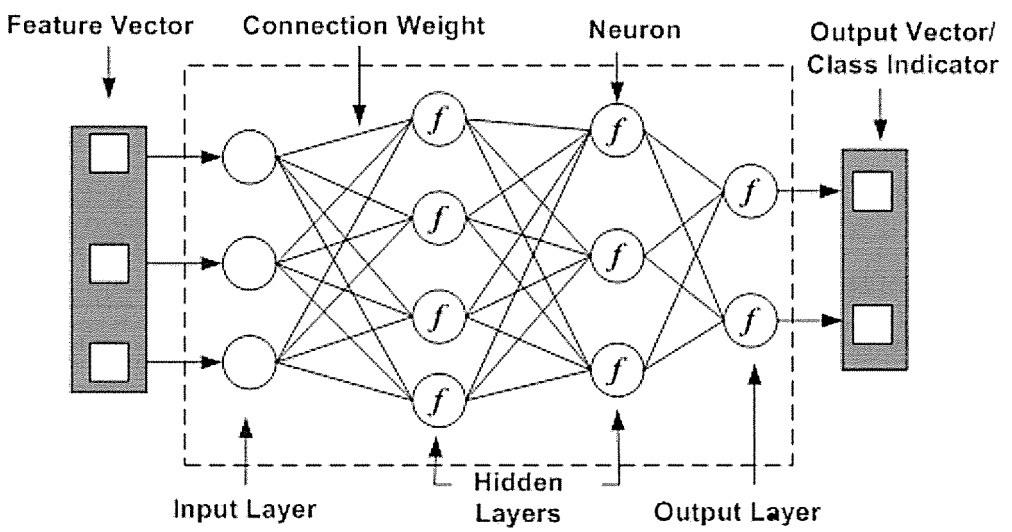


Figure 3.12: Multilayer perceptron architecture.

In pattern classification applications, the network input is a feature vector describing the pattern to be classified, and the network output indicates the class of the feature vector.

[9] This reduces to a scalar if the output layer contains a single neuron.

The network can generate meaningful output only after it is trained. The training process is iterative in nature, during which the network is presented with several training examples. Each training example is a pair of a feature vector and the desired (or target) output vector. During training, the network's adjustable parameters, namely the connection weights and biases, are systematically updated so that the network produces the expected output vectors for the given feature vectors in the training set. Updating network parameters is typically, but not necessarily, done through the *error back-propagation algorithm* (see, e.g. [24]). MLP networks are remarkable in that they have the ability to generalize from the training examples to correctly classify previously unseen feature vectors.

B. Color Pixel Classifier Using the MLP Networks

The color pixel classification technique presented is an extension of the neural-network approach we reported in [84]. In the following discussion, the RGB color space is used. However, this technique can be extended easily to any color space.

In the proposed MLP approach, the color vector **x** is normalized linearly so that its elements are in the interval [0, 1]. The structure of the MLP is chosen as follows. The input layer accepts the normalized color vector; the output layer has one neuron. The network has two hidden layers, each of which consists of a few neurons. The network output is a scalar that indicates if the network input is a skin color. The activation function is the hyperbolic tangent *tanh*. This activation function is differentiable and monotonically increasing, and it ensures that the network output is always in the interval [-1, 1]:

$$tanh(t) = (e^{t} - e^{-t})/(e^{t} + e^{-t}) \tag{3.22}$$

The training set consists of 4,096 training samples. During training, the expected outputs for skin and nonskin colors are set to +0.9 and -0.9, respectively. The training algorithm is Levenberg-Marquardt [38]. A validation set consisting of 32,768 samples is used to monitor the generalization performance of the network during training and to determine the appropriate network size.

We use a technique known as the *committee machine* to improve the robustness and the stability of the MLP classifier. Basically, the classification results of several individual networks are fused to form the final classification. In our approach, a large number of networks are trained, and N networks ($N = 5$) with the highest classification rates on the validation set are selected. Let r_i, $i = 1, 2, \ldots, N$, be the classification rate of the ith network on the validation set, and $y_i(\mathbf{x})$ be the output of the ith network. The output of the committee machine is the weighted average of the outputs of the individual networks:

$$y(\mathbf{x}) = \frac{\sum_{i=1}^{N} r_i y_i(\mathbf{x})}{\sum_{i=1}^{N} r_i}. \tag{3.23}$$

The input vector $\mathbf{x}$ is considered as skin color if

$$y(\mathbf{x}) \geq \theta_{\text{mlp}}, \tag{3.24}$$

where θ_{mlp} is a threshold that controls how conservative the classifier is. The various parameters in our MLP approach are summarized in Table 3.4.

Table 3.4: Settings of the MLP classifier.

Network Training	
Training set	4,096 samples
Validation set	32,768 samples
Training algorithm	Levenberg-Marquardt
Maximum training epochs	5,000
Target mean squared error	0.01
Network Structure	
Number of MLPs	5
Arbitration function	weighted average
Network input vector	(R G B) range [0, 1]
Network output vector	Scalar y in range [-1, 1]
Network size	First hidden layer: 6 neurons Second hidden layer: 6 neurons

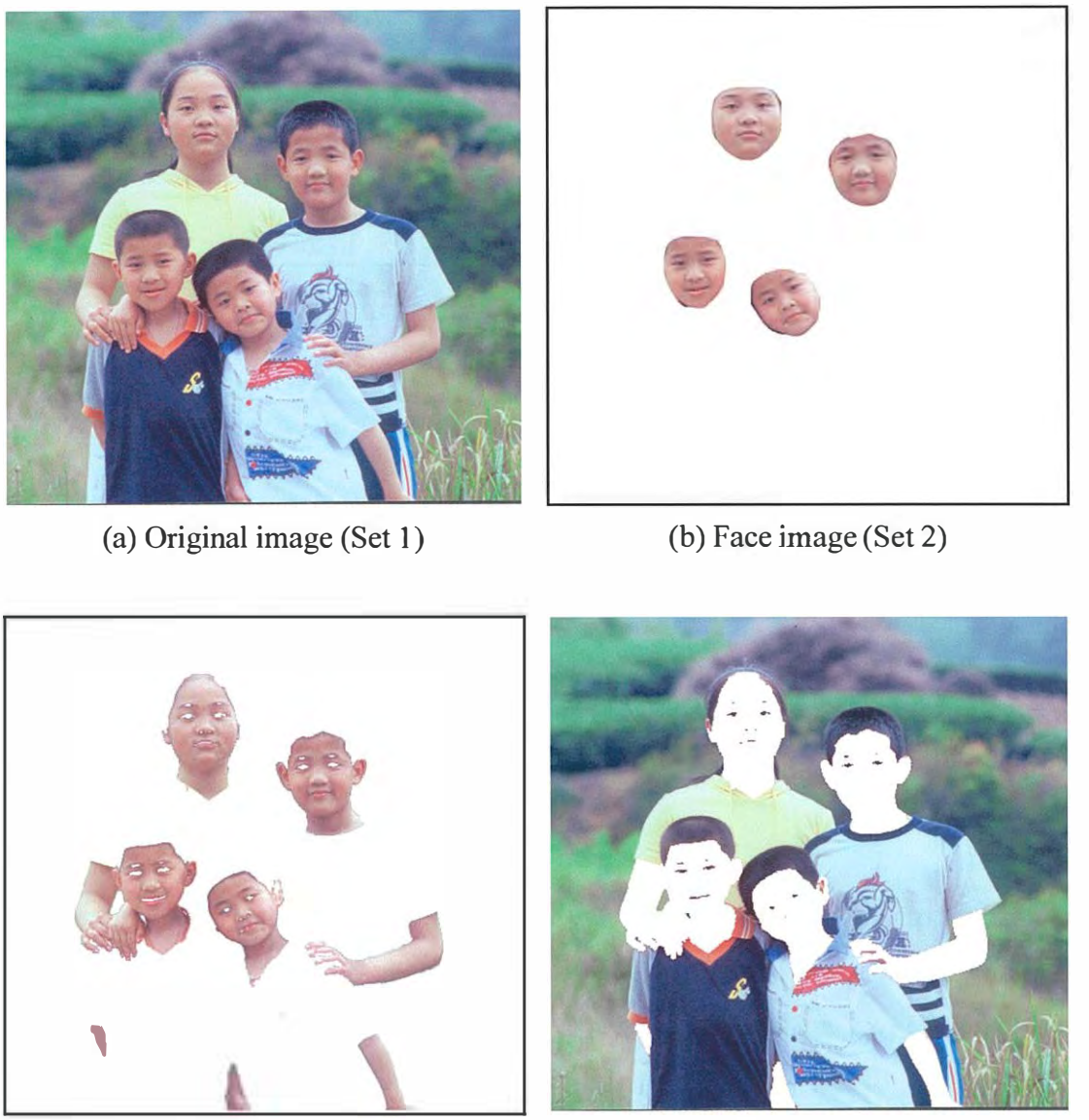
3.5 Results and Analysis

This section presents the results of several experiments that were carried out to analyze the skin detection performance of the color pixel classification algorithms described in the previous section. An overview of the experimental setup is given in Subsection 3.5.1. The Bayesian classifier with nonparametric density estimation and the effects of the histogram size and the color space on skin detection are analyzed in Subsection 3.5.2. Finally, the color pixel classification algorithms are compared in Subsection 3.5.3.

3.5.1 Data Preparation and Performance Measures

The data used in this section are taken from the ECU face detection database [79] that we have created at Edith Cowan University. This database differs from existing online face detection databases, such as the CMU and MIT databases, in several aspects. First, the ECU database contains color images, and hence supports color-based approaches to face detection. Second, the database is comprehensive in terms of both the number of images (4,000 images at the time of writing) and the image contents (varied background scenes, lighting conditions, faces, skin etc.). Third, we have segmented manually the images for *face* and *skin* regions (using the Adobe Photoshop software). The availability of a large number of ground-truth images enables the construction and reliable testing of skin detection and face detection algorithms. The segmented skin images consist of not only facial skin regions but also other skin regions, such as exposed neck, arms, and hands. More information about the database can be found in Appendix A, or at the database website [79].

We used 2500 images (images 1 to 2500) for training. The skin pixels (colors) were taken from database set 3, which consists of manually segmented skin images (Fig. 3.13c). The nonskin pixels were taken from complement images of the skin-segmented regions (Fig. 3.13d). The training set consisted of 116.6 million skin colors and 564.7 million nonskin colors.



(a) Original image (Set 1) (b) Face image (Set 2)

(c) Skin image (Set 3) (d) Nonskin image

Figure 3.13: Example images from ECU face detection database.

We used 500 images (images 2501 to 3000) to test skin detection performance of the color pixel classification algorithms described in Section 3.4. These images contained a wide range of skin types including whitish, blackish, yellowish, and brownish. The lighting conditions in these images varied from indoor office lightings to outdoor daylight. Images in the test set consisted of 27.1 million skin pixels and 97.5 million nonskin pixels.

The color pixel classifiers described in Section 3.4 were applied to the test images. The results reported in this section are from the raw classifier outputs; no extra processing

steps were applied. Each output image generated by a classifier was compared pixel-wise with the manually segmented skin image. The segmentation performance is measured in terms of the *correct detection rate* (CDR), *false detection rate* (FDR), and *classification rate* (CR):

$$\text{CDR} = \frac{\text{No. of skin pixels detected}}{\text{No. of skin pixels}} \times 100\% \tag{3.25}$$

$$\text{FDR} = \frac{\text{No. of nonskin pixels falsely classified}}{\text{No. of nonskin pixels}} \times 100\% \tag{3.26}$$

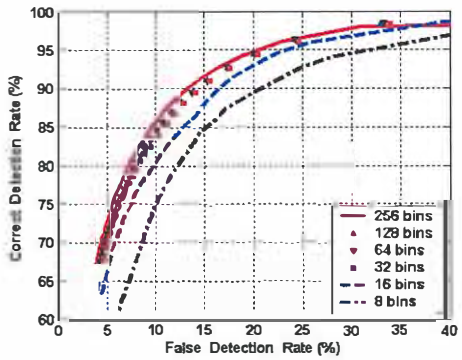
$$\text{CR} = \frac{\text{No. of pixels correctly classified}}{\text{No. of pixels}} \times 100\% \tag{3.27}$$

3.5.2 Analysis of Bayesian Color Pixel Classifier

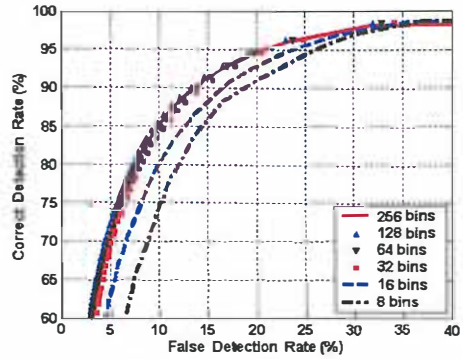
A. Histogram Size

We experimented with six dyadic histogram sizes: 256, 128, 64, 32, 16, and 8. The highest histogram size was so chosen because almost all hardware-realizable colors are encoded in the RGB color space with 8 bits per channel, which gives 256 different levels per channel. The *receiver operating characteristic*[10] (ROC) curves of the Bayesian classifier for different histogram sizes across 8 choices of feature vectors are shown in Fig. 3.14. The results show that there are only small differences in the classification rates when the histogram size n varies from 64 to 256. For the RGB, HSV, and YCbCr color spaces, the classification rates remain more or less the same down to a histogram size of 32. There is a decrease in the classification performance when n drops below 32; the chrominance-plane feature vectors (e.g. rg, HS, CbCr, and ab) are among the worst affected. The RGB, YCbCr and HSV feature vectors are more robust to changes in the histogram size compared to the CIE-Lab feature vector or chrominance-based feature vectors.

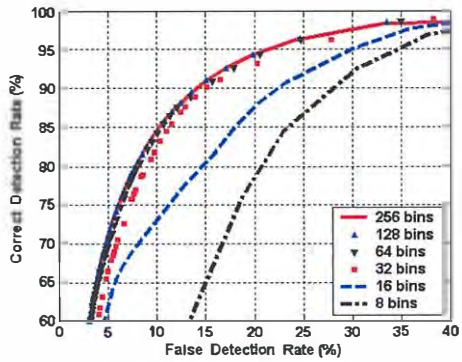
[10] Terminology borrowed from radar where a receiver is characterized by its ability to detect a transmitted signal.



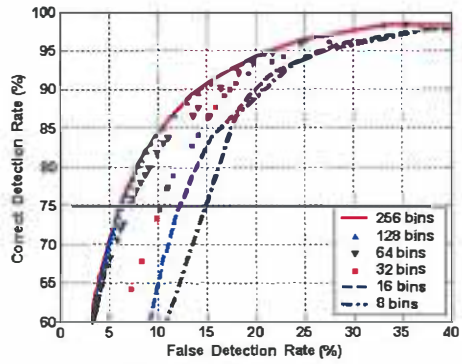
(a) RGB color space



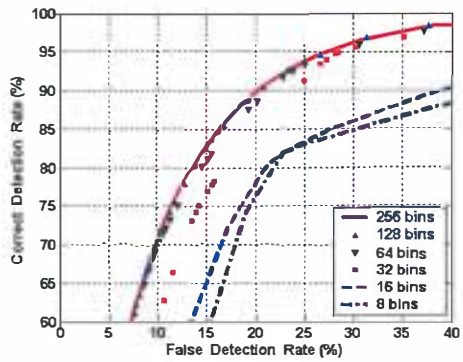
(b) HSV color space



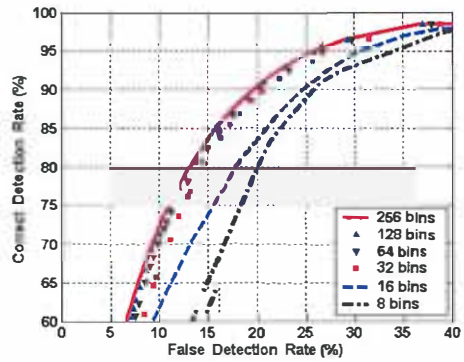
(c) YCbCr color space



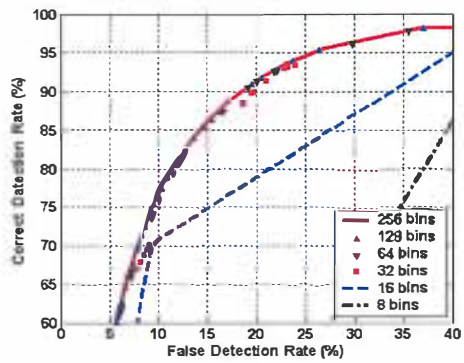
(d) CIE-Lab color space



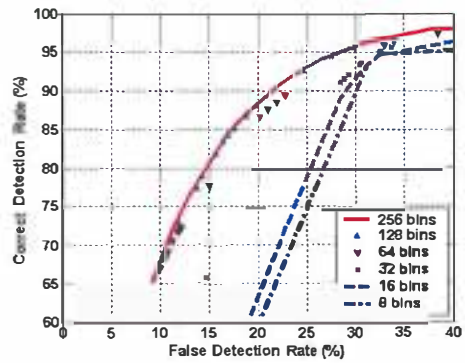
(e) rg plane (normalized RGB)



(f) HS plane



(g) CbCr plane

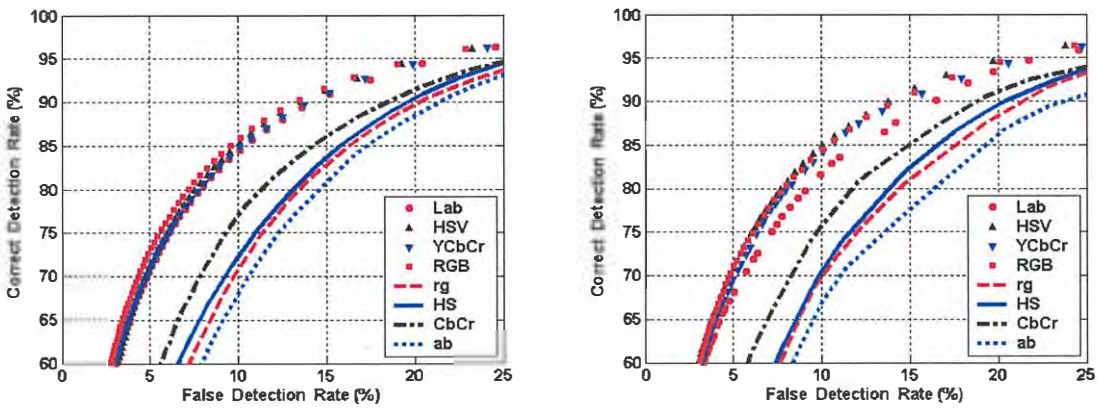


(h) ab plane

Figure 3.14: Effects of histogram size on skin detection.

B. Color Space

We compared eight choices of feature vectors RGB, HSV, YCbCr, CIE-Lab, normalized rg, HS, CbCr and ab. The ROC curves of the Bayesian classifier for these feature vectors are shown in Fig. 3.15. The results show that regardless of the histogram size, feature vectors consisting of all channels (i.e. RGB, HSV, YCbCr, and CIE-Lab) outperform feature vectors consisting of only chrominance channels (i.e. normalized rg, HS, CbCr, and ab). It has been suggested by many authors that skin color classification becomes more robust to the lighting intensity if the luminance is not included in the feature vector. We believe that such robustness to the lighting intensity is essentially the result of expanding the skin color decision boundary to cover the entire luminance channel. This comes at the cost of more false detections, which reduces the effectiveness of skin detection. Therefore, we propose that all color channels be used for accurate detection of skin pixels.



(a) histogram size $n = 256$ (b) histogram size $n = 64$

Figure 3.15: Effects of feature vector on skin detection.

Figure 3.15 shows that, for high histogram sizes ($n \geq 64$), the classification performances are almost the same for all tested color spaces (i.e. RGB, HSV, YCbCr, and CIE-Lab). This is an interesting result because the distributions of skin colors appear to be more compact in CIE-Lab and YCbCr colors spaces (see Fig. 3.9) than in other color spaces, such as RGB and HSV. This result shows that the overlap between skin and nonskin is not reduced by transforming to compact (skin-wise) color spaces, such as CIE-Lab or YCbCr. Therefore, skin color pixel classification can be done in any color space. The color space should be selected according to the format of the input

image or the need of subsequent processing so that the color conversion step is eliminated. Our conclusion agrees with a result published recently by Shin *et al.* [101]. The difference is that Shin *et al.* measured the skin and nonskin separability for different color spaces using metrics derived from the class scatter matrix and histograms, whereas in our work, the performances of different color spaces are directly evaluated on a large test set. Figure 3.15 also shows that the CbCr feature vector is significantly better than other chrominance-based feature vectors.

The classification rates of the Bayesian classifier in the RGB color space with $n = 64$ are listed in Table 3.5. Sample results of skin detection using the Bayesian classifier are shown in Figs. 3.17 and 3.18 in Section 3.6 of this chapter.

Table 3.5: Results of skin detection using the Bayesian classifier (RGB, $n = 64$).

False Detection Rate (%)	5	5.8	10	15	20
Correct Detection Rate (%)	71.0	74.0	84.4	91.0	94.5
Classification Rate (%)	89.7	89.8 [a]	88.7	86.1	83.1

[a] Maximum classification rate.

3.5.3 Comparison of Color Pixel Classification Algorithms

In this subsection, we compare the skin detection performances of the pixel color classifiers that are presented in Section 3.4:

- the rectangular decision boundary (**fixed-range**) in the CbCr plane of the YCbCr color space, proposed by Chai and Ngan [15].
- the **piece-wise linear** decision boundary in the YCbCr color space, used by Garcia and Tziritas [33].
- the **Gaussian** models. The 2-D Gaussian model in the CbCr plane, proposed by Menser and Wien [72], was used as an example.
- the **SOM** classifier proposed by Brown *et al.* [8]. Similarly to Brow *et al.*'s approach, we used a SOM that operated on the normalized rg plane and consists of 100 neurons arranged in a hexagonal grid. The winning neuron was selected based on the Euclidean distance. The label (i.e. skin or nonskin) of

the winning neuron was determined by the ratio of the number of skin training samples to the number of nonskin training samples that the neuron responded to.

- the **Bayesian** classifier using nonparametric density estimation. The performance of this classifier in different color spaces and at different histogram sizes is thoroughly investigated in the previous subsection. In this comparative study of classifiers, we used the Bayesian classifier in the RGB color space with a histogram size n of 64 bins per channel. At this histogram size, the storage requirement is 2MB. More accurate Bayesian classifiers are possible but at the cost of much larger memory requirements.
- the multilayer perceptron classifier (**MLP**), which is an extension our approach described in [84].

A. Skin Detection Accuracy

The ROC curves of the classifiers on the test set of 500 images are shown in Fig. 3.16. Because the parameters of the fixed-range and piece-wise linear classifiers are fixed, the corresponding ROC curves have only a single point. For the other classifiers, the classification rates were obtained by varying the decision thresholds on the continuous classifier outputs. The results in Fig. 3.16 show that the Bayesian classifier clearly outperforms all other classifiers in terms of classification accuracy. It has CDRs of 84.4% and 94.5% at FDRs of 10% and 20%, respectively (see also Table 3.5). For false detection rates between 5% and 20%, there is a difference of more than 5% between the correct detection rates of the Bayesian classifier and the other classifiers.

The MLP classifier is the next best in terms of classification accuracy. It performs consistently better than the SOM and the Gaussian classifiers. At the same false detection rates, the MLP classifier is better than the fixed range classifier, and almost the same as the piece-wise linear classifier. The fixed-range and piece-wise linear classifiers both operate in the region of high false detection rates (over 15%). The classification measures are a CDR of 94.2% at a FDR of 29.8% for the fixed-range classifier, and a CDR of 87.2% at a FDR of 19.0% for the piece-wise linear classifier. We can conclude that at high false detection rates, the manually constructed decision

boundaries of the fixed-range and piece-wise linear classifiers are comparable to the decision boundary produced by the MLP classifier.

The SOM classifier and the Gaussian classifier do not perform as well as the Bayesian and MLP classifiers. For the SOM classifier, this can be attributed to a number of factors. First, the class memberships of the training vectors are not taken into account during the unsupervised clustering step of SOM. Second, more neurons are probably needed to cover the input space. Lastly, as mentioned in Section 3.4, the learning vector quantization architecture is probably more suitable than the SOM for classification problems. Results of the Gaussian classifier indicate that the approximation of the skin color distribution as a Gaussian has some merit but is not sufficiently accurate for skin detection. In literature, the Gaussian model of the human skin color is usually applied to chrominance planes. However, we show in the previous subsection that classification using only two chrominance channels is not as good as classification using all three color channels. The CDRs of the Gaussian classifier are only 58.9% and 79.5% at FDRs of 10% and 20%, respectively.

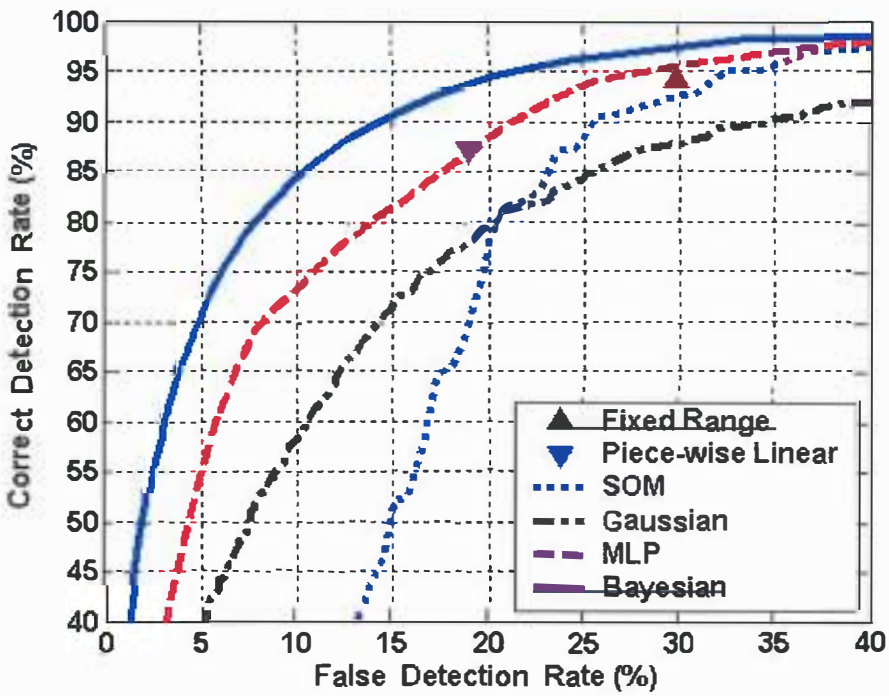


Figure 3.16: Comparison of color pixel classifiers in skin detection.

B. Memory Requirement

The memory requirements of the color pixel classifiers are shown in Table 3.6. The memory requirements are listed for the run-time classifiers that produce continuous outputs (i.e. continuous skin color scores). We find it desirable to keep continuous skin color scores rather than a binary 0/1 because in post-processing stages, the relative classification confidence reflected in these scores can be valuable. In our calculations, we assume that each double-precision floating-point value takes 8 bytes. The memory requirements for the fixed-range, piece-wise linear and Gaussian classifiers are almost negligible. The SOM and MLP classifiers require medium amounts of memory (between 2-3KB). In contrast, a good-performance Bayesian classifier with nonparametric density estimation requires at least 2MB of memory to store the matrix of skin color likelihood ratios (histogram size $n = 64$ bins/channel).

Table 3.6: Comparison of color pixel classifiers in terms of memory and speed.

Color Pixel Classifier	Memory (bytes)	Normalized Processing Time
Fixed-range [15]	32	1.0[d]
Piece-wise linear [33]	130	5.1
Gaussian [72]	50	35.6
SOM [8]	3,200[a]	186.1
Bayesian [52, 83, 121]	2,097,152[b]	2.1
MLP [84]	2,600[c]	73.7

[a] 100 neurons.
[b] 3-D color vector, n = 64 bins/channel.
[c] 5 MLPs, 1st and 2nd hidden layers have 6 and 5 neurons, respectively.
[d] Takes 0.11s to process an image of size 352×288.

C. Processing Time

Table 3.6 also shows the processing times of the color pixel classifiers. The classifiers were applied to process a color image of size 352×288, and the average processing times (over 10 runs) were recorded. In our experiments, MATLAB implementations of the classifiers running on a Pentium III 600MHz were used. However, to obtain a more machine-independent comparison of the algorithm speeds, the processing times were normalized against the shortest processing time. Results in Table 3.6 show that the

fixed-range classifier is the fastest classifier taking only 0.11s to process the 352×288 image. The Bayesian classifier is the next fastest, and is twice as fast as the Gaussian classifier and 36 times faster than the MLP classifier. The SOM classifier has the longest processing time.

3.6 Skin Detection Using Color

In this section, we formalize the skin detection approach that uses the Bayesian color pixel classifier. The analysis in the previous section has shown that the Bayesian classifier (with nonparametric density estimation) outperforms the other color pixel classifiers in terms of classification accuracy and speed. In addition, the Bayesian classifier is largely unaffected by the choice of the color space, provided that all three color channels are used. The Bayesian classifier has only one drawback in that it requires a significant amount of memory (≈ 2MB) for good performance. For the rest of this thesis, the Bayesian color pixel classifier, operating on the RGB color space and having a histogram size of 64 bins per color channel, is used. However, as mentioned earlier, the Bayesian classifier can be applied to any color space.

A. Formulation of the Skin Detection Approach

The skin detection approach can be described as follows. Let $\mathbf{I}$ be a color input image, and $\mathbf{I}(x,y)$ be the color of a pixel at location (x, y)[11]. Let $f_s(\mathbf{x})$ be the likelihood ratio function defined as follows:

$$f_s(\mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_1)}{p(\mathbf{x} \mid \omega_2)}, \tag{3.28}$$

where $p(\mathbf{x}|\omega_1)$ and $p(\mathbf{x}|\omega_2)$ are the probability density functions of skin and nonskin classes.

For each image pixel $\mathbf{I}(x, y)$, a skin score $\mathbf{S}(x, y)$ is computed as follows

[11] In this thesis, the image coordinate system is used. The coordinate origin is the top left of the screen. The horizontal coordinate, denoted by x or the first index, increases from left to right. The vertical coordinate, denoted by y or the second index, increases from top to bottom.

$$\mathbf{S}(x, y) = f_s(\mathbf{I}(x,y)). \tag{3.29}$$

The higher this score is, the more likely the pixel is a skin pixel. The skin scores $\mathbf{S}(x, y)$ of all pixels in the input image $\mathbf{I}$ can be treated as a gray-scale image $\mathbf{S}$ (i.e. image having only one channel). We use the term 'skin score image' to refer to this image. The skin score image is thresholded to obtain a binary mask $\mathbf{B}_{sc}$ for skin-colored regions[12]:

$$\mathbf{B}_{sc}(x, y) = \begin{cases} 1, & \text{if } \mathbf{S}(x, y) \geq \theta_{sc} \\ 0, & \text{otherwise} \end{cases}. \tag{3.30}$$

In our work, a threshold $\theta_{sc} = 1$ is used.

B. Skin Detection Results

The results of skin detection using the Bayesian color pixel classifier, in terms of classification rates, have been presented previously in Subsection 3.5.2. In this subsection, we present some visual results of the skin detection approach. The skin-detected images for three test images in the ECU face detection database are shown in Fig. 3.17. The test image contains different skin types, namely blackish, yellowish and whitish. In Fig. 3.17, we also report, for each test image, the amount of image search space that is reduced as a result of skin color pixel classification. This figure shows that the proposed Bayesian classifier can detect different skin color types very reliably. Furthermore, it reduces the search space for skin significantly.

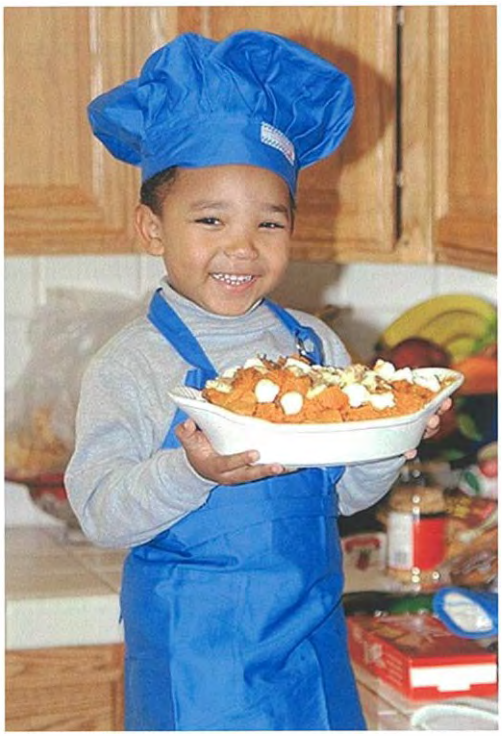
Results of skin detection on two additional test images are shown in Fig. 3.18. Again, all skin pixels are reliably detected. However, some nonskin regions that have skin-like colors, such as the wooden surface behind the boy in Fig. 3.18b, are incorrectly identified as skin. Because of this type of false detection, the amount of search space reduction becomes smaller. Nevertheless, the proposed skin detection technique proves to be very effective as an attention-focus step, considering that it relies on only one visual cue that is skin color.

[12] The subscript "sc" signifies that this is a binary mask created based on skin color.

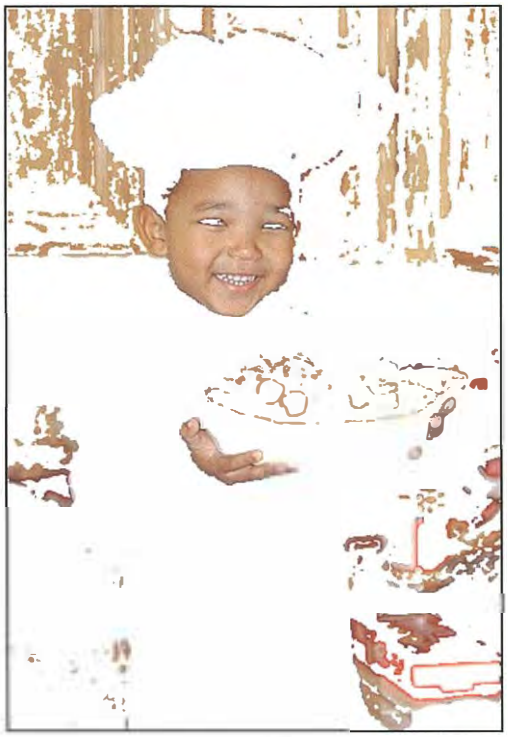
	Input Images	Skin Detection Outputs
Blackish Skin Color	 (a)	 (b) search space reduced by 87%
Yellowish Skin Color	 (c)	 (d) search space reduced by 90%
Whitish Skin Color	 (e)	 (f) search space reduced by 70%

Figure 3.17: Results of skin detection using Bayesian color pixel classifier – Part I.

Input Images	Skin Detection Outputs



(a)



(b) search space reduced by 81.1%

(c)

(d) search space reduced by 56.6%

Figure 3.18: Results of skin detection using Bayesian color pixel classifier – Part II.

3.7 Chapter Summary

This chapter addresses the skin detection approach that uses color pixel classification. The approach is based on the observation that the human skin has a very consistent and distinct color compared to other objects. In this chapter, we presented a comprehensive comparative analysis of several color pixel classifiers including classifiers with piece-wise linear decision boundaries, Gaussian-based classifiers, Bayesian classifier using nonparametric density estimation, and neural network classifiers. The Bayesian classifier has been found to outperform other classifiers in terms of segmentation accuracy and speed, but requires significantly more memory for good performance. The study of several color spaces has revealed that the skin detection approach is largely unaffected by the choice of color space, provided that all three color channels are used. Experimental results on a large database have demonstrated the viability of skin detection through color pixel classification.

Skin detection can be made more robust if other features of the human skin are taken into account. Nevertheless, the color-based approach to pixel-wise skin detection presented in this chapter is a crucial stepping stone towards efficient skin detection and face detection algorithms described in later chapters. We believe that the analysis of color pixel classification algorithms presented in this chapter will be useful not only for skin and face detection but also for the other computer vision tasks that need color-based object detection and recognition.

Chapter 4

Skin Region Segmentation

4.1 Introduction

In the previous chapter, we presented an approach to skin detection using color pixel classification, in which image pixels are divided into skin and nonskin pixels entirely on the basis of their colors. Extensive experiments have shown that this skin detection approach is attractive because skin pixels can be identified with almost minimal amount of computation. In the previous chapter, it was pointed out that false detections can occur because there are background objects with skin-like color. This chapter addresses post-processing techniques that aim to improve the results of pixel-wise skin detection by taking into account the texture property of the human skin. The work presented in this chapter has three major motivations: (i) to reduce false skin detections, (ii) to group skin pixels into homogenous skin regions for the face candidate selection stage (described in Chapter 5), and (iii) to adhere to the attention-focus strategy we have used so far, that is reducing search space significantly using computation-efficient image operations.

The current chapter is organized into five sections. A brief review of image segmentation techniques is given in Section 4.2. The proposed techniques for skin segmentation enhancement are presented in Section 4.3. An experimental analysis of

these techniques is provided in Section 4.4. Finally, the chapter summary is given in Section 4.5. Part of the work in this chapter has been reported in [81, 83].

4.2 Image Segmentation Techniques

Image segmentation is defined as the partition of a digital image into non-overlapping regions that are meaningful with respect to an application [13, 40]. General image segmentation techniques can be divided into four main categories [23]: thresholding, boundary-based, region-based, and hybrid techniques:

- **Thresholding techniques** assign pixels to different classes according to their gray level. They assume that the objects of interest and the image background have distinct ranges of gray level.
- **Boundary-based techniques** perform segmentation by locating the boundaries between regions. The region boundaries are often constructed through edge detection and edge linking.
- **Region-based techniques**, most notably *split-and-merge* and *region-growing* approaches, are generalizations of the thresholding techniques. In the split-and-merge approach, the input image is first divided recursively into non-overlapping and homogenous regions using visual features such as intensity, color or texture. An example of this step is the quadtree image decomposition. Next, the split regions are merged iteratively as long as some region homogeneity criteria are satisfied. In the region-growing approach, some initial pixels are identified as seed regions, and at each subsequent step, unlabeled pixels are added to these regions according a growth mechanism and a region homogeneity rule.
- **Hybrid techniques** use a combination of the above segmentation techniques.

4.2.1 Thresholding Segmentation

This technique works best when the object of interest and the background have homogenous but distinct gray levels. Suppose $\mathbf{F}(x, y)$ is an image that contains a bright object resting on relatively dark background. Such image can be segmented as follows:

$$G(x, y) = \begin{cases} 1, & \text{if } F(x, y) \geq T \\ 0, & \text{otherwise} \end{cases}. \tag{4.1}$$

The threshold T, which can be either fixed or adaptive, is often determined from the gray level histogram. For example, if an image contains one object and the background scene both having uniform gray levels, the image histogram will be bimodal; therefore, the object can be segmented with a threshold chosen between the two peaks of the histogram. A well-known adaptive thresholding technique is the watershed algorithm [13]. Thresholding is less effective for complex images that have wide-varying gray levels. However, if the objects of interest are different from the background in terms of other features such as color and texture, we can transform such features to gray-scale measures so that segmentation can be done in the new gray-scale domain.

4.2.2 Boundary-based Segmentation

As described above, boundary-based segmentation attempts to find and link edges to form region boundaries. Edges are modeled as abrupt transitions in the gray-scale image, which can be found by searching for either high gradient magnitudes or zero-crossings of the second-order derivatives. Well-known techniques for locating edges include Sobel, Prewitt, Roberts, Laplacian of Gaussian (LOG) and Canny edge detectors. Except for images of low complexity, detected edges are often noisy and do not form a closed curve. In most cases, only part of the detected edges belongs to region boundaries. Therefore, an important task of boundary-based segmentation is to form perceptual object boundary from the edge mask, and this often involves a boundary tracking process. Boundary tracking techniques, such as maximum gradient tracer and the tracking bug, are described in [13]. Boundary-based segmentation is often used in conjunction with other segmentation techniques.

4.2.3 Region-based Segmentation

An example of region-based image segmentation is the algorithm by Salgodo *et al.* [95]. The algorithm consists of two stages – a detailed global segmentation followed by a merging procedure. In the first stage, the original image is repeatedly subsampled by a

factor of 2 to form an image pyramid. The image at the top of the pyramid is segmented into different classes using a probabilistic relaxation labeling technique.

The probabilistic relaxation labeling technique can be described as follows. The number of classes M and the class-representative intensities are determined automatically from the image histogram. Each image pixel (x_n, y_n) is assigned an M-dimensional *probability vector* $\mathbf{P}_n$:

$$\mathbf{P}_n = (P_{n1}, P_{n2}, \ldots, P_{nM}), \tag{4.2}$$

where $P_{n\alpha}$ is the probability that the pixel belongs to class α. Similarly, to each image pixel, an M-dimensional *compatibility vector* is assigned; each element of this vector measures the degree of support received from the neighboring pixels to classify the pixel into a particular class. The probability vector is initialized according to the differences between the pixel intensity and the class-representative intensities. The compatibility vector is computed as the average probability vector of all pixels in the neighborhood of the current pixel. The probability and compatibility vectors for all image pixels are updated iteratively until they are stabilized. Each image pixel is then assigned to the class with the highest entry in the probability vector.

The segmented image at the pyramid top is projected iteratively to lower levels of the pyramid. At each level, the uncertainty area caused by downward projection is handled using the same probabilistic relaxation labeling technique described above. In the second stage, transitional or frontier regions are identified based on a global contrast measure, and then merged to the appropriate adjacent region according to a local contrast measure. A region having a low global contrast and surrounded by a single region is considered as a noise region; it is subsequently merged into its surrounding region. Finally, adjacent regions are joined if their frontiers are short and the average gradient over their frontiers are small.

4.2.4 Hybrid Segmentation

An example of hybrid image segmentation is the algorithm proposed by Fan *et al.* [23]. First, color edge pixels are detected using gradient operators and entropic thresholding. The edge masks of three image channels Y, U and V are combined to form the final

edge mask. Connected edge pixels are grouped into edge regions, and adjacent edge regions are identified. The midpoints between these edge regions are taken as seed regions, which are then grown pixel-by-pixel. At each step, a pixel is assigned to the adjacent region that corresponds to the minimum distance between pixel color and region centroid color. Next, the set of pixels on a region boundary is compared with the set of color-edge pixels in that region. Pixels that belong to only one of these two sets are deemed as uncertain pixels, which are then refined on the basis of their neighbors.

4.2.5 Existing Skin Segmentation Techniques

As mentioned in the last chapter, the vast majority of skin segmentation techniques are based on skin color [4, 8, 25, 43, 127]. There are a number of techniques that extend the color-based pixel-wise segmentation by taking into account the labels of adjacent pixels. For example, Chen and Chiang [18] considered an image pixel as a skin pixel if more than 4 of its neighbors have skin colors. In addition, skin regions with size smaller than 5×5 are deleted. Feris *et al.* [26] used a morphological closing operator and a median filter to refine skin-colored regions. Regions with a size smaller than 1% of the input image are eliminated. Cai *et al.* [10] selected an optimal threshold for skin color likelihood by monitoring the change in the skin-colored region size as the threshold varies. These authors suggested that the change in the region size reaches a minimum near the optimal threshold.

The face detection algorithm proposed by Albiol *et al.* [1] consists of a skin segmentation stage, in which skin colors in the input image are first detected using a Gaussian mixture model in the Cb-Cr plane (YCbCr color space). Once skin-colored pixels are detected, a 2-D histogram of these pixels in the Cb-Cr plane is constructed and treated as an image. This image is segmented using a watershed algorithm with markers being all local maxima in the histogram. Results of histogram segmentation are used to partition skin-colored pixels into homogenous regions.

More complex techniques use skin texture for segmentation. Fleck *et al.* [27] considered only skin-colored pixels that have small texture amplitude as skin pixels. Garcia and Tziritas [33, 34] proposed a skin segmentation approach that involves color quantization

and region merging. In their approach, pixel colors in the input image are mapped to a set of 16 dominant colors (in the HSV color space), which are found through a color clustering algorithm. Next, skin colors are detected using a piece-wise linear model of skin color. Adjacent skin-colored regions are iteratively merged if their dissimilarity is small. The following dissimilarity measure between two regions was used:

$$Dr(R_i, R_j) = \alpha|H_i - H_j| + \beta|S_i - S_j| + \gamma|H_i - H_j|, \tag{4.3}$$

where R_i and R_j are two adjacent regions, (H_i, S_i, V_i) is the average color vector of region R_i, and α, β, and γ are scalar weights. The color vector quantization technique increases the homogeneity of skin regions, but it leads to poor detection of skin colors (we showed in Chapter 3 that using fewer than 32 quantization bins per color channel results in low skin detection rates). In addition, using the global measure in (4.3) two different regions may have a very low dissimilarity measure because of the averaging of colors in each region.

4.3 Proposed Skin Region Segmentation Algorithm

The techniques presented in this section take into account the texture property of the human skin. It is found that the skin has a smooth texture. In addition, we find that the face candidate selection approach described in Chapter 5 is most effective when detected skin (including possibly false detections) is separated into homogenous regions. That way, facial features, namely the eyes, can be identified with fewer false detections. The proposed skin region segmentation algorithm consists of three main stages: *color-based skin detection*, *skin region verification*, and *skin region refinement*.

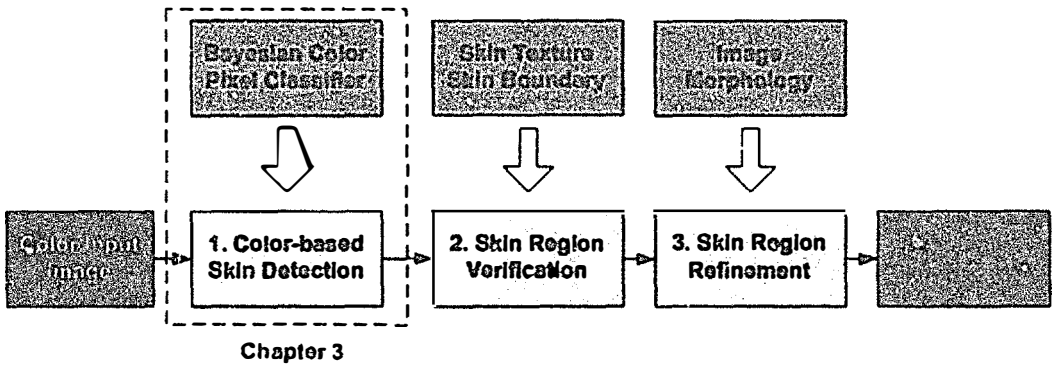


Figure 4.1: Block diagram of the proposed skin segmentation algorithm.

The color-based skin detection stage is the focus of Chapter 3. In this section, we describe the last two stages of the proposed skin region segmentation algorithm. Let **I** be a color input image. The outcomes of the color-based skin detection stage are:

- a skin color score image **S** whose entry $\mathbf{S}(x, y)$ of the color-based skin score image indicates the likelihood of pixel $\mathbf{I}(x, y)$ having a skin color; and
- a color-based skin mask $\mathbf{B}_{sc}$ whose entry $\mathbf{B}_{sc}(x, y)$ has a value of either 1 or 0, indicating whether or not the pixel $\mathbf{I}(x, y)$ is a skin color. This mask is obtained from the score image **S** using a threshold $\theta_{sc} = 1$.

4.3.1 Skin Region Verification

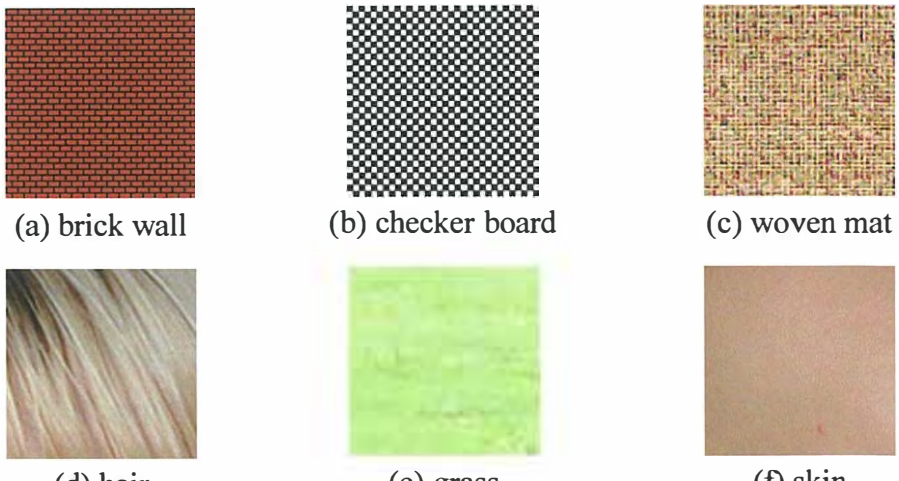
This stage is based on two properties of the human skin: (i) the human skin has a relatively smooth texture; and (ii) there exists, in most cases, a boundary between the actual skin region and the image background. Two skin verification techniques based on these properties are presented next.

A. Skin Texture Verification

Texture is an intuitive property of the surface of an object, yet there is no precise and unique definition of texture [107]. This is because there is a wide variability in textures. Pitas [87] defined texture as "a measure of image coarseness, smoothness and regularity", whereas Forsyth and Ponce [28] considered image textures as "organized patterns of quite regular subelements". Texture is made up from a large number of

primitive elements or *textels*[13]. Some textels are well-defined, and follow a regular pattern; others are less well-defined, and arranged in a less ordered pattern. Two features are often associated with a texture: *tone* and *structure*. Texture tone is defined based on color or intensity properties of pixels in each textel, whereas texture structure is related to the placement of textels. For regular textures, such as a brick wall and checker board (Fig. 4.2a,b), texture structure is the dominant feature, and the textures are known as macro-textures. In contrast, for irregular and unordered textures, such as hair and grass (Fig. 4.2d,e), texture tone is the dominant feature, and the textures are known as micro-textures. Examples of different textures are shown in Fig. 4.2.

The different approaches to texture representation can be divided into three main categories: *statistical*, *structural* and *filter-based*. Statistical approaches (see e.g. [40]) characterize texture using statistical measures such as mean, variance, skewness, kurtosis, and entropy. These measures are computed from a variety of sources including the gray-level histogram, the gray-level difference histogram, the gray-level co-occurrence matrices, the autocorrelation function, or the spectral power density function. Structural approaches describe textures in terms of textels and the placement rules [66]. Filter-based approaches apply a series of linear filters on the image. Texture features are then extracted from the filter responses. Recently, there has been a growing emphasis on the use of wavelets for multi-scale representations of texture [74, 137].



(a) brick wall (b) checker board (c) woven mat

(d) hair (e) grass (f) skin

Figure 4.2: Texture examples.

[13] The term textel stands for texture element.

The human skin texture is due to the relatively unordered grouping of several pixels that are similar in color and intensity. Therefore, the dominant feature of the skin texture is texture tone, for which simple statistics computed from local windows are sufficient to describe. Our method can be described as follows. Let **C** be a channel of the input color image. For each pixel location (x, y) in **C**, the standard deviation of its $W \times W$ neighborhood is computed:

$$\sigma(x, y) = \sqrt{\frac{1}{W \times W} \sum_{(i,j) \in N(x,y)} (C(i, j) - \mu)^2}\ , \tag{4.4}$$

where μ is the mean intensity in the neighborhood $N(x,y)$ at of pixel (x, y). The standard deviations for all pixels in **C** is treated as an image $\boldsymbol{\sigma}$. This image is thresholded to form a binary mask $\mathbf{B}_h(x, y)$ for homogeneous regions:

$$\mathbf{B}_h(x, y) = \begin{cases} 1, & \text{if } \boldsymbol{\sigma}(x, y) \le \theta_h \\ 0, & \text{otherwise} \end{cases}. \tag{4.5}$$

The standard deviation is very small for the interior of a skin region. However, this measure can be unstable for skin pixels near the skin region boundary because the local windows at these pixels may straddle over the inhomogeneous regions including skin, skin boundary, and possibly nonskin regions. To address this issue, we select a small window size of 3×3. This window size is sufficiently large to describe intensity variations in the local neighborhood of each pixel. The standard deviations are computed using an efficient algorithm that is based on linear filtering (see Table 4.1).

Table 4.1: Computing standard deviation image.

Input:	Color channel $\mathbf{C}(x, y)$, window size $W \times W$
Output:	Image of standard deviations $\boldsymbol{\sigma}(x, y)$.
Algorithm:	
Step 1: Construct an averaging filter mask **h** of size $W \times W$.	
Step 2: Compute image of squared intensities: $\mathbf{C}_2(x, y) = \mathbf{C}(x, y)^2$	
Step 3: Apply the averaging mask to the gray-scale and squared-intensity images:	$\boldsymbol{\mu} = \mathbf{C} * \mathbf{h}$, and $\mathbf{m}_2 = \mathbf{C}_2 * \mathbf{h}$
Step 4: Compute the image of standard deviations:	$\boldsymbol{\sigma}(x, y) = \sqrt{\mathbf{m}_2(x, y) - \boldsymbol{\mu}^2(x, y)}$

So far, the standard deviation image is computed only for one color channel. We find that non-uniform region detection can be improved if all three color channels are used. Figure 4.3 illustrates that using three color channels we can locate non-uniformity that is not evident in the gray-scale version of the input image. A homogeneity binary mask is computed for each image channel as in (4.5). The final homogeneity mask is the combination, through logical AND, of the three homogeneity masks. The result of detecting smooth regions in another test image is shown in Fig. 4.4.

(a) input image

(b) skin texture using gray-scale

(c) skin texture using all color channels

Figure 4.3: Comparison of using gray scale and all channels for finding skin-textured regions. White pixels indicate homogenous regions.

(a) input image (b) skin-texture region

Figure 4.4: Finding skin-textured regions. White pixels indicate homogenous regions.

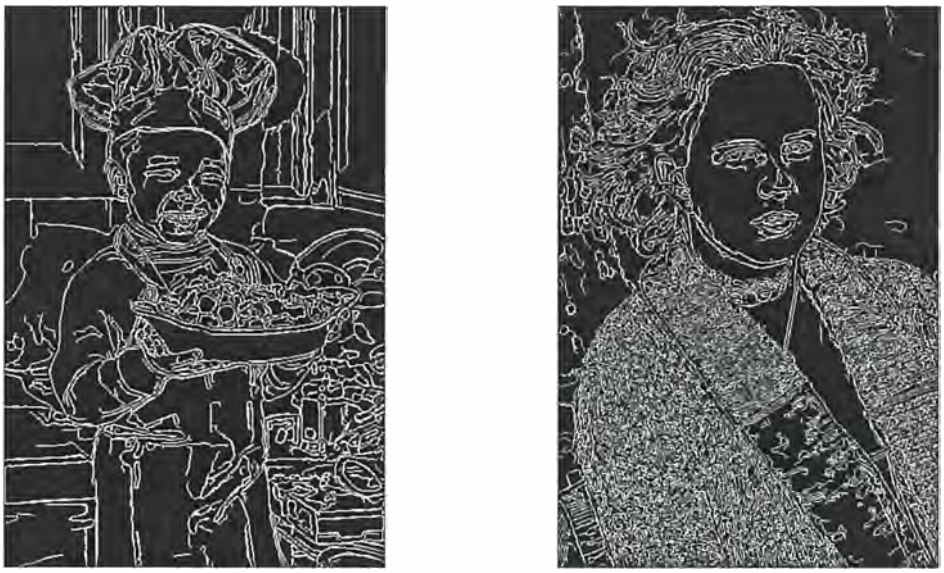
B. Finding Skin Region Boundary

The skin texture test in (4.5) eliminates most non-uniform image regions. In an unconstrained input image, there may be nonskin regions that have relatively low standard deviations $\boldsymbol{\sigma}(x, y)$. Common examples of such regions are smooth painted walls and wooden surfaces. We believe that a much more complex texture model is needed to distinguish reliably the texture of such nonskin regions from the skin texture. Here, we adopt an alternative strategy that aims to separate skin regions from regions of similar textures. The separation is achieved by first detecting edge pixels and then removing such pixels from the skin mask image. The end results of combining the skin color test, the skin texture test, and the skin boundary test are skin regions that can be treated separately during later stages of face detection.

In our work, edge detection is done using the Canny edge detector [12], which works as follows. The input image is convolved with two Gaussian derivative filters (x and y direction). The gradient magnitude and direction at each pixel location are computed from two filter outputs. Edges are found by looking for local maxima of the gradient

magnitude. Two magnitude thresholds are used to detect strong and weak edges. Weak edges are included in the output only if they are connected to strong edges.

We find that using all color channels in edge detection is better than using only one color channel or gray-scale image. The reason is that weak edges in one color channel can still be visible in other channels. Therefore, the Canny edge detector is applied to the three color channels of the input image. A final edge mask $\mathbf{B}_{edge}(x, y)$ is formed by combining the three edge masks via logical OR operator. Results of detecting skin boundary in two test images are shown in Fig. 4.5.



(a) for input image in Fig. 4.3a (b) for input image in Fig. 4.4b

Figure 4.5: Finding skin region boundary.

An image pixel is considered as skin pixel if it satisfies the following conditions:

(i) it has skin colors (binary mask $\mathbf{B}_{sc}$)

(ii) its local neighborhood is homogenous (binary mask $\mathbf{B}_h$)

(iii) it is not an edge pixel (complement of binary mask $\mathbf{B}_{edge}$)

Mathematically, a binary mask $\mathbf{B}_s$ for skin regions can be defined as:

$$\mathbf{B}_s = \mathbf{B}_{sc} \cap \mathbf{B}_h \cap \overline{\mathbf{B}_{edge}} \tag{4.6}$$

The edge pixels are removed from the skin binary mask in order to separate skin-colored regions that are dissimilar in texture and color.

4.3.2 Skin Region Refinement

A. Connected-Component Labeling

The skin binary mask $\mathbf{B}_s$ consists of 1's and 0's for skin and nonskin pixels, respectively. Objects in the binary mask are labeled using a connected-component labeling algorithm [40]. This labeling operation can be considered as the partitioning of the binary mask into a set of smaller disjoint regions $\mathbf{B}_{s,i}$. The disjoint regions $\mathbf{B}_{s,i}$ are processed separately in the next stage (Chapter 5) of our face detection algorithm.

B. Noise Removal in Skin Binary Mask

Noise in the binary mask, which is caused by a few skin-colored background pixels, tends to scatter and has small area. This suggests the following noise removal techniques. Any skin region $\mathbf{B}_i$ with area, compared to the largest skin area, that is below a threshold is removed. We find that an area threshold of $\theta_{area} = 1\%$ works quite well in most cases. In addition, regions whose area reduces to less than $\theta_{erosion} = 10\%$ after a morphological erosion operation are removed. This technique can deal with regions of sparse density.

C. Region-growing Using Image Morphology

To locate the boundaries between skin and nonskin regions, and to remove non-homogenous regions, aggressive thresholds are used in the skin verification stage. A side-effect of this approach is that a significant number of pixels within each skin region are removed during the verification of skin homogeneity and the detection of skin boundary. These skin pixels can be recovered as follows. For each detected skin region $\mathbf{B}_i$, the corresponding skin color mask $\mathbf{B}_{sc,i}$ is identified. The skin region mask $\mathbf{B}$ is enlarged within its bounding rectangle through a morphological dilation operation. All holes within the enlarged skin mask are filled. The enlarged and filled skin mask is combined with the skin color mask $\mathbf{B}_{sc,i}$, using the logical AND operator, to form the output skin mask for the region.

Another post-processing step is to remove skin regions that fall mostly inside another skin region. These enclosed regions may be present due to over-segmentation of a skin region. The complete skin segmentation algorithm is given in Table 4.2.

Table 4.2: Propose skin region segmentation algorithm.

Input Image $\mathbf{I}(x, y)$ in the RGB color space (or any other color space)

Output Binary masks $\mathbf{B}_i$, $i = 1, 2, \ldots$ for skin regions

Algorithm

STAGE I - Skin Color Detection

Step 1: Compute skin color score image $\mathbf{S}(x,y)$ and skin color mask $\mathbf{B}_{sc}$ as described in Chapter 3.

STAGE II – Skin Region Verification

Step 2: Compute standard deviation images $\sigma(x, y)$ for 3 image channels. Threshold these images with $\theta_h = 12$ to obtain three homogeneity masks. Combine these masks using logical AND operator to form the final homogeneity mask $\mathbf{B}_h$.

Step 3: Apply the Canny edge detector on three image channels. Combine the three edge masks using logical OR operator to obtain the final edge mask $\mathbf{B}_{edge}$.

Step 4: Form the skin binary mask $\mathbf{B}_s = \mathbf{B}_{sc} \cap \mathbf{B}_h \cap \overline{\mathbf{B}_{edge}}$

STAGE III – Skin Region Refinement

Step 5: Label connected components of the skin mask $\mathbf{B}_s$. Remove regions smaller than $\theta_a = 1\%$ of the largest region, and regions whose area reduces to less than $\theta_{erosion} = 10\%$ after a morphological erosion operation.

Step 6: For each remaining region $\mathbf{B}_{s,i}(x, y)$:

- Obtain the corresponding skin color mask $\mathbf{B}_{sc,i}(x, y)$,
- Morphologically dilate $\mathbf{B}_{s,i}(x, y)$ within its bounding rectangle and fill all holes in the dilated mask.
- Obtain refined mask for the skin region:

$$\mathbf{B}_{s,i}(x, y)\ (refined) = \mathbf{B}_{sc,i}(x, y) \cap \mathbf{B}_{s,i}(x, y)\ (enlarged)$$

Step 7: Remove skin regions that fall mostly inside another skin region.

4.4 Results and Discussion

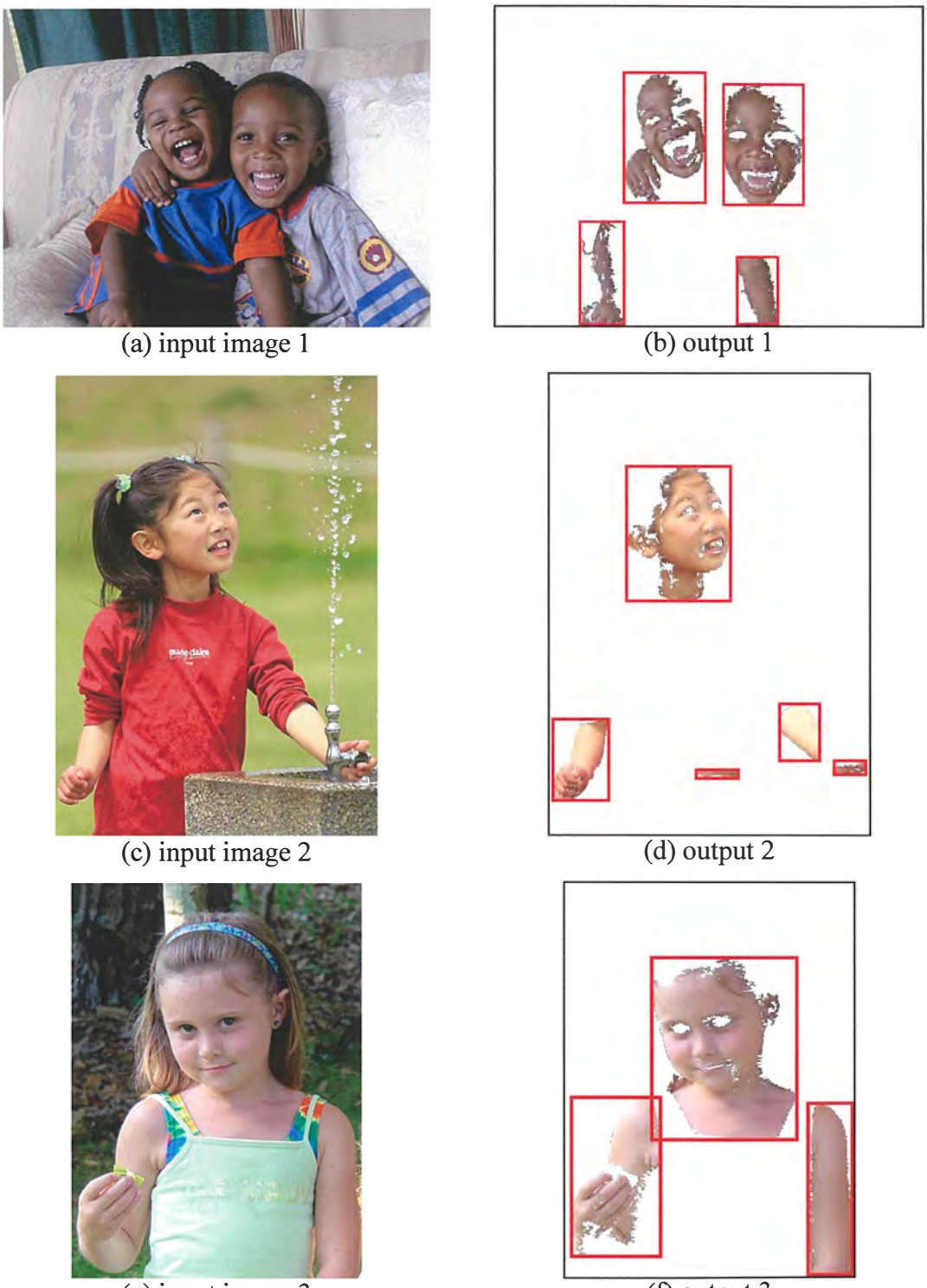
The proposed skin region segmentation algorithm was tested on 500 images in the ECU database (images 2501-3000). These images were segmented manually for skin regions so a pixel-wise comparison between the algorithm outputs and the segmentation ground-truths is possible. The various classification rates are defined in the same way as in (3.25)-(3.27). The classification rates for the proposed segmentation algorithm are listed in Table 4.3. The classification rates of the skin detection approach using only color pixel classification (i.e. the approach described in Chapter 3) are also included in the Table for comparison. The table shows that the proposed skin region segmentation algorithm, by using the texture property of the human skin, does improve skin detection. For example, at a FDR of near 10%, the proposed algorithm has a CDR of 89.1% compared to a CDR of 84.4% by the approach of using only skin color. The classification rate of the proposed algorithm (the last column) is higher than both cases of skin detection using only color pixel classification.

Table 4.3: Comparison of skin region segmentation approaches.

	CDR (%)	FDR (%)	CR (%)
Skin detection using color pixel classification: $\tau = 2$	84.4	10.1	88.7
Skin detection using color pixel classification: $\tau = 1.2$	89.4	13.7	86.9
Proposed skin region segmentation algorithm	89.1	9.9	89.9

Results of skin region segmentation performed on three test images are shown in Fig. 4.6. Each segmented skin region is enclosed by a red rectangle. In all cases, skin regions are correctly detected, and the actual skin regions are well separated from false detections. This is a very desirable property of the proposed algorithm because we can treat each detected skin region (enclosed in a red rectangle) independently of other skin regions. Additional results on two test images are shown in Fig. 4.7. There are a number of falsely detected skin regions in the output in Fig. 4.7b. These false detections can be

removed if more aggressive thresholds are used in our algorithm. However, we decide to keep these false detections at this stage: they can be discarded more easily in later stages of face detection using more *a priori* information about the face pattern.



(a) input image 1 (b) output 1

(c) input image 2 (d) output 2

(e) input image 3 (f) output 3

Figure 4.6: Skin region segmentation examples – Part I.

Input Images	Skin Region Segmentation Results

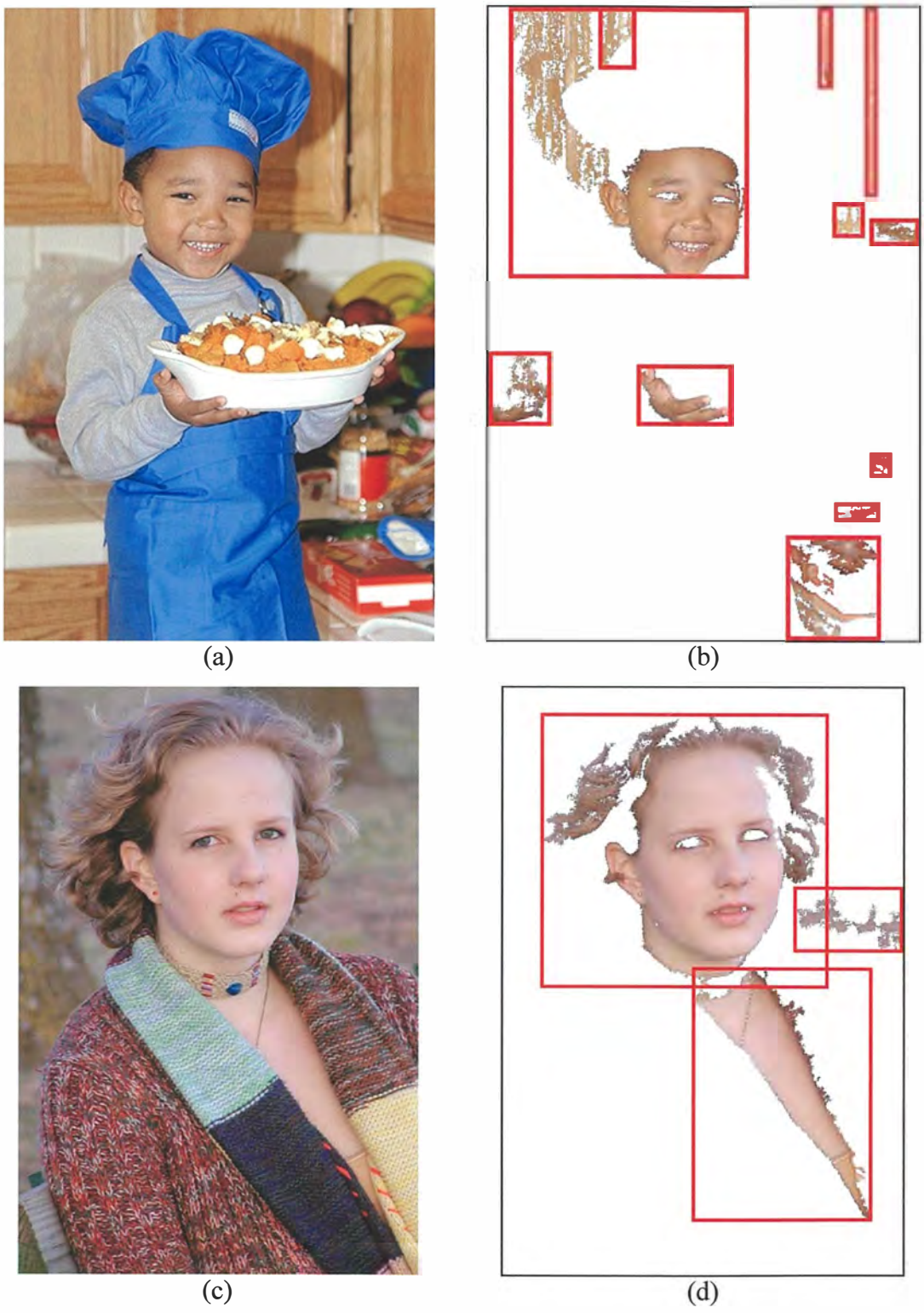


Figure 4.7: Skin region segmentation examples – Part II.

4.5 Chapter Summary

In this chapter, an algorithm has been proposed for segmenting human skin regions in color images. The proposed algorithm, which combines region-based and boundary-based segmentation techniques, is novel compared to existing skin detection approaches in that it takes into account both color and texture properties of the human skin. The proposed algorithm consists of three main stages: color-based skin detection, skin region verification, and skin region refinement. Color-based skin detection is done using the Bayesian classifier described in the previous chapter. In the skin region verification stage, a statistical homogeneity measure is used to select only skin-colored regions that have smooth texture. In addition, the Canny edge detector is applied to all color channels of the input image in order to enforce the outline of skin regions. In the skin refinement stage, a number of post-processing techniques are employed to remove noise and to cope better with the problem of over-segmentation.

Chapter 5

Feature-based Face Candidate Selection

5.1 Introduction

The last two chapters address the problem of segmenting regions in unconstrained color images. The segmented skin regions provide an initial estimation of face locations in an input image, and through skin segmentation as much as 90% of the face search space can be eliminated. In this chapter, we focus on the task of finding face candidates in the detected skin regions. In addition, several heuristic techniques are proposed for a preliminary verification of the face candidates. More sophisticated techniques for final verification of the remaining candidates will be discussed in the next chapter.

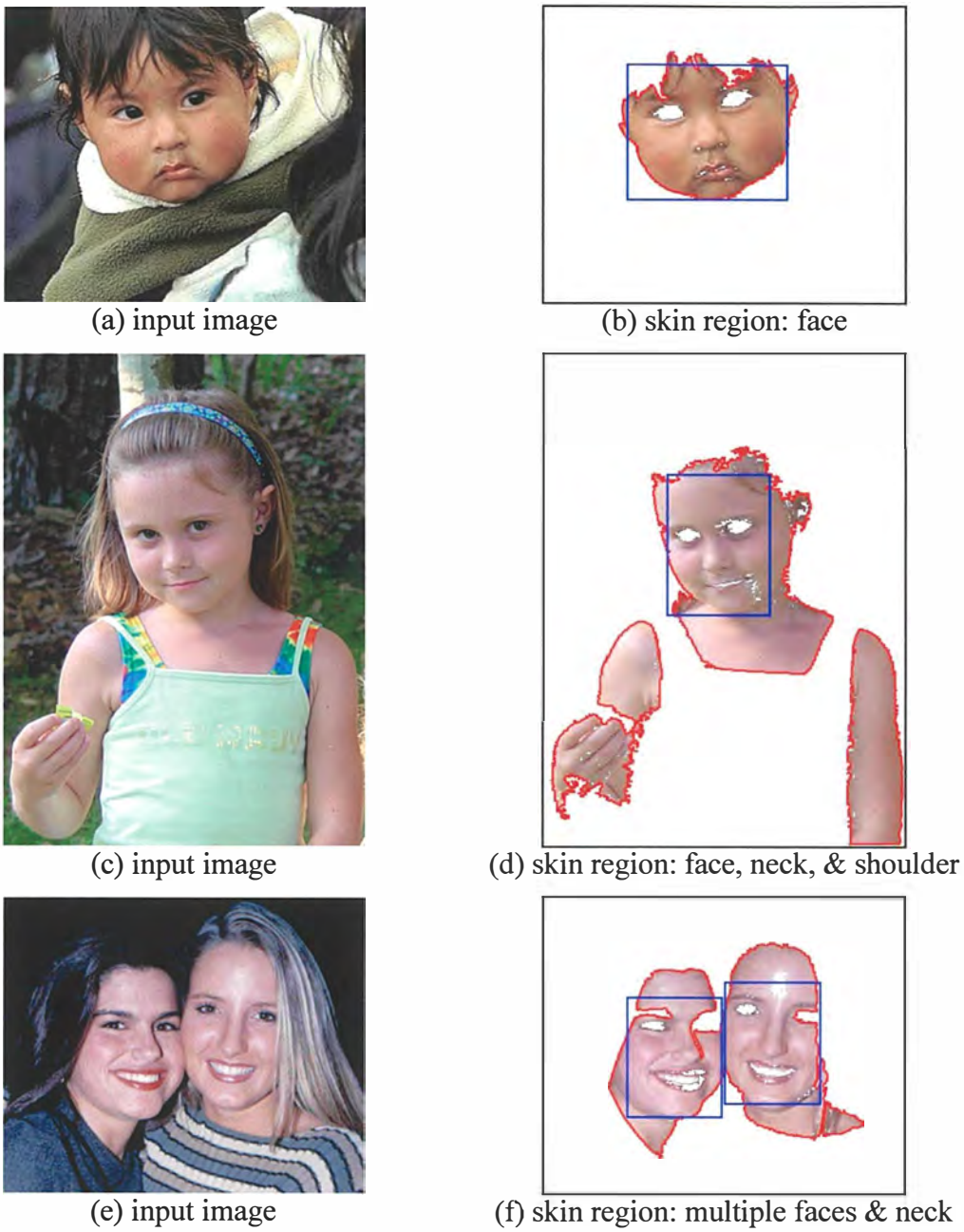
The majority of skin color-based face detection techniques consider each detected skin region as one face candidate [15, 23, 105, 127, 135]. The face candidate is checked against heuristic rules about the face shape [23, 105], compared with face templates [127, 135], or processed by face/nonface pattern classifiers [72]. This approach works based on the assumption that the facial skin region is separated neatly from other skin regions because of the presence of nonskin-colored image background (Fig. 5.1a). This assumption is reasonable in applications such as face segmentation for videophone sequence coding [15] and face tracking with a desktop camera, in which the input image

typically consists of a head-and-shoulder view of the person. In more general imagery, this assumption is no longer valid because the facial skin region may be merged with other exposed skin regions, especially the neck, shoulders and arms (Fig. 5.1c), or one skin region may consist of multiple faces (Fig. 5.1e). Clearly, in these cases, verification of face candidates will fail if each detected skin region is treated in its entirety as one face candidate. Therefore, there is a clear need for identifying face candidates in each segmented skin region.

An obvious approach to finding face candidates in a segmented skin region is to use a window-scanning technique [25]. In this approach, rectangular windows of fixed size in the skin region are scanned for faces. For each window centered at a given pixel location, a pattern classification algorithm is employed to determine if the window contains a face. In fact, the window-scanning approach is commonly taken in the holistic approach to face detection that we described in Chapter 2. However, there remain a number of issues to be resolved with the window-scanning approach. First, because the face can appear at any location in the skin region, an exhaustive scan of a large number of windows in the skin region is required. Second, because a face can have any size, whereas the scanning window has a fixed size, several scales of the skin region must be examined. Third, there is a problem of multiple detections of the same face across different scales and at nearby pixel locations.

In this chapter, we propose a novel feature-based approach to identifying face candidates in a skin region. In our approach, potential eye regions in the skin region are first located. For each pair of eye regions, two face candidates are constructed based on a geometric model of the human face. These candidates are then subjected to a series of preliminary tests in order to remove obvious nonfaces. All remaining face candidates will be further examined using pattern classification techniques that are presented in Chapter 6. The current chapter is organized as follows. A novel technique for detecting potential eye regions in a skin region is presented in Section 5.2. The formation of face candidates from a pair of eye regions is described in Section 5.3. In that section, we also discuss the validity of our face-from-eyes approach. Several techniques for preliminary verification of the face candidates are proposed in Section 5.4. Experimental results of

the feature-based approach to face candidate selection are discussed in Section 5.4. The chapter summary is presented in Section 5.5.



(a) input image (b) skin region: face

(c) input image (d) skin region: face, neck, & shoulder

(e) input image (f) skin region: multiple faces & neck

Figure 5.1: The need for finding face candidates in each segmented skin region: (b), (d), and (f) are segmented skin regions produced by the algorithm in Chapter 4; skin region perimeters are shown in red; the ground-truth faces are shown as blue rectangles.

5.2 Eye Region Detection

5.2.1 Existing Eye Detection/Extraction Techniques

The eye is a prominent feature of the human face, and it has been used as an important visual cue in face recognition. Compared to other facial features such as mouth and chin, the eye changes very little even in different facial expression. For example, the distance between the corners of the two eyes remains almost fixed for a person, and this fact has been used to normalize two face images before comparison. Many techniques for extracting eye and other facial features have been studied in the context of face recognition. These techniques are mostly limited to processing frontal facial images that contain typically a mug-shot of a person on a relatively simple and uniform background. Under this assumption, the problem of extracting the eyes is simplified. For example, De Silva and Win [103] used horizontal and vertical projection of the edge map to locate eyes; Lin *et al.* [64] proposed an algorithm for detecting eyes in facial images using fractal dimension; Lin and Wu [63] used genetic algorithms to locate eyes as well as other facial feature points (eyebrows, nose, and mouth).

A number of eye detection techniques have been proposed in the context of face detection. These techniques are designed to work on unconstrained images. Yow [134] proposed that facial features such as eyebrows, eyes, nose and mouth can be modeled as dark bars on light background, which can be located using bar detectors. In Yow's approach, the following elongated 2-D Gaussian derivative filters are used as bar detectors:

$$f(x,y) = g(x)\frac{d^2}{dy^2}g(y), \tag{5.1}$$

where $g(x)$ is the Gaussian function with mean μ_x and variance σ_x^2:

$$g(x) = \frac{1}{\sqrt{2\pi\sigma_x^2}}\exp\left(-\frac{(x-\mu_x)^2}{2\sigma_x^2}\right). \tag{5.2}$$

The elongation of the filter (σ_x/σ_y = 3:1) is chosen to correspond to the aspect ratios of the eye and nose. This approach to facial feature extraction has a number of practical limitations. First, to detect facial features in an image containing in-plane rotated faces,

rotated versions of the Gaussian derivative filters are needed. Second, different scales, σ, are required to detect facial features of different sizes. Third, the feature detection approach using Gaussian derivative filters has been found to have a high number of false detections, even for images of moderate background complexity [77].

Yuan *et al.* [135] modeled facial features (eye and mouth) as image valleys, which are detected using templates of fixed sizes. The feature detector is designed for frontal upright faces where the eye and mouth are almost horizontal. Wong *et al.* [125] proposed an eye detection approach, in which eye regions are emphasized by subtracting, from the original image, a valley image that is obtained through gray-scale morphological operations. Sobottka and Pitas [105] modeled the eye and mouth regions as dark image regions, which are located by analyzing horizontal and vertical projections of the intensity image. Jeon *et al.* [49] observed that (i) among facial features, the eyes and eyebrows are always darker than the rest of the facial region; (ii) the eye and eyebrow regions occupy about 20% of the circular face mask. Based on these observations, an intensity threshold is selected so that only 20% of image pixels are smaller than it. The intensity threshold is applied on the input window of size 21×21 to form a binary image for eye and eyebrow regions. Face orientation is determined by comparing, using the Hausdorff distance, the binary image with rotated versions of a binary face model, which consists of two horizontal lines representing the two eyes. Wu and Zhou [128] developed a face candidate selection technique that relies on the detection of eye regions. In their approach, eye-analogue segments are first detected as regions that are darker than their neighborhood. Detected segments are paired to form face candidates if they satisfy a number of size, orientation and distance constraints. Their experimental results showed that the technique may fail if the face rotation angle is too large.

Sung [110] proposed an example-based learning approach for object detection. The proposed approach, which is described in Section 2.3.3, is applied to the detection of human faces and eyes, and the recognition of hand-printed digits. The distribution-based model consists of 16 Gaussian "eye" clusters and 8 "noneye" clusters. The clusters are generated by performing elliptical k-means clustering on a set of 6684 human eye patterns, and 8335 noneye patterns. The noneye clusters are carefully formed so that

they are close to the eye clusters. Because the eye-detection system was trained with only frontal eye patterns, it fails at large rotation angles.

Hsu *et al.* [43] presented a color-based eye detection approach, which is based on the observation that the eye has high Cb and low Cr (Cb and Cr are chrominance channels of the YCbCr color space). In their approach, two eye maps are first computed:

$$EyeMapC = \frac{1}{3}\left\{Cb^2 + (255 - Cr)^2 + (Cb / Cr)\right\}, \tag{5.3}$$

$$EyeMapL = \frac{Y \oplus g}{Y \ominus g + 1}, \tag{5.4}$$

where $\oplus$ and $\ominus$ are, respectively, gray-scale morphological dilation and erosion operators, g is the structuring element. Each of the three terms in (5.3) is normalized to the range [0, 255]. The chrominance eye map *EyeMapC* is enhanced with histogram equalization, and then multiplied with the luminance eye map *EyeMapL*. The resulting eye map is dilated, masked and normalized to brighten the eyes and suppress other facial areas. In this color-based approach to eye detection, the eye color is modeled using scores that are highly heuristic. In contrast, in the eye detection algorithm we propose below, the eye color is modeled using a robust statistical technique.

5.2.2 Proposed Eye Region Detection Technique

We present in this subsection a novel technique for locating eye regions in a segmented skin region for the purpose of face candidate construction. Our assumptions and goals in eye region detection are stated below:

- **Lighting conditions**: the assumption regarding the lighting condition is essentially the same as with skin segmentation; that is, the focus is on typical outdoor daylight and office lighting conditions.
- **Face orientation**: the eye must be detected regardless of the face pose, provided that the eye is visible in the face pose.
- **Facial expression**: the eye must be detected regardless of the facial expression (e.g. eye open or closed). This implies that detection methods based on

geometric assumptions, such as the eye is made up of elliptic curves, are unlikely to be effective.

- **Eye outside segmented skin regions**: in some cases, because of artifacts of the skin segmentation process, for example due to the presence of hairlines, the actual eye may fall outside the corresponding segmented skin region (see, e.g., Fig. 5.1f). It is important that eye regions are detected even when they are not enclosed within a segmented skin region.
- **Eye detection accuracy**: because the centers of detected eye regions are used later steps to construct face candidates (Section 5.3), it is important that these centers are pinpointed with reasonable accuracy. In addition, while falsely detected eye regions cannot be avoided entirely, it is necessary to keep the number of false eye detections to a minimum.
- **Eye spectacles**: spectacles pose a difficult challenge to eye detection. In our work, eye region detection in the presence of spectacles should still be possible, as long as the spectacles do not alter too much the eye color.

The above goals suggest a color-based approach to detecting the eye because color, unlike shape, is a visual feature that is less sensitive to changes in pose. The proposed technique locates eye regions by first looking for pixels that have eye color. We use the term '*eye color*' to represent the color of eye-pupil, iris, eyelash, and the white of the eye. This definition of eye color allows the eye to be located even when it is closed. Our experiments have shown that the eye has very consistent but distinctive color. Eye color is generally different from skin color; this makes color-based detection of the eye in a segmented skin region possible.

A. Color-based Eye Score

In the following description, the notation follows with that of Chapters 3 and 4. Let ω_1, ω_2, and ω_3 represent the skin color, nonskin color, and eye color classes respectively. Let $p(\mathbf{x}|\omega_i)$ be the class-conditional probability density function (pdf) of class ω_i. The class-conditional pdf of skin color $p(\mathbf{x}|\omega_1)$ is estimated as in Subsection 3.4.4. The class-conditional pdf $p(\mathbf{x}|\omega_3)$ of eye color can be determined in a similar way using a training set of eye pixels. We use Set 4 of the ECU face detection database; this set consists of

manually segmented eye images. An eye score is defined as the likelihood ratio of the eye and skin color classes:

$$f_e(\mathbf{x}) = \frac{p(\mathbf{x}|\omega_3)}{p(\mathbf{x}|\omega_1)}. \tag{5.5}$$

Because our aim is to detect eye regions in segmented skin regions, a score that differentiates the *eye* and the *skin* is needed. Hence, the skin color pdf is used in (5.5) to compute the eye score. The eye score is based on the Bayesian decision rule for minimum cost (see Subsection 3.4.4), and it measures how likely a pixel can be considered as an eye pixel. Based on the analysis of color space and histogram size in Chapter 3, we decide to use the RGB color space and a histogram size of 64 bins per color channel in estimating the eye color pdf $p(\mathbf{x}|\omega_3)$. Note that any color space can also be used.

Let $\mathbf{I}_i$ be the rectangular region of the input image that corresponds to a segmented skin region, $\mathbf{I}_i(x, y)$ be the color of the pixel at image location (x, y). An eye score image $\mathbf{E}_i$ is obtained by computing the eye score for every pixel in region $\mathbf{I}_i$:

$$\mathbf{E}_i(x, y) = f_e(\mathbf{I}_i(x, y)). \tag{5.6}$$

If the eye score for a pixel is greater than a predefined threshold θ_{eye}, we can consider the pixel as a potential eye pixel. Further processing is needed for eye detection, but before discussing that, we investigate the properties of the eye score defined in (5.5).

The images in Fig. 5.1 are used as test images, and a threshold $\theta_{eye} = 1.0$ is applied on the corresponding eye score images. The detected eye pixels are shown in Fig. 5.2. We show only the detected eye pixels that are inside the bounding boxes of segmented skin regions. The following observations can be made about the color-based eye score:

- In all three cases, the eye pixels are picked up. This holds regardless of whether the eye lies completely inside (Fig. 5.2b,d) or outside a segmented skin region (Fig. 5.2f).
- The eye score is very effective within a skin region. There are very few false eye detections inside each skin region.

- The eye score is not very effective outside a skin region. There are many false eye detections outside the skin region or nearby the skin region perimeter. This problem can be addressed through post-processing.
- If the actual eye region does not lie inside the segmented skin region (Fig. 5.2f), it still remains close the segmented skin region. In fact, the actual eye region often corresponds to a concave part of the segmented skin region.
- The eye regions belong to image regions that are detected as nonskin. More precisely, the eye regions tend to have very low skin scores. Therefore a detected eye pixel that has a high skin score can be considered as a false detection.

It is fair to say that the color-based eye score produces many eye candidates. For the eye detection to be applicable to the proposed face candidate selection approach, it is necessary to limit the number of detected eye candidates. This can be done by incorporating some of the existing techniques described in Subsection 5.2.1 for eye verification. Alternatively, we can rank the detected eye candidates, and then select only a fixed number of eye candidates. The following ranking scheme is used in our work. Let μ_c be the average eye score, and μ_s be the average skin score for pixels in an eye candidate region. The rank of this eye candidate is determined by the ratio of these two scores:

$$eye\ rank = \mu_c / \mu_s . \tag{5.7}$$

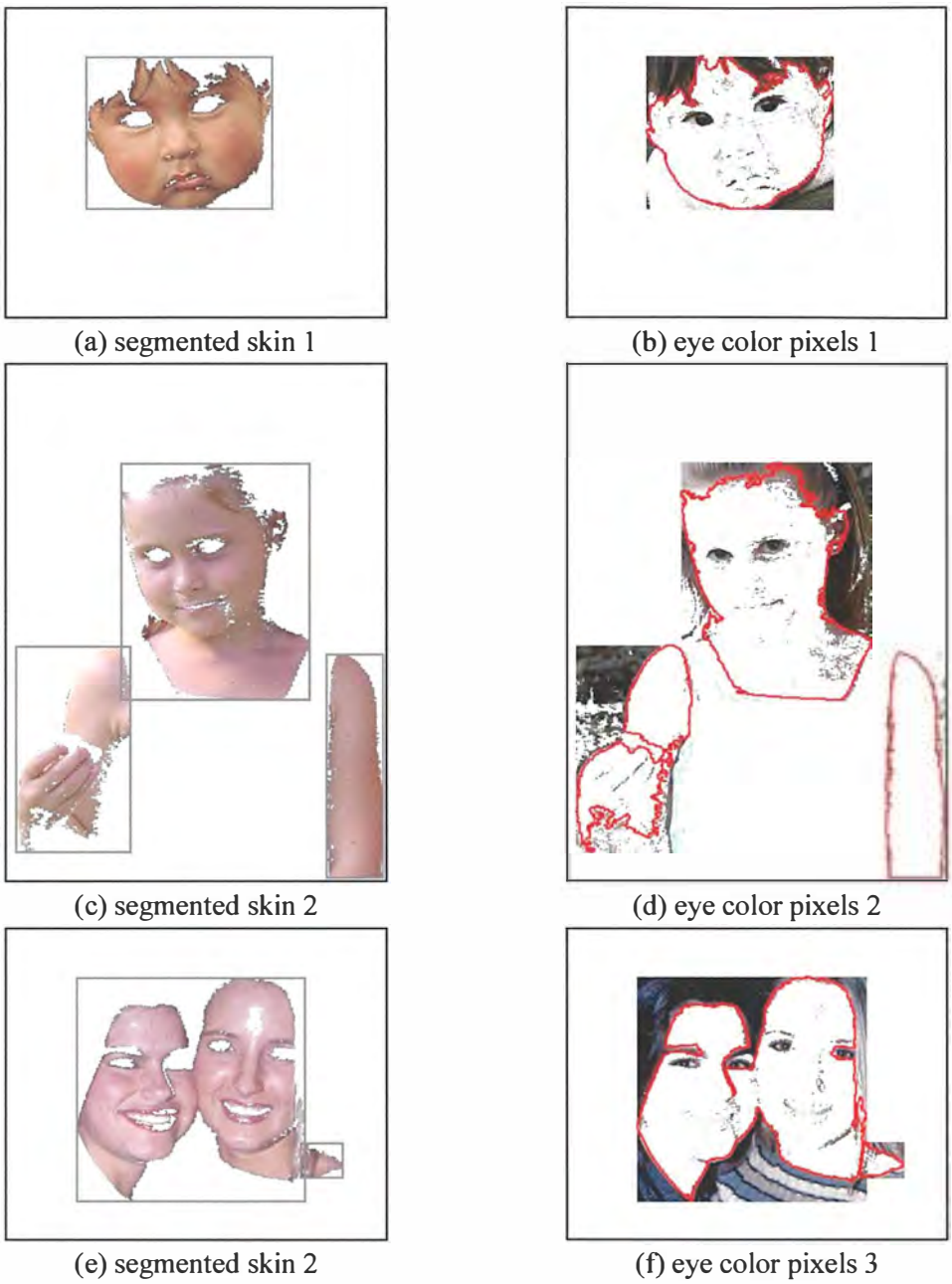


Figure 5.2: Eye detection using a color-based eye score. In (b), (d), (f), the perimeters of the segmented skin regions are shown in red.

B. Eye Region Detection Algorithm

Potential eye pixels are considered as pixels with high eye scores and low skin scores. We define an eye search region that consists of two parts (Fig. 5.3). The inner part is the skin mask eroded by a circular structuring element with a radius of 8 pixels. The outer part is obtained from the skin mask and its convex region. The convex region is eroded,

by a circular structuring element of radius 6 pixels, and combined with a strip outside the skin mask. The strip is the difference between two dilated versions of the skin mask by circular structuring elements with radii of 4 and 7 pixels. This definition of the eye search region allows the detection of eye pixels that fall outside the skin mask.

Two different sets of thresholds (for skin score and eye score) are used to find potential eye pixels in the inner part and the outer part. This strategy is necessary because the eye score is more effective inside the skin region. For the inner part, eye pixels must have eye score greater than 0.8, and skin score smaller than 2. For the outer part, eye pixels must have eye score greater than 6, and skin score smaller than 2. The detected eye pixels are grouped into eye regions through connected-component labeling. Eye regions that do not satisfy some constraints are removed. We check the following constraints for each eye region:

- region area compared to the largest eye region must be larger than 1%;
- region aspect ratio must be within the interval [0.1, 10];
- region area compared to the bounding box area must be greater than 0.3.

A step-by-step description of the proposed algorithm is given in Table 5.1. We should point out that the parameters used in our algorithm are empirically determined through experimentation with a wide range of input images.

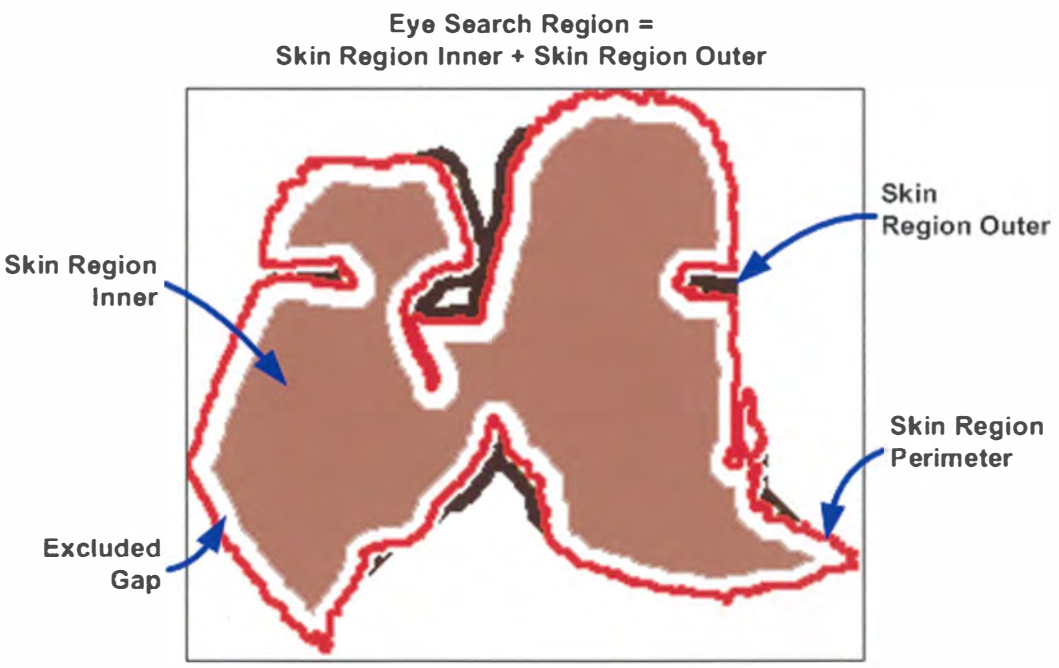


Figure 5.3: Defining eye search region based on the skin mask.

Table 5.1: Proposed eye detection algorithm.

Input: $\mathbf{B}_{s,i}$: Binary mask of a skin region.

$\mathbf{I}_i$: Rectangular part of the input image for the skin mask $\mathbf{B}_{s,i}$.

$\mathbf{S}_i$: Rectangular part of the skin score image for the skin mask $\mathbf{B}_{s,i}$.

Output: A list of eye regions.

Algorithm:

Step 1: Compute the eye score image $\mathbf{E}_i$ for the region as in (5.5) and (5.6).

Step 2: Obtain binary masks for potential eye pixels

for inner part: $\mathbf{B}_{\text{color outer},i} = (\mathbf{E}_i \geq 0.8) \cap (\mathbf{S}_i \leq 2)$

for outer part: $\mathbf{B}_{\text{color inner},i} = (\mathbf{E}_i \geq 6) \cap (\mathbf{S}_i \leq 2)$

Step 3: Define the eye search region as described above.

$\mathbf{B}_{\text{eye search},i} = \{\mathbf{B}_{\text{eye search inner},i}, \mathbf{B}_{\text{eye search outer},i}\}$

Step 4: Obtain the final eye mask:

$\mathbf{B}_{\text{eye},i} = (\mathbf{B}_{\text{eye search outer},i} \cap \mathbf{B}_{\text{color outer},i}) \cup (\mathbf{B}_{\text{eye search inner},i} \cap \mathbf{B}_{\text{color inner},i})$

Step 5: Perform connected-component labeling of $\mathbf{B}_{\text{eye},i}$. Remove noise regions using constraints about eye region area and aspect ratio, as described above.

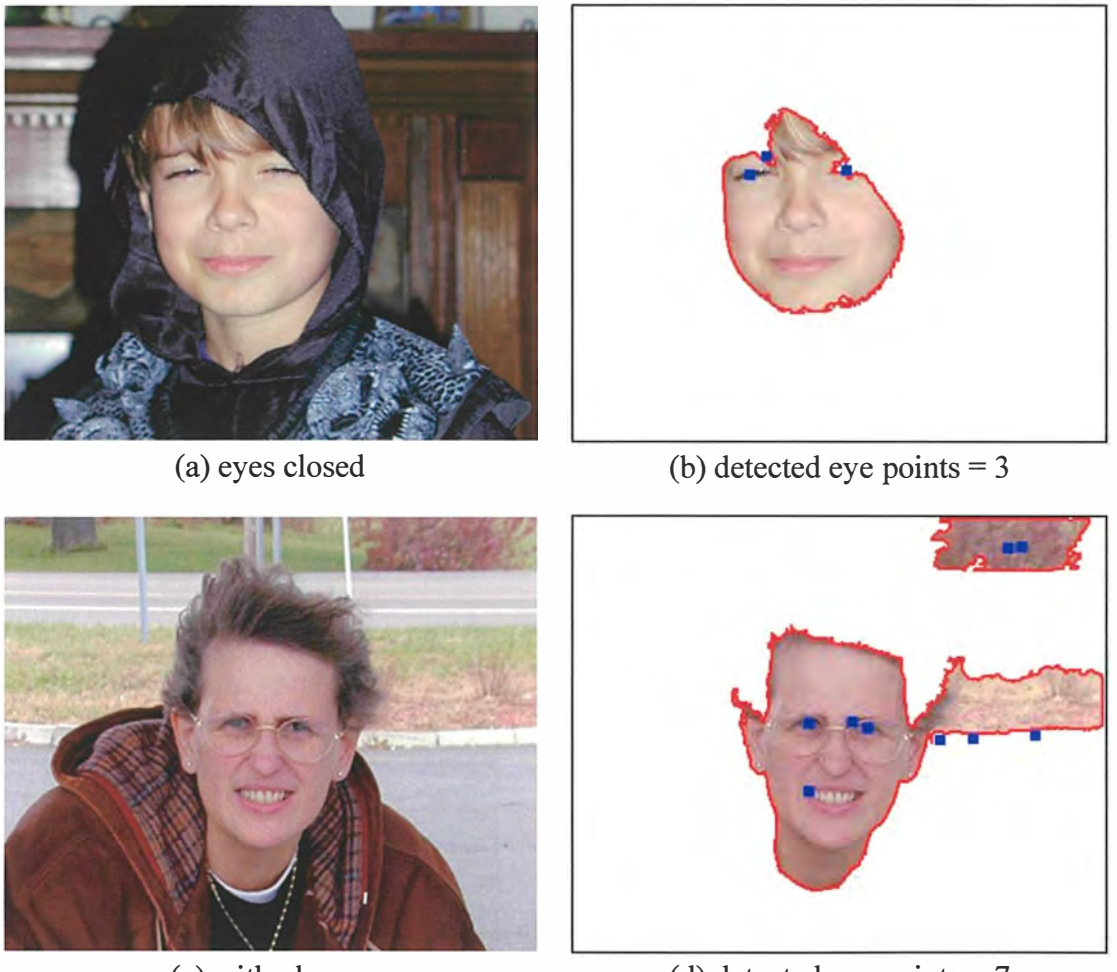
Step 6: Select a fixed number of eye regions according to the eye rank defined in (5.7). Each selected eye region is represented by its centroid.

C. Results of Eye Detection

The results of eye detection on the test images of Fig. 5.1 are shown in Fig. 5.4. The perimeters of skin regions are shown in red, and the detected eyes are shown as blue dots. The proposed algorithm can detect all eyes, even when the eyes fall outside the segmented skin regions (Fig. 5.4c). The number of false eye detections is typically between 4 and 18 in a skin region. Because for each pair of eyes, two face candidates are formed (see Section 5.3), we can expect fewer than 400 face candidates per skin region. Results of eye detection for different input images are shown in Fig. 5.5 and Fig. 5.6.

(a) for input of Fig. 5.1a (b) for input of Fig. 5.1c (c) for input of Fig. 5.1e

Figure 5.4: Results of eye detection – Part I. The perimeters of segmented skin regions are shown in red, and the detected eye points are marked as blue dots.

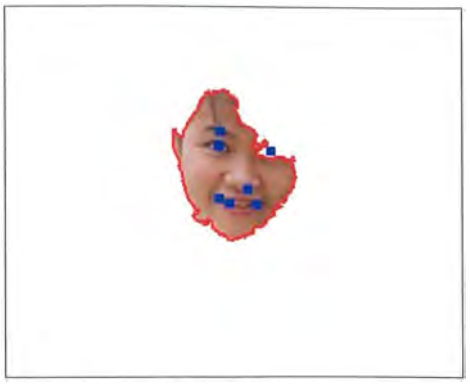


(a) eyes closed (b) detected eye points = 3

(c) with glasses (d) detected eye points = 7

Figure 5.5: Results of eye detection – Part II. The perimeters of segmented skin regions are shown in red, and the detected eye points are marked as blue dots.

(a) eye point outside segmented skin

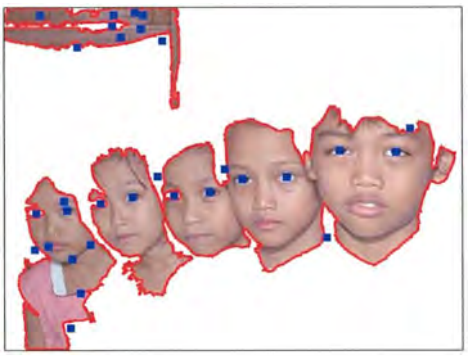


(b) detected eye points = 7

(c) rotated face

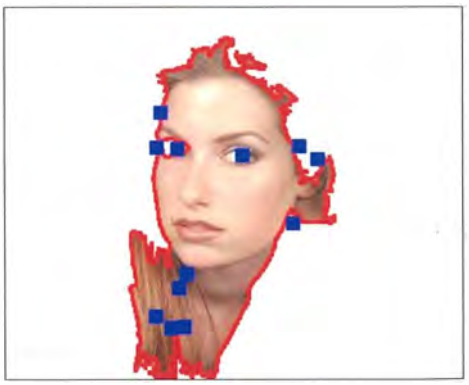
(d) detected eye points = 14

(e) multiple faces



(f) detected eye points = 30

(g) different eye colors



(h) detected eye points = 16

Figure 5.6: Results of eye detection – Part III. The perimeters of segmented skin regions are shown in red, and the detected eye points are marked as blue dots.

5.3 Face Candidate Selection

In this section, we present a new approach to face candidate selection that is based on the results of eye detection described in the previous section. The geometric face model is addressed in Subsection 5.3.1. The steps of constructing face candidates from a pair of eye points are described in Subsection 5.3.2.

5.3.1 Geometric Face Model

We propose a geometric face model, shown in Fig. 5.7, to determine the spatial extent of the face once the eyes have been detected. The geometric face model is an extension of a model we described in [86]. The model reflects roughly the relative anthropometric distances between the facial landmarks. The key property of this geometric face model can be stated as follows. Let D be half of the distance between the two eye points, where each eye point is taken as the center of an eye region. The central part of the face can be considered as the rectangular region of size $4D \times 4D$ highlighted in Fig. 5.7. The four corners of the region are completely determined from the two eye points. Hereafter, this approach is termed as the *face-from-eyes* approach. Since in later face/nonface verification stages, we are interested only on the central portion of the face, the aspect ratio of 1:1 is used in the geometric face model. This aspect ratio of the face is also used by many other authors in face/nonface classification [91, 111].

A similar face model is also used by Samad *et al.* [96] in a hybrid rule-based approach to eye detection. However, Samad *et al.* used the face model to locate eyes in a face image so that the face image can be normalized for face recognition purpose. In contrast, we used the face model to determine the spatial extent of the face in an unconstrained image where the eye positions are known. Furthermore, Samad *et al.* assumed the face is in a frontal upright position whereas in our approach, provision is given for face rotations about different axes.

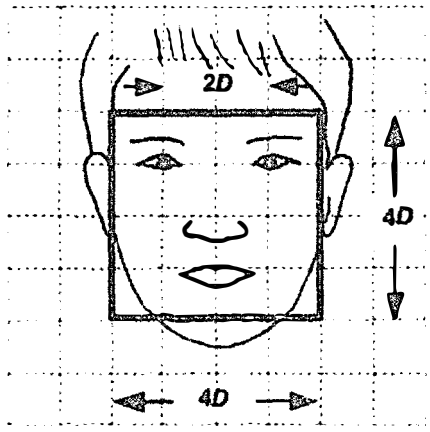


Figure 5.7: Geometric face model.

The geometric face model becomes less accurate when the face is rotated. We need to determine the extent of rotation, under which the face-from-eyes approach is still reasonably accurate. Because the face is a 3D object, three types of rotation need to be considered:

(1) *roll* is the rotation about the axis which is perpendicular to the image plane (also known as the *in-plane* rotation);

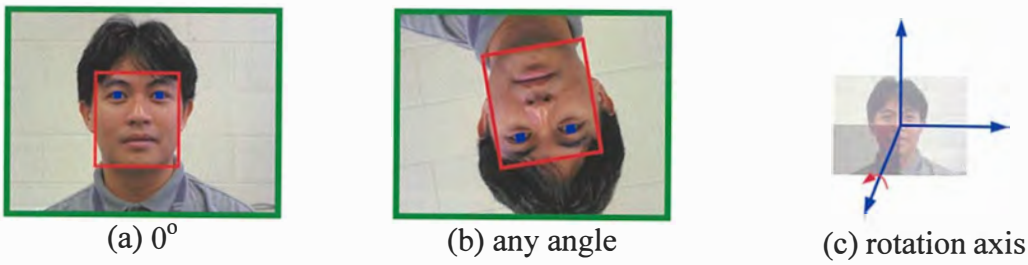
(2) *yaw* is the left-and-right rotation;

(3) *pitch* is the up-and-down rotation.

We conducted an experiment to investigate the effects of the above types of rotation on the face-from-eyes approach. The experiment involved five human subjects. During the experiment, the subjects' faces were captured at various rotation angles. The rotation axes are shown in Fig. 5.8(c), Fig. 5.9(l), and Fig. 5.10(h). The orientations of these rotation axes are chosen according to the *right-hand-rule*: if we curve our right palm following the rotation arrow, then the thumb will point along the direction of the rotation axis. In reporting the rotation angles, we follow the following convention: the rotation angle is the angle that a face image must be rotated in the positive direction, as indicated by the rotation arrow, so that it becomes a frontal upright face.

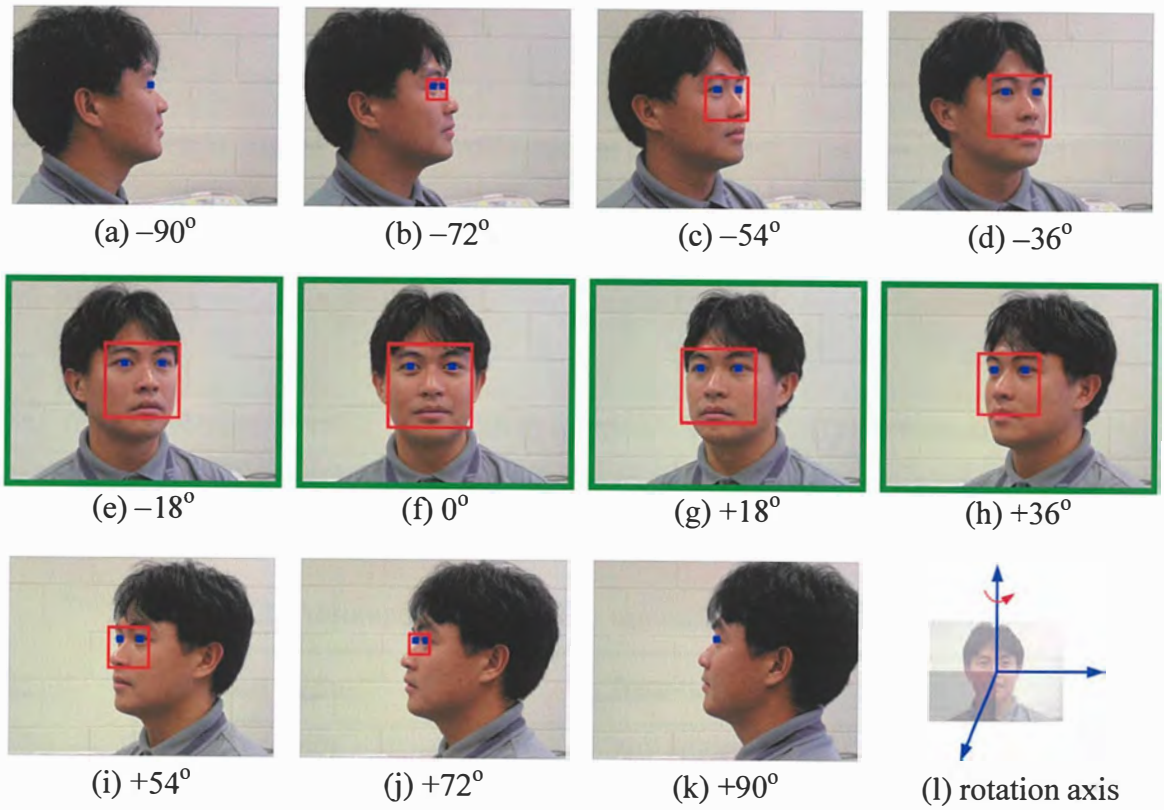
Clearly, the face-from-eyes approach is valid for *all* in-plane rotations (Fig. 5.8). Results of the experiment indicated that the face-from-eyes approach is also applicable for yaw rotations where the rotation angle is between -18° and 18° (Fig. 5.9), and for pitch rotations where the rotation angle is between -36° and 36° (Fig. 5.10). In

Figs. 5.8-10, the rotation angles, for which the face-from-eyes approach is valid, are indicated by images with green bounding rectangles. For illustration purpose, the face bounding rectangles (in red), which are generated automatically according to the geometric face model in Fig. 5.5, are drawn on top of the original images.



(a) 0° (b) any angle (c) rotation axis

Figure 5.8: Effects of in-plane rotation on the face-from-eyes approach.



(a) –90° (b) –72° (c) –54° (d) –36°

(e) –18° (f) 0° (g) +18° (h) +36°

(i) +54° (j) +72° (k) +90° (l) rotation axis

Figure 5.9: Effects of yaw rotation on the face-from-eyes approach.

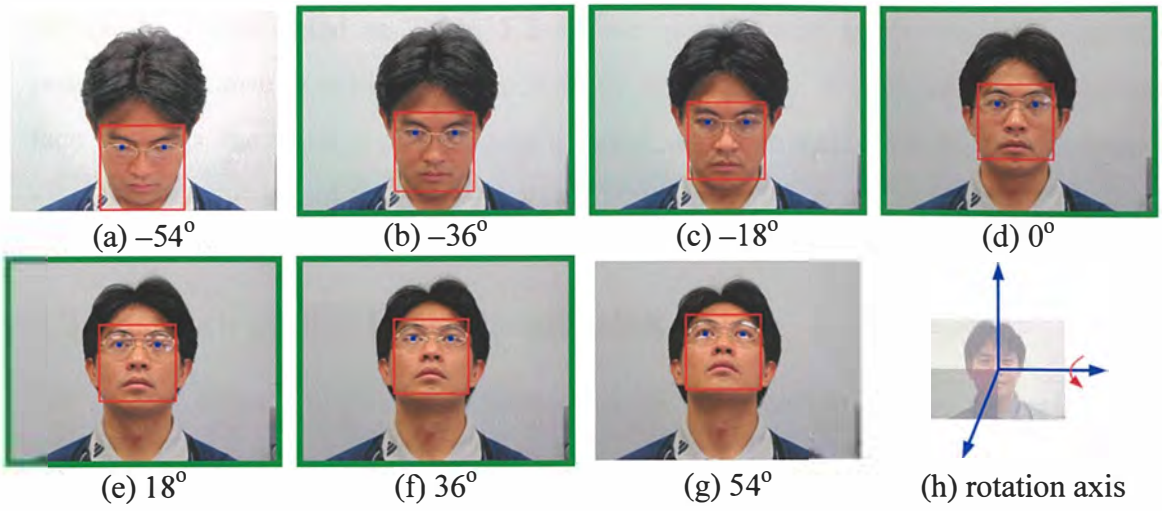


Figure 5.10: Effects of pitch rotation on the face-from-eyes approach.

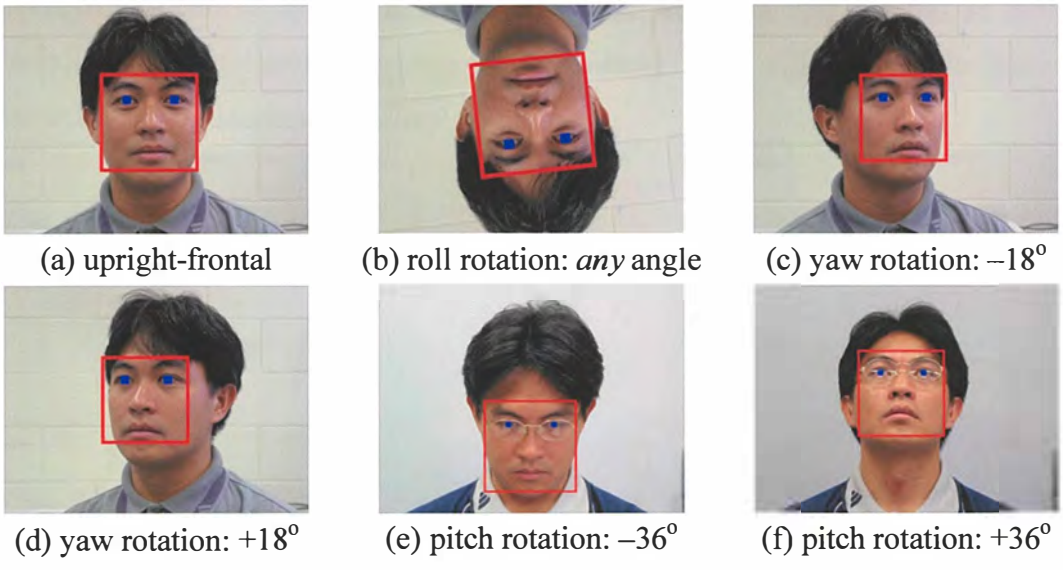


Figure 5.11: Rotation angle limits of quasi-frontal faces.

Table 5.2: Assumptions about face orientation in quasi-frontal views.

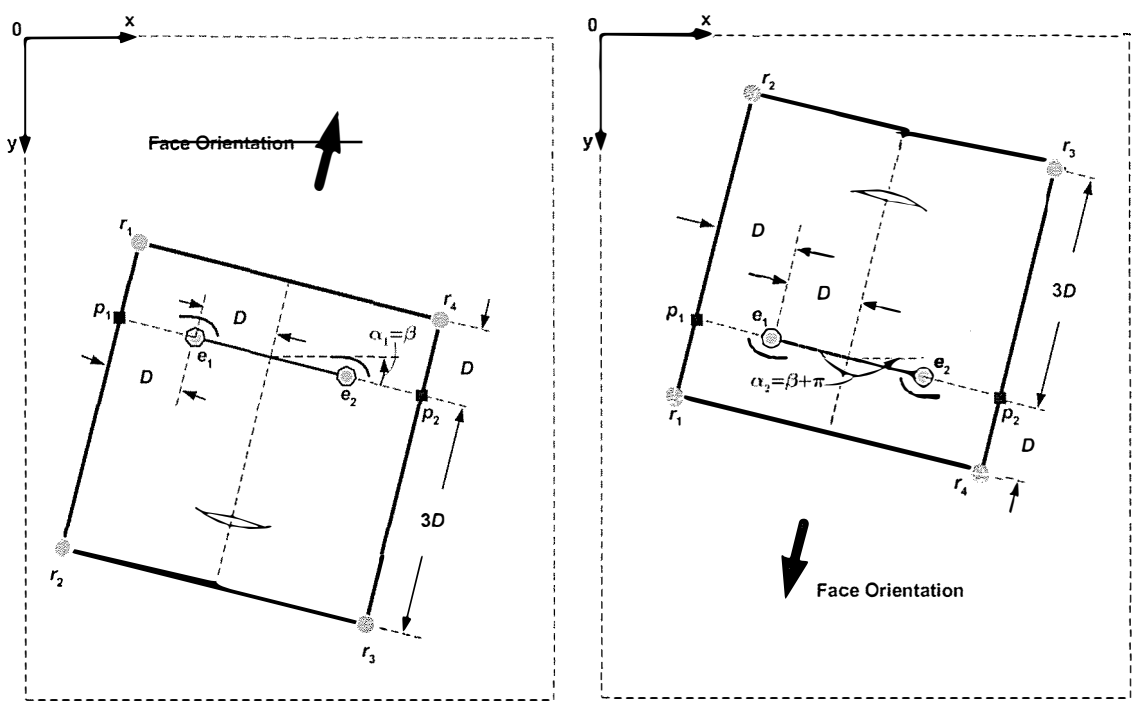
Rotation Type	Rotation Angle
Roll (in-plane)	any angle
Yaw	–18° to +18°
Pitch (tilt)	–36° to +36°

Based on the above analysis, we summarize in Table 5.2 the assumptions about the quasi-frontal views. We can expect that the face-from-eyes approach to work also for face orientations that are formed by a combination of roll, yaw, and pitch rotations

within the limits listed in Table 5.2. These quasi-frontal assumptions are not too restrictive because in applications such as face recognition or surveillance, the human face contains the most discriminative information when it is seen in a quasi-frontal view. The constraints of the quasi-frontal face views are illustrated in Fig. 5.11.

5.3.2 Constructing Face Candidates

The task of constructing face candidates can be described as follows: given two eye points, find the spatial boundary of face candidates that satisfy the geometric face model described in the previous subsection. In Fig. 5.7, the face is in the frontal upright position, and the eye line is horizontal. The method of finding the face spatial boundary proposed here can be applied to the general case when the eye line has an arbitrary orientation. In the following description, for an image point p, p_x denotes its horizontal coordinate and p_y denotes its vertical coordinate.



(a) Candidate 1: face angle $\alpha_1 = \beta$ (b) Candidate 2: face angle $\alpha_2 = \beta \pm \pi$

Figure 5.12: Determining the face boundary for a given eye pair.

For a given pair of eye points, there are two possible face candidates, as shown in Fig. 5.12. The two candidates have opposite orientations. The face orientation can be

characterized by *face angle*, which is defined as angle in the range [0, 2π) that a face must rotate in the anti-clockwise direction to become an upright frontal face. The face angle is actually the (directed) angle between the eye line and the horizontal axis. Clearly, if one candidate has a face angle of β, then the other candidate has a face angle of either $\beta + \pi$ or $\beta - \pi$.

Let e_1 and e_2 be the two eye points. The coordinates of the four corners r_1, r_2, r_3, and r_4 of the face bounding box can be computed as follows:

Step 1: *Compute half the distance between the two eyes.*

$$D = \frac{1}{2}\sqrt{(e_{2x} - e_{1x})^2 + (e_{2y} - e_{1y})^2} \tag{5.8}$$

Step 3: *Compute the two points p_1 and p_2 along the eye line.*

The positions of the two points p_1 and p_2 with respect to the two eye points are shown in Fig. 5.12. The points p_1 and p_2 can be located as follows:

$$p_{1x} = (3e_{1x} - e_{2x})/2, \quad p_{1y} = (3e_{1y} - e_{2y})/2$$

$$p_{2x} = (3e_{2x} - e_{1x})/2, \quad p_{2y} = (3e_{2y} - e_{1y})/2. \tag{5.9}$$

Step 4: *Compute the four face corners r_1, r_2, r_3, and r_4.*

The following formulae are applicable to both cases shown in Fig. 5.12a and Fig. 5.12b, assuming that the face angle is α:

$$r_{1x} = p_{1x} + D\sin\alpha, \qquad r_{1y} = p_{1y} - D\cos\alpha$$

$$r_{2x} = p_{1x} - 3D\sin\alpha, \quad r_{2y} = p_{1y} + 3D\cos\alpha$$

$$r_{3x} = p_{2x} - 3D\sin\alpha, \quad r_{3y} = p_{2y} + 3D\cos\alpha \tag{5.10}$$

$$r_{4x} = p_{2x} + D\sin\alpha, \qquad r_{4y} = p_{2y} - D\cos\alpha.$$

Step 5: *Form face candidates.*

For a given face bounding rectangle, a *face mask* is formed by selecting only the part of the rectangle that lies inside the bounding box of the skin region. A face candidate is created by extracting the original image region that corresponds to the face mask. This face candidate is rotated by an angle α (α is the face angle) so that the eye line is horizontal. The normalized (i.e. rotated) face candidate is then said to be in an upright position. The process of constructing face candidates is illustrated in Fig. 5.13.

(a) an image region

(b) segmented skin region and a pair of detected eye points

(c) face candidate 1 from the eye pair

(d) face candidate 2 from the eye pair

(e) normalized face candidate 1 (rotated to the upright position)

(f) normalized face candidate 2 (rotated to the upright position)

Figure 5.13: Steps of constructing face candidates from eyes.

5.4 Preliminary Face Candidate Verification

In this section, we present a number of simpie verification techniques designed to remove obvious nonfaces from the face candidate list. More sophisticated verification techniques will be presented in the next chapter.

5.4.1 Eye Distance

The distance between two eye points cannot be too short. Let d_{emax} be the largest distance between two eye points in the eye list. Face candidates are constructed for the pair of eye points only if their distance d_e satisfies the following conditions:

$$\frac{d_e}{d_{emax}} > \theta_{ed1} \quad \text{and} \quad d_e > \theta_{ed2}, \tag{5.11}$$

where $\theta_{ed1} = 0.02$ and $\theta_{ed2} = 10$.

5.4.2 Face Bounding Rectangle

The face bounding rectangle is constructed based entirely on the two eye points, and it may extend beyond the bounding box of the corresponding skin region. Let f_{fr} be the proportion of the face bounding rectangle that lies inside the skin region bounding box. The face candidate is removed if:

$$f_{fr} < \theta_{fr}, \tag{5.12}$$

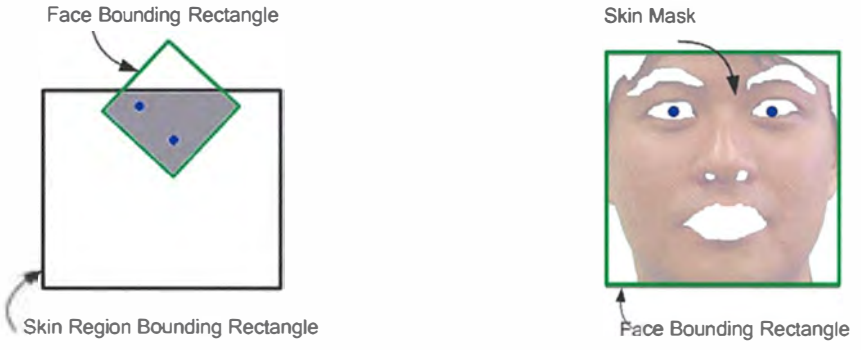
where θ_{fr} is a threshold, $\theta_{fr} = 0.8$. This verification step eliminates many face candidates having eye points that either are far apart or lie near the skin region boundary (Fig. 5.14a).

5.4.3 Skin Proportion

The skin region must form a significant part of the face region. Let f_{sp} be the ratio of the number of skin pixels inside the face mask to the number of image pixels inside the face bounding rectangle (Fig. 5.14b). The candidate is removed if

$$f_{sp} < \theta_{sp}, \tag{5.13}$$

where $\theta_{sp} = 0.3$. This threshold is set lower than an anthropologically reasonable value to account for errors in skin detection.



(a) face bounding rectangle test (b) skin proportion test

Figure 5.14: Face mask verification

5.4.4 Face Inhomogeneity

The human face consists of regions (the eye, mouth, and chin) that differ significantly in intensity, and consequently has a high intensity variance. Let f_{fi} be the standard deviation of the pixel intensities in the face mask. The face candidate is removed if:

$$f_{fi} < \theta_{fi}, \tag{5.14}$$

where $\theta_{fi} = 40$. This verification technique can remove false face candidates that are formed by mostly uniform skin pixels.

5.4.5 Face Rotation Angle

In this subsection, we present a technique to estimate the in-plane rotation angle of the face. This technique has application not only in our preliminary face candidate verification but also in the general setting of face pose estimation. The problem of estimating in-plane face rotation angles can be described as follows. Given an arbitrary quasi-frontal face image, determine the angle β_{ur} that the image has to be rotated in the anti-clockwise direction so that it becomes most similar to a quasi-frontal *upright* face. It must be stressed that in this problem formulation, no other information about the face is known *a priori*.

The problem of estimating the in-plane face rotation angle has been studied by a number of authors in the context of face detection. Most notably, Rowley [90] trained neural networks to estimate face rotation angle for a given input window of size 20×20. Once the face angle is estimated, the input window is then rotated back to the upright position. The corrected input image is then processed by another neural network face detector to determine if it is indeed a near frontal face pattern. In this subsection, we present a novel technique for estimating the in-plane face rotation angle, and discuss how this technique can be applied for face candidate verification.

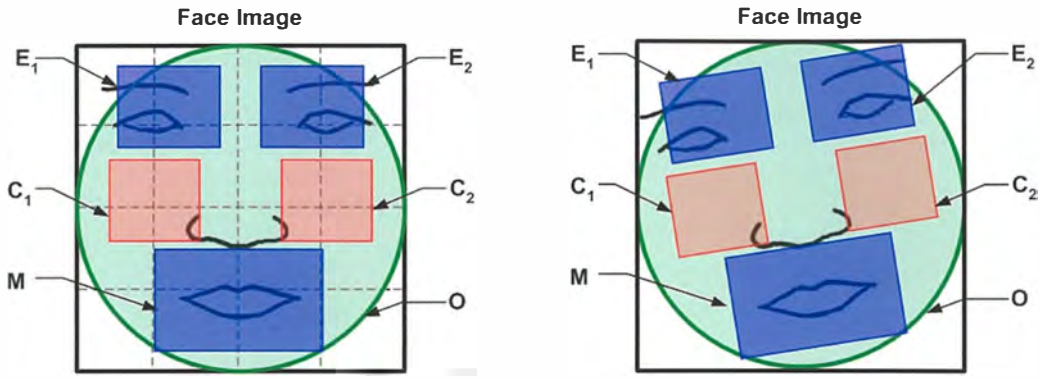
A. In-plane Face Rotation Angle Estimation

The proposed technique relies on the fact that for a human face, the eye and mouth are relatively darker compared to the cheeks. This fact is used to find the angle to rotate the input face so that it becomes closest to a quasi-frontal upright face. For a quasi-frontal upright face, the eye, mouth and cheek regions can be approximately located as shown in Fig. 5.15a. Let E_1, E_2, M, C_1, and C_2 denote the two eyes, the mouth and the two cheeks, respectively. For a region X, let μ_X be the average intensity in the *intersection* of region X and the green circle O. The circle O is centered on the face image rectangle, and it is used to exclude irrelevant pixels from face angle estimation. For each angle β, a rotation score is defined to measure the degree of match between the input face image and the image obtained by rotating the upright frontal face by an angle $-\beta$.

The rotation score for angle $\beta = 0$ is computed as follows:

$$R(\beta = 0) = \alpha_E \mu_{E_1} + \alpha_E \mu_{E_2} + \alpha_C \mu_{C_1} + \alpha_C \mu_{C_2} + \alpha_M \mu_M, \tag{5.15}$$

where $\alpha_E < 0$, $\alpha_C > 0$, and $\alpha_M < 0$ are scalar weighting factors for the eye, cheek and mouth regions, respectively. We find that the values $\alpha_E = -0.75$, $\alpha_C = 1$, and $\alpha_M = -0.25$ work quite satisfactorily.



(a) rotation angle $\beta = 0$ (b) arbitrary rotation angle β

Figure 5.15: Rotation score calculation.

To generalize the rotation score definition for other angles, we need to consider the rotation score in (5.15) from the template-matching perspective. We construct a face template $\mathbf{T}_0$ of size 32×32 (i.e. having the same aspect ratio as the face). Except for the eye, mouth and cheek regions that fall inside the circle O, all coefficients of the template are set to zero. For the intersection region $E_1 \cap O$, the template coefficients are

$$\mathbf{T}_{0,E_1 \cap O} = \frac{\alpha_e}{|E_1 \cap O|}, \tag{5.16}$$

where |X| denotes the total number of pixels in region X. The template coefficients for other regions are defined in a similar fashion. Given an input face image, it is first resized to form an image **F** that has same size as the face template $\mathbf{T}_0$. The rotation score for angle $\beta = 0$ for the face image **F** can be expressed as:

$$R(\beta = 0) = \sum_{(i,j)} \mathbf{F}(i,j)\mathbf{T}_0(i,j)\,. \tag{5.17}$$

The rotation score for an arbitrary angle β can be computed as follows (see Fig. 5.15b). The $\mathbf{T}_0$ template is rotated by an angle $-\beta$ to obtain a rotated template $\mathbf{T}_\beta$ of the same size (by keeping the part inside of the green circle). The rotation score for angle β is computed similarly to (5.17):

$$R(\beta) = \sum_i \mathbf{F}_i \mathbf{T}_{\beta,i}\,. \tag{5.18}$$

For a given face image **F**, the rotation scores are computed for $2K + 1$ angles β that are equally spaced in the interval [–60°, 60°], where $K = 16$ in our work. This interval is so

chosen because our technique for face rotation angle estimation is designed to work on near-upright face candidates (similar to those in Fig. 5.13e,f). Next, the maximum rotation score R_{max} and the corresponding angle β_{max} are identified:

$$R_{max} = \max_{i=1}^{2K+1} R(\beta_i), \tag{5.19}$$

$$\beta_{max} = \beta_i \text{ such that } R(\beta_i) = R_{max}. \tag{5.20}$$

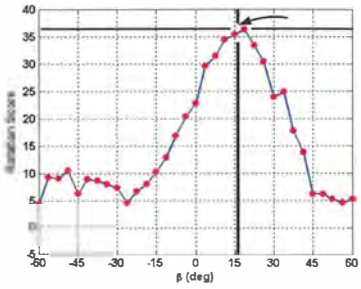
We use two input faces in Fig. 5.16a and Fig. 5.16d to illustrate the effects of the rotation score. The rotation scores at various rotation angles for the two input images are shown in Fig. 5.16b and Fig. 5.16e, respectively. The plots show that in both cases, the rotation score reaches its maximum near the actual face angle. Therefore, the angle β_{max} can be used as an estimate for the rotation angle β_{ur} of the face image F. However, we can obtain a more robust estimate of β_{ur} by using a set of angles near β_{max}. This set, denoted A, consists of angles that are close to β_{max} and have high rotation scores compared to R_{max}:

$$A = \left\{\beta_i \middle| R(\beta_i) \geq \min(0.95R_{max}, 1.05R_{max}) \text{ and } |\beta_i - \beta_{max}| < K\Delta_\beta / 8\right\}, \tag{5.21}$$

where $\Delta_\beta = |\beta_i - \beta_{i+1}|$ is the difference between two adjacent angles[14]. The estimated face angle is the mean of set A. Once this angle is found, the face can be rotated to the upright position. The normalized faces for the two input images are shown in Fig. 5.16c and Fig. 5.16f, respectively.

[14] The *min* function and the term $1.05R_{max}$ are introduced in (5.21) to handle the case when R_{max} is negative.

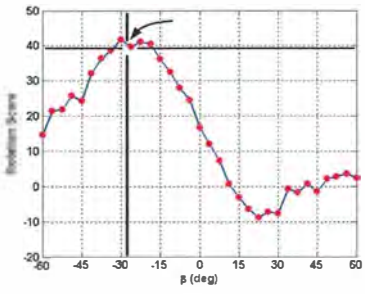
(a) input face image: actual face angle β_{ur} =16°



(b) estimated face angle β = 16.9°

(c) corrected image after rotation by the angle β

(d) input face image: actual face angle β_{ur} = –28°



(e) estimated face angle β = -28.1°

(f) corrected image after rotation by the angle β

Figure 5.16: Template-based face angle estimation.

B. Face Angle Verification

For normalized face candidates produced by the proposed face candidate selection algorithm, the eye line is horizontal and the face rotation angle should be close to zero. This suggests the following technique for face candidate verification. Let β_{ur} be the face rotation angle estimated using the template-based technique presented above. If β_{ur} is significantly different from zero, the face candidate should be rejected. In other words, a face candidate is removed if

$$|\beta_{ur}| > \theta_{ur}, \tag{5.22}$$

where θ_{ur} =10° is a threshold.

5.4.6 Face Template Matching

We use the template matching technique to compare the normalized face candidate with a prototype quasi-frontal upright face. The face template is generated by averaging aligned quasi-frontal upright face patterns. In our work, 1521 face patterns extracted from the BioID database [50] are used. This database is chosen because it provides the eye coordinates for each face, which can be used together with our geometric face model to extract well-aligned faces[15]. The face template of size 32×32 is shown in Fig. 5.17. Comparison between a normalized face candidate and the face template is based on the cross-correlation measure. Let **F** be the normalized face candidate, and **T** be the face template. The template matching face score is computed as follows:

$$f_{\text{tm}} = \frac{\sum_i (F_i - \mu_{\text{F}})(T_i - \mu_{\text{T}})}{\sigma_{\text{F}}\sigma_{\text{T}}}, \tag{5.23}$$

where μ_{F} and μ_{T} are the means, and σ_{F} and σ_{T} are the standard deviations of **F** ad **T**, respectively. The face score in (5.23) is undefined for input pattern **F** with $\sigma_{\text{F}} = 0$ (i.e. all elements in the pattern are equal). In this case, the pattern is considered as nonface, and the face score is set to 0. The higher the face score, the more likely the input pattern is a face.

Figure 5.17: Quasi-frontal upright face template.

We computed the template matching scores for a test set of 3,000 face patterns and 6,000 nonface patterns. The histograms of template matching scores for face and nonface classes are normalized and shown in Fig. 5.18. The figure shows that face patterns have high matching scores, where nonface patterns have low matching scores.

[15] faces where the eye points are at the same predefined pixel positions.

However, the figure also shows that there is a significant amount of overlap between the template matching scores for face and nonface classes. The ROC curve (Fig. 5.19a) of the template-based face/nonface classifier shows that template matching is not sufficient to separate between face and nonface patterns. Nevertheless, we use template matching as a preliminary verification of face candidates. A pattern that has a face score below τ_{tm} = 0.05 is considered as a nonface. At the chosen threshold, the classifier has a correct detection rate of 97.1%, and a correct rejection rate of 26.7%. The patterns that pass this template matching test will be further examined using the pattern classification techniques presented in the next chapter.

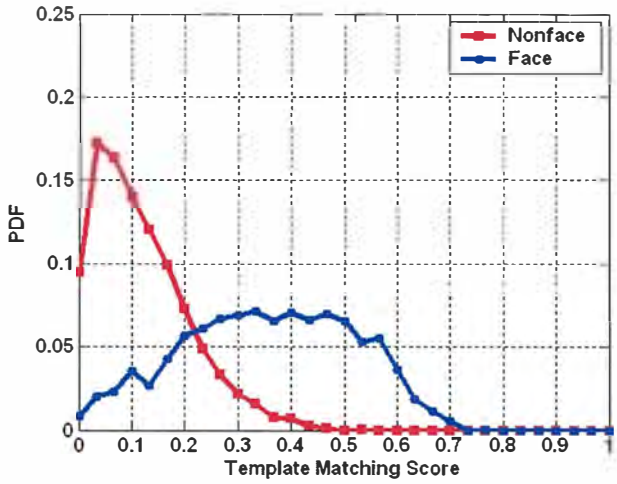
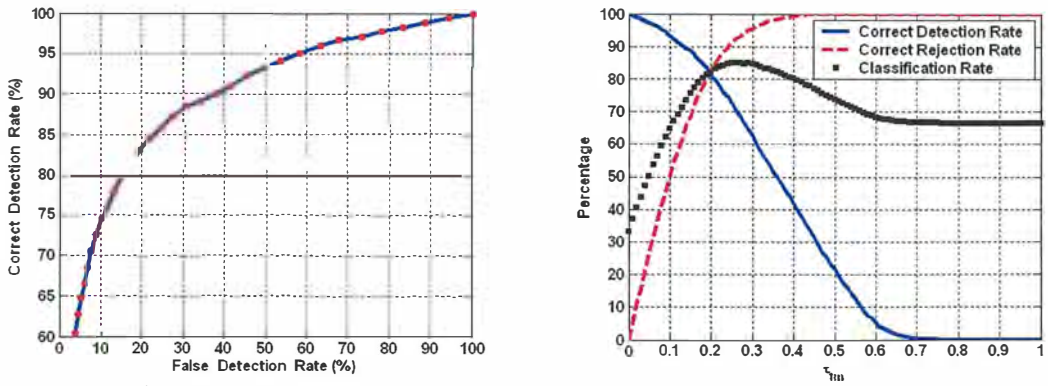


Figure 5.18: Normalized histograms of template matching score for face and nonface classes.



(a) the ROC curve (b) classification rates

Figure 5.19: Classification performance of the template matching face/nonface classifier.

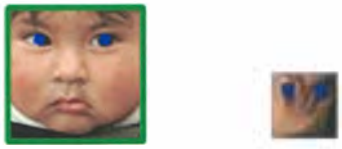
5.5 Results and Discussion

The proposed eye detection algorithm and face candidate selection algorithm were applied to process 100 images from the ECU database (images 1 to 100). The detection results are presented in Table 5.3. In this test set, there were 108 faces and 216 eye regions. The eye detection algorithm could locate 202 eye regions correctly (correct detection rate of 93.5%). On average, 12.5 candidate eye regions were generated for an actual eye region. This large number of false detections is necessary so that only few true eye regions are dismissed. From these eye points, 329.2 face candidates were generated, on average, for each actual face. After preliminary verification steps, the number of face candidates dropped to 89.8 face candidates for each actual face.

We show some visual results of running the face candidate selection algorithm. Images in Fig 5.1a, 5.1c, and 5.1e are used as inputs. The face candidates for the three input images that remain after the preliminary verification steps are shown in Fig. 5.20. Each face candidate has been rotated so that the eye line is horizontal. In the next stage of our face detection algorithm, these candidates will be verified if they are indeed quasi-frontal upright faces.

Table 5.3: Results of eye detection and face candidate selection.

Test Data	Test images	100	Images 1-100
	Number of true faces	108	
	Number of true eye points	216	
Eye Detection	Number of eye points generated	2742	12.5 eye candidates per true eye region
	Number of eye points correctly identified	202	CDR = 93.5%
Face Candidate Selection	Number of face candidates generated (without preliminary tests)	35554	329.2 face candidates per true face
	Number of face candidates generated (after preliminary verification steps)	9698	89.8 face candidates per true face



(a) face candidates for input image of Fig. 5.1a

(b) face candidates for input image of Fig. 5.1c

(c) face candidates for input image of Fig. 5.1e

Figure 5.20: Results of face candidate selection. The true face is indicated by a green box; the eye points, from which a face candidate is formed, are marked by blue dots. In (b), face candidates from the same segmented skin region are grouped inside a dashed box.

The proposed color-based eye detection technique has a number of important advantages. First, since only color is used, eye regions can be detected regardless of whether the face is in frontal upright or rotated positions. Second, the eye points, taken as the centers of eye regions, are localized with very high spatial accuracy. This enables the application of the geometric face model to define the face extent. However, the color-based eye detection technique also has a number of drawbacks:

- First, because very few geometric constraints are imposed on the eye, the proposed eye detection approach produces many false detections. It is possible to reduce the number of false detections further by incorporating some of the existing eye detection techniques described in Section 5.2.1 to verify the eye candidates. However, we find it better to maintain a high correct positive rate for eye detection at the cost of high false positive rates. Face candidates from false eye points will be further examined by a later stage of face detection.
- Second, for the proposed eye detection technique to work, it is necessary to have input images of high quality. We find that the necessary condition for the eye detection technique is that the face has at least 40 pixels in one dimension.

The face-from-eyes approach presented in this chapter is better than the window-scanning approach to face candidate selection in a number of ways:

- First, in the window-scanning approach, a face candidate is a rectangular window aligned with the image axes, whereas in the proposed approach, a face candidate is a rectangular window explicitly defined according to the eye line, which can have an arbitrary orientation. Therefore, it is reasonable to expect that faces detected with our approach align better to the true face than do faces detected with the window-scanning approach.

- Second, our approach provides an explicit face angle and accurate eye positions for each detected face, whereas in the window-scanning approach, this information needs to be estimated from the candidate window (e.g., Rowley [90] used neural networks to estimate the face rotation angle for a given window). Accurate face angle and eye positions are needed to normalize the detected faces in applications such as face recognition.

- Third, the face-from-eyes approach simplifies the task of face/nonface classification in later face verification. Each face candidate can be normalized (given the two eye points) so that the eye line is horizontal; subsequently, we only need to determine if the normalized face candidate is actually an upright face. In other words, the face/nonface classifier needs to distinguish only quasi-frontal upright faces from all other patterns.

5.6 Chapter Summary

In this chapter, a color-based technique for detecting eye regions in the segmented skin regions was proposed. The proposed eye detection technique can locate the eyes very precisely even for different face orientations. The detected eye regions are used with a geometric face model to construct face candidates in a face-from-eyes approach. We stated in the chapter the necessary conditions for the face-from-eyes approach to be applicable. This approach to face candidate selection can provide accurate information about face angle and eye positions for the detected faces. It also simplifies the task of face/nonface classification.

Chapter 6

Face/Nonface Classification

6.1 Introduction

In the last chapter, we presented an approach for identifying face candidates in segmented skin regions. Several preliminary verification techniques based on heuristics about the face pattern were proposed to remove obvious nonfaces among these candidates. However, there remain nonfaces among these candidates for which simple heuristics cannot reliably remove. In this chapter, we focus on more sophisticated classification techniques to perform final verification of the remaining face candidates. This final verification of face candidates is treated as a pattern classification problem, in which image patterns are classified into either *face* or *nonface* classes. The organization of this chapter is as follows. A concise problem statement and major assumptions are given in Section 6.2. Several preprocessing techniques for reducing image variations caused by different imaging conditions are described in Section 6.3. A new and accurate face and nonface classifier based on the naive Bayesian model is proposed in Section 6.4. Strategies for improving the performance of the Bayesian classifier are presented in Section 6.5. Experimental results and discussion are provided in Section 6.6. The chapter summary is given in Section 6.7.

6.2 Problem Statement and Assumptions

The face/nonface classification problem can be described as follows. Given a rectangular image window, determine whether or not the window *as a whole* can be considered as a face pattern. Because in the previous stages of our face detection system, color information has been used extensively in skin segmentation, face candidate selection, and preliminary face verification, we expect that color information provides little distinction between the remaining face candidates and a true face. Therefore, in this chapter we focus on the classification of face and nonface on the basis of image intensity only. Other assumptions are listed below:

- **Input size:** The size of classification window is 64×64. This window size is found to be sufficiently large to capture the detailed appearance of the human face, yet small enough to be computationally viable. The aspect ratio of the window is the same as that of the face candidates generated in the candidate selection stage. A face candidate of arbitrary size is first resized to this standard size before classification takes place. In addition, windows of smaller size (e.g. 16×16) can be constructed from this base window depending on the particular need of a classification scheme.
- **Face pose:** The face class should include all *quasi-frontal upright* faces. Examples of quasi-frontal faces are shown in Fig. 5.11. We only have to consider the upright position because all face candidates have been normalized so that the eye line is horizontal.
- **Lighting conditions:** The face/nonface classifier should be able to accept face patterns under different lighting conditions.

6.3 Image Preprocessing Techniques

The principal aim of face/nonface classification is to learn the concept of "human face", or in other words, to identify visual attributes that distinguish the human face from all other patterns. A major challenge of this task is coping with possible variations in the face pattern, both intrinsic and extrinsic. Examples of intrinsic variations are different people, different facial expressions, the presence or absence of natural facial features

such as moustaches and beards. Examples of extrinsic variations are different lighting conditions, viewing angle, and scale, and the presence of artificial accessories such as eye glasses. A simple solution is to collect a training set that includes all these variations. However, this solution is impractical because such a training set would be huge, and training a classifier to learn salient features from such a set is computationally prohibitive. Alternatively, we can use image normalization techniques to reduce the variations due to some imaging factors such as lighting intensity, and use a comprehensive data set that covers most of the intrinsic variations; this is, in practice, a more plausible strategy.

The goal of image normalization is to bring the input image to some "standard" condition, and to extract invariant features for classification. In the last chapter, we showed how the detection of two eyes is used to normalize the face candidate so that we can now focus on differentiating upright faces (where the eye line is almost horizontal) from all other 2-D patterns. In this section, we continue to discuss a number of preprocessing techniques that have been developed for coping with variations in the imaging conditions, and excluding irrelevant image portion from classification.

6.3.1 Image Normalization

A. Mean Normalization

This technique, which involves the removal of the mean pixel intensity from the input image, can correct a constant shift in the image intensity. Let $\mathbf{I}$ be the input image, and I_{mean} be the mean of all pixel intensities in the image. The corrected image $\mathbf{I}_c$ using this technique is:

$$\mathbf{I}_c = \mathbf{I} - I_{mean}. \tag{6.1}$$

B. Range Normalization

The input image is normalized (i.e. stretched) linearly so that its range is [0, 255]. Let I_{max} and I_{min} respectively be the maximum and minimum intensity level in **I**. The corrected image $\mathbf{I}_c$ is obtained as follows:

$$\mathbf{I}_c = 255 \times \frac{\mathbf{I}_c - I_{min}}{I_{max} - I_{min}}. \tag{6.2}$$

It is easy to show that this normalization technique, like the mean normalization technique, cancels out the effects of constant shifts in the image intensity.

C. Standard Deviation Normalization

The input image is scaled so that it has a standard deviation of 1. Let I_{mean} and I_σ be, respectively, the mean and the standard deviation of pixel intensities for the input image **I**. The image is normalized as follows:

$$\mathbf{I}_c = \frac{\mathbf{I} - I_{mean}}{I_\sigma}. \tag{6.3}$$

This technique is similar to the range normalization technique above. It handles both additive and multiplicative variations in pixel intensity.

D. Illumination Gradient Correction

Under a strong directional lighting, some facial regions may cast heavy shadows on other facial regions. In that case, the following technique, which was first used by Sung [110], and later on by Rowley [90], can be applied. The change in the intensity of image pixel (x, y) due to directional lighting can be modeled approximately as a linear function of the pixel coordinates:

$$\Delta_I(x, y) = ax + by + c, \tag{6.4}$$

where a, b, and c are constant coefficients. These coefficients are found as the least-square-error solution to the following over-determined system of linear equations:

$$(\mathbf{x}\ \mathbf{y}\ \mathbf{1})(a\ \ b\ \ c)^{\mathrm{T}} = \mathbf{I}, \tag{6.5}$$

where **x** is a column vector consisting of horizontal coordinates of all image pixels, **y** is a column vector consisting of vertical coordinates of all image pixels, **1** is a column vector of all 1's, **I** is a column vector consisting of intensities of all image pixels. The normalized image $\mathbf{I}_c$ is obtained as follows:

$$I_c(x, y) = I(x, y) - (ax + by + c). \tag{6.6}$$

Because the number of equations in (6.5) is much large than the number of unknown variables, the normalized image in (6.6) can have a quite large dynamic range. To correct this, we propose to apply a range normalization on $\mathbf{I}_c$ to limit its range to [0, 255].

E. Histogram Equalization and Modification

The histogram of an image consists of the counts of intensity levels in the image. In histogram equalization, an intensity mapping function is applied on image pixels so that the resultant image has approximately uniform histogram. Suppose there are $L + 1$ intensity levels: 0, 1,..., L. Let $h(i)$ be the histogram count of intensity level i in the input image **I**. The probability density function (pdf) and the cumulative density function (cdf) for intensity level are defined respectively as:

$$\text{pdf:} \quad p(i) = \frac{h(i)}{\sum_{j=0}^{L} h(j)}, \tag{6.7}$$

$$\text{cdf:} \quad P(i) = \sum_{j=0}^{i} p(j), \tag{6.8}$$

The intensity mapping function f required for histogram equalization transforms an original intensity level i to a new intensity level $f(i)$ such that:

$$f(i) = [L \times P(i)], \tag{6.9}$$

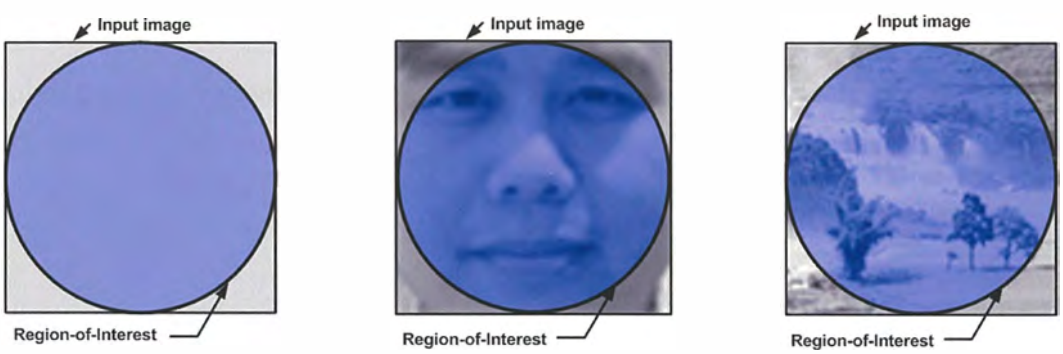
where [.] denotes the nearest-integer operator. It is a well-known fact that histogram equalization increases the image contrast, but at the same time tends to amplify noise [87]. An extension of histogram equalization is histogram modification, in which the intensity mapping function is chosen so that the resultant image has a histogram that is similar to a predefined histogram. The predefined histogram can be created from, for example, the mean of several "well lit" faces [53].

The above list of normalization techniques is by no means exhaustive; other possibilities such as normalization in the DCT domain [22] also exist. However, in this work, we consider only the above normalization techniques. A comparative analysis of these techniques will be presented in Subsection 6.4.2.

6.3.2 Background Masking

A technique widely used in face and nonface classification is background masking [90, 110], in which the pixels near the four corners of the rectangular window are excluded from classification. These pixels either belong to the image background or lie near the face contour, which is highly variable from one person to another. Therefore, excluding these pixels not only reduces the dimensionality of the input but also improves classification accuracy.

In our work, a circular mask centered on the input image is used for background masking (Fig. 6.1a). When this circular mask is applied on a human face, only the central region of the face is selected (Fig. 6.1b). We find that face/nonface classification is entirely possible even when only this central part of the face is used. In addition, variations in this central region can be more easily characterized than variations in the face contour, especially for different individuals or views. Using the circular mask, as much as 17% of the window can be excluded. The masked region can be treated as a 2-D image pattern, or alternatively as a vector through lexicographic ordering of the pixels.



(a) circular mask (b) example 1: face (c) example 2: nonface

Figure 6.1: Background masking.

6.4 Naive Bayesian Classifier

This is a statistical-based approach to face and nonface classification. The two classes are *face* and *nonface*. Let **x** be a feature vector: $\mathbf{x} = (x_1, x_2, \ldots, x_N)$; the elements $x_1, x_2, \ldots, x_N$ are called features. Features can simply be pixel intensities; they also can be generated with sophisticated feature extraction techniques. Let $P(face|\mathbf{x})$ and $P(nonface|\mathbf{x})$ be the respective *a posteriori* probabilities for face and nonface classes. Let $p(\mathbf{x}|face)$ and $p(\mathbf{x}|nonface)$ be the respective class-conditional pdfs, and $P(face)$ and $P(nonface)$ be the respective *a priori* probabilities. Let λ_{fd} be the cost of a false detection[16], and λ_{fr} be the cost of a false rejection[17]; we assume the cost of a correct classification decision to be zero.

For an arbitrary input pattern **x**, the cost of classifying **x** into face class is

$$R_{face}(\mathbf{x}) = \lambda_{fd}\, P(nonface|\mathbf{x}). \tag{6.10}$$

Similarly, the cost of classifying **x** into nonface class is

$$R_{nonface}(\mathbf{x}) = \lambda_{fr}\, P(face|\mathbf{x}). \tag{6.11}$$

Clearly, the pattern should be classified into the class that gives the lowest classification cost. In other words, **x** is classified as a face if

$$R_{face}(\mathbf{x}) \le R_{nonface}(\mathbf{x}). \tag{6.12}$$

Combining with (6.10) and (6.11), we can express this classification criterion as

$$\frac{P(face\,|\,\mathbf{x})}{P(nonface\,|\,\mathbf{x})} \ge \frac{\lambda_{fd}}{\lambda_{fr}}. \tag{6.13}$$

Applying the Bayes theorem, which states that

$$P(face\,|\,\mathbf{x}) = \frac{p(\mathbf{x}\,|\,face)P(face)}{p(\mathbf{x})}, \tag{6.14}$$

and

[16] when a nonface pattern is incorrectly classified as a face pattern.

[17] when a face pattern is incorrectly classified as a nonface pattern.

$$P(nonface \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid nonface)P(nonface)}{p(\mathbf{x})}, \tag{6.15}$$

where $p(\mathbf{x})$ is the probability of observing the pattern $\mathbf{x}$, yields

$$\frac{p(\mathbf{x} \mid face)}{p(\mathbf{x} \mid nonface)} \geq \frac{\lambda_{Td}}{\lambda_{Tr}} \frac{P(nonface)}{P(face)}, \tag{6.16}$$

The left-hand-side term of (6.16) is the *likelihood ratio*

$$L(\mathbf{x}) = \frac{p(\mathbf{x} \mid face)}{p(\mathbf{x} \mid nonface)}. \tag{6.17}$$

The right-hand-side term of (6.16), on the other hand, is independent of $\mathbf{x}$, and can be treated as a threshold.

6.4.1 Density Estimation Using Naive Bayesian Model

To use decision rule (6.16), the class-conditional pdfs must be estimated. This is a difficult problem to tackle, especially in high-dimensional spaces when only limited data are available. There are a number of different approaches to solving this problem. Yang *et al.* [133] presented two different approaches based on mixture density. In the first approach, a mixture of factor analyzers are used, and the parameters of the mixture are estimated using the expectation-maximization (EM) algorithm. In the second approach, the Kohonen's self-organizing map [57] is used to divide face and nonface samples to subclasses (i.e. clusters). Fisher's linear discriminant analysis (FLD) is then applied to project the samples to a lower-dimensional feature space. The class-conditional pdfs are modeled as mixture of Gaussians, and each subclass (in the feature space) is modeled by a Gaussian. The parameters of the Gaussian mixture are again estimated using the EM algorithm. We presented in [80, 85] a face/nonface classification approach that is based on PCA for dimensionality reduction, and Gaussian mixtures for density estimation. A commonality in the above approaches is that they are extremely computation-intensive. The training usually takes long time, and it is not easy to update or retrain the resulting classifier to cope with new false classification.

We present an approach to estimating the class-conditional pdfs $p(\mathbf{x}|face)$ and $p(\mathbf{x}|nonface)$, which is based on the naive Bayesian model. In the naive Bayesian model, the statistical dependency between elements of the feature vector is not modeled, i.e. the elements are assumed to be statistically independent. With this assumption, the problem of estimating the probability densities is transformed from a problem in N-dimensional space to a problem in 1-D space, which is more manageable. The class-conditional pdfs for a given feature vector $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ are estimated as follows:

$$p(\mathbf{x} \mid face) = \prod_{i=1}^{N} p_i(x_i \mid face), \tag{6.18}$$

and

$$p(\mathbf{x} \mid nonface) = \prod_{i=1}^{N} p_i(x_i \mid nonface), \tag{6.19}$$

where $p_i(x|face)$ and $p_i(x|nonface)$ are, respectively, the marginal class-conditional pdf of the ith feature in $\mathbf{x}$. Although the assumption of independent features is not entirely accurate (e.g. certain regions of the face have correlated intensity), in practice the class-conditional densities estimated based on this assumption can form a basis for face/nonface discrimination. A similar naive Bayesian approach has been used by Schneiderman and Kanade [100]. In their approach, each image window (of size 64×64) is decomposed in to a set of overlapping sub-regions (of size 16×16), and the overall class-conditional pdfs are estimated as the products of the class-conditional pdfs of the subregions. The class-conditional pdfs for a subregion are determined according to both its appearance (i.e., subregion intensity) and position (i.e., subregion coordinates). In comparison, in our approach different choices of the feature vector $\mathbf{x}$ are possible, and one of our goals is to investigate the suitability of such choices for the naive Bayesian model.

It is convenient for implementation purposes to express the decision rule (6.16) in logarithmic form. We define the *log likelihood ratio* as

$$\mathcal{L}(\mathbf{x}) = \log L(\mathbf{x}), \tag{6.20}$$

which can be expressed, using (6.17), as follows:

$$\mathcal{L}(\mathbf{x}) = \log p(\mathbf{x} \mid face) - \log p(\mathbf{x} \mid nonface). \tag{6.21}$$

The pattern **x** is classified into the face category if

$$\mathcal{L}(\mathbf{x}) \geq \theta_{face}, \tag{6.22}$$

where θ_{face} is a threshold with a theoretical value of

$$\theta_{face} = \log\left(\frac{\lambda_{fd}}{\lambda_{fr}} \frac{P(nonface)}{P(face)}\right). \tag{6.23}$$

Using (6.18) and (6.19), we can rewrite (6.21) to emphasize the role of individual features of **x** in the final classification:

$$\begin{aligned}\mathcal{L}(\mathbf{x}) &= \sum_{i=1}^{N} \log p_i(x_i \mid face) - \sum_{i=1}^{N} \log p_i(x_i \mid nonface) \\ &= \sum_{i=1}^{N} \{\log p_i(x_i \mid face) - \log p_i(x_i \mid nonface)\} \quad .\end{aligned} \tag{6.24}$$

The term inside the last summation sign can be expressed as a scalar function:

$$\mathcal{L}_i(x) = \{\log p_i(x \mid face) - \log p_i(x \mid nonface)\}\Big|_{x_i = x} \tag{6.25}$$

We can consider $\mathcal{L}_i(x)$ as a marginal "face score" produced when the ith feature of **x** has a value of x. Even though each score by itself is small, the final face/nonface discrimination score $\mathcal{L}(\mathbf{x})$ can be very robust as a result of combining the marginal scores of a large number of features.

The marginal class-conditional pdfs $p_i(x|face)$ and $p_i(x|nonface)$ are estimated using the traditional histogram technique. From a large set of face and nonface samples, two histograms are generated for each feature x_i:

- $h_{face,i}(x)$ is the number of face samples whose ith element $x_i = x$;
- $h_{nonface,i}(x)$ is the number of nonface samples whose the ith element $x_i = x$.

The individual class-conditional pdfs are simply the normalized histograms:

$$p_i(x \mid face) = \frac{h_{face,i}(x)}{\sum_x h_{face,i}(x)}, \tag{6.26}$$

$$p_i(x \mid nonface) = \frac{h_{face,i}(x)}{\sum_x h_{face,i}(x)}, \tag{6.27}$$

If the number of features (i.e. number of elements of the feature vector **x**) is N, and the number of histogram bins per feature is K, then each histogram will have $N \times K$ entries. This number is many magnitudes smaller than the number of entries (K^N) required for a full histogram in N-dimensional space. Apart from memory consideration, there are usually not enough training samples for a direct (full histogram) estimation of probability density in a high dimensional space. These are manifests of the *curse of dimensionality* of pattern recognition [48].

6.4.2 Preparation of Face and Nonface Patterns

In the following study of face/nonface classification using the naive Bayesian classifier, we use a dataset of over 9,000 different face windows (Set 5 of the ECU database). To create this dataset, we collected images containing faces from the following sources:

- the AR face recognition database [70],
- the AT&T face database [97],
- the BioID face detection database [50],
- JAFEE face database [68],
- the Stirling university face database [39],
- the UMIST face database [36],
- Web images, and images we took with digital cameras.

The face windows were manually cropped from these images. The windows contain faces that are different in terms of persons, facial expressions, and/or lighting conditions. We also prepared nonface windows by selecting random rectangular regions in 500 landscape and scenery digital photos (Set 7 of the ECU database) that are known to contain no faces. Examples of face and nonface windows are shown in Fig. 6.2 and Fig. 6.3. These figures are screenshots of the GUI program written to manage the ECU

database. Although some of the windows are in full color, face/nonface classification described in this chapter is always done in the gray-scale domain.

Figure 6.2: Examples of face patterns.

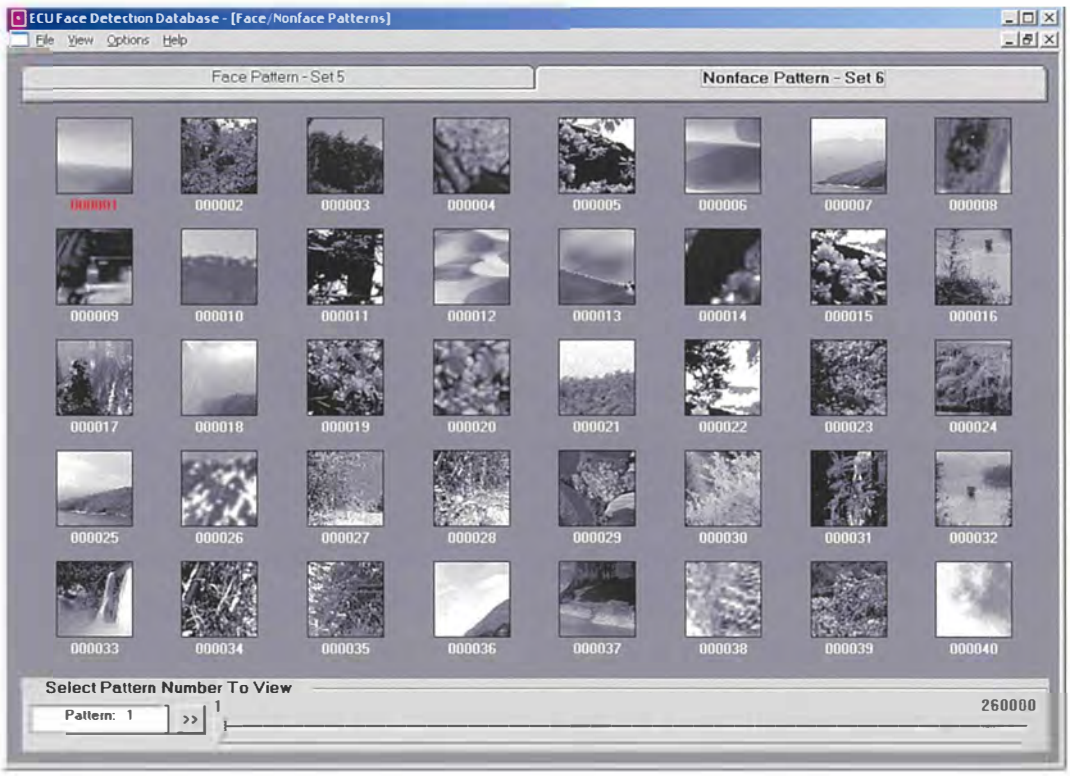


Figure 6.3: Examples of nonface patterns.

The set of 9,000 face windows was partitioned into two parts: part 1 of 6,000 face windows was used for training, and part 2 of 3,000 face windows was used for testing. For training purpose, each training face window was rotated, scaled and shifted by small random amounts to generate 10 extra face windows. Although we focused on detecting quasi-frontal upright faces, these artificial variations would make the classifier more robust to small variations in orientation, scale and position. The final data sets were:

- "Train 1": 60,000 face patterns and 120,000 nonface patterns, used for training;
- "Test 1": 3,000 face patterns and 6,000 nonface patterns, used for testing.

6.4.3 Intensity Feature Vector

In this feature extraction approach, the feature vector **x** consists of pixel intensities (after an image normalization step). The input window is rescaled to 16×16 size. A circular mask is applied (i.e. background masking), and the image pixels within the circular mask are collected to form a feature vector **x** that has $N = 216$ features. The number of histogram bins used for the estimation of marginal class-conditional pdfs is $K = 64$ bins. In this subsection, we first analyze the performance of different image normalization techniques. The best normalization technique is selected for use throughout this chapter. We then analyze thoroughly the performance of the intensity feature vector.

A. Analysis of Image Normalization Techniques

Image normalization has been used in almost all existing face/nonface classification algorithms. However, to the best of our knowledge, there has been no reported investigation of different image normalization techniques. It is known that image normalization techniques can reduce the variations in the visual pattern caused by the lighting conditions. However, it is not clear how such techniques will affect the separability between the face and nonface classes. Quite often, the normalization technique is heuristically chosen by the classifier designer.

We present here a study into the effectiveness of the normalization techniques that have been described in Subsection 6.3.1. In our experiment, these normalization techniques are individually applied to the input window before the feature vector **x** is extracted.

The classification rates of the naive Bayesian classifier are then recorded and used as comparative measures. We used set "Train 1" for training, and set "Test 1" for testing. It should be pointed out that the two sets covered a wide range of lighting possibilities. The ROC curves of the naive Bayesian classifier for different normalization techniques are shown in Fig. 6.4. The results clearly show that face/nonface classification is significantly improved with image normalization. For example, for a false detection rate between 5% and 30%, there is an increase of more than 5% in the correct detection rate (compared to no normalization) when the range normalization is used. The increase in the correct detection rate is even much higher (in some case more than 25%) for histogram equalization. These experimental results also show that the histogram equalization technique is significantly better than the other four normalization techniques, namely mean, range, standard deviation normalization and gradient illumination correction. The standard deviation normalization technique is the next best in performance. This technique, which can be implemented through linear filtering, is less computation-intensive compared to the histogram equalization technique.

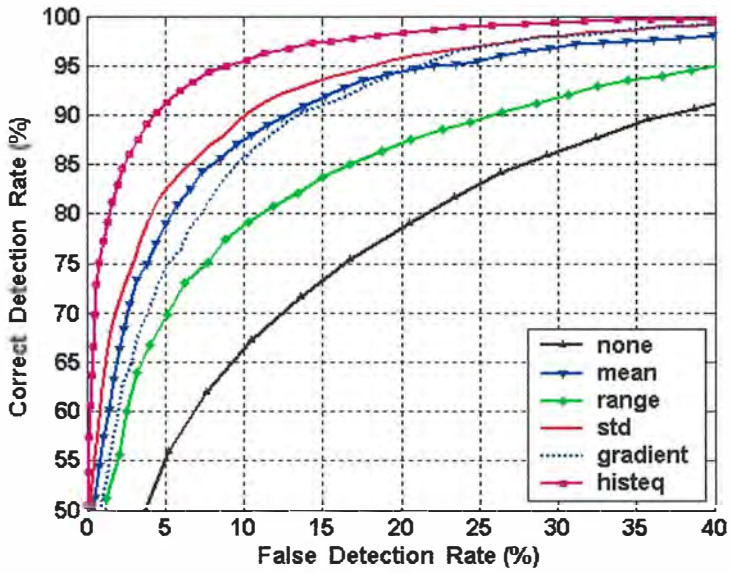
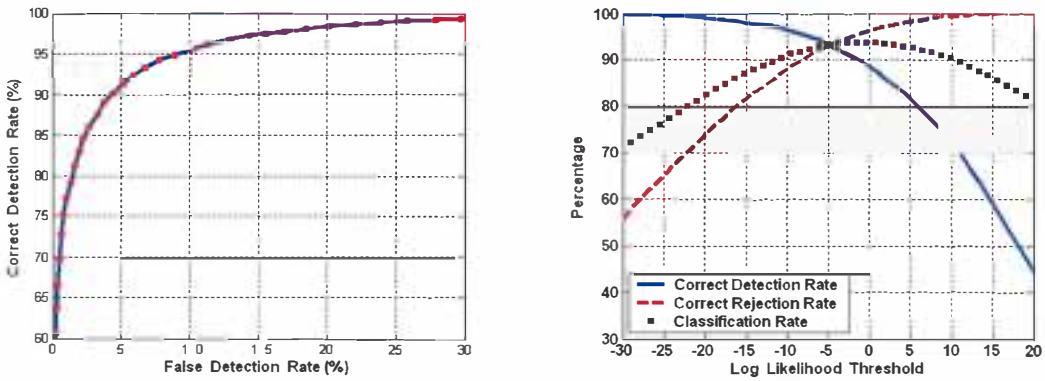


Figure 6.4: Comparison of image normalization techniques.

B. Analysis of Intensity Feature Vector

The input image is preprocessed using the histogram equalization technique, and an intensity feature vector of size 216 is extracted. The naive Bayesian classifier was

trained with set "Train 1" and tested on set "Test 1". The ROC curve in Fig. 6.5a shows that the classifier can achieve correct detection rates of 91% and 96% for false detection rates of 5% and 10%, respectively. The classification rates at various log-likelihood thresholds are shown in Fig. 6.5b. At threshold $\theta_{\text{face}} = 0$, the correct detection rate is 89.1% and the correct rejection rate is 96.2%. At threshold $\theta_{\text{face}} = -5$, the correct detection rate is 93.4% and the correct rejection rate is 93.2%. We can conclude that reasonable classification performance can be achieved even when no statistical dependency among features is modeled.



(a) ROC curve (b) classification rates

Figure 6.5: Performance of naive Bayesian classifier with intensity feature vector.

Once the marginal class-conditional pdfs are found, the pattern that produces the highest face score $\mathcal{L}(\mathbf{x})$ can be identified as follows. For $i = 1, 2, \ldots, N$, we find the value of the ith feature that maximizes the marginal face score defined in (6.25):

$$x_i^* = \arg\max_x F_i(x). \tag{6.28}$$

The 2-D pattern reconstructed from the feature vector $\mathbf{x}^* = \left(x_1^*, x_2^*, \ldots, x_N^*\right)$ has the highest face score, and can be considered as the "face concept" learned by the classifier. The "face concept" for the naive Bayesian classifier with intensity feature vector is shown in Fig. 6.6a. The two eyes in the "face concept" are very distinctive; the mouth and nose regions have a blurry but still visible boundary. This can be attributed to the fact that the classifier has been trained using quasi-frontal upright faces. The upright condition specifies a horizontal eye line, but the quasi-frontal condition allows for certain yaw and pitch rotations. The 2-D pattern, for which the classifier produces the

lowest face score, is called the "nonface concept" learned by the classifier. The "nonface concept" produced by the naive Bayesian classifier is shown in Fig. 6.6b.

(a) face concept: $\mathcal{L}_{max} = 72.1$

(b) nonface concept: $\mathcal{L}_{min} = -325.3$

Figure 6.6: The face and nonface concepts learned by the naive Bayesian classifier with intensity feature vector.

6.4.4 Projection onto Face Subspace Feature Vector

This choice of feature vector, namely projection onto face subspace (PFS) feature vector, is motivated by the eigenface approach in face recognition, which was described in Chapter 2. Each input pattern is represented by its linear projection onto the face subspace. This face subspace is spanned by the eigenvectors that are computed from a training set of face patterns. We are interested in experimenting with the PFS feature vector because of two reasons: (i) the dimensionality of the feature vector is reduced through PCA; (ii) the elements of the feature vector are uncorrelated for the face class.

Suppose there are M training face patterns $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_M$, each of which is a D-dimensional column vector formed through lexicographic ordering of the face images. First, the average face pattern $\mathbf{F}_m$ is calculated:

$$\mathbf{F}_m = \frac{1}{M}\sum_{i=1}^{M}\mathbf{F}_i . \tag{6.29}$$

The covariance matrix of face patterns is estimated as:

$$\mathbf{C} = \frac{1}{M-1}\sum_{i=1}^{M}(\mathbf{F}_i - \mathbf{F}_m)(\mathbf{F}_i - \mathbf{F}_m)^{\mathrm{T}} . \tag{6.30}$$

N eigenvectors of this covariance matrix that correspond to the N largest eigenvalues are selected. The selected eigenvectors, also known as the principal components, form a basis $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N)$ for a face subspace. The PFS feature vector $\mathbf{x}$ is computed for each input pattern $\boldsymbol{I}$ (face or nonface) as follows:

$$\mathbf{x} = \mathbf{V}^T(\mathbf{I} - \mathbf{F}_m). \tag{6.31}$$

For the computation of the PFS feature vector, we use a window size of 16×16. Background masking and histogram equalization are applied as pre-processing techniques. The 6,000 face patterns in set "Train 1" were used to generate a set of N = 100 principal components. This set accounts for more than 97% of the variances in the training set.

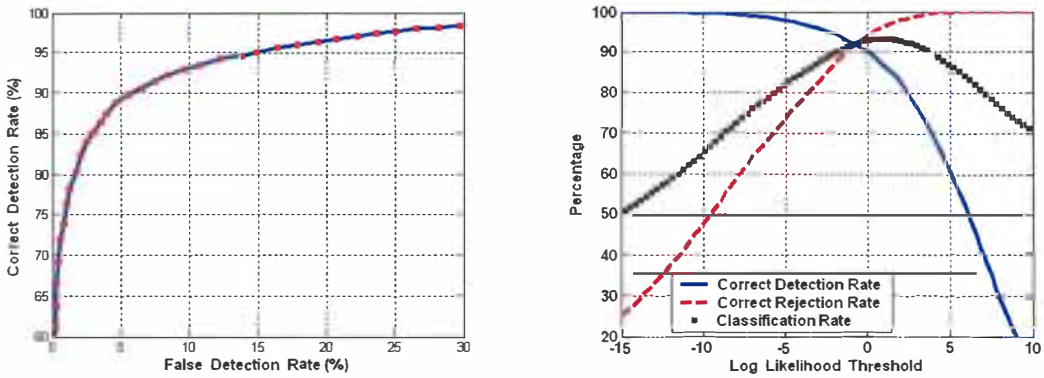
The PFS feature vector extracted is processed by the Bayesian classifier, which was developed using set "Train 1". As before, we identify the input pattern, for which the classifier produces the most positive response. This pattern, shown in Fig. 6.7a, appears to be more realistic than the pattern produced by the Bayesian classifier that accepts intensity feature vector (Fig. 6.6a). The "nonface concept" produced by the naive Bayesian classifier with PFS feature vector is shown in Fig. 6.7b.

The ROC curve and the classification rates of the naive Bayesian classifier with PFS feature vector are shown in Fig. 6.8. Set "Test 1" was used for testing. The classifier achieves correct detection rates of 88.8% and 92.9% for false detection rates of 5% and 10%, respectively. The classification rates at thresholds $\theta_{face} = 0$, $\theta_{face} = -5$, and $\theta_{face} = 5$ are 92.9%, 81.7%, and 86.8%, respectively. Performance comparison between the PFS feature vector and the intensity feature vector will be presented later in this chapter.

(a) face concept: $\mathcal{L}_{max} = 62.4$

(b) nonface concept: $\mathcal{L}_{min} = -454.3$

Figure 6.7: The "face concept" and "nonface concept" learned by the naive Bayesian classifier with PFS feature vector.



(a) ROC curve (b) classification rates

Figure 6.8: Performance of naive Bayesian classifier with PFS feature vector.

6.4.5 Edge-based Feature Vector

In this feature extraction approach, face patterns are characterized by image edges. Compared to pixel intensities, image edges are visual features that are less sensitive to the scene illumination. In the past, face detection approaches based on comparison between the edge mask of the input window and an edge-based face model have been proposed [50, 108, 123]. In these approaches, the similarity score is computed, through some variant of the Hausdorff distance, without reference to nonface patterns. In this subsection, we present a face/nonface classifier that is based on the naive Bayesian model and uses an edge-based feature vector. The difference between our approach and the existing edge-based approaches is that the face similarity score is constructed from both face and nonface patterns.

(a) pattern 1: face (b) edge mask 1 (c) pattern 2 - nonface (d) edge mask 2

Figure 6.9: Edge masks of face and nonface windows.

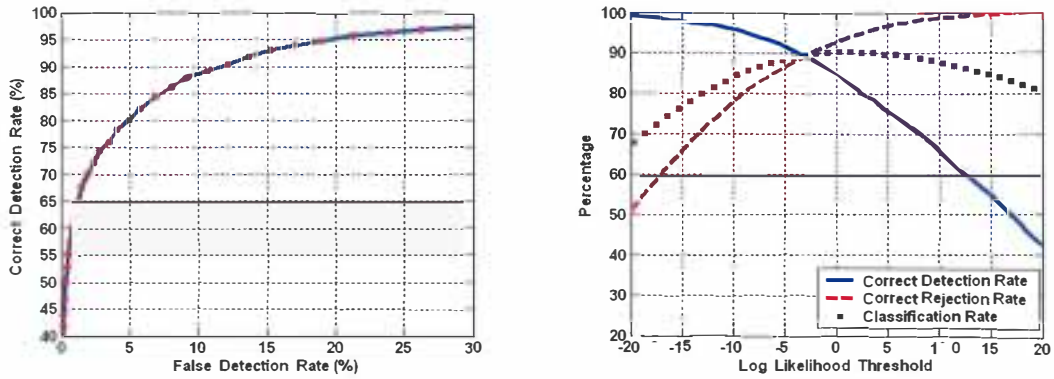
In our approach, the Canny edge detector is applied on the input window to produce an edge mask (Fig. 6.9). Entries of the edge mask are binary: 1's represent edge pixels, and 0's represents nonedge pixels. A feature vector is then extracted through lexicographic ordering of the edge mask. Only edge mask entries that are inside the central circular region are selected (see Fig. 6.1 on background masking). Because image edges exhibit strong invariance to the lighting condition, we decide not to apply any lighting correction technique. In addition, we find that for edge detection to be effective, the input window must have a reasonably large size; we use a window size of 64×64. Each input window is represented by a feature vector **x** of 3,300 binary elements.

The class-conditional densities $p(\mathbf{x}|face)$ and $p(\mathbf{x}|nonface)$ are estimated using set "Train 1" as described in Subsection 6.4.1. Using a similar method as described in Subsections 6.4.3 and 6.4.4, we identify the edge patterns, for which the face/nonface classifier produces the highest and the lowest face scores. These patterns (i.e. face and nonface concepts) are shown in Fig. 6.10a and Fig. 6.10b, respectively. We can conclude that the "face concept" learned by the classifier has edge pixels in the eye, nose and mouth regions. The shapes of these regions in Fig. 6.10a are relatively wide, which can be explained by spatial variations in the facial edges. The "nonface concept" in Fig. 6.10b has edge pixels at spatial positions that correspond to smooth regions of the face (e.g. the cheek and the chin).

The naive Bayesian classifier with edge-based feature vector was run on set "Test 1". The performance indicators of the classifier on this test set are shown in Fig. 6.11. The classifier had correct detection rates of 81.03% and 88.9% for false detection rates of 5% and 10%, respectively. The classifier using edge-based feature vector is quite accurate, but is not as good as the classifier using intensity or PFS feature vector.

(a) face concept: $\mathcal{L}_{max} = 254.1$ (b) nonface concept: $\mathcal{L}_{min} = 402.2$

Figure 6.10: The face and nonface concepts (edge masks) learned by the naive Bayesian classifier with PFS feature vector.



(a) ROC curve (b) classification rates

Figure 6.11: Performance of naive Bayesian classifier with edge-based feature vector.

6.4.6 Discussion of Naive Bayesian Classifier

A. Comparison of Feature Vectors

We compare the three different feature vectors presented in the previous subsections, namely intensity feature vector, PFS feature vector, and edge-based feature vector. The ROC curves of the Bayesian classifiers using these feature vectors on set "Test 1" are shown in Fig. 6.12. For comparison purpose, we also show the ROC curve of the template matching classifier, described in Subsection 5.4.6, on the the same test set. The figure shows that all three naive Bayesian classifiers outperform the template matching classifier by large margins. At any false detection rate between 5% and 25%, there is a difference of more than 10% between the CDR of the naive Bayesian classifiers and the CDR of the template matching classifier. Among the three Bayesian classifiers, the classifier with intensity feature vector has the best performance. However, there is only a small difference in the performances of the classifier using intensity and PFS feature vectors. The edge-based feature vector does not perform as well as the other two feature vectors.

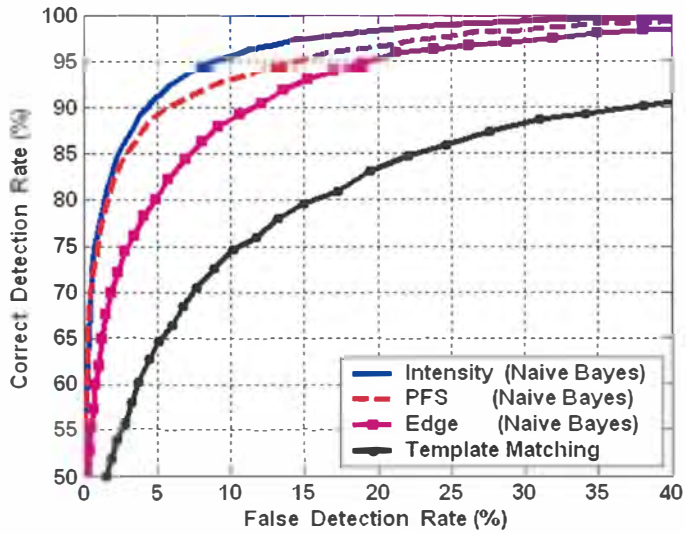


Figure 6.12: Comparison of feature vectors used in the naive Bayesian classifier: intensity, PFS, and edge-based. The ROC curve of the template matching classifier is included as a baseline for comparison.

B. Advantages of Naive Bayesian Classifier

The experiment results presented in this section have shown that the naive Bayesian classifier can achieve quite high classification rates. We also find that the classifier is robust to various changes in facial expression, lighting conditions, and viewing angles (within the assumptions of quasi-frontal upright faces). Compared to other classifiers, the naive Bayesian classifier has three important advantages regarding computation efficiency. First, it is fast in classifying input windows because classification involves mostly table lookup operations. Second, it requires a relatively small amount of memory. For example, a classifier using intensity feature vector needs, assuming that each floating-point value takes 8 bytes, only $8N \times K = 8 \times 216 \times 64 = 108$ KB of memory. Third, training the naive Bayesian classifier is much faster than training other classifiers, such as neural networks and parametric density models. Because of the simplicity of the training algorithm (i.e. the histogram-based density estimation), very large data sets can be used in training the Bayesian classifier. For example, for set "Train 1" with a total of 180,000 patterns, it takes only about 5 hours on a Pentium III 600 MHz to generate the class-conditional pdfs (i.e. 0.1s per training sample). In comparison, for a feed-forward neural network, a single pass through the same training set may take days of computation; for such a large training set, it requires typically

thousands of passes for the network to converge. Furthermore, it was pointed out that in [90], for large training sets, by the time the network reaches the last pattern of the training set, it may have forgotten the characteristics of the first pattern. Classifiers that require clustering in a high-dimensional space or parametric density estimation using the expectation-maximization algorithm must face similar dilemmas.

C. Other Feature Extraction Techniques

Besides FPS and edge-based techniques, other feature extraction techniques can also be used with the Bayesian classifier. One of such techniques is independent component analysis (ICA). Unlike PCA, which identifies components that best explain the variance in the data, thereby allowing a least-square-error reconstruction of the samples, ICA seeks directions that are most independent from each other. ICA has many applications in the domain of *blind source separation*[18], which aims to extract original signals from observed data. The observed data are mixtures of the original sources, which are assumed to be independent. ICA is also a promising tool for feature extraction task in pattern classification because the extracted features are highly independent, and as such the naive Bayesian model can be readily applied to compute the multivariate density function as in (6.18) and (6.19). ICA has been applied in tasks such as face recognition and facial expression analysis [2], and cork-stopper classification [88]. However, ICA often involves a gradient-descent optimization of a multi-dimensional target function (e.g. the entropy) [21], which is very computation-intensive. In this thesis, ICA is not used for feature extraction primarily due to computational concerns. Other possible feature extraction techniques are projections to vertical and horizontal axes, and wavelets.

[18] The terms *blind* signifies the fact that the mapping between the original sources and the observed data is unknown.

6.5 Improving Face/Nonface Classification

In this section, we discuss a number of techniques for improving the performance of the naive Bayesian classifier. In general, there are three main strategies for classifier enhancement. The first strategy involves progressive training and updating of the classifier; it is often called the *bootstrap* strategy. The second strategy is *classifier combination,* in which the classification problem is solved by combining multiple classifiers that can be different in terms of classification architecture, feature vector, training set, and even performance. The third strategy makes use of *contextual* information in forming the final classification decision. These three classifier enhancement strategies will be examined below.

6.5.1 Bootstrap Strategy

With the bootstrap strategy, a classifier is first trained with a limited training set of face and nonface patterns. The partially-trained classifier is then run on a larger validation set of face and nonface patterns, and false detections and false rejections are collected to be used as further training examples for the classifier. The bootstrap strategy was described by Sung [110], and since then it has been used in many other face/nonface classification algorithms [75, 90]. There are two major justifications for bootstrapping:

- First, it is a means to overcome the difficulty in selecting a representative training set. Like in many other object detection problems, it is easy to collect representative face patterns, but it is harder to collect representative nonface patterns. In addition, the nonface class is magnitudes larger than the face class; for example, in an image of size 640×480 there can be hundreds of thousands nonface patterns, whereas there are typically at most a few dozens face patterns. Therefore, we may run the risk of omitting nonface patterns that are essential for learning the decision function. This problem does not exist in training algorithms that use only positive training examples (i.e. face patterns), but such algorithms are very poor in defining the decision boundary between face and nonface classes.

- Second, bootstrapping allows the classifier to be trained progressively. Presenting a huge training set to the classifier is computationally prohibitive in terms of storage and processing time. It is far more efficient to train the classifier using small chunks of samples, and to focus the training efforts on "hard" examples in the feature space.

The implementation of bootstrap strategy for the naive Bayesian classifier is quite straightforward. It involves updating the feature histograms, and then the marginal pdfs. It is, therefore, necessary to keep the histograms for both face and nonface classes. In our work, the updated classifier is evaluated on a validation set of 1,000 faces and 5,000 nonfaces to prevent overfitting, and to identify the point where bootstrapping is no longer effective. In this thesis, bootstrapping is implemented only to this extent. However, there are two possibilities for extending the bootstrap strategy:

- adding new Bayesian classifiers to the classification system to handle new classification errors. These new classifiers focus on sub-regions of the feature space. Issues such as avoiding overfitting and combining individual classification scores must be considered.
- training a cascade of Bayesian classifiers using boosting algorithms such as AdaBoost [29, 30]. This approach has been taken by Schneiderman [99].

6.5.2 Classifier Combination

Classifier combination has been an active research topic in pattern recognition [55, 56, 58]. By exploiting complementary capabilities of several classifiers, classifier combination aims to achieve a better performance than that of an individual classifier, in terms of accuracy and robustness. Jain *et al.* [48] described three main architectures for classifier combination, namely *parallel*, *series*, and *hierarchical*:

- **Parallel architecture**: Individual classifiers are activated independently, and their outputs are fused by a combiner to form the final output. Simple combiners, such as majority voting, mean, weighted average, max, min, and median operators, are commonly used. More sophisticated combiners require extensive training but they can operate adaptively according to the input pattern. It can be shown that the mean combiner reduces classifier variance.

- **Series architecture**: Individual classifiers are arranged in a sequential fashion; the range of classes that the input pattern may belong to is reduced at each stage along the classifier chain. The series architecture is functionally equivalent to the parallel architecture with the max and min combiners. However, the series architecture can be more computationally efficient in some cases.
- **Hierarchical architecture**: Individual classifiers are arranged in a tree-like structure.

It is also possible to use these architectures in a hybrid classification system. Classifier combination has been used for face/nonface classification by many authors (see Chapter 2). For example, Rowley *et al.* [91] used a combination of two feed-forward neural networks; Viola and Jones [118-120], Lienhart *et al.* [60, 61], Fröba *et al.* [31], Wang *et al.* [122] used the AdaBoost algorithm to train a cascade of classifiers; Elad *et al.* [22] constructed as series of linear classifiers through the maximal rejection mechanism.

We propose an approach of combining the three different naive Bayesian classifiers that were described and analyzed in Section 6.4. These classifiers use intensity, PFS and edge-based feature vectors. The schematic diagram of the proposed combination approach is shown in Fig. 6.13. An input window is processed by three Bayesian classifiers, and three log likelihood scores $\mathcal{L}^2_{\text{int}}$, $\mathcal{L}^2_{\text{pfs}}$, and $\mathcal{L}^2_{\text{edge}}$ are produced. These scores are mapped to the range [0, 1] using the following mapping function:

$$f_{(H,L)}(x) = \begin{cases} 0, & x \le L \\ 1, & x \ge H \\ (x-L)/(H-L), & L < x < H \end{cases}, \tag{6.32}$$

where H and L are the upper and lower limits of mapping. The mapping function is necessary because the scores produced by the three classifiers have different dynamic ranges. We use (H = 40, L = -60) for the intensity feature vector, (H = 20, L = -40) for the PFS feature vector; and (H = 60, L = -60) for the edge-based feature vector. The mapped face scores p_{int}, p_{pfs}, and p_{edge} in range [0, 1] can be considered as pseudo *a posteriori* probabilities. They are fused into the final score p using the mean operator; classification into face or nonface is done by thresholding this final score.

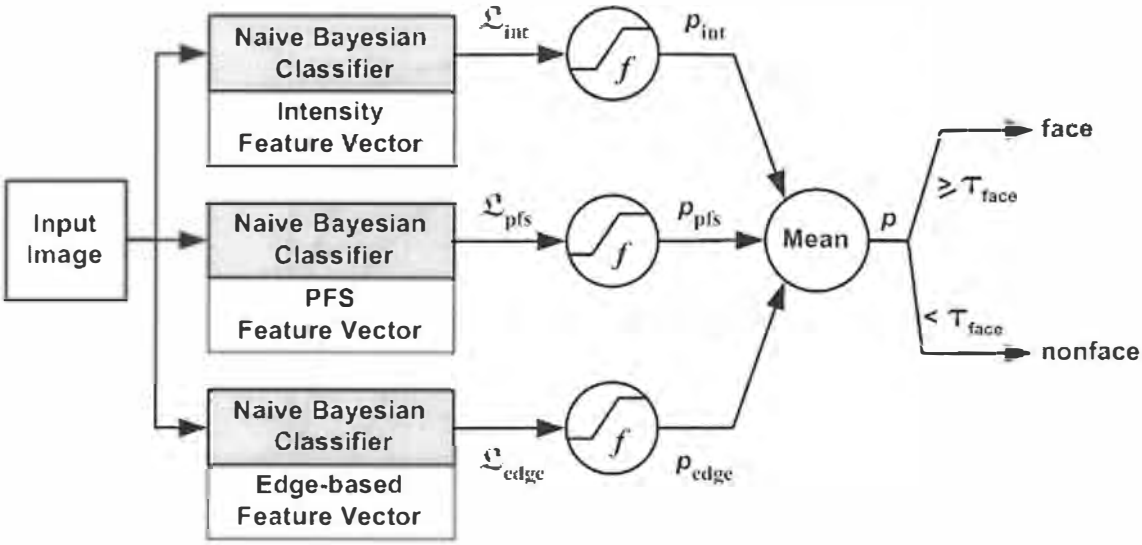


Figure 6.13: Classifier combination scheme.

The combination of classifiers was tested on set "Test 1". Results of comparison with the three individual classifiers are shown in Fig. 6.14. The figure confirms that classifier combination does improve face/nonface classification. At a FDR of 5%, the classifier ensemble has a CDR of 96.6% compared to a CDR of 91.0% of the best individual classifier. At a FDR of 10%, the classifier ensemble has a CDR of 98.6% compared to a CDR of 96.0% of the best individual classifier. A reason for this improvement is the diversity in the classifier ensemble, which is enforced through the use of different feature vectors. We believe that higher classification rates can be achieved by using more Bayesian classifiers that rely on new feature extraction schemes. The classification rates of the classifier ensemble at different face score thresholds are shown in Fig. 6.15.

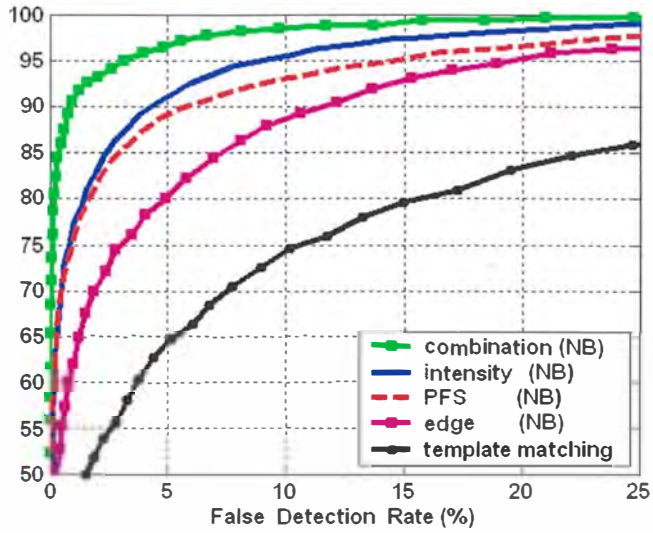


Figure 6.14: Comparison of classifier combination and three individual classifiers.

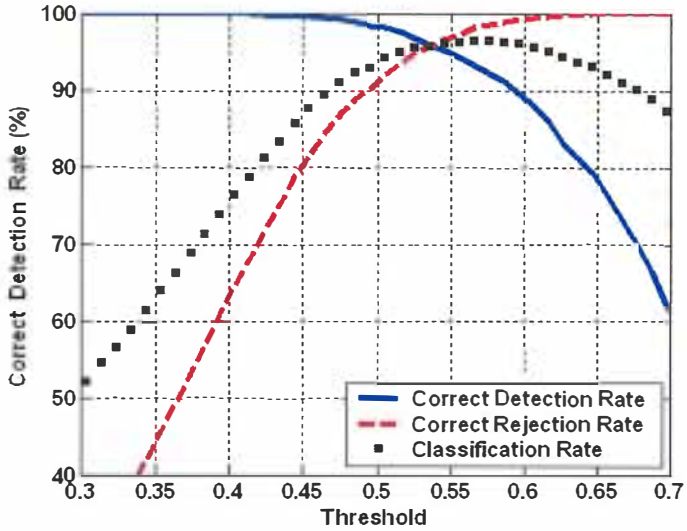


Figure 6.15: Classification rates of the classifier ensemble.

6.5.3 Using Contextual Information

It is widely agreed that more accurate classification decisions can be made by incorporating contextual information. For example, Rowley [90] observed that the face is often detected at multiple nearby scales and positions in the image, while false detections tend to occur with less consistency. In his approach, for a given detection, if

the number of detections within its neighborhood is below a certain threshold, the detection is rejected. Rowley also suggested that if a particular location is correctly identified as a face, then all other detections that overlap the location are likely to be errors. It is fair to say that these heuristics contribute very significantly to the classifier performance. For example, the false detection rate of a network trained in [90] is 1 in 89,546 without heuristics, and 1 in 161,044 with heuristics (i.e. an improvement of 1.8 times). For almost any classifier, it is possible to boost the detection results simply by tuning the parameters in the post-processing heuristics; however, this fact is often overlooked in reporting classifier performance.

We present here a technique of using contextual information to improve face/nonface classification, which is more suitable to our face detection strategy. The candidate selection algorithm proposed in Chapter 5 produces many face candidates that partially overlap; quite often the overlapping candidates belong to the same segmented skin region. Clearly, if two face candidates overlap, one of them must be removed; it is natural to reject the face candidate with a lower face score. For each face candidate F_i, the set of all face candidates that overlap it is identified. The face candidate F_i is kept only if its face score is greater than the face scores of the overlapping candidates.

A. Identification of Overlapping Candidates

We use a simple and fast technique to determine if two face candidates F_i and F_j overlap. First, overlapping is possible only if the two candidates are formed from the same segmented region. This means candidates from the same segmented skin regions must be processed together. Second, overlapping happens only if the bounding boxes of the two candidates overlap. Third, overlapping can occur only if the face masks of the two candidates overlap. These three tests are performed in series so that the costly computation in the third test is avoided unless absolutely necessary.

We do not consider overlaps where the amount of overlapping is small. That is, let $|F_i|$ and $|F_i|$ be the size of the two face masks, and $|F_i \cap F_i|$ be the size of the intersection between the face masks. The amount of overlapping is defined as:

$$o(F_i, F_j) = \frac{|F_i \cap F_j|}{\min(|F_i|, |F_j|)}. \tag{6.33}$$

When this overlap factor is below a threshold, the two candidates are not considered to overlap. Overlapping between bounding boxes in the second test is handled in a similar way. The only differences are that the overlap threshold is higher and $|F_i \cap F_j|$ can be evaluated very quickly given the box coordinates.

B. Selecting Face Score

The overlapping elimination post-processing technique is deterministic in the sense that only the best of the overlapping candidates is kept. However, for this technique to work, the face/nonface classifier must produce a face score that has a higher value for a pattern that is more similar to a face. For this purpose, we use the face score produced by the ensemble of Bayesian classifiers described in the previous subsection.

To illustrate the effect of the ensemble face score, we process the 19 face candidates that are shown in Fig. 5.20c. These face candidates belong to the same segmented skin region; some of the candidates overlap. There are two true faces among these candidates. In this case, the candidates are divided neatly into two groups, each of which consists of candidates that overlap with a true face. In Fig. 6.16, the face score produced by the classifier ensemble is shown next to each face candidate. The figure shows that:

- Using a face score threshold[19] of $\tau_{face} = 0.47$, there are 3 detections (candidates 1, 3, and 14), of which candidates 3 and 14 are correct detections, and candidate 1 is a false detection.
- Applying the overlapping candidate elimination technique, the false detection in candidate 1 is removed because it overlaps with two face candidates (3 and 14) that have higher face scores than its face score. Only two correct detections remain.

[19] At this threshold, the CDR of the classifier ensemble is 98.6% and the FDR is 10%.

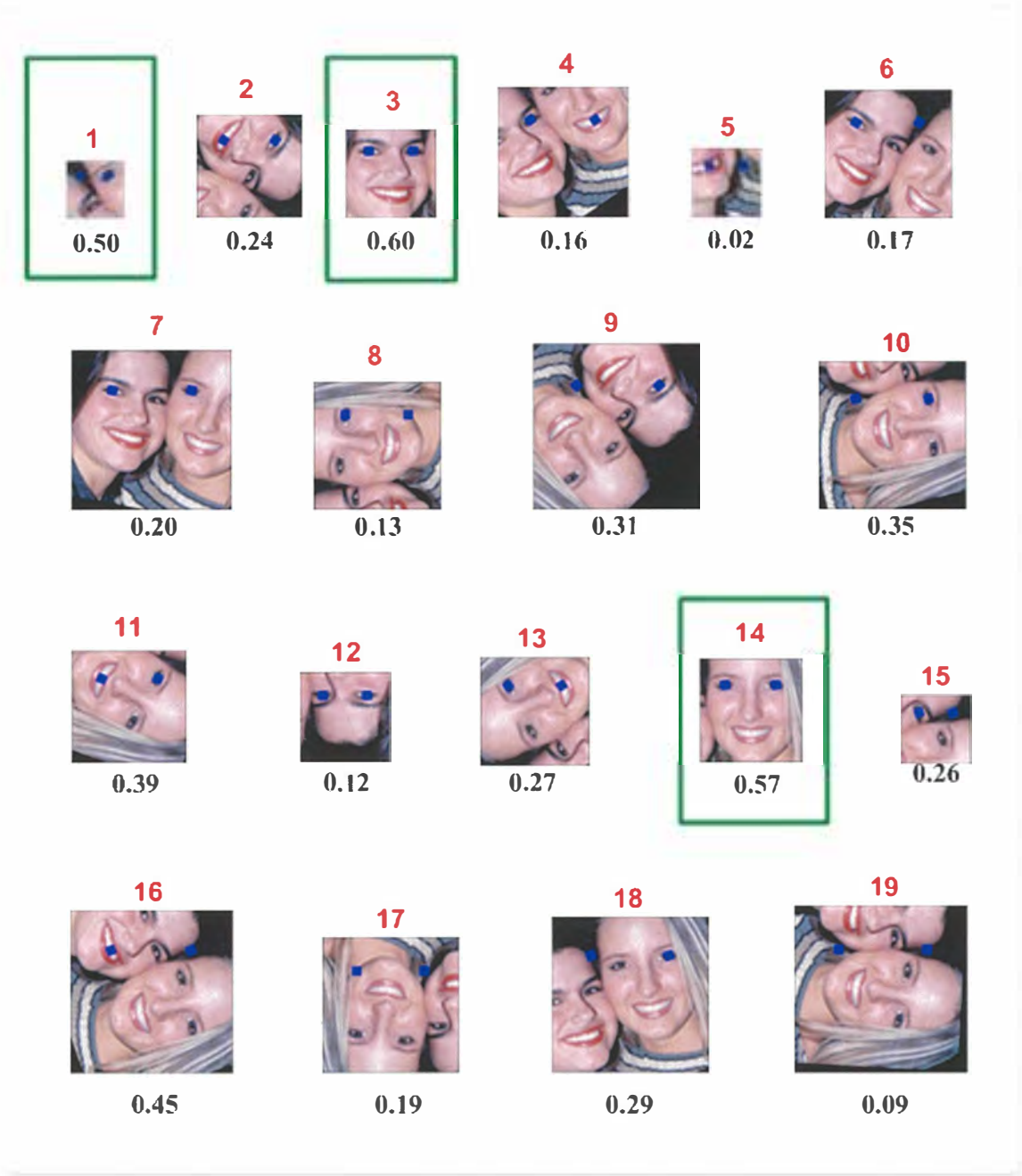


Figure 6.16: False detection elimination using contextual information. A face candidate is kept only if its face score is greater than face scores of all candidates that overlap with it.

6.6 Chapter Summary

We have presented an algorithm for face/nonface classification that is based on the naive Bayesian classifier and nonparametric estimation of marginal density. Because of the normalization of face candidates in the previous stage of our face detection algorithm, the classifier has to differentiate only quasi-frontal upright faces from all other patterns. Experiment results have shown that the naive Bayesian classifier has a very high accuracy. For the classifier using intensity feature vector, the correct detection rates are 91% and 96% at false detection rates of 5% and 10%, respectively. In addition, the naive Bayesian classifier has three important advantages compared to other classifier architectures: fast classification, small memory requirements, and fast training. We investigated also three choices of the feature vectors, namely intensity, PFS and edge-based; and five different image normalization techniques, namely mean, standard deviation, range normalization, histogram equalization, and gradient illumination correction. It is found that the intensity feature vector is slightly better than the PFS feature vector, but the latter requires less memory due to the dimensionality reduction capability of PCA. Both intensity and PFS feature vector perform better than the edge-based feature vector. We found that, with the intensity feature vector, image normalization improves classification performance very significantly, and the histogram equalization technique outperforms other normalization techniques. In this chapter, strategies for improving the performance of the naive Bayesian classifier were also discussed. They include bootstrapping, classifier combination, and using contextual information. Based on the analysis of these classifier enhancement techniques, we developed a final face/nonface classifier that combines three naive Bayesian classifiers. These naive Bayesian classifiers use intensity, PFS and edge-based feature vectors. The combined classifier is used to perform the last verification of face candidates.

Chapter 7

The Face Detector and Its Applications

7.1 Introduction

In this chapter, we present a complete system for detecting human faces in color images that integrates the various components described in the previous chapters of the thesis. The organization of the chapter is as follows. Section 7.2 describes the complete face detector and its major stages. Section 7.3 provides a comprehensive analysis of the face detector performance. Section 7.4 describes a number of applications of the face detector. Section 7.5 gives the chapter summary.

7.2 System Description

The face detector consists of five main stages as shown in its block diagram in Fig. 7.1:

- **Stage I**: *Skin Detection using Color Pixel Classification* (Chapter 3)
  Pixels of the input color images are classified as skin or nonskin pixels on the basis of their color. The classifier based on the Bayesian decision rule for minimum cost is used.
- **Stage II**: *Skin Region Segmentation* (Chapter 4)
  Skin region verification and skin refinement techniques are combined to enhance the results of skin detection in the previous stage.
- **Stage III**: *Feature-based Face Candidate Selection* (Chapter 5)
  In this stage, color-based eye detection and a geometric face model are used to construct face candidates from segmented skin regions. Preliminary face/nonface tests are also performed.
- **Stage IV**: *Face Candidate Verification* (Chapter 6)
  The face candidates produced in the previous stage are verified using face/nonface pattern classifiers that are based on the naive Bayesian model.
- **Stage V**: *Face Detection Output*
  In this final stage, the detection results are presented in a form that is suitable for particular needs of face detection.

Stages I to IV have been described thoroughly in chapters 3 to 6, respectively. Stage V is included in the face detector so that appropriate information about the detected faces can be produced according to the user needs. The record of a detected face consists of the following fields:

- **Face mask**: a rectangular binary mask that indicates the spatial extent of the face.
- **Face angle**: the angle by which the face needs to be rotated counter-clockwise so that it will be in the upright position.
- **Eye points**: the coordinates of the two eyes.
- **Face score**: a confidence score of the face detection.

Depending on the user needs, the face detector can optionally produce the following information regarding each detected face:

- **Skin mask**: a free-shape binary mask of the skin region that corresponds to the face. This skin mask can be combined with the face mask to generate a more precise face mask that follows the face contour. Accurate face contour is needed in applications such as facial image compression.
- **Normalized face**: the detected face is normalized so that it is in the frontal-upright position and has a size of 64×64. This normalized face is often used as input to a face recognition system.

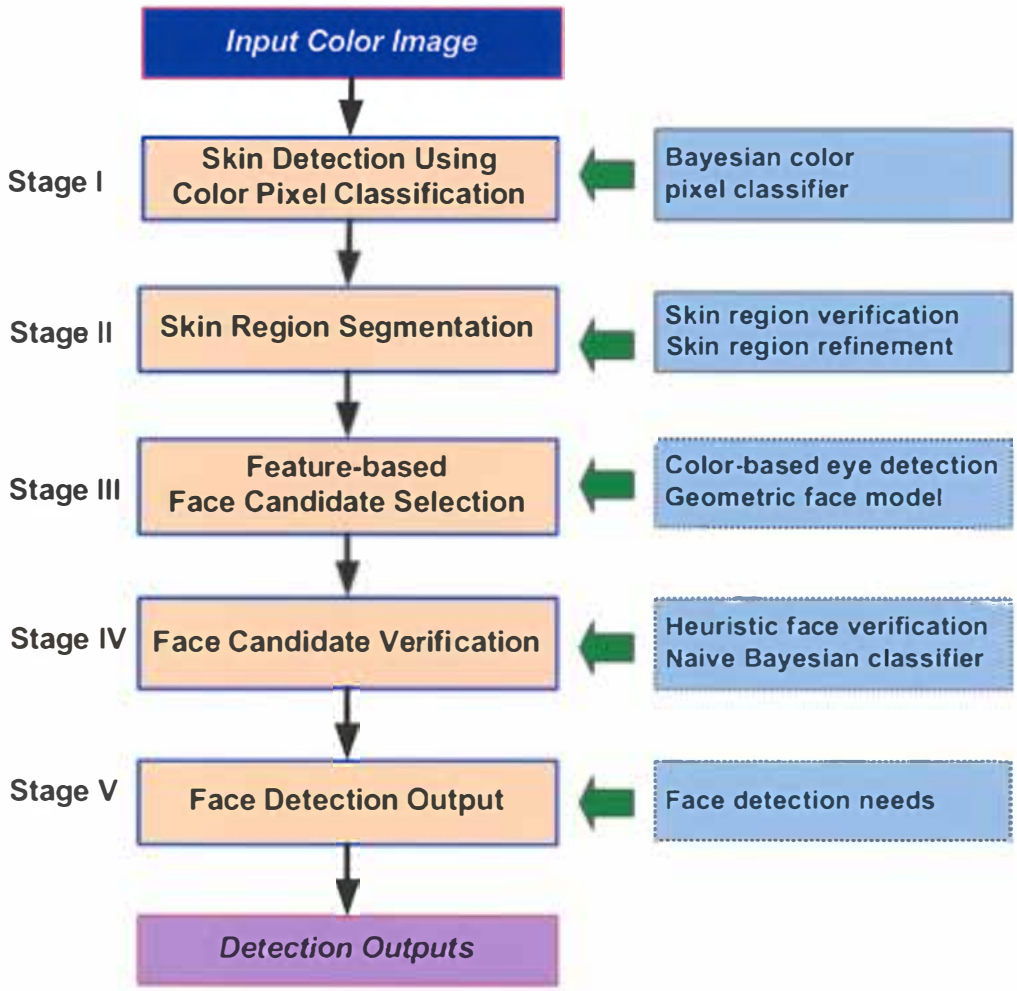


Figure 7.1: Block diagram of the proposed face detector.

7.2.1 Face Detection Database

Because of the color-based nature of the proposed face detection algorithm, it is necessary to evaluate the algorithm on a face detection database that consists of color images. To the best of our knowledge, no large and comprehensive database with such property exists online at the time of this writing. This deficiency has motivated us to develop the ECU face detection database to support the development and evaluation of both intensity-based and color-based approaches to face detection. The ECU database has been mentioned a number of times in this thesis, and its full description can be found in Appendix A.

7.2.2 Training the Face Detector

We used 2,500 color images for training (images 1 to 2,500 in the ECU database). Such a large number of training images were needed so that very large sets of skin, nonskin and eye pixels could be extracted. These sets were used in the estimation of the class-conditional densities of the skin, nonskin, and eye classes, as described in Chapters 3 and 5. Fewer images were actually used in determining appropriate parameters for the skin region segmentation algorithm and the face candidate selection algorithm. We excluded all training images from our tests below. A separate training set of face and nonface patterns was used for constructing the naive Bayesian face/nonface classifiers presented in Chapter 6. The patterns were collected from various online face databases as described in that chapter. The settings for individual stages of the face detector are described in the previous respective chapters.

7.2.3 Speed Optimization

In this section, we discuss a technique to improve the processing speed of the proposed face detector. In our analysis, the MATLAB profiling tool was used to assist in identifying the bottlenecks in the algorithm. We have found that one of the most time-consuming steps in the algorithm is normalizing the face candidate to the upright position after it is constructed from an eye pair. This normalization involves an image

rotation operation, which could be quite computation-intensive if the entire face candidate in its original size has to be rotated. However, we note that in the subsequent verification steps, the analysis window sizes are very small: 32×32 for face angle estimation and template matching classifier, 16×16 for naive Bayesian classifiers that use intensity and PFS feature vector, and 64×64 for naive Bayesian classifier with edge-based feature vector. Therefore, significant speed-up can be achieved if the face candidate is subsampled to a smaller size before rotation so that the normalized face candidate will have a size of 64×64. This technique is illustrated in Fig. 7.2. The bounding box (aligned with the vertical and horizontal axes) of the face candidate is identified, and the corresponding square image region (width W) is resized through subsampling to a square region of width $W' = 64(\cos\alpha + \sin\alpha)$, where α is the face angle. This resized region is rotated by an angle α, and the 64×64 pixel region at the center of the rotated region is taken as the normalized face candidate. This technique is not applied when the distance between the two eye points is smaller than 32.

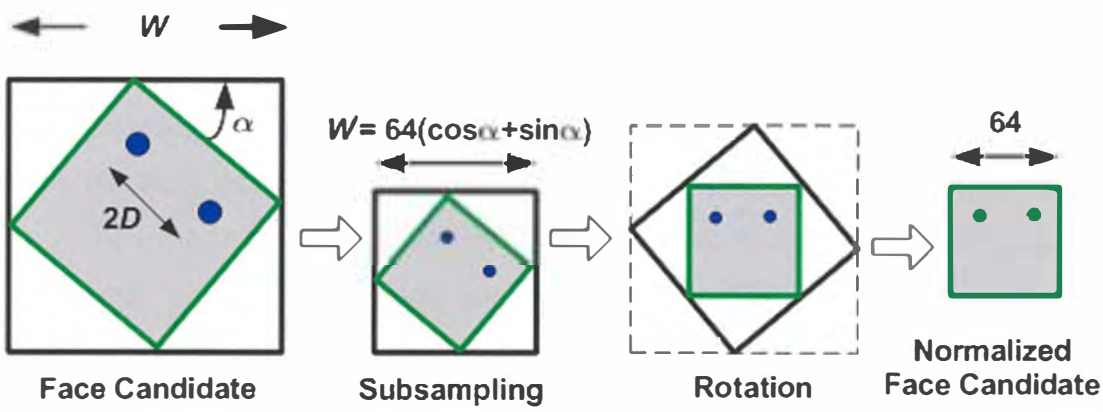


Figure 7.2: Rotation speed-up through window subsampling.

7.2.4 Software Implementation

The face detector was implemented using the MATLAB software – a mathematical package from MathWorks Inc. Most experiments in this thesis were performed on a Pentium III 600MHz with 512 RAM running Windows XP. Based on the software implementation, we have developed a MATLAB library for face detection, and a Web-based demo of the face detector. Both tools will be released online at http://www.soem.ecu.edu.au/~sphung/face_detection/.

7.3 Analysis and Discussion

7.3.1 Performance Measures

A face detection algorithm can be evaluated in terms of three major criteria: computation requirements, detection accuracy, and detection quality:

- Computation requirements refer to both processing time and memory storage needed by the algorithm to handle an image of fixed size.
- Detection accuracy is usually measured in terms of the correct detection rate (CDR) and the false detection rate (FDR). A correct detection is when a face is correctly identified, whereas a false detection is when a nonface image region is incorrectly identified as a face. It is widely accepted that the correct detection rate is the percentage of faces in an image test set that are detected by the algorithm. However, there is not yet a consensus on how to report the false detection rate. It is sometimes defined, especially in holistic face detection approaches, as the ratio of the number of false detections to the number of image windows examined by the classifier. This measure puts more emphasis on the accuracy of face/nonface classifier. However the measure is not suitable for reporting face detection performance because there is significant redundancy in these image windows. A uniform background patch, for instance, will generate a large number of obvious nonface windows. We suggest that it is more appropriate to define false detection rate as the ratio of the number of false detections to the number of faces. This measure is useful because it indicates the average number of false detections that the face detector makes per true face. We should point out that this FDR is a conservative measure as the number of face candidates tested usually far exceeds the number of faces in the image. In summary, we use the following formal definitions:

$$CDR = \frac{\text{Number of correct detections}}{\text{Total number of faces}} \times 100\% \tag{7.1}$$

$$FDR = \frac{\text{Number of false detections}}{\text{Total number of faces}} \times 100\% \tag{7.2}$$

- *Detection quality* is measured in terms of how well a detected face aligns with the corresponding true face. This performance criterion is often overlooked in face detection literature. Because the ECU database includes also manually segmented face images (Set 3), it can be used to evaluate the detection quality of face detection algorithms. We propose the following region-based alignment measure. Let F_{gt} be the ground-truth mask for a face and F be the mask produced by the face detector. The ground-truth mask can be obtained using the code example in Table A.3 (Appendix A). An alignment score can be defined as follows:

$$a = \frac{|F_{gt} \cap F|}{|F_{gt} \cup F|}, \tag{7.3}$$

where $|x|$ represents the number of 1-pixels in a binary mask x. This alignment score is the ratio between areas of the intersection and the union of the two masks F_{gt} and F; it is independent of the face size. A perfect alignment has a score of 1; a total miss has a score of 0. We find that an alignment score of 0.6 or more can be considered as a good detection.

Another approach to measuring face alignment is based on the normalized distances between the detected and the actual facial landmarks. The most common facial landmarks used for this purpose are the two eyes. For example, the eye-based alignment score is recommended with the BioID face detection database [50]. It is possible to use this alignment score for our face detector because it also locates the two eye points in each face. However, because the eye segmented images in the ECU database have been prepared *only* for eye color modeling purpose, we decided to adopt the region-based alignment score in (7.3).

7.3.2 Detection of Faces in the ECU Database

The face detector was applied on a test set of 200 images from the ECU database. The images in this test set contain faces that have the following properties:

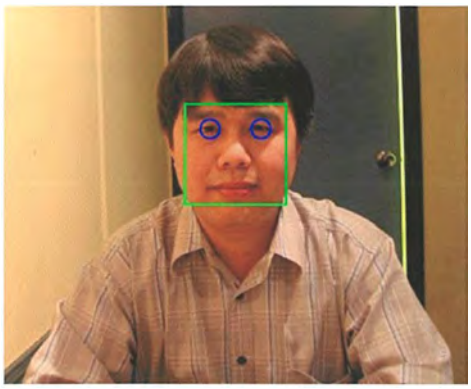
- different skin color types,

- different people,
- various facial expressions,
- quasi-frontal faces with arbitrary in-plane rotations,
- out-of-plane rotations with small angles,
- taken under different but moderate lighting conditions,
- with and without glasses.

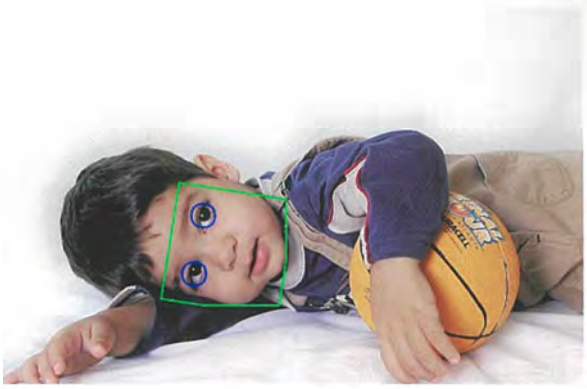
We should point out that the test images have quite high resolution, which is an essential requirement of the proposed eye detection algorithm. The image sizes range from 352×288 to 1280×960 pixels. The face sizes range from 40 to 300 pixels in width and height.

The detection results are listed in Table 7.1. The face detector can detect 208 out of 231 faces and make 10 false detections. That is, it has a correct detect rate of 90.04% and a false detection rate of 4.3%. The average alignment score between the detected faces and the true faces is 0.71. Example outputs of the face detector are shown in Fig. 7.3 and Fig. 7.4. Detected faces are indicated by green rectangles, and detected eyes by blue circles. Although, the face mask defined by the geometric face model is rectangular, the final face mask can be a polygon after combining with the rectangular box of the corresponding skin region. Results in Fig 7.3 and 7.4 show that the detected faces are well aligned with the true faces (the alignment scores vary between 0.6 and 0.7). The alignment score can be improved if we combine the polygonal face mask with the skin mask.

The complete results produced by our face detector for the above test set are included in Appendix B. Referring to Appendix B, false detections in images 2521, 2527, 2528, 2551, and 2563 are caused by the face/nonface classifiers' failure to reject the nonface patterns. False detection in image 2659 is due to the face score of the true face being lower than the face score of an overlapping nonface. False rejection in image 2654 is because the true face is rejected by the naïve Bayesian classifier with intensity feature vector.



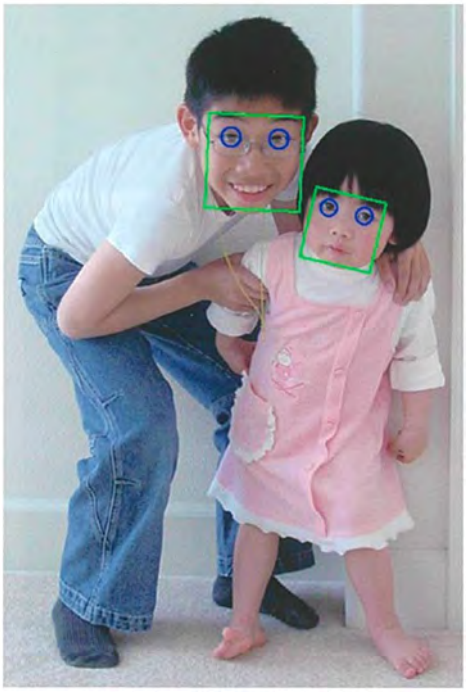
(a) TF = 1, CD = 1, FD = 0, a = 0.78



(a) TF = 1, CD = 1, FD = 0, a = .77

(c) TF = 5, CD = 5, FD = 0, a = 0.75

(d) TF = 1, CD = 1, FD = 0, a = 0.68

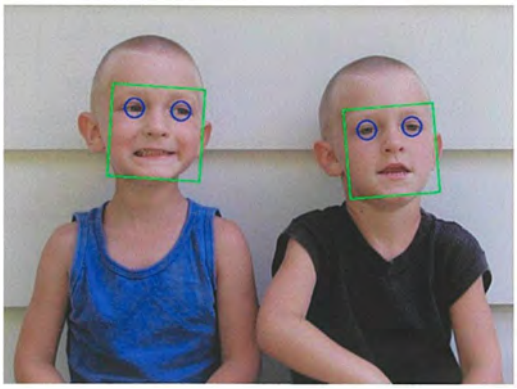


(e) TF = 2, CD = 2, FD = 0, a = 0.71

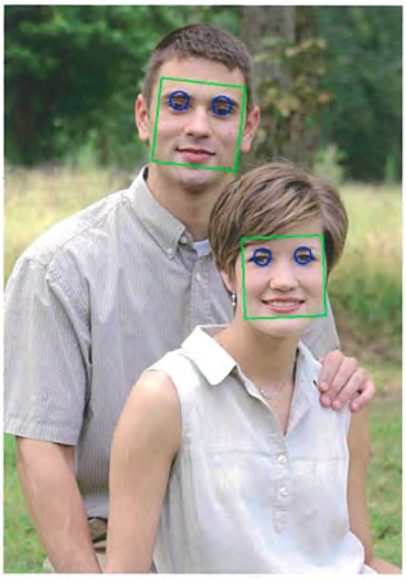
(f) TF = 1, CD = 1, FD = 0, a = 0.70

Figure 7.3: Visual results of face detection – Part I: TF = number of true faces, CD = correct detections, FD = false detections, a = average alignment score.

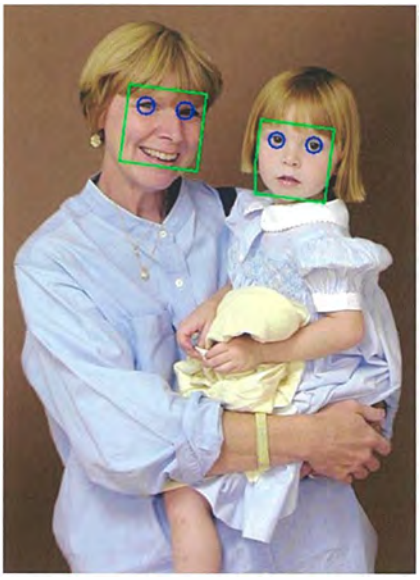
(a) TF = 1, CD = 1, FD = 0, a = 0.64



(a) TF = 1, CD = 1, FD = 0, a = .67

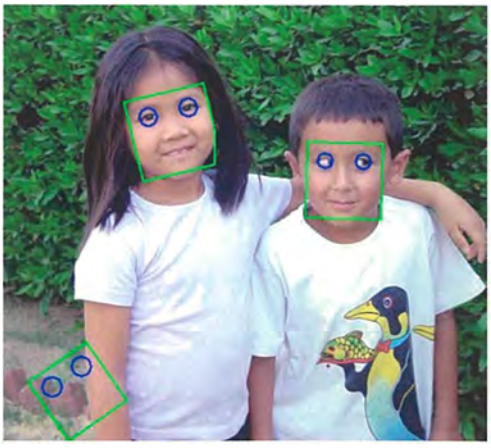


(c) TF = 2, CD = 2, FD = 0, a = 0.75



(d) TF = 2, CD = 2, FD = 0, a = 0.77

(e) TF = 4, CD = 4, FD = 0, a = 0.75



(f) TF = 2, CD = 2, FD = 1, a = 0.79

Figure 7.4: Visual results of face detection – Part II: TF = number of true faces, CD = correct detections, FD = false detections, a = average alignment score.

Table 7.1: Face detection results.

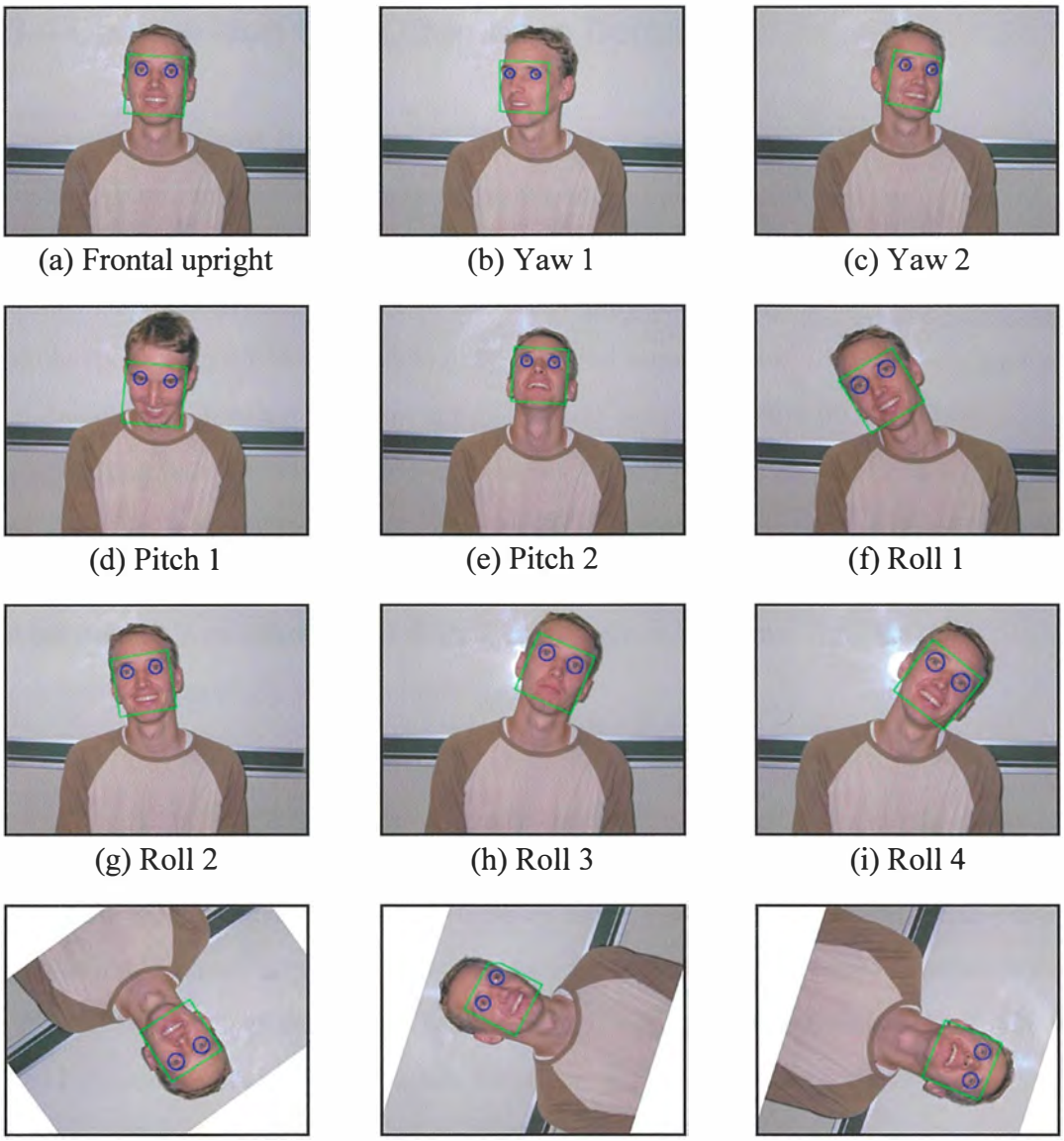
Test Images	Number of images	200 Images 2501-2700	
	Number of faces	231	
Test Results	Number of faces detected	218	
	Number of correct detections	208	CDR = 90.04%
	Number of false detections	10	FDR = 4.3%
	Average alignment score	0.71	

7.3.3 Detection of Rotated Faces

We present here an analysis of the face detector's performance for rotated faces. In this experiment, we collected images of rotated faces of 3 people in our lab. The face of each person was captured at 9 different orientations.

- frontal upright
- two yaw rotations: -18° and 18°
- two pitch rotations: -36° and 36°
- four roll rotations: -60°, -30°, 30°, and 60°

The yaw and pitch rotations are within the requirements of quasi-frontal faces that we showed in Table 5.2. Because the aim of this experiment is to evaluate detection performance for rotated faces, the images were taken against relatively simple background. Each of the five frontal, yaw and pitch face images are rotated in-plane by 9 angles $k\times36^\circ$, $k = \{-5,-4,...,-1, 1, 2...,4\}$ using software. Together, there were 49 images per person, and a total of 147 images in the test set. The results of face detection are shown in Table 7.2. A high detection rate of 94.35% on this set shows that the proposed algorithm is robust in detecting in-plane rotated faces. In fact, the way in which face candidates are constructed in our algorithm automatically compensates the effect of in-plane rotation. The 9 cases of false detections were actually caused by the failure in locating the eyes. Results of detecting rotated faces for one person are shown in Fig. 7.5.



(a) Frontal upright (b) Yaw 1 (c) Yaw 2

(d) Pitch 1 (e) Pitch 2 (f) Roll 1

(g) Roll 2 (h) Roll 3 (i) Roll 4

(j) Software roll 1 (k) Software roll 2 (l) Software roll 3

Figure 7.5: Detecting quasi-frontal faces with large in-plane rotation

Table 7.2: Results of detecting rotated faces in images of simple background

Test Data	Number of images	147
	Number of faces	147
	Yaw rotations	-18°, 18°
	Pitch rotations	-36°, 36°
	Roll rotations	from -180° to 180°
Test Results	Number of faces detected	147
	Number of correct detections	138 CDR = 94.35%
	Number of false detections	9 FDR = 5.65%

7.3.4 Comparison with Other Face Detectors

Several reasons make it difficult to obtain a fair comparison between different face detectors. First, the different face detectors must be evaluated on the same test set. Common test sets do exist online, for example the CMU, MIT, and BioID databases. Unfortunately, these databases consist of only gray-scale images, and therefore are not suitable for testing color-based face detectors like ours. Second, the implementations of face detection algorithms are not usually made available publicly. We may add that implementing published algorithms is often a time-consuming and difficult task.

In this section, the proposed face detector is compared with a number of face detectors that are available as either online demo or stand-alone program. The face detectors used in our comparison study are listed below:

- **BuFaLo face detector**

  The acronym BuFaLo stands for *b*ase-*u*nit for *fa*ce *lo*calization. This face detector, written by Frank Fritze, is available as a stand-alone program and can be downloaded from http://www.geocities.com/fritzfra2001/. The face detector is based on algorithms proposed by Viola and Jones [118-120] and Rainer Lienhart *et al.* [60, 61]. These algorithms, which are based on the AdaBoost algorithm and Harr-like features, are described in Chapter 2.

- **University of Crete (UC) face detector**

  This detector is based on the face detection approach by Garcia and Delakis [32]. In their approach, face/nonface classification is done by a convolutional neural network (see Chapter 2). The face detector is an interactive online demo that accepts input images submitted over the Web.

- **Carnegie Mellon University (CMU) face detector**

  This face detector is based on the work of Rowley *et al.* [90-92] (neural network) and Schneiderman *et al.* [99, 100] (naive Bayesian classifier). This face detector is an online demo that allows people to submit input images over the Web. However, the demo is not interactive.

The same test set of 200 images, described in Subsection 7.3.2, was used. We submitted this test set over the Internet to the two online face detectors at CMU and the University of Crete. The test set was also processed by the stand-alone program BuFaLo. We should point out that these three face detectors use only gray-scale images as inputs.

Table 7.3: Comparison with other face detectors.

Number of images	200			
Number of faces	231			
	Proposed Detector	CMU [90, 91, 99, 100]	BuFaLo [60, 61, 118-120]	University of Crete [32]
Correct Detection	218 faces 90.04%	218 faces 90.04%	180 faces 77.92%	228 faces 98.7%
False Detection	10 4.30%	283 122.5%	5 2.15%	19 faces 8.2%
Average Alignment Score	0.71	*	*	*
Processing Time	120-140s	*	4-5s	9-10s

The test results are shown in Table 7.3. The following observations can be made:

- Compared to the CMU face detector, our face detector has the same correct detection rate. However, the CMU makes 28.3 times more false detections. In fact, the CMU face detector has the highest number of false detections (283), among the four face detectors. We notice that the CMU face detector makes many false detections in textured image regions consisting of tree leaves, grass, or painted shirts.

- Compared to the BuFaLo face detector, our face detector has a much higher correct detection rate. The BuFaLo face detector can locate only 78% of the faces, whereas our face detector can detect 90.04% of the faces. However, the BuFaLo face detector makes only 5 false detections, compared to 10 false

detections produced by our face detector. In fact, among the four face detectors, the BuFaLo face detector makes the least number of false detections. We use the following simple approach to compare our face detector and the BuFaLo face detector at a similar false detection rate. The threshold on the face score of our face detector is raised so that only 5 of its 10 false detections remain (i.e. a false detection rate of 2.15%, which is same as that of the BuFaLo face detector). At the new threshold setting of 0.53, only 199 of the original 218 correct detections produced by our face detector remain (i.e. a correct detection rate of 86.1%).

- Compared to the UC face detector, our face detector has a lower correct detection rate. The UC face detector can locate 98.7% of the faces. However, the UC face detector makes almost twice as many false detections as our face detector does. We were not able to adjust our face detector to produce similar false detection rate as the UC face detector. The current implementation of the UC face detector has a limitation in that it cannot detect faces rotated in-plane by more than 20°. The CMU and BuFaLo face detectors also cannot detect face rotated in-plane by very large angles. In comparison, our face detector is designed to handle arbitrary in-plane rotation, i.e., any angle from -180° to 180°.

In terms of processing time, our face detector takes longer to process a file compared to the BuFaLo and UC face detectors. Approximate measures indicated that, to process a file in the test set (including file I/O), the BuFaLo and UC detectors take on average less than 10s whereas our face detector takes more than 120s. Processing time of the CMU face detector is not made available, but a version of the detector has been reported to take about 383s, running on a 200MHz R4400 SGI Indigo 2, to process a 320×240 pixel image [91]. Strictly speaking, a fair comparison of the processing times requires face detection algorithms to be implemented in the same programming language and run on the same hardware platform. Our face detector is implemented in MATLAB; the core of the BuFaLo face detector is implemented in C++; implementation information of the CMU and UC detectors is not known to us. We believe that the staged MATLAB implementation of our face detector can be optimized to run faster.

7.4 Applications

In this section, we describe two applications of the proposed face detector. The first application is face localization and normalization for face recognition purposes. The second application is facial region segmentation for foreground/background video coding. We also discuss necessary adjustments to the proposed face detector to take advantage of *a priori* knowledge about the input images in these application domains.

7.4.1 Face Normalization for Face Recognition

A face recognition system for biometric personal identification typically consists of two stages: face localization and face identification [78]. In the first stage, the face is extracted from an input image, and then normalized for factors such as face size, face pose, and the lighting condition. In the second stage, the normalized face is compared with stored face images in a database. In face recognition, the face image of a person is usually taken in the frontal upright pose, and against a relatively simple background. It is fair to assume that the face lies approximately in the center of the image. However, its precise location and size usually varies. In addition, the face may not be exactly frontal upright because the person may tilt his or her head slightly. Therefore, even in this simple setting, it is necessary to extract and normalize the face for recognition purposes. Although some face recognition algorithms can handle imprecisely located faces [69], it is quite clear that accurate face localization is essential for good face recognition.

The proposed face detector is particularly suited for a fully automatic face recognition system because of the following reasons. First, it detects not only the face but also the in-plane rotation angle of the face (based on the eye angle); therefore, detected faces can be rotated to the exact frontal upright position. Second, the face detector also outputs precise locations of the two eye points, which can be used to normalize the face for subsequent matching. Third, the geometric face model used in the face detector allows the central portion of the face to be extracted. Chen *et al.* [19] pointed out that statistics-based face recognition should use only the central face portion, which includes internal facial features such as the eyes, nose and mouth. They showed that if extra portions

such as the hair, neck, shoulder and the background are also used in recognition, these nonface portions can actually dominate the face recognition process.

For the face recognition application, we modify the proposed face detector based on the following assumptions, which are reasonable in most face recognition settings:

(i) the face to be detected is in an almost frontal upright position.

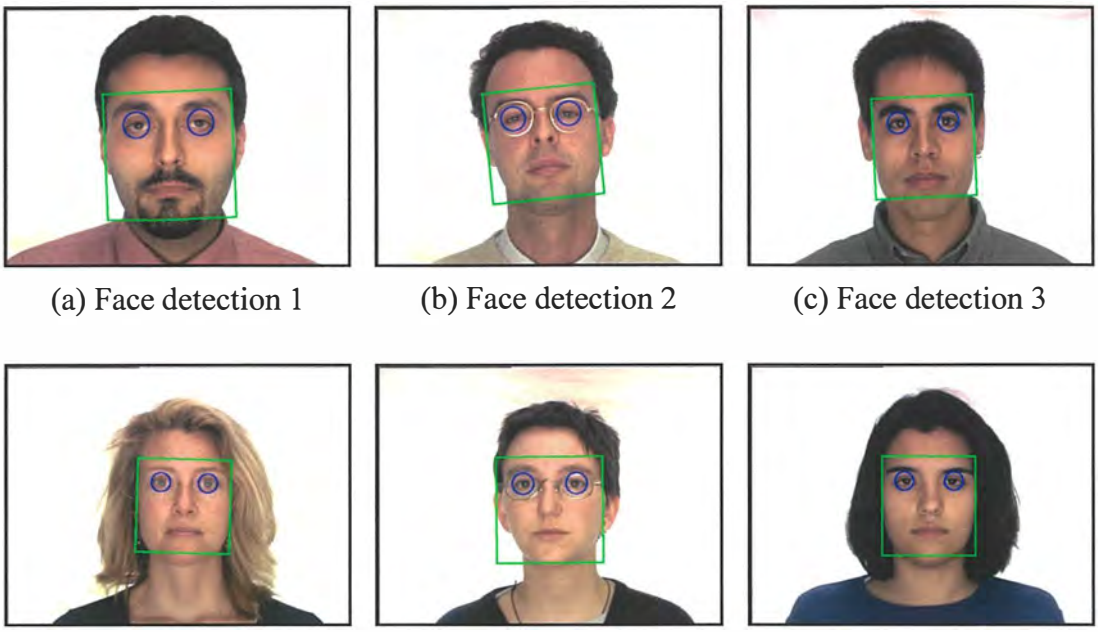
(ii) there is only one face in the image.

The first assumption allows us to ignore eye pairs that have large angles. The second assumption means that only the highest-scored face should be kept. With these simplifications, we are able to implement a version of the face detector that takes less than 20s to process an image of size 568×576 (running on a 600 MHz PC).

This modified face detector was applied to locate faces and correct small in-plane rotations for face recognition purpose. In our experiment, the AR face recognition database [70] was used. This database has been created by Aleix Martinez and Robert Benavente at Purdue University, and at the time of this writing, it consists of over 4,000 facial images of 134 people (75 males and 59 females). From this database, a test set of 134 images containing faces of different people was used. The detection results are shown in Table 7.4. The face detector can correctly locate and normalize 124 faces out of 134 faces (i.e. a correct detection rate of 92.54%). Some of the correct detections and the corresponding normalized faces are shown in Fig. 7.6 and Fig. 7.7, respectively. The detector make no false detections. However, it fails to detect faces in 10 images. In these 10 images, the faces are either darkly lit or taken with eye glasses that reflect lights strongly (see Fig. 7.8). Under these conditions, the eyes can not be located correctly by our algorithm.

Table 7.4: Face localization/normalization on the AR face database.

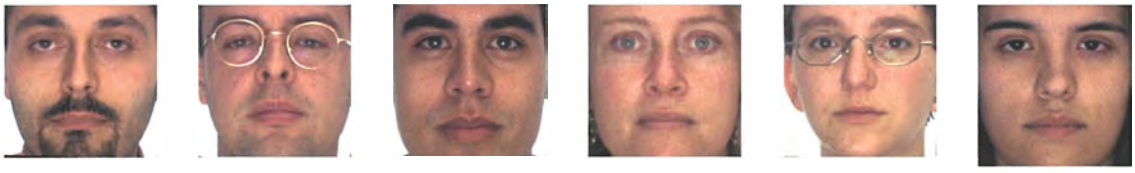
Test Data	Number of images	134
	Image size	568×576
	Number of faces	134
Test Results	Number of faces detected	124
	Number of correct detections	124 CDR = 92.54%
	Number of false detections	0 FDR = 0%



(a) Face detection 1 (b) Face detection 2 (c) Face detection 3

(d) Face detection 4 (e) Face detection 5 (f) Face detection 6

Figure 7.6: Visual results of face detection in AR face database.



(a) (b) (c) (d) (e) (f)

Figure 7.7: Normalized faces after detection for images of Fig. 7.6

(a) Darkly lit face (c) Reflective eye glasses

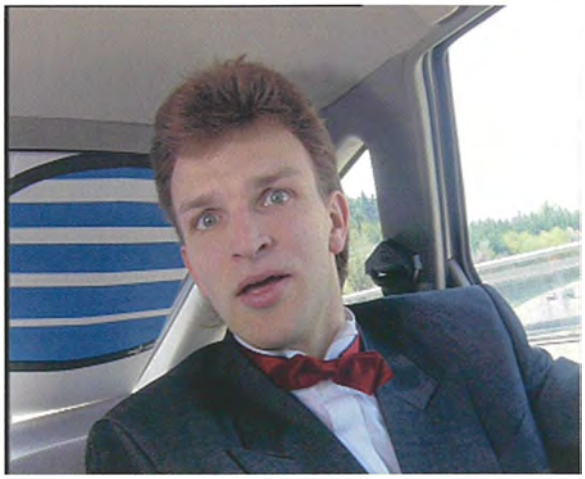
Figure 7.8: Cases of face detection failure in the AR test set. The face is either darkly lit or taken with eye glasses that reflect lights very strongly.

7.4.2 Face Segmentation for Videophone Image Coding

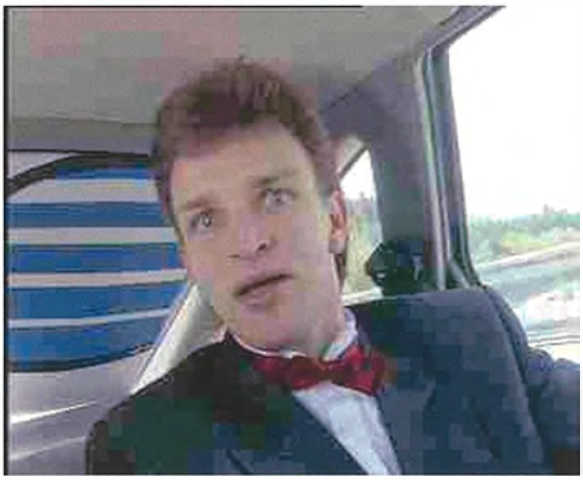
Face segmentation has been applied together with a foreground/background coding scheme to improve the subjective quality of videophone sequences [14, 15]. The low bitrate used in videophone coding results in what commonly known as the *blocking artifacts* or discontinuities at the block boundary, which are visually displeasing to the viewers. The blacking artifacts are present in the entire video frame if all regions are coded with a constant bitrate. However, if the region of interest (ROI) to the viewer is coded with a higher bitrate at the cost of lower bitrate available for other regions, then overall subjective quality of the video is improved. It is found that in a videophone application, it is the face of the speaker that is of more importance to the viewers. Hence, face segmentation can be used to extract the facial regions, which then are coded with higher fidelity compared to the background [14]. We should stress that in this psychovisual coding scheme, the overall bitrate stays the same, but the subjective video quality is enhanced (see Fig. 7.9).

We applied to proposed face detector to locate and extract the facial region in a video sequence. In the experiment, the *carphone* video sequence was use. The video sequence consists of 382 frames, each of size 352×288. The face detector was applied to individual video frames, i.e. without taking into account temporal cues. This experiment is useful to see how the proposed face detector performs for different facial expressions because the same person in the 382 frames of the video shows a wide range of facial actions (eye open/close, mouth movement due to speaking, eyebrow raise, smile, etc.). The face detection results are shown in Table 7.5. The face detector can locate correctly 91.36% of the faces. Some visual results of face detection are shown in Fig. 7.10. The face detector takes, on average, 45.1 seconds to process one frame (including file I/O). This processing time is currently much longer than what required for a practical application of face segmentation. Nevertheless, we outline below a number of ways that our face detector can be improved:

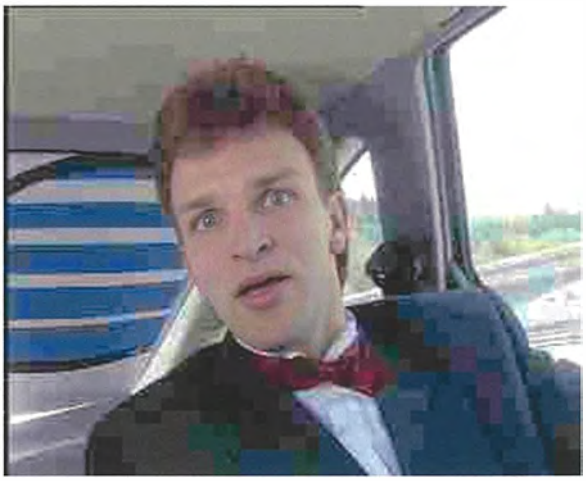
- **Using temporal information.** A motion filter can be applied to limit the search space for face because the facial region is usually under constant motion (the person speaks or move his or her head). In addition, there are strong correlations between the face locations/orientations in one frame and the next.



(a) original *carphone* image coded at 24 bits/pixel



(b) DCT coded image at a bitrate of 0.41 bits/pixel



(c) DCT coded image at an overall bitrate of 0.41 bits/pixel. The background and the segmented facial region are coded at 0.35 bits/pixel and 0.81 bits/pixel, respectively.

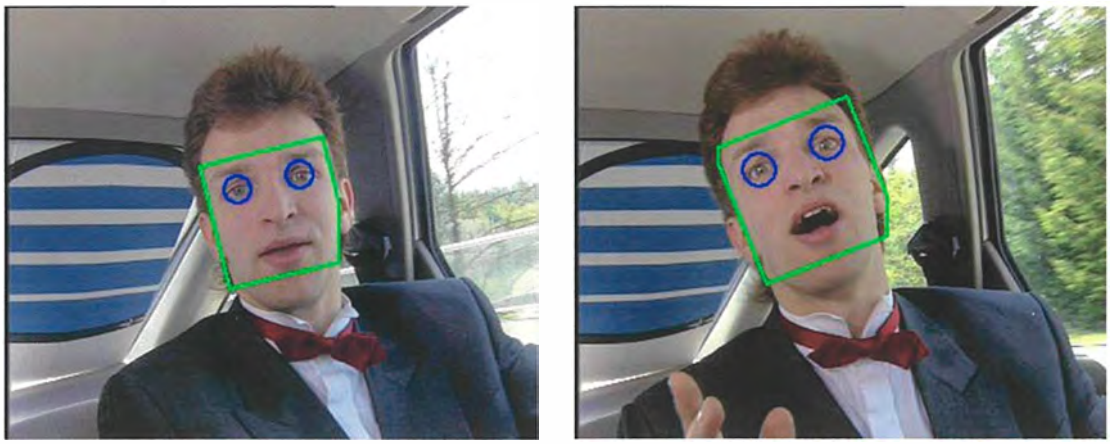
Figure 7.9: Improving perceptual image quality through region-of-interest coding. Notice how the perceptual image quality in (c) is improved compared to (b).

These facts can be used to speed up face tracking and eliminate quickly false detections. For example, we enforced the following temporal constraint on the face orientation: the difference between the face angles in two consecutive frames must not exceed 20°. This constraint removes many false detections that overlap with the true face; these false detections win over the true faces in the overlapping elimination step. Using this temporal constraint, the face detector can correctly locate faces in 368 frames of the total 382 frames (CDR = 96.3%).

- **Using pixel-wise skin segmentation for face tracking.** The Bayesian approach to skin color detection presented in Chapter 3 is very accurate and sufficiently fast for real-time tracking of skin regions. We can perform all stages of face detection for video frames at regular intervals. When a face is detected, the system can go into a tracking mode where it is sufficient to perform only skin segmentation and minimal face verification.

Table 7.5: Face detection in video

Test Data	Video sequence	*carphone*
	Number of video frames	382
	Frame size	352×288 CIF
	Number of faces in all frames	382
	Face characteristics	Different facial expressions
Test Results	Number of faces correctly detected	349 CDR = 91.36%
	Number of faces incorrectly detected	33 FDR = 8.64%
	Processing time per frame (without using temporal information)	45.1 ± 11.5 seconds



(a) *Carphone* frame 1 (b) *Carphone* frame 288

Figure 7.10: Face segmentation for videophone image coding.

7.5 Chapter Summary

In this chapter, we present a complete face detection system that integrates the various components described in the previous chapters of this thesis. The face detector was analyzed using the ECU data detection database. On a test set of 200 images, the face detector achieved a correct detection rate of 90.04% and a false detection rate of 4.30%. The face detector was compared with three existing face detectors using the same test set. Comparison results have indicated that the proposed face detector outperforms the BuFaLo and the CMU face detector. Compared to the UC face detector, the proposed face detector has a lower detection rate. However, the online implementation of the UC cannot detect faces rotated by more than 20°, whereas our face detector can handle faces rotated in-plane by an arbitrary angle. In fact, detecting in-plane rotated faces is one of the major strength of the proposed face detector. In this chapter, we also discussed two possible applications of the proposed face detectors, namely face localization and normalization for recognition purpose, and face segmentation for perceptual video coding.

Chapter 8

Conclusions and Further Work

8.1 Thesis Summary

This thesis addresses the problem of automatic human face detection in color images. The principal goal of the thesis is to develop a robust algorithm for detecting quasi-frontal faces with arbitrary in-plane rotation, which combines the strengths of both analytic and holistic approaches to face detection. To this end, several research objectives are defined and systematically addressed in the preceding chapters of the thesis. This section summarizes important findings from our study, and highlights the major contributions of the work reported herein.

The thesis is organized into 8 chapters. In Chapter 1, we describe the problem of automatic face detection, and its significance in the broader field of automated facial image understanding. We show that automatic face detection is indeed a difficult problem in computer vision. In face detection, visual attributes common to all faces that differentiate the human face from all other objects must be identified and implemented in an efficient way. In doing so, one realizes that there is a tremendous degree of intrinsic and extrinsic variations in the face pattern. In Chapter 1, we also describe important applications of face detection in several domains including face recognition for biometric personal identification, face segmentation for region-of-interest image and

video coding, perceptual human computer interface (HCI), and multimedia content management.

In Chapter 2, we present a comprehensive and systematic review of the state-of-the-art in face detection. Our literature review reveals that research interests in face detection have been strong, especially in the last few years. This can be explained in a number of ways. First, despite several years of research and impressive progress in the field, there remain many unexplored paths and unsolved problems in face detection. For example, we find that intrinsic face variations can be handled quite adequately by many existing face detection algorithms, whereas coping with many extrinsic variations (face pose, scale, and lighting) is still open for improvement. In this thesis, we attempt to tackle two important sources of extrinsic variations: in-plane rotation and scale. Second, novel applications of face detection and tracking are being identified, for example perceptual HCI, video surveillance, and video coding and indexing. These applications have led to a new perspective on face detection, and necessitated many domain-specific performance requirements of face detection algorithms. Two major categories of face detection approaches, namely analytic and holistic, are described in Chapter 2. In our literature review, we focus on several recent developments in the field that were not described in earlier survey papers [42, 132].

Chapter 3 presents a comprehensive study into skin detection using color pixel classification. Because a significant part of the facial region is skin, we use skin detection as the first stepping stone towards face detection. This skin detection approach works at pixel level, in which image pixels are classified as either skin or nonskin on the basis of their color. This is possible because the human skin has very consistent colors that are distinct from the color of many other objects. It is fair to say that many authors have used skin color for face detection. However, each author uses a different color space or classification algorithm, and there has been no established confirmation of which method has better performance in terms of accuracy and robustness. In our study, we attempt to evaluate the skin detection performance of different color spaces and color pixel classification algorithms. For this purpose, we use a set of 3,000 color images and manually segmented these images for skin and nonskin regions. This image set represents a wide variation in the lighting conditions and the skin color types

(blackish, whitish, yellowish, brownish). Overall, a training set of over 650 million color pixels and a test set of over 120 million color pixels have been collected and used in the study.

In this chapter, we compare four main families of color spaces, namely RGB, YCbCr (representing class Y color spaces), HSV (representing hue-saturation-value color spaces), and CIE-Lab (representing perceptual uniform color spaces), and six different classification algorithms, namely fixed range skin color map [15], piece-wise linear decision boundary [33], self-organizing map [8], Gaussian densities [72], multilayer perceptron [84], and Bayesian classifier [83]. Based on comparison results, we can conclude that if all three color channels are used for classification, there is no clear difference in performance to favor the use of one color space over another. In addition, the segmentation approach of using all color channels is consistently better than the approach of using only two chrominance channels (normalized rg, HS, CbCr, or ab). Regarding classification algorithms, we find that the Bayesian classifier is superior to the other five classifiers in terms of both segmentation accuracy and speed. Segmentation accuracy of the Bayesian classifier can be attributed to accurate estimation of class-conditional densities using a large labeled set of skin and nonskin pixels. In Chapter 3, experiments are also performed to determine the optimum trade-off between the histogram size (i.e. memory storage) and segmentation accuracy of the Bayesian classifier. We find that histogram sizes of more than 64 bins per channel are necessary for good skin detection performance. Using the histogram size of 64 bins, the CDRs are 84.4% and 94.5% for FDRs of 10% and 20%, respectively. We believe that our study on color spaces and color pixel classification algorithms is important not only for skin detection but also for the general task of color-based object detection. For example, the Bayesian color pixel classifier can be used to segment objects with consistent colors, such as fruits, coal, and timber in production lines, regardless of the object orientation and shape.

Chapter 4 focuses on skin detection at region level, in which detected skin pixels are grouped into homogenous skin regions. In Chapter 4, we propose a complete skin region segmentation algorithm that consists of three stages: *color-based skin detection*, *skin region verification*, and *skin region refinement*. Texture property of the human skin

is taken into consideration in order to remove false detections. We find that skin has a special texture that results from the grouping of many pixels that have similar colors. As a result, the skin texture can be considered a type of micro-texture, for which texture tone is a dominant factor. Hence, simple texture model based on local homogeneity is suitable for describing the human skin texture. Based on this model, skin-colored pixels detected by the Bayesian classifier are removed if their local neighborhood is not homogenous. The proposed approach of skin texture verification using local homogeneity measure is valid for a wide range of skin textures. Skin regions are also verified using edge information. Edge pixels are removed from the skin binary mask in order to separate true skin region from background regions having skin colors and skin-like texture. Experimental results indicate that these verification techniques can remove false detections quite significantly. The skin region segmentation algorithm presented in this chapter allows us to treat each segmented skin regions independently from other skin regions in the next stage of face detection.

Chapter 5 presents a novel approach to face candidate selection that gives our face detection algorithm scale-invariant and in-plane rotation-invariant capabilities. The proposed approach eliminates the need for scanning the input image window-by-window, as is commonly done in holistic approaches to face detection. In addition, it addresses a major deficiency in many existing skin color-based approaches to face detection: they treat each segmented skin region in their entirety as one face candidate. In Chapter 5, we propose a new technique for detecting eyes in the segmented skin region. Our eye detection technique is based on eye color. Unlike the technique developed by Hsu *et al.* [43], in which the eye is detected using rather heuristic rules about the chrominance and luminance of eye color, our technique models the eye color using the Bayesian color pixel classifier described in Chapter 3. We are able to handle automatically different eye color types, including blackish, bluish, brownish, etc. The eye color pixel classifier takes advantage of the fact that we need to differentiate only the eye and the skin. The proposed color-based technique to eye detection is very attractive compared to the filter-based [134] or distribution-based [110] techniques because the eye can be detected regardless of its orientation.

For each detected eye pair, two face candidates are constructed using a geometric face model. This geometric model is found to be valid for the class of quasi-frontal faces that our face detector is design to handle. The face class includes the frontal upright face and its variations caused by all in-plane rotations and certain out-of-plane rotation (see Table 5.2). Because the face orientation is known explicitly from the eye-to-eye line, each face candidate can be rotated (normalized) to the upright position. That way, in later verification of the face candidates, we only need to check if the normalized candidate is indeed similar to a frontal upright face. In Chapter 5, we also propose several new techniques for preliminary verification of the face candidates. Applying the proposed face candidate selection approach, we found that on average, we need to examine only about 90 face candidates per true face. This number is much smaller than the number of windows to be examined in a window scanning approach. Actually in the latter approach, the number of windows is basically proportional to the number of pixels in the image, which is in the order of hundreds of thousands for a typical image.

Chapter 6 addresses pattern classification techniques to perform final verification of the remaining face candidates. We focus on face/nonface classifiers that are based on the naive Bayesian model. The naive Bayesian model, which assumes statistical independence between variables in a feature vector, is a well known model in statistical pattern recognition [21]. Our approach is motivated by the simplicity and the accuracy of the naive Bayesian model. We need to acknowledge that this model has been used in the past by Schneiderman and his colleague [99, 100]. However, there are a number of differences between our approach and theirs. Schneiderman *et al.* modeled both the appearance (i.e. intensity) and the position of the window subregions, whereas in our approach the position information is implicitly captured in the lexicographic ordering of window pixels. Furthermore, we investigate different feature extraction techniques (intensity, PFS, and edge-based) and their suitability for face/nonface classification. The intensity feature vector performs better than the PFS feature vector. The edge-based feature vector, although not as good as the intensity and PFS feature vectors, has quite good classification rates. At false detection rate of 10%, the correct detection rates of the intensity, PFS, and edge-based feature vectors are 96.0%, 92.9% and 89.9%, respectively.

In Chapter 6, we describe different image preprocessing techniques, and analyze the effects they have on the separability between face and nonface classes. To the best of our knowledge, there is, to date, no reported comparative study of different preprocessing techniques. Experimental results in Chapter 6 indicate that image preprocessing indeed improves face/nonface classification significantly. For example, at the same false detection rate, the correct detection rate with histogram equalization increases by more than 20% compared to no preprocessing. Among the five preprocessing techniques, namely mean, range, standard deviation normalization, histogram equalization and gradient illumination correction, histogram equalization is found to have the best performance. In Chapter 6, three strategies to improve classification performance are presented: bootstrapping, classifier combination, and using contextual information. We show that face/nonface classification is improved by fusing the classification results of naive Bayesian classifiers, which use different feature vectors. On the same test set of face and nonface patterns, the ensemble of three classifiers using intensity, PFS and edge-based feature vectors has a CDR of 98.6% at a FDR of 10%, compared to a CDR of 96% of the best individual classifier. As for using contextual information, we show that many false face detections can be removed using the overlapping elimination technique, which keeps a face candidate only if it has a higher face score than all the face candidates that overlap with it.

In Chapter 7, the complete face detector integrating the components described in Chapters 3 to 6 is presented. This face detector is analyzed using a test set of 200 color images taken from the ECU face detection database. On this test set, the proposed face detector has a correct detection rate of 90.4% and a false detection rate of 4.3%. We also perform comparison of the proposed face detector and three existing face detectors on the same test set. Results show that the proposed face detector has better detection rates compared to the CMU and the BuFaLo face detectors. The proposed face detector has a lower CDR (90.04%) compared to the UC face detector (98.7%). However, our face detector makes only 10 false detections compared to 19 false detections by the UC face detector. In addition, experiments have shown that the proposed face detector can handle large in-plane rotations, where the UC face detector cannot locate faces rotated by large angles. Compared to the three face detectors, the proposed face detector has the following important advantages: (i) it does not require the exhaustive scanning of the

(without using sparse-matrix coding, a high-performance Bayesian classifier needs at least 2MB for this table). We suggest that memory requirement can be reduced further while maintaining the accuracy and speed advantages by training a small neural network to approximate the color pdfs of the Bayesian classifier.

3. Integration with other eye detection techniques

The eye detection technique developed in Chapter 5 is very effective in finding eye regions that are enclosed within a segmented skin region. We find that the proposed eye score based on the eye and skin color pdfs can discriminate well between the eye and the skin. However, the eye score is less effective in detecting eye regions that fall outside a segmented skin region. This problem can be addressed in two ways. First, a complementary eye score based on the eye and the background color pdfs can be used. Here, the term "background color" represents the color of objects other than the eye and the skin. Second, we can incorporate other eye detection techniques based on multi-scale and multi-orientation filters to detect eye regions that are outside but close to a segmented skin region.

4. Combining multiple pattern classification techniques for face candidate verification

In Chapter 6, we show how multiple naive Bayesian classifiers are combined to yield a classifier ensemble with higher classification rates, compared to even the best individual face/nonface classifier. In addition, we show how the face scores produced by different classifiers are combined into a robust face score that can be used to rank and eliminate overlapping detections. The successes of these two techniques prove the feasibility of a face/nonface classification system combining multiple classifiers that may have different architectures or use different features. Although we focus on the naive Bayesian classifier in this thesis, other classifiers, such as neural networks and support vector machines, do exist and can be used for more robust face candidate verification.

5. Face detection and tracking in video

Real-time face detection and tracking in video play a key role in applications such as vision-based human-computer interactivity and video surveillance. While many components of the proposed face detector such as skin segmentation, in-plane rotation invariant and scale-invariant face candidate selection are attractive for a real-time

application, there exist aspects of the face detector that can be improved or optimized by incorporating temporal information.

8.3 Closing Remarks

Automatic human face detection is an important research topic in computer vision. It has an established application in automatic face recognition systems, and many emerging applications in perceptual human computer interface, region-of-interest video coding, and multimedia content management. This thesis presents a complete automatic face detection algorithm that integrates both analytic and holistic approaches to face detection. The algorithm combines a color-based skin detection technique, a face candidate selection scheme using a geometric face model and color-based eye detection, and a face/nonface classification method based on the naïve Bayes model. The proposed approach eliminates the computation-intensive step of window-scanning commonly adopted in holistic face detection approaches. The proposed approach addresses two important problems in face detection, namely coping with arbitrary in-plane rotation and detecting faces of different sizes. It is our hope that the work in this thesis represents continued progress towards an automatic face detection system that is comparable to the human vision system.

APPENDIX

Appendix A: The ECU Face Detection Database

A crucial requirement for research in automated facial image understanding is the availability of a large and comprehensive image database, based on which algorithms can be constructed and reliably tested. For face recognition task, large databases exist, for example, the FERET database with over 14,000 images of 1,199 individuals [78], and the AR database with over 4,000 images of 126 individuals [70]. However, images in face recognition databases are not adequate for face detection because they often have very simple background. There are a number of online databases specific to face detection, but the numbers of images in such databases are usually quite small. For example, the CMU database [91], which is often used for reporting face detection performance, has only 130 images. Recently, BioID made available a large image database for face detection research [50]. The database consists of 1521 original images with face detection ground-truths. However, the images are in gray-scale, and therefore are not suitable for color-based approaches to face detection.

We have created, as part of this Ph.D. research, a large and comprehensive face detection database at Edith Cowan University. Compared to existing face detection datasets, this database is unique in three important aspects. Firstly, the database consists of color images, and therefore supports both color-based and intensity-based face detection algorithms. If face detection needs to be done in the gray-scale domain, the images can be converted to gray-scale. Secondly, the database consists of a large number of images (4,000 images at the time of this writing). These images are widely varied in terms of the image size, the background, the person, the lighting condition, the facial expression, and the face pose. Therefore, the database provides a wide spectrum of face detection scenarios. Thirdly, the database contains ground-truth images that have been manually produced. This appendix describes the ECU database, its datasets, and how it can be used. We also suggest a method for evaluating the performance of face detection algorithms on the database.

A. The Datasets

The ECU face detection database consists for the following datasets:

- **Set 1** consists of *original color images*. The images are stored in the JPEG format. The image files are named sequentially in the form for *im***nnnnn***.jpg*, where **nnnnn** is a five-digit sequence number starting from 1.
- **Set 2** consists of the *skin segmented images* for the images in Set 1. The segmentation is done manually. In Set 2, nonskin regions are replaced with white pixels (R = G = B = 255). Note that we segment not only the facial skin regions but also all other exposed skin regions such as the neck, the hands, and the arms. This set is very valuable for skin color models and skin segmentation algorithms.
- **Set 3** consists of the *face segmented images* for the images in Set 1. In Set 3, nonface regions are replaced with white pixels. We segment the face manually by following closely the face contours and not including any parts such as the hair, the ears, or background regions. This dataset can be used to evaluate face detection algorithms.
- **Set 4** consists of the *eye segmented images* for the images in Set 1. In Set 4, noneye regions are replaced with white pixels. We consider the eye region to include the iris, the pupil, the eyelashes, and the white of eye.
- **Set 5** consists of *face patterns*. These face patterns are obtained by manually cropping the central region of the face in images obtained from various online sources. The face patterns in this set are mainly for face/nonface classification task.
- **Set 6** consists of *nonface patterns*. These patterns are extracted from nonface images in Set 7. The nonface patterns are needed for the task of classifying face and nonface.
- **Set 7** consists of *nonface images*. These images are known to contain no faces. This set consists of mostly landscape and scenery photos.

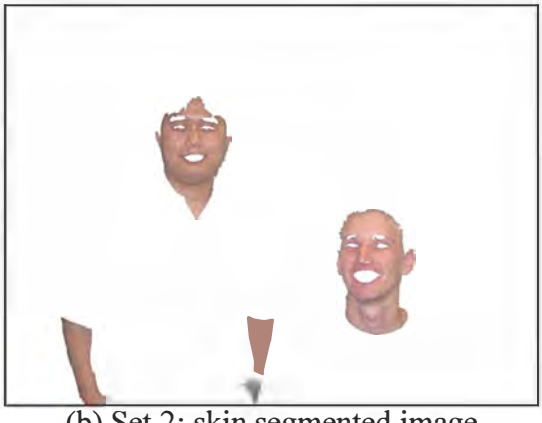
Summaries of the datasets are given in Table A.1. Examples of the images in Set 1, 2, 3, and 4 are shown in Fig. A.1. The ECU database comes with a Windows program that can be used to browse its datasets. Screenshots of the program are shown in Fig. A.2.

Table A.1: ECU face detection database.

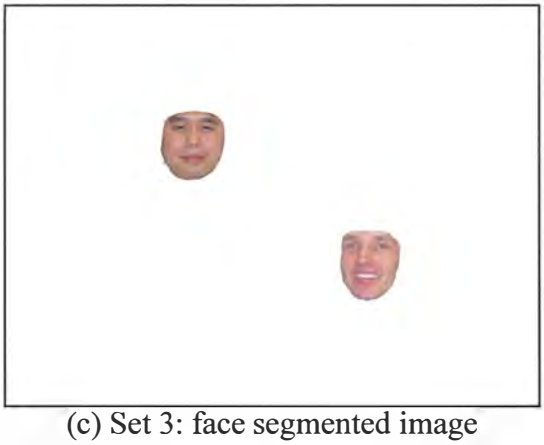
Set	Description	Size	Filename	File Format	Other Info.
1	Original color images	4,000	im**nnnnn**.jpg	JPEG	
2	Skin segmented images	3,300	im**nnnnn**_s.bmp	Bitmap*	
3	Face segmented images	4,000	im**nnnnn**_f.bmp	Bitmap*	
4	Eye segmented images	1,000	im**nnnnn**_e.bmp	Bitmap*	
5	Face patterns	12,000	Face_**nnnnn**.bmp	Bitmap	Image size = 64×64
6	Nonface patterns	260000	Nonface_**nnnnn**.bmp	Bitmap	Image size = 64×64
7	Nonface images	1845	LS**nnnn**.jpg	JPEG	Landscapes

* To save storage, these ground-truths can be converted into binary images.

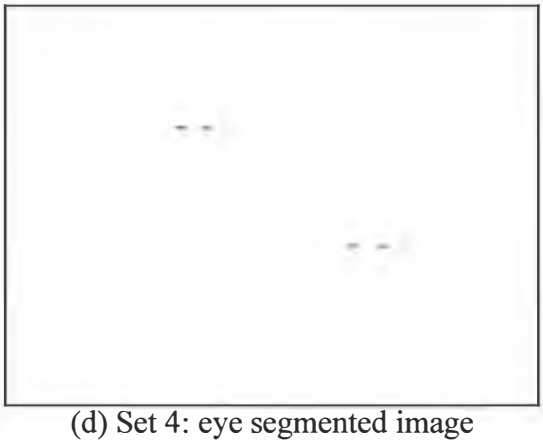
(a) Set 1: original image im03168.jpg



(b) Set 2: skin segmented image im03168_s.bmp

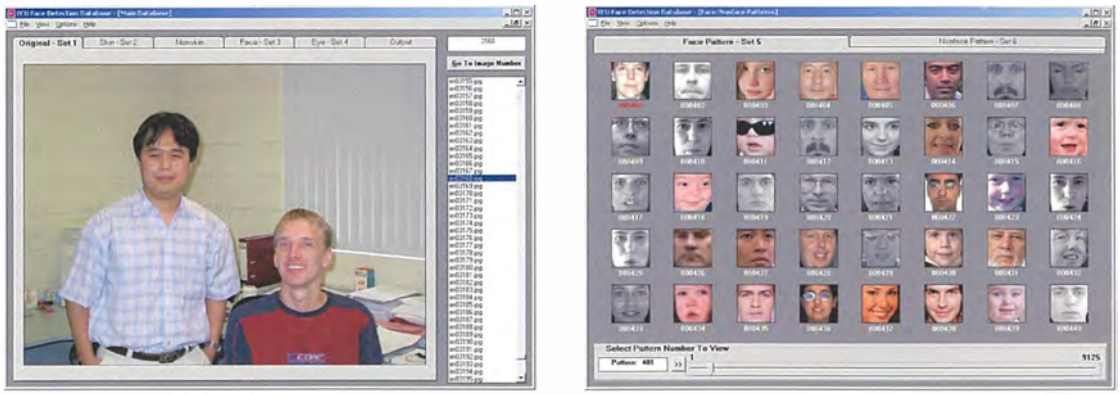


(c) Set 3: face segmented image im03168_f.bmp



(d) Set 4: eye segmented image im03168_e.bmp

Figure A.1: Example images in the ECU face detection database.



(a) viewing sets 1-4 (b) viewing sets 5-6

Figure A.2: A GUI program for viewing the ECU database.

B. Using the Datasets

In this subsection, we show how to extract data from the ECU database for various tasks including skin color modeling, skin segmentation, and face detection. In the following examples, the MATLAB environment and its Image Processing toolbox are used. Users need to adapt these examples to their programming environments. The essential idea is that non-object pixels in the segmented images are replaced by white pixels (R = G = B = 255). Here, the term "object" refers to skin, face, or eye. Once object pixels are identified, they can be grouped into objects through connected-component labeling. Two simple MATLAB code examples are shown in Tables A.2 and A.3.

Table A.2: Using skin segmented images – MATLAB example

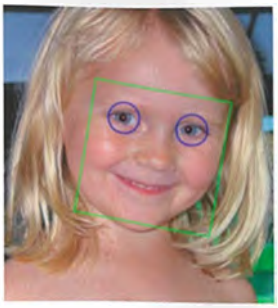
```
% Read skin segmented image
im_skin = imread('im00001_s.bmp');

% Create skin mask - skin pixels are non-white pixels
skin_mask = (im_skin(:,:,1) ~= 255) | ...
            (im_skin(:,:,2) ~= 255) | ...
            (im_skin(:,:,3) ~= 255);

% Extract skin pixels and nonskin pixels
im_org = imread('im00001.jpg'); % Read original image
r_org = im_org(:,:,1);          % Red component
g_org = im_org(:,:,2);          % Green component
b_org = im_org(:,:,3);          % Blue component
r_skin = r_org(skin_mask);      % Red values for skin pixels
g_skin = g_org(skin_mask);      % Green values for skin pixels
b_skin = b_org(skin_mask);      % Blue values for skin pixels
r_nonskin = r_org(~skin_mask);  % Red values for nonskin pixels
g_nonskin = g_org(~skin_mask);  % Green values for nonskin pixels
b_nonskin = b_org(~skin_mask);  % Blue values for nonskin pixels
```

Appendix B: Results of the Proposed Face Detector


im02501.jpg


im02502.jpg


im02503.jpg


im02504.jpg


im02505.jpg


im02506.jpg


im02507.jpg


im02508.jpg

im02509.jpg


im02510.jpg

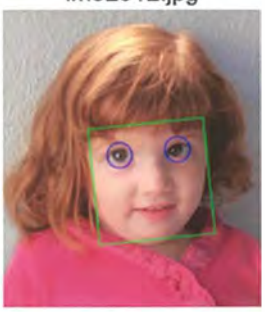

im02511.jpg


im02512.jpg


im02513.jpg


im02514.jpg


im02515.jpg


im02516.jpg


im02517.jpg


im02518.jpg


im02519.jpg


im02520.jpg


im02521.jpg


im02522.jpg


im02523.jpg


im02524.jpg

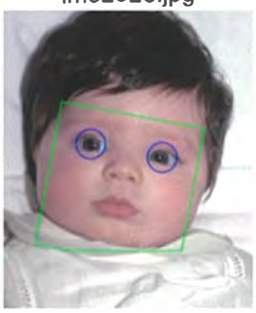

im02525.jpg

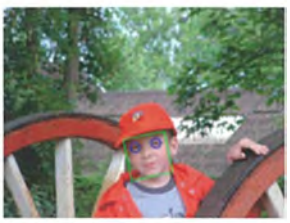

im02526.jpg

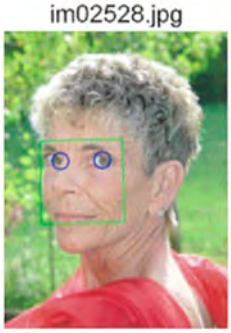

im02527.jpg

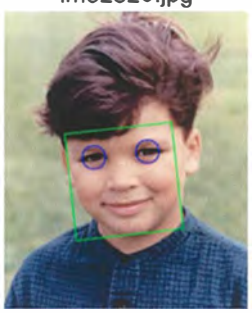

im02528.jpg


im02529.jpg


im02530.jpg


im02531.jpg

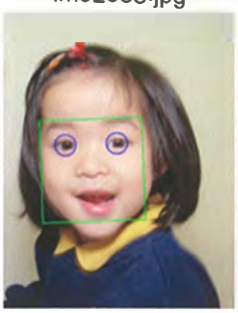

im02532.jpg


im02533.jpg


im02534.jpg


im02535.jpg

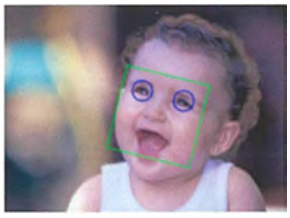

im02536.jpg


im02537.jpg


im02538.jpg


im02539.jpg


im02540.jpg

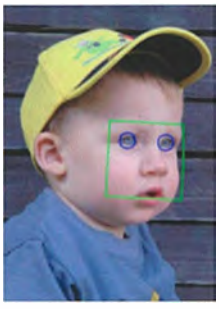

im02541.jpg


im02542.jpg


im02543.jpg


im02544.jpg


im02545.jpg


im02546.jpg


im02547.jpg


im02548.jpg


im02549.jpg


im02550.jpg


im02551.jpg


im02552.jpg


im02553.jpg

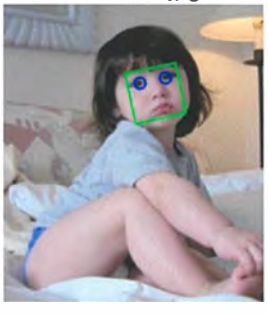

im02554.jpg


im02555.jpg


im02556.jpg


im02557.jpg


im02558.jpg


im02559.jpg


im02560.jpg

im02561.jpg

im02562.jpg

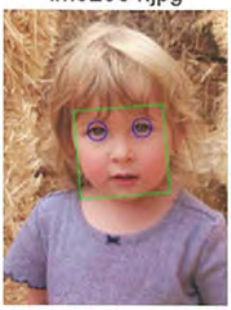
im02563.jpg

im02564.jpg

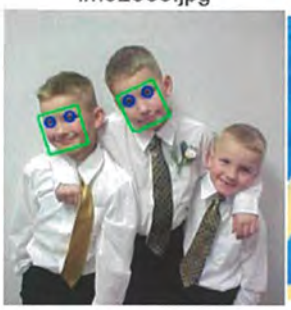
im02565.jpg

im02566.jpg

im02567.jpg



im02568.jpg

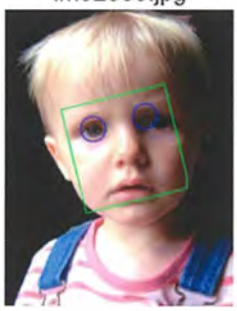


im02569.jpg

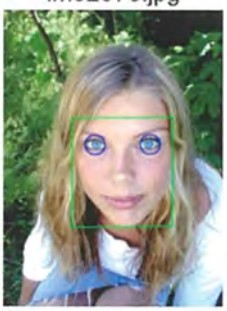
im02570.jpg

im02571.jpg

im02572.jpg



im02573.jpg



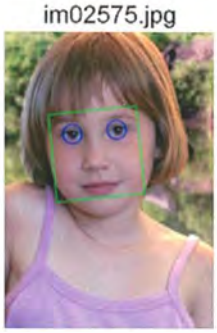
im02574.jpg

im02575.jpg

im02576.jpg

im02577.jpg

im02578.jpg



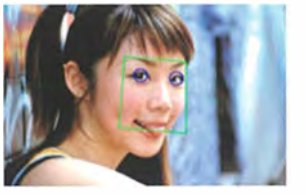
im02579.jpg

im02580.jpg

im02581.jpg

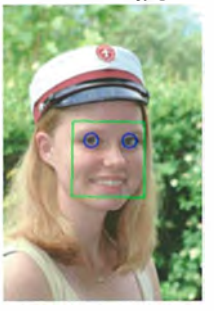
im02582.jpg

im02583.jpg

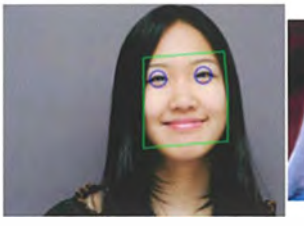


im02584.jpg

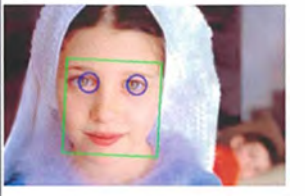
im02585.jpg



im02586.jpg



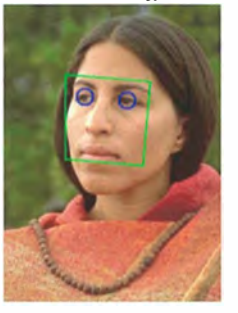
im02587.jpg



im02588.jpg

im02589.jpg

im02590.jpg

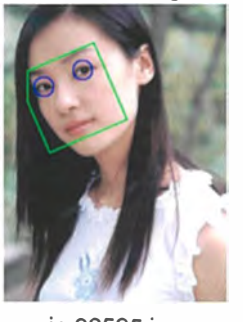


im02591.jpg

im02592.jpg

im02593.jpg

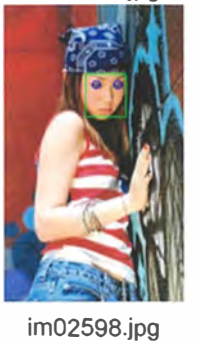
im02594.jpg



im02595.jpg

im02596.jpg

im02597.jpg



im02598.jpg

im02599.jpg

im02600.jpg

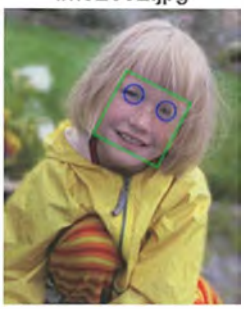

im02601.jpg

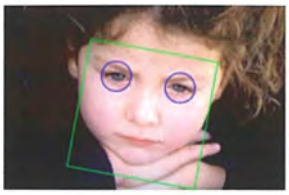

im02602.jpg

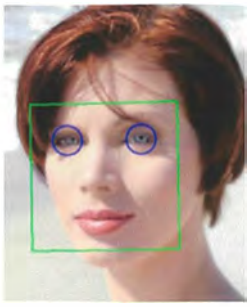

im02603.jpg


im02604.jpg

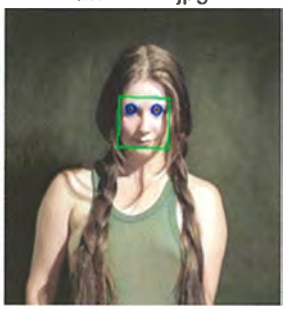

im02605.jpg


im02606.jpg


im02607.jpg

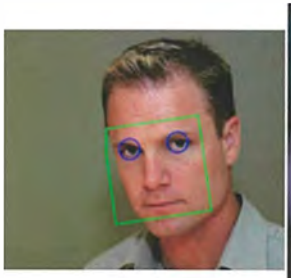

im02608.jpg

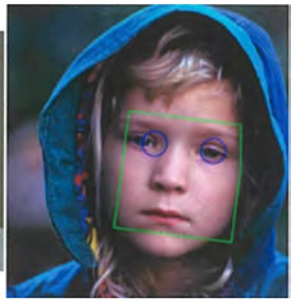

im02609.jpg

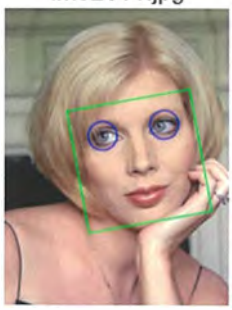

im02610.jpg


im02611.jpg


im02612.jpg


im02613.jpg


im02614.jpg


im02615.jpg


im02616.jpg


im02617.jpg

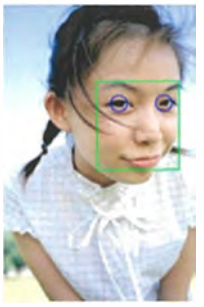

im02618.jpg


im02619.jpg


im02620.jpg


im02621.jpg

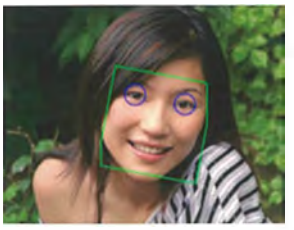

im02622.jpg

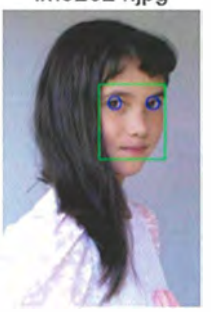

im02623.jpg


im02624.jpg

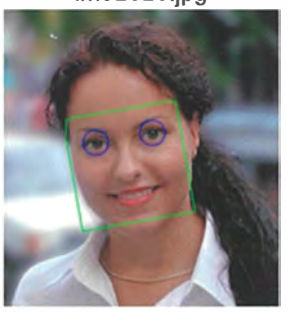

im02625.jpg

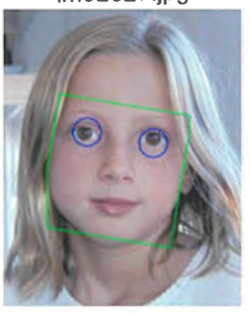

im02626.jpg


im02627.jpg


im02628.jpg


im02629.jpg


im02630.jpg


im02631.jpg


im02632.jpg


im02633.jpg


im02634.jpg

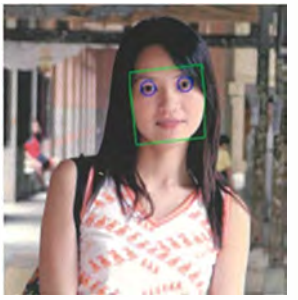

im02635.jpg


im02636.jpg

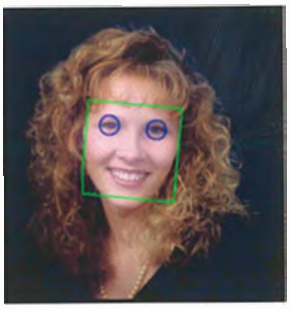

im02637.jpg

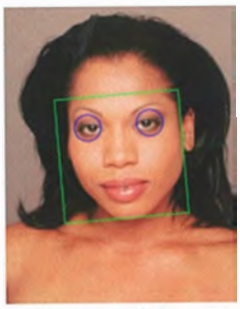

im02638.jpg

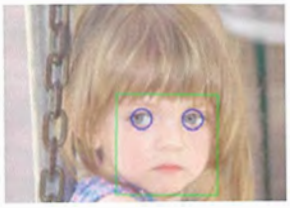

im02639.jpg


im02640.jpg

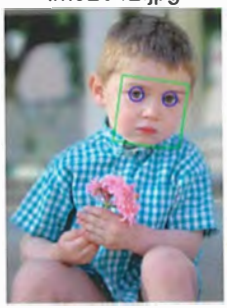

im02641.jpg


im02642.jpg


im02643.jpg


im02644.jpg


im02645.jpg


im02646.jpg

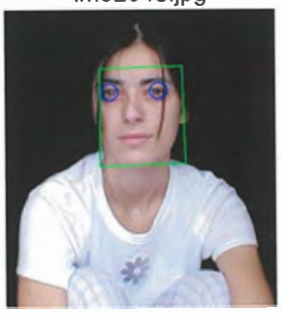

im02647.jpg

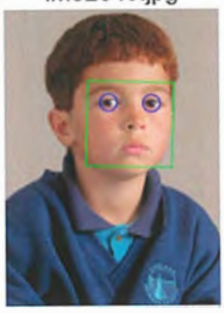

im02648.jpg


im02649.jpg


im02650.jpg

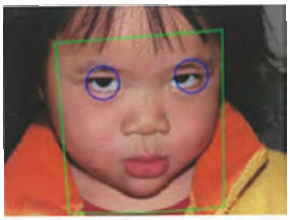

im02651.jpg

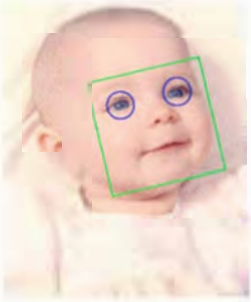

im02652.jpg

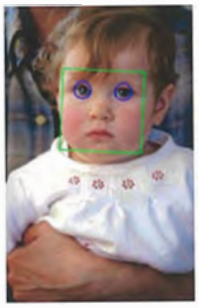

im02653.jpg

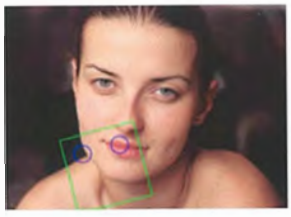
im02654.jpg

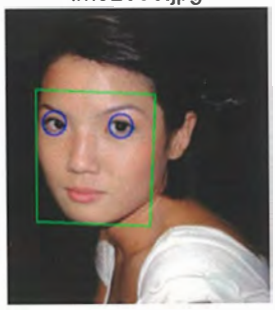

im02655.jpg


im02656.jpg


im02657.jpg


im02658.jpg


im02659.jpg


im02660.jpg


im02661.jpg

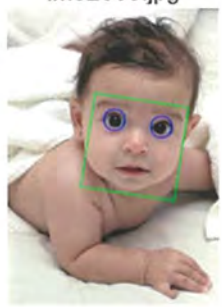

im02662.jpg


im02663.jpg

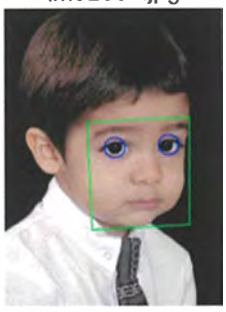

im02664.jpg


im02665.jpg


im02666.jpg

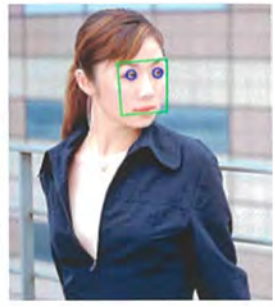

im02667.jpg


im02668.jpg


im02669.jpg


im02670.jpg


im02671.jpg


im02672.jpg


im02673.jpg


im02674.jpg

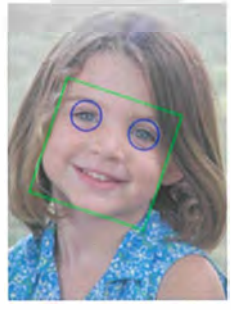

im02675.jpg

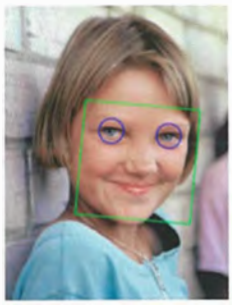

im02676.jpg


im02677.jpg


im02678.jpg

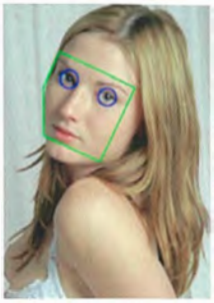

im02679.jpg

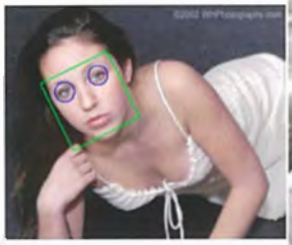

im02680.jpg

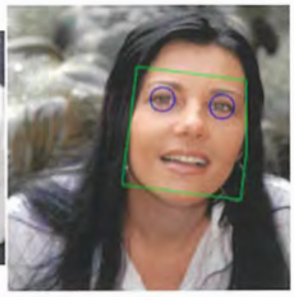

im02681.jpg


im02682.jpg


im02683.jpg


im02684.jpg


im02685.jpg


im02686.jpg


im02687.jpg


im02688.jpg

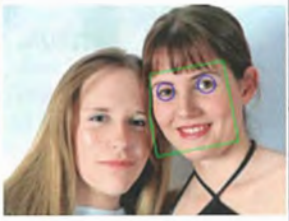

im02689.jpg

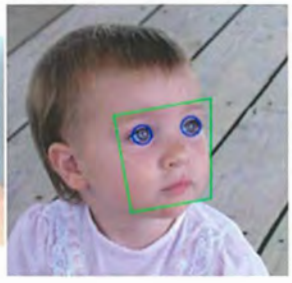

im02690.jpg


im02691.jpg


im02692.jpg


im02693.jpg


im02694.jpg


im02695.jpg


im02696.jpg


im02697.jpg


im02698.jpg


im02699.jpg


im02700.jpg

Appendix C: List of Publications

Below is the list of publications arising from this research study. The relevant objectives stated in Chapter 1 are shown next the publication.

1. S. L. Phung, D. Chai, and A. Bouzerdoum, "A universal and robust human skin color model using neural networks," in *Proceedings of INNS-IEEE International Joint Conference on Neural Networks*, Washington DC, July 2001, vol. 4, pp. 2844 - 2849. [*Objective 4*]
2. S. L. Phung, D. Chai, and A. Bouzerdoum, "Skin color based face detection," in *Proceedings of the Seventh Australian and New Zealand Intelligent Information Systems Conference*, Perth, Australia, Nov. 2001, pp. 171-176. [*Objective 3*]
3. S. L. Phung, A. Bouzerdoum, and D. Chai, "A novel skin color model in YCbCr color space and its application to human face detection," in *Proceedings of IEEE International Conference on Image Processing*, Rochester, New York, Sep. 2002, vol. 1, pp. 289-292. [*Objectives 3, 4*]
4. S. L. Phung, D. Chai, and A. Bouzerdoum, "A distribution-based face/nonface classification technique," *Australian Journal of Intelligent Information Processing Systems*, vol. 7, no. 3/4, pp. 132-137, June 2002. [*Objectives 3, 7*]
5. S. L. Phung, A. Bouzerdoum, and D. Chai, "A distribution-based face/nonface classifier using Gaussian mixtures," in *Proceedings of the Fourth IASTED International Conference on Signal and Image Processing*, Hawaii, Aug. 2002, pp. 485-490. [*Objectives 3, 7*]
6. S. L. Phung, D. Chai, and A. Bouzerdoum, "Adaptive skin segmentation in color images," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Hong Kong, Apr. 2003, pp. 353-356. [*Objective 5*]
7. S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin segmentation using color and edge information," in Proceedings of *IEEE International Symposium on Signal Processing and Applications*, Paris, France, July 2003. [*Objective 5*]

BIBLIOGRAPHY

[1] A. Albiol, L. Torresz, C. A. Bouman, and E. J. Delp, "A simple and efficient face detection algorithm for video database applications," in *Proceedings of IEEE International Conference on Image Processing*, Vancouver, Canada, Sep. 2000, pp. 239-242.

[2] M. S. Bartlett, *Face Image Analysis by Unsupervised Learning*. Massachusetts: Kluwer Academic Publishers, 2001.

[3] N. Bassiou, C. Kotropoulos, T. Kosmidis, and I. Pitas, "Frontal face detection using support vector machines and backprogagation neural networks," in *Proceedings of IEEE International Conference on Image Processing*, Thessaloniki, Greece, Oct. 2001, pp. 1026-1029.

[4] N. Bojic and K. K. Pang, "Adaptive skin segmentation for head and shoulder video sequence," in *Proceedings of SPIE Visual Communications and Image Processing*, Perth, Australia, June 2000, pp. 704-711.

[5] A. Bouzerdoum, "Nonlinear lateral inhibitory neural networks: analysis and application to motion detection," University of Washington, Washington D.C., Ph.D. dissertation, 1991.

[6] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," *Intel Technology Journal*, no. 2, 1998.

[7] J. Brand and J. Mason, "A comparative assessment of three approaches to pixel-level human skin detection," in *Proceedings of IEEE International Conference on Pattern Recognition*, Barcelona, Spain, Sep. 2000, pp. 1056 -1059.

[8] D. Brown, I. Craw, and J. Lewthwaite, "A SOM-based approach to skin detection with application in real time systems," in *Proceedings of British Machine Vision Conference*, Manchester, UK, July 2001, pp. 491-500.

[9] M. C. Burl, T. K. Leung, and P. Perona, "Face localization via shape statistics," in *Proceedings of IEEE International Workshop on Automatic Face and Gesture Recognition*, Zurich, Switzerland, June 1995, pp. 154-159.

[10] J. Cai, A. Goshtasby, and C. Yu, "Detecting human faces in color images," in *Proceedings of IEEE International Workshop on Multimedia Database Management Systems*, Dayton, Ohio, Aug. 1998, pp. 124-131.

[11] T. E. d. Campos, R. S. Feris, and R. M. C. Junior, "Improved face and non-face discrimination using Fourier descriptors through feature selection," in *Proceedings of Brazilian Symposium on Computer Graphics and Image Processing*, Gramado, Brazil, Oct. 2000, pp. 28-35.

[12] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679-698, 1986.

[13] K. R. Castleman, *Digital Image Processing*. Englewood Cliffs, NJ: Prentice Hall, 1989.

[14] D. Chai, "Face segmentation and coding of videophone images," University of Western Australia, Perth, Australia, Ph.D. dissertation, 1999.

[15] D. Chai and K. N. Ngan, "Face segmentation using skin color map in videophone applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 4, pp. 551-564, 1999.

[16] D. Chai, S. L. Phung, and A. Bouzerdoum, "Face localization based on color and shape information in a neural network approach," in *Proceedings of IEEE International Conference on Information, Communications and Signal Processing*, Singapore, Oct. 2001.

[17] R. Chellappa, C. L. Wilson, and S. Sirohey, "Human and machine recognition of faces: A survey," *Proceedings of the IEEE*, vol. 83, no. 5, pp. 705-740, 1995.

[18] C. Chen and S. P. Chiang, "Detection of human faces in colour images," *IEE Proceedings on Vision, Image and Signal Processing*, vol. 144, no. 6, pp. 384-388, 1997.

[19] L.-F. Chen, H.-Y. M. Liao, J.-C. Lin, and C.-C. Han, "Why recognition in a statistics-based face recognition system should be based on the pure face portion: a probabilistic decision-based proof," *Pattern Recognition*, vol. 34, no. 7, pp. 1393-1403, 2001.

[20] A. J. Colmenarez and T. S. Huang, "Face detection with information-based maximum discrimination," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Puerto Rico, June 1997, pp. 782-787.

[21] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York: John Wiley & Sons, Inc., 2001.

[22] M. Elad, Y. Hel-Or, and R. Keshet, "Rejection based classifier for face detection," *Pattern Recognition Letters*, vol. 23, no. 12, pp. 1459-1471, Oct. 2002.

[23] J. Fan, D. K. Y. Yau, A. K. Elmagarmid, and W. G. Aref, "Automatic image segmentation by integrating color-edge extraction and seed region growing," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1454-1466, 2001.

[24] L. Fausett, *Fundamentals of Neural Networks - Architecture, Algorithms, and Applications*. New Jersey: Prentice Hall, Inc., 1994.

[25] R. Féraud, O. J. Bernier, J.-E. Viallet, and M. Collobert, "A fast and accurate face detector based on neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 1, pp. 42-53, Jan. 2001.

[26] R. S. Feris, T. E. d. Campos, and R. M. Cesar, "Detection and tracking of facial features in video sequences," vol. 1793, *Lecture Notes in Artificial Intelligence*: Springer-Verlag Press, 2000, pp. 197-206.

[27] M. Fleck, D. Forsyth, and C. Bregler, "Finding naked people," in *Proceedings of European Conference on Computer Vision*, Cambridge, UK, Apr. 1996, pp. 592-602.

[28] D. A. Forsyth and J. Ponce, *Computer Vision - A Modern Approach*: Prentice Hall, 2002.

[29] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119-139, 1995.

[30] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proceedings of the Thirteenth International Conference on Machine Learning*, Morgan Kuafmann, 1996, pp. 148-156.

[31] B. Fröba, A. Ernst, and C. Kübleck, "Real-time face detection," in *Proceedings of the Fourth IASTED International Conference on Signal and Image Processing*, Hawaii, Aug. 2002, pp. 497-502.

[32] C. Garcia and M. Delakis, "A neural architecture for fast and robust face detection," in *Proceedings of IEEE International Conference on Pattern Recognition*, Quebec, Canada, Aug. 2002, pp. 44-47.

[33] C. Garcia and G. Tziritas, "Face detection using quantized skin color regions merging and wavelet packet analysis," *IEEE Transactions on Multimedia*, vol. 1, no. 3, pp. 264-277, 1999.

[34] C. Garcia and G. Tziritas, "Face detection in color images using wavelet packet analysis," in *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, Florence, Italy, June 1999, pp. 703 -708.

[35] V. Govindaraju, "Locating human faces in photographs," *International Journal of Computer Vision*, vol. 19, no. 2, pp. 129-146, 1996.

[36] D. B. Graham and N. M. Allinson, "Characterizing virtual eigensignatures for general purpose face recognition," in *Face Recognition: From Theory to Applications, NATO ASI Series F, Computer and Systems Sciences*, vol. 163, H. Wechsler, P. J. Phillips, V. Bruce, F. Fogelman-Soulie, and T. S. Huang, Eds., 1998, pp. 446-456.

[37] S. R. Gunn and M. S. Nixon, "A dual active contour for head and boundary extraction," in *Proceedings of IEEE International Colloquium on Image Processing for Biometric Measurement*, London, 1994, pp. 6/1-6/4.

[38] M. T. Hagan and M. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989-993, 1994.

[39] P. Hancock, "Psychological image collection at Stirling (PICS)," University of Stirling, Psychology Department, Available: http://pics.psych.stir.ac.uk/, 2002

[40] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*. Reading, MA: Addison-Wesley Publishing Company, Inc., 1992.

[41] B. Heisele, T. Poggio, and M. Pontil, "Face detection in still gray images," Massachusetts Institute of Technology, Artificial Intelligent Laboratory, A.I. Memo No. 1687, 2000.

[42] E. Hjelmås and B. K. Low, "Face detection: a survey," *Computer Vision and Image Understanding*, vol. 83, no. 3, pp. 236-274, 2001.

[43] R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 696-707, 2002.

[44] F. J. Huang and T. Chen, "Tracking multiple faces for human-computer interfaces and virtual environment," in *Proceedings of IEEE International Conference on Multimedia and Expo*, New York, Jul. 2000, pp. 1563 -1566.

[45] L. Huang, A. Shimizu, and H. Kobatake, "Face detection using a modified radial basis function neural network," in *Proceedings of IEEE International Conference on Pattern Recognition*, Quebec, Canada, Aug. 2002, pp. 342 -345.

[46] L. L. Huang, A. Shimizu, Y. Hagihara, and H. Kobatake, "Robust face detection using a modified radial basis function network," *IEICE Transactions on Information and Systems*, vol. E85-D, no. 10, pp. 1654-1662, Oct. 2002.

[47] L.-L. Huang, A. Shimizu, Y. Hagihara, and H. Kobatake, "Face detection from cluttered images using a polynomial neural network," in *Proceedings of IEEE International Conference on Image Processing*, Thessaloniki, Greece, Oct. 2001, pp. 669 -672.

[48] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4-37, Jan. 2000.

[49] B. H. Jeon, S. U. Lee, and K. M. Lee, "Face detection using the 1st-order RCE classifier," in *Proceedings of IEEE International Conference on Image Processing*, Rochester, New York, Sep. 2002, pp. 125 -128.

[50] O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz, "Robust face detection using the Hausdorff distance," in *Proceedings of International Conference Audio- and Video-Based Biometric Person Authentication*, Halmstad, Sweden, June 2001, pp. 90-95.

[51] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," Compaq Cambridge Research Laboratory, Cambridge, MA, Technical Report CRL 98/11, 1998.

[52] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Ft. Collins, CO, June 1999, pp. 274-280.

[53] S. Karungaru, M. Fukumi, and N. Akamatsu, "Detection of human faces in visual scenes," in *Proceedings of the Seventh Australian and New Zealand Intelligent Information Systems Conference*, Perth, Australia, Nov. 2001, pp. 165-170.

[54] M. Kirby and L. Sirovich, "Application of the Karhunen-Loeve procedure for the characterization of human faces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 103-108, January, 1990.

[55] J. Kittler and F. M. Alkoot, "Sum versus vote fusion in multiple classifier systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 1, pp. 110-115, Jan. 2003.

[56] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226-239, Mar. 1998.

[57] T. Kohonen, *Self-Organizing Maps*, 2nd ed. Berlin: Springer, 1997.

[58] L. I. Kuncheva, "A theoretical study on six classifier fusion strategies," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 281-286, Feb. 2002.

[59] K.-M. Lam and H. Yan, "Fast algorithm for locating head boundaries," *Journal of Electronic Imaging*, vol. 3, no. 4, pp. 351-359, Oct. 1994.

[60] R. Lienhart, A. Kuranov, and V. Pisarevsky, "Empirical analysis of detection cascades of boosted classifiers for rapid object detection," Microprocessor Research Lab, Intel Corporation, Santa Clara, CA, MRL Technical Report, 2002.

[61] R. Lienhart, A. Kuranov, and V. Pisarevsky, "An extended set of Haar-like features for rapid object detection," in *Proceedings of IEEE International Conference on Image Processing*, Rochester, NY, Sep. 2002, pp. 900-903.

[62] R. Lienhart, S. Pfeiffer, and W. Effelsberg, "Video abstracting," *Communications of the ACM*, vol. 40, no. 12, pp. 55-62, Dec. 1997.

[63] C.-H. Lin and J.-L. Wu, "Automatic facial feature extraction by genetic algorithms," *IEEE Transactions on Image Processing*, vol. 8, no. 6, pp. 834-845, 1999.

[64] K.-H. Lin, K.-M. Lam, and W.-C. Siu, "Locating the eye in human face image using fractal dimension," *IEE Proceedings on Vision, Image and Signal Processing*, vol. 148, no. 6, pp. 413-421, 2001.

[65] C. Liu, "A Bayesian discriminating features method for face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 6, pp. 725-740, 2003.

[66] S. Y. Lu and K. S. Fu, "A syntactic approach to texture analysis," *Computer Graphics and Image Processing*, vol. 7, pp. 303-330, 1978.

[67] H. Luo and A. Eleftheriadis, "On face detection in the compressed domain," in *Proceedings of ACM international conference on Multimedia*, California, U.S.A., 2000, pp. 285 - 294.

[68] M. J. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, "Coding facial expression with Gabor wavelets," in *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*, Nara, Japan, Apr. 1998, pp. 200-205.

[69] A. M. Martinez and R. Benavente, "Recognizing imprecisely localized, partially occluded and expression variant faces from a single sample per class," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 748-763, Jul. 2002.

[70] A. M. Martinez and R. Benavente, "The AR face database," Technical Report CVC No. 24, June 1998.

[71] B. Menser and M. Wien, "Automatic face detection and tracking for H.263 compatible region-of-interest coding," in *Proceedings of SPIE Image and Video Communications and Processing*, San Jose, California, USA, 2000, pp. 882-891.

[72] B. Menser and M. Wien, "Segmentation and tracking of facial regions in color image sequences," in *Proceedings of SPIE Visual Communications and Image Processing 2000*, Perth, Australia, June 2000, pp. 731-740.

[73] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 696-710, 1997.

[74] B. W. Ng, "Wavelet based image texture segmentation using a modified K-means algorithm," University of Adelaide, Adelaide, Ph.D. dissertation, 2003.

[75] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Puerto Rico, June 1997, pp. 130 -136.

[76] M. Pantic and L. J. M. Rothkrantz, "Automatic analysis of facial expressions: The state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1424-1445, Dec. 2000.

[77] V. T. Pham and M. Worring, "Face detection methods: a critical evaluation," Department of Computer Science, University of Amsterdam, Amsterdam, The Netherlands, Technical Report 11, Nov. 2000.

[78] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The FERET evaluation methodology for face-recognition algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, Oct. 2000.

[79] S. L. Phung, "ECU face detection database," Edith Cowan University, Available: http://www.soem.ecu.edu.au/~sphung/face_detection/, 2002

[80] S. L. Phung, A. Bouzerdoum, and D. Chai, "A distribution-based face/nonface classifier using Gaussian mixtures," in *Proceedings of the Fourth IASTED International Conference on Signal and Image Processing*, Hawaii, Aug. 2002, pp. 485-490.

[81] S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin segmentation using color and edge information," in *Proceedings of IEEE International Symposium on Signal Processing and Applications*, Paris, France, July 2003

[82] S. L. Phung, A. Bouzerdoum, and D. Chai, "A novel skin color model in YCbCr color space and its application to human face detection," in *Proceedings of IEEE International Conference on Image Processing*, Rochester, New York, Sep. 2002, pp. 289-292.

[83] S. L. Phung, D. Chai, and A. Bouzerdoum, "Adaptive skin segmentation in color images," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Hong Kong, Apr. 2003, pp. 353-356.

[84] S. L. Phung, D. Chai, and A. Bouzerdoum, "A universal and robust human skin color model using neural networks," in *Proceedings of INNS-IEEE International Joint Conference on Neural Networks*, Washington DC, July 2001, pp. 2844 - 2849.

[85] S. L. Phung, D. Chai, and A. Bouzerdoum, "A distribution-based face/nonface classification technique," *Australian Journal of Intelligent Information Processing Systems*, vol. 7, no. 3/4, pp. 132-137, June 2002.

[86] S. L. Phung, D. Chai, and A. Bouzerdoum, "Skin color based face detection," in *Proceedings of the Seventh Australian and New Zealand Intelligent Information Systems Conference*, Perth, Australia, Nov. 2001, pp. 171-176.

[87] I. Pitas, *Digital Image Processing Algorithms*. New York: Prentice Hall, 1993.

[88] R. Radeva, M. Bressan, A. Tobar, and J. Vitrià, "Real-time inspection of cork stoppers using parametric methods in high dimensional spaces," in *Proceedings of the Fourth IASTED International Conference on Signal and Image Processing*, Hawaii, Aug. 2002, pp. 480-484.

[89] T. D. Rikert, M. J. Jones, and P. Viola, "A cluster-based statistical model for object detection," in *Proceedings of IEEE International Confenrence on Computer Vision*, Corfu, Greece, Sep. 1999, pp. 1046-1053.

[90] H. A. Rowley, "Neural network-based face detection," Carnegie Mellon University, Pittsburgh, PA, Ph.D. dissertation, May 1999.

[91] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23-38, 1998.

[92] H. A. Rowley, S. Baluja, and T. Kanade, "Rotation invariant neural network-based face detection," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Santa Barbara, California, June 1998, pp. 963 -963.

[93] T. Sakai, M. Nagao, and T. Kanade, "Computer analysis and classification of photographs of human faces," in *Proceedings of First USA-JAPAN Computer Conference*, 1972, pp. 55-62.

[94] T. Sakai, M. Nagao, and T. Kanade, "Computer analysis of human faces," *Transactions of Institute of Electronics and Communication Engineers of Japan*, vol. 56-D, no. 4, pp. 226-233, 1973.

[95] L. Salgado, N. García, J. M. Menéndez, and E. Rendón, "Efficient image segmentation for region-based motion estimation and compensation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 7, pp. 1029-1039, 2000.

[96] S. A. Samad, A. Hussain, and A. Teoh, "Eye detection using hybrid rule based approach and contour mapping," in *Proceedings of IEEE International Symposium on Signal Processing and its Applications*, Kuala-Lampur, Malaysia, 2001, pp. 631-634.

[97] F. Samaria and A. Harter, "Parameterisation of a stochastic model for human face identification," in *Proceedings of the second IEEE Workshop on Applications of Computer Vision*, Sarasota, FL, Dec. 1994, pp. 138-142.

[98] S. Satoh, Y. Nakamura, and T. Kanade, "Name-It: Naming and detecting faces in news videos," *IEEE Multimedia*, vol. 6, no. 1, pp. 22 - 35, 1999.

[99] H. Schneiderman, "A statistical approach to 3D object detection applied to faces and cars," Carnegie Mellon University, Ph.D. dissertation, 2000.

[100] H. Schneiderman and T. Kanade, "Probabilistic modeling of local appearance and spatial relationships for object recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1998, pp. 45-51.

[101] M. C. Shin, K. I. Chang, and L. V. Tsap, "Does colorspace transformation make any difference on skin detection?," in *Proceedings of IEEE Workshop on Applications of Computer Vision*, Orlando, Florida, Dec. 2002, pp. 275-279.

[102] G. C. D. Silva, M. J. Lyons, S. Kawato, and N. Tetsutani, "Human factors evaluation of a vision-based facial gesture interface," in *Proceedings of IEEE Workshop on Applications of Computer Vision and Pattern Recognition in Human-Computer Interface*, Madison, Wisconsin, Jun. 2003

[103] L. C. D. Silva and K. K. Win, "Automatic facial feature detection for model based coding," in *Proceedings of SPIE Visual Communications and Image Processing 2000*, June 2000, pp. 1126-1137.

[104] S. A. Sirohey, "Human face segmentation and identification," University of Maryland, Center for Automation Research, Technical Report CS-TR-3176, 1993.

[105] K. Sobottka and I. Pitas, "Segmentation and tracking of faces in color images," in *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*, Killington, Vermont, 1996, pp. 236-241.

[106] K. Sobottka and I. Pitas, "A novel method for automatic face segmentation, facial feature extraction and tracking," *Signal Processing: Image Communication*, vol. 12, no. 3, pp. 263-281, 1998.

[107] M. Sonka, V. Hlavav, and R. Boyle, *Image Processing, Analysis and Machine Vision*. London: Chapman & Hall Computing, 1994.

[108] S. Srisuk and W. Kurutach, "A new Hausdrorff distance-based face detection," in *Proceedings of IEEE International Conference on Image Processing*, Thessaloniki, Greece, Oct. 2001, pp. 1022 -1025.

[109] M. Störring, H. J. Andersen, and E. Granum, "Estimation of the illuminant colour from human skin colour," in *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*, Grenoble, France, 2000, pp. 64-69.

[110] K. K. Sung, "Learning and example selection for object and pattern recognition," Massachusetts Institute of Technology, Massachusetts, Ph.D. dissertation, Jan. 1996.

[111] K. K. Sung and T. Poggio, "Example-based learning for view-based human face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 39-51, 1998.

[112] Y.-H. Tasi, Y.-S. Huang, and T. Poggio, "Fast face detection using support vector machine," in *Proceedings of the Fourth IASTED International Conference on Signal and Image Processing*, Hawaii, Aug. 2002, pp. 455-459.

[113] J.-C. Terrillon, Mahdad N. Shirazi, H. Fukamachi, and S. Akamatsu, "Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images," in *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*, Grenoble, France, Mar. 2000, pp. 54 -61.

[114] F. H. C. Tivive and A. Bouzerdoum, "A new class of convolutional neural networks (SICoNNets) and their application to face detection," in *Proceedings of*

International Joint Conference on Neural Networks, Portland, Oregon, July 2003.

[115] K. Toyama, "'Look, Ma - No Hands!' hands-free cursor control with real-time 3D face tracking," in *Proceedings of Workshop on Perceptual User Interfaces*, San Francisco, Nov. 1998, pp. 49-54.

[116] M. A. Turk and A. P. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71-96, 1991.

[117] V. N. Vapnik, *The nature of statistical learning theory*. New York: Springer-Verlag, 1995.

[118] P. Viola and M. Jones, "Fast and robust classification using asymmetric AdaBoost and a detector cascade," in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2002, pp. 1311-1318.

[119] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of Computer Vision and Pattern Recognition*, Kauai, Hawaii, Dec. 2001, pp. 511-518.

[120] P. Viola and M. Jones, "Robust real-time object detection," Cambridge Research Laboratory, Technical Report CRL 2001/01, Feb. 2001.

[121] H. Wang and S. F. Chang, "A highly efficient system for automatic face detection in MPEG video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 4, pp. 615-628, 1997.

[122] H. Wang, F. Liu, and Y. Wang, "View-based face detection system," in *Proceedings of the Fourth IASTED International Conference on Signal and Image Processing*, Hawaii, Aug. 2002, pp. 451-454.

[123] G. Wei, D. Li, and I. K. Sethi, "Detection of side-view faces in color images," in *Proceedings of IEEE International Workshop on Applications of Computer Vision*, Palm Springs. California, Dec. 2000, pp. 79-84.

[124] G. Wei and I. K. Sethi, "Omni-face detection for video/image content description," in *Proceedings of International Workshop on Multimedia Information Retrieval*, California, Nov. 2000, pp. 185 - 189.

[125] K.-W. Wong, K.-M. Lam, and W.-C. Siu, "An efficient algorithm for human face detection and facial feature extraction under different conditions," *Patttern Recognition*, vol. 34, no. 2001, pp. 1993-2004, 2001.

[126] K.-W. Wong, K.-M. Lam, and W.-C. Siu, "A robust scheme for live detection of human faces in color images," *Signal Processing: Image Communication*, vol. 18, no. 2, pp. 103-114, Feb. 2003.

[127] H. Wu, Q. Chen, and M. Yachida, "Face detection from color images using a fuzzy pattern matching method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 6, pp. 557-563, 1999.

[128] J. Wu and Z.-H. Zhou, "Efficient face candidates selector for face detection," *Pattern Recognition*, vol. 36, no. 5, pp. 1175-1186, May 2003.

[129] G. Xu and T. Sugimoto, "Rits Eye: A software-based system for realtime face detection and tracking using pan-tilt-zoom controllable camera," in *Proceedings of IEEE International Conference on Pattern Recognition*, Brisbane, Australia, Aug. 1998, pp. 1194-1197.

[130] J. Yang and A. Waibel, "A real-time face tracker," in *Proceedings of IEEE Workshop on Applications of Computer Vision*, Sarasota, Florida, Dec. 1996, pp. 142 -147.

[131] M.-H. Yang and N. Ahuja, "Gaussian mixture model for human skin color and its applications in image and video databases," in *Proceedings of SPIE Storage and Retrieval for Image and Video Databases*, San Jose, CA, Jan. 1999, pp. 458-466.

[132] M.-H. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34-58, Jan. 2002.

[133] M.-H. Yang, D. Kriegman, and N. Ahuja, "Face detection using multimodal density models," *Computer Vision and Image Understanding*, vol. 84, no. 2, pp. 264-284, Nov. 2001.

[134] K. C. Yow, "Automatic face detection," University of Cambridge, Cambridge, U.K., Ph.D. dissertation, 1998.

[135] Q. Yuan, W. Gao, and H. Yao, "Robust frontal face detection in complex environment," in *Proceedings of IEEE International Conference on Pattern Recognition*, Quebec, Canada, Aug. 2002, pp. 25-28.

[136] A. L. Yuille, P. W. Hallinan, and D. S. Coden, "Feature extraction from faces using deformable templates," *International Journal of Computer Vision*, vol. 8, no. 2, pp. 99-111, 1992.