Handbook of Face Recognition
Face recognition has a large number of applications, including security, person verification, Internet communication, and computer entertainment. Although research in automatic face recognition has been conducted since the 1960s, this problem is still largely unsolved. Recent years have seen significant progress in this area owing to advances in face modeling and analysis techniques. Systems have been developed for face detection and tracking, but reliable face recognition still offers a great challenge to computer vision and pattern recognition researchers.

There are several reasons for recent increased interest in face recognition, including rising public concern for security, the need for identity verification in the digital world, and the need for face analysis and modeling techniques in multimedia data management and computer entertainment. Recent advances in automated face analysis, pattern recognition, and machine learning have made it possible to develop automatic face recognition systems to address these applications.

This book was written based on two primary motivations. The first was the need for highly reliable, accurate face recognition algorithms and systems. The second was the recent research in image and object representation and matching that is of interest to face recognition researchers.

The book is intended for practitioners and students who plan to work in face recognition or who want to become familiar with the state-of-the-art in face recognition. It also provides references for scientists and engineers working in image processing, computer vision, biometrics and security, Internet communications, computer graphics, animation, and the computer game industry. The material fits the following categories: advanced tutorial, state-of-the-art survey, and guide to current technology.

The book consists of 16 chapters, covering all the subareas and major components necessary for designing operational face recognition systems. Each chapter focuses on a specific topic or system component, introduces background information, reviews up-to-date techniques, presents results, and points out challenges and future directions.

Chapter 1 introduces face recognition processing, including major components such as face detection, tracking, alignment, and feature extraction, and it points out the technical challenges of building a face recognition system. We emphasize the importance of subspace analysis and learning, not only providing an understanding of the challenges therein but also the most suc-
cessful solutions available so far. In fact, most technical chapters represent subspace learning-based techniques for various steps in face recognition.

Chapter 2 reviews face detection techniques and describes effective statistical learning methods. In particular, AdaBoost-based learning methods are described because they often achieve practical and robust solutions. Techniques for dealing with nonfrontal face detection are discussed. Results are presented to compare boosting algorithms and other factors that affect face detection performance.

Chapters 3 and 4 discuss face modeling methods for face alignment. These chapters describe methods for localizing facial components (e.g., eyes, nose, mouth) and facial outlines and for aligning facial shape and texture with the input image. Input face images may be extracted from static images or video sequences, and parameters can be extracted from these input images to describe the shape and texture of a face. These results are based largely on advances in the use of active shape models and active appearance models.

Chapters 5 and 6 cover topics related to illumination and color. Chapter 5 describes recent advances in illumination modeling for faces. The illumination invariant facial feature representation is described; this representation improves the recognition performance under varying illumination and inspires further explorations of reliable face recognition solutions. Chapter 6 deals with facial skin color modeling, which is helpful when color is used for face detection and tracking.

Chapter 7 provides a tutorial on subspace modeling and learning-based dimension reduction methods, which are fundamental to many current face recognition techniques. Whereas the collection of all images constitutes high dimensional space, images of faces reside in a subspace of that space. Facial images of an individual are in a subspace of that subspace. It is of paramount importance to discover such subspaces so as to extract effective features and construct robust classifiers.

Chapter 8 addresses problems of face tracking and recognition from a video sequence of images. The purpose is to make use of temporal constraints present in the sequence to make tracking and recognition more reliable.

Chapters 9 and 10 present methods for pose and illumination normalization and extract effective facial features under such changes. Chapter 9 describes a model for extracting illumination invariants, which were previously presented in Chapter 5. Chapter 9 also presents a subregion method for dealing with variation in pose. Chapter 10 describes a recent innovation, called Morphable Models, for generative modeling and learning of face images under changes in illumination and pose in an analysis-by-synthesis framework. This approach results in algorithms that, in a sense, generalize the alignment algorithms described in Chapters 3 and 4 to the situation where the faces are subject to large changes in illumination and pose. In this work, the three-dimensional data of faces are used during the learning phase to train the model in addition to the normal intensity or texture images.

Chapters 11 and 12 provide methods for facial expression analysis and synthesis. The analysis part, Chapter 11, automatically analyzes and recognizes facial motions and facial feature changes from visual information. The synthesis part, Chapter 12, describes techniques on three-dimensional face modeling and animation, face lighting from a single image, and facial expression synthesis. These techniques can potentially be used for face recognition with varying poses, illuminations, and facial expressions. They can also be used for human computer interfaces.
Chapter 13 reviews 27 publicly available databases for face recognition, face detection, and facial expression analysis. These databases provide a common ground for development and evaluation of algorithms for faces under variations in identity, face pose, illumination, facial expression, age, occlusion, and facial hair.

Chapter 14 introduces concepts and methods for face verification and identification performance evaluation. The chapter focuses on measures and protocols used in FERET and FRVT (face recognition vendor tests). Analysis of these tests identifies advances offered by state-of-the-art technologies for face recognition, as well as the limitations of these technologies.

Chapter 15 offers psychological and neural perspectives suggesting how face recognition might go on in the human brain. Combined findings suggest an image-based representation that encodes faces relative to a global average and evaluates deviations from the average as an indication of the unique properties of individual faces.

Chapter 16 describes various face recognition applications, including face identification, security, multimedia management, and human-computer interaction. The chapter also reviews many face recognition systems and discusses related issues in applications and business.

Acknowledgments

A number of people helped in making this book a reality. Vincent Hsu, Dirk Colbry, Xiaoguang Lu, Karthik Nandakumar, and Anoop Namboodiri of Michigan State University, and Shiguang Shan, Zhenan Sun, Chenghua Xu and Jiangwei Li of the Chinese Academy of Sciences helped proofread several of the chapters. We also thank Wayne Wheeler and Ann Kostant, editors at Springer, for their suggestions and for keeping us on schedule for the production of the book. This handbook project was done partly when Stan Li was with Microsoft Research Asia.

December 2004

Stan Z. Li
Beijing, China

Anil K. Jain
East Lansing, Michigan
# Contents

Chapter 1. Introduction  
*Stan Z. Li, Anil K. Jain*  .....................................................  1

Chapter 2. Face Detection  
*Stan Z. Li*  ................................................................  13

Chapter 3. Modeling Facial Shape and Appearance  
*Tim Cootes, Chris Taylor, Haizhuang Kang, Vladimir Petrović*  .................  39

Chapter 4. Parametric Face Modeling and Tracking  
*Jörgen Ahlberg, Fadi Dornaika*  ...........................................  65

Chapter 5. Illumination Modeling for Face Recognition  
*Ronen Basri, David Jacobs*  ..................................................  89

Chapter 6. Facial Skin Color Modeling  
*J. Birgitta Martinkauppi, Matti Pietikäinen*  ....................................  113

Color Plates for Chapters 6 and 15  
.................................................................  137

Chapter 7. Face Recognition in Subspaces  
*Gregory Shakhnarovich, Baback Moghaddam*  ..................................  141

Chapter 8. Face Tracking and Recognition from Video  
*Rama Chellappa, Shaohua Kevin Zhou*  .........................................  169

Chapter 9. Face Recognition Across Pose and Illumination  
*Ralph Gross, Simon Baker, Iain Matthews, Takeo Kanade*  .........................  193

Chapter 10. Morphable Models of Faces  
*Sami Romdhani, Volker Blanz, Curzio Basso, Thomas Vetter*  ....................  217
Chapter 11. Facial Expression Analysis
Ying-Li Tian, Takeo Kanade, Jeffrey F. Cohn ........................................... 247

Chapter 12. Face Synthesis
Zicheng Liu, Baining Guo ................................................................. 277

Chapter 13. Face Databases
Ralph Gross ..................................................................................... 301

Chapter 14. Evaluation Methods in Face Recognition
P. Jonathon Phillips, Patrick Grother, Ross Micheals ............................. 329

Chapter 15. Psychological and Neural Perspectives on Human Face Recognition
Alice J. O’Toole ............................................................................... 349

Chapter 16. Face Recognition Applications
Thomas Huang, Ziyue Xiong, Zhenqiu Zhang .................................... 371

Index .............................................................................................. 391
Chapter 1. Introduction

Stan Z. Li¹ and Anil K. Jain²

¹ Center for Biometrics Research and Testing (CBRT) and National Laboratory of Pattern Recognition (NLPR), Chinese Academy of Sciences, Beijing 100080, China. szli@nlpr.ia.ac.cn
² Michigan State University, East Lansing, MI 48824, USA. jain@cse.msu.edu

Face recognition is a task that humans perform routinely and effortlessly in their daily lives. Wide availability of powerful and low-cost desktop and embedded computing systems has created an enormous interest in automatic processing of digital images and videos in a number of applications, including biometric authentication, surveillance, human-computer interaction, and multimedia management. Research and development in automatic face recognition follows naturally.

Research in face recognition is motivated not only by the fundamental challenges this recognition problem poses but also by numerous practical applications where human identification is needed. Face recognition, as one of the primary biometric technologies, became more and more important owing to rapid advances in technologies such as digital cameras, the Internet and mobile devices, and increased demands on security. Face recognition has several advantages over other biometric technologies: It is natural, nonintrusive, and easy to use. Among the six biometric attributes considered by Hietmeyer [12], facial features scored the highest compatibility in a Machine Readable Travel Documents (MRTD) [18] system based on a number of evaluation factors, such as enrollment, renewal, machine requirements, and public perception, shown in Figure 1.1.

A face recognition system is expected to identify faces present in images and videos automatically. It can operate in either or both of two modes: (1) face verification (or authentication), and (2) face identification (or recognition). Face verification involves a one-to-one match that compares a query face image against a template face image whose identity is being claimed. Face identification involves one-to-many matches that compares a query face image against all the template images in the database to determine the identity of the query face. Another face recognition scenario involves a watch-list check, where a query face is matched to a list of suspects (one-to-few matches).

The performance of face recognition systems has improved significantly since the first automatic face recognition system was developed by Kanade [14]. Furthermore, face detection, facial feature extraction, and recognition can now be performed in “realtime” for images captured under favorable (i.e., constrained) situations.

* Part of this work was done when Stan Z. Li was with Microsoft Research Asia.
Although progress in face recognition has been encouraging, the task has also turned out to be a difficult endeavor, especially for unconstrained tasks where viewpoint, illumination, expression, occlusion, accessories, and so on vary considerably. In the following sections, we give a brief review on technical advances and analyze technical challenges.

Fig. 1.1. A scenario of using biometric MRTD systems for passport control (left), and a comparison of various biometric features based on MRTD compatibility (right, from Hietmeyer [12] with permission).

1 Face Recognition Processing

Face recognition is a visual pattern recognition problem. There, a face as a three-dimensional object subject to varying illumination, pose, expression and so on is to be identified based on its two-dimensional image (three-dimensional images e.g., obtained from laser may also be used). A face recognition system generally consists of four modules as depicted in Figure 1.2: detection, alignment, feature extraction, and matching, where localization and normalization (face detection and alignment) are processing steps before face recognition (facial feature extraction and matching) is performed.

Face detection segments the face areas from the background. In the case of video, the detected faces may need to be tracked using a face tracking component. Face alignment is aimed at achieving more accurate localization and at normalizing faces thereby whereas face detection provides coarse estimates of the location and scale of each detected face. Facial components, such as eyes, nose, and mouth and facial outline, are located; based on the location points, the input face image is normalized with respect to geometrical properties, such as size and pose, using geometrical transforms or morphing. The face is usually further normalized with respect to photometrical properties such illumination and gray scale.

After a face is normalized geometrically and photometrically, feature extraction is performed to provide effective information that is useful for distinguishing between faces of different persons and stable with respect to the geometrical and photometrical variations. For face matching, the extracted feature vector of the input face is matched against those of enrolled
faces in the database; it outputs the identity of the face when a match is found with sufficient confidence or indicates an unknown face otherwise.

Face recognition results depend highly on features that are extracted to represent the face pattern and classification methods used to distinguish between faces whereas face localization and normalization are the basis for extracting effective features. These problems may be analyzed from the viewpoint of face subspaces or manifolds, as follows.

2 Analysis in Face Subspaces

Subspace analysis techniques for face recognition are based on the fact that a class of patterns of interest, such as the face, resides in a subspace of the input image space. For example, a small image of $64 \times 64$ has 4096 pixels can express a large number of pattern classes, such as trees, houses and faces. However, among the $256^{4096} > 10^{9864}$ possible “configurations,” only a few correspond to faces. Therefore, the original image representation is highly redundant, and the dimensionality of this representation could be greatly reduced when only the face pattern are of interest.

With the eigenface or principal component analysis (PCA) [9] approach [28], a small number (e.g., 40 or lower) of eigenfaces [26] are derived from a set of training face images by using the Karhunen-Loeve transform or PCA. A face image is efficiently represented as a feature vector (i.e., a vector of weights) of low dimensionality. The features in such subspace provide more salient and richer information for recognition than the raw image. The use of subspace modeling techniques has significantly advanced face recognition technology.

The manifold or distribution of all faces accounts for variation in face appearance whereas the nonface manifold accounts for everything else. If we look into these manifolds in the image space, we find them highly nonlinear and nonconvex [4, 27]. Figure 1.3(a) illustrates face versus nonface manifolds and (b) illustrates the manifolds of two individuals in the entire face manifold. Face detection can be considered as a task of distinguishing between the face and nonface manifolds in the image (subwindow) space and face recognition between those of individuals in the face manifold.

Figure 1.4 further demonstrates the nonlinearity and nonconvexity of face manifolds in a PCA subspace spanned by the first three principal components, where the plots are drawn from
Fig. 1.3. (a) Face versus nonface manifolds. (b) Face manifolds of different individuals.

real face image data. Each plot depicts the manifolds of three individuals (in three colors). There are 64 original frontal face images for each individual. A certain type of transform is performed on an original face image with 11 gradually varying parameters, producing 11 transformed face images; each transformed image is cropped to contain only the face region; the 11 cropped face images form a sequence. A curve in this figure is the image of such a sequence in the PCA space, and so there are 64 curves for each individual. The three-dimensional (3D) PCA space is projected on three 2D spaces (planes). We can see the nonlinearity of the trajectories.

Two notes follow: First, while these examples are demonstrated in a PCA space, more complex (nonlinear and nonconvex) curves are expected in the original image space. Second, although these examples are subject to the geometric transformations in the 2D plane and pointwise lighting (gamma) changes, more significant complexity is expected for geometric transformations in 3D (e.g., out-of-plane head rotations) transformations and lighting direction changes.

3 Technical Challenges

As shown in Figure 1.3, the classification problem associated with face detection is highly nonlinear and nonconvex, even more so for face matching. Face recognition evaluation reports (e.g., [8, 23]) and other independent studies indicate that the performance of many state-of-the-art face recognition methods deteriorates with changes in lighting, pose, and other factors [6, 29, 35]. The key technical challenges are summarized below.

Large Variability in Facial Appearance. Whereas shape and reflectance are intrinsic properties of a face object, the appearance (i.e., the texture look) of a face is also subject to several other factors, including the facial pose (or, equivalently, camera viewpoint), illumination, facial expression. Figure 1.5 shows an example of significant intrasubject variations caused by these
Fig. 1.4. Nonlinearity and nonconvexity of face manifolds under (from top to bottom) translation, rotation, scaling, and Gamma transformations.

factors. In addition to these, various imaging parameters, such as aperture, exposure time, lens aberrations, and sensor spectral response also increase intrasubject variations. Face-based person identification is further complicated by possible small intersubject variations (Figure 1.6). All these factors are confounded in the image data, so “the variations between the images of the same face due to illumination and viewing direction are almost always larger than the image variation due to change in face identity” [21]. This variability makes it difficult to extract the
intrinsic information of the face objects from their respective images.

Fig. 1.5. Intrasubject variations in pose, illumination, expression, occlusion, accessories (e.g., glasses), color, and brightness. (Courtesy of Rein-Lien Hsu [13].)

Fig. 1.6. Similarity of frontal faces between (a) twins (downloaded from www.marykateandashley.com); and (b) a father and his son (downloaded from BBC news, news.bbc.co.uk).

**Highly Complex Nonlinear Manifolds.** As illustrated above, the entire face manifold is highly nonconvex, and so is the face manifold of any individual under various change. Linear methods such as PCA [26, 28], independent component analysis (ICA) [2], and linear discriminant analysis (LDA) [3]) project the data linearly from a high-dimensional space (e.g., the image space) to a low-dimensional subspace. As such, they are unable to preserve the nonconvex variations of face manifolds necessary to differentiate among individuals. In a linear subspace, Euclidean distance and more generally Mahalanobis distance, which are normally used for template matching, do not perform well for classifying between face and nonface manifolds and...
between manifolds of individuals (Figure 1.7(a)). This crucial fact limits the power of the linear methods to achieve highly accurate face detection and recognition.

**High Dimensionality and Small Sample Size.** Another challenge is the ability to generalize, illustrated by Figure 1.7(b). A canonical face image of $112 \times 92$ resides in a 10,304-dimensional feature space. Nevertheless, the number of examples per person (typically fewer than 10, even just one) available for learning the manifold is usually much smaller than the dimensionality of the image space; a system trained on so few examples may not generalize well to unseen instances of the face.

![Fig. 1.7. Challenges in face recognition from subspace viewpoint. (a) Euclidean distance is unable to differentiate between individuals: In terms of Euclidean distance, an interpersonal distance can be smaller than an intrapersonal one. (b) The learned manifold or classifier is unable to characterize (i.e., generalize to) unseen images of the same individual face.](image)

### 4 Technical Solutions

There are two strategies for dealing with the above difficulties: feature extraction and pattern classification based on the extracted features. One is to construct a “good” feature space in which the face manifolds become simpler i.e., less nonlinear and nonconvex than those in the other spaces. This includes two levels of processing: (1) normalize face images geometrically and photometrically, such as using morphing and histogram equalization; and (2) extract features in the normalized images which are stable with respect to such variations, such as based on Gabor wavelets.

The second strategy is to construct classification engines able to solve difficult nonlinear classification and regression problems in the feature space and to generalize better. Although good normalization and feature extraction reduce the nonlinearity and nonconvexity, they do not solve the problems completely and classification engines able to deal with such difficulties
are still necessary to achieve high performance. A successful algorithm usually combines both strategies.

With the geometric feature-based approach used in the early days [5, 10, 14, 24], facial features such as eyes, nose, mouth, and chin are detected. Properties of and relations (e.g., areas, distances, angles) between the features are used as descriptors for face recognition. Advantages of this approach include economy and efficiency when achieving data reduction and insensitivity to variations in illumination and viewpoint. However, facial feature detection and measurement techniques developed to date are not reliable enough for the geometric feature-based recognition [7], and such geometric properties alone are inadequate for face recognition because rich information contained in the facial texture or appearance is discarded. These are reasons why early techniques are not effective.

The statistical learning approach learns from training data (appearance images or features extracted from appearance) to extract good features and construct classification engines. During the learning, both prior knowledge about face(s) and variations seen in the training data are taken into consideration. Many successful algorithms for face detection, alignment and matching nowadays are learning-based.

The appearance-based approach, such as PCA [28] and LDA [3] based methods, has significantly advanced face recognition techniques. Such an approach generally operates directly on an image-based representation (i.e., array of pixel intensities). It extracts features in a subspace derived from training images. Using PCA, a face subspace is constructed to represent “optimally” only the face object; using LDA, a discriminant subspace is constructed to distinguish “optimally” faces of different persons. Comparative reports (e.g., [3]) show that LDA-based methods generally yield better results than PCA-based ones.

Although these linear, holistic appearance-based methods avoid instability of the early geometric feature-based methods, they are not accurate enough to describe subtleties of original manifolds in the original image space. This is due to their limitations in handling nonlinearity in face recognition: there, protrusions of nonlinear manifolds may be smoothed and concavities may be filled in, causing unfavorable consequences.

Such linear methods can be extended using nonlinear kernel techniques (kernel PCA [25] and kernel LDA [19]) to deal with nonlinearity in face recognition [11, 16, 20, 31]. There, a nonlinear projection (dimension reduction) from the image space to a feature space is performed; the manifolds in the resulting feature space become simple, yet with subtleties preserved. Although the kernel methods may achieve good performance on the training data, however, it may not be so for unseen data owing to their more flexibility than the linear methods and overfitting thereof.

Another approach to handle the nonlinearity is to construct a local appearance-based feature space, using appropriate image filters, so the distributions of faces are less affected by various changes. Local features analysis (LFA) [22], Gabor wavelet-based features (such as elastic graph bunch matching, EGBM) [15, 30, 17] and local binary pattern (LBP) [1] have been used for this purpose.

Some of these algorithms may be considered as combining geometric (or structural) feature detection and local appearance feature extraction, to increase stability of recognition performance under changes in viewpoint, illumination, and expression. A taxonomy of major face recognition algorithms in Figure 1.8 provides an overview of face recognition technology based on pose dependency, face representation, and features used for matching.
Fig. 1.8. Taxonomy of face recognition algorithms based on pose-dependency, face representation, and features used in matching (Courtesy of Rein-Lien Hsu [13]).

A large number of local features can be produced with varying parameters in the position, scale and orientation of the filters. For example, more than 100,000 local appearance features can be produced when an image of $100 \times 100$ is filtered with Gabor filters of five scales and eight orientation for all pixel positions, causing increased dimensionality. Some of these features are effective and important for the classification task whereas the others may not be so. AdaBoost methods have been used successfully to tackle the feature selection and nonlinear classification problems [32, 33, 34]. These works lead to a framework for learning both effective features and effective classifiers.

5 Current Technology Maturity

As introduced earlier, a face recognition system consists of several components, including face detection, tracking, alignment, feature extraction, and matching. Where are we along the road of making automatic face recognition systems? To answer this question, we have to assume some given constraints namely what the intended situation for the application is and how strong constraints are assumed, including pose, illumination, facial expression, age, occlusion, and facial hair. Although several chapters (14 and 16 in particular), provide more objective comments, we risk saying the following here: Real-time face detection and tracking in the normal indoor environment is relatively well solved, whereas more work is needed for handling outdoor scenes. When faces are detected and tracked, alignment can be done as well, assuming the image resolution is good enough for localizing the facial components, face recognition works well for
cooperative frontal faces without exaggerated expressions and under illumination without much shadow. Face recognition in an unconstrained daily life environment without the user’s cooperation, such as for recognizing someone in an airport, is currently a challenging task. Many years’ effort is required to produce practical solutions to such problems.

Acknowledgment

The authors thank Jörgen Ahlberg for his feedback on Chapters 1 and 2.

References

Chapter 2. Face Detection

Stan Z. Li

Microsoft Research Asia, Beijing 100080, China. *

Face detection is the first step in automated face recognition. Its reliability has a major influence on the performance and usability of the entire face recognition system. Given a single image or a video, an ideal face detector should be able to identify and locate all the present faces regardless of their position, scale, orientation, age, and expression. Furthermore, the detection should be irrespective of extraneous illumination conditions and the image and video content.

Face detection can be performed based on several cues: skin color (for faces in color images and videos), motion (for faces in videos), facial/head shape, facial appearance, or a combination of these parameters. Most successful face detection algorithms are appearance-based without using other cues. The processing is done as follows: An input image is scanned at all possible locations and scales by a subwindow. Face detection is posed as classifying the pattern in the subwindow as either face or nonface. The face/nonface classifier is learned from face and nonface training examples using statistical learning methods.

This chapter focuses on appearance-based and learning-based methods. More attention is paid to AdaBoost learning-based methods because so far they are the most successful ones in terms of detection accuracy and speed. The reader is also referred to review articles, such as those of Hjelmas and Low [12] and Yang et al. [52], for other face detection methods.

1 Appearance-Based and Learning Based Approaches

With appearance-based methods, face detection is treated as a problem of classifying each scanned subwindow as one of two classes (i.e., face and nonface). Appearance-based methods avoid difficulties in modeling 3D structures of faces by considering possible face appearances under various conditions. A face/nonface classifier may be learned from a training set composed of face examples taken under possible conditions as would be seen in the running stage and nonface examples as well (see Figure 2.1 for a random sample of 10 face and 10 nonface subwindow images). Building such a classifier is possible because pixels on a face are highly correlated, whereas those in a nonface subwindow present much less regularity.

* Stan Z. Li is currently with Center for Biometrics Research and Testing (CBRT) and National Laboratory of Pattern Recognition (NLPR), Chinese Academy of Sciences, Beijing 100080, China. szli@nlpr.ia.ac.cn
However, large variations brought about by changes in facial appearance, lighting, and expression make the face manifold or face/nonface boundaries highly complex [4, 38, 43]. Changes in facial view (head pose) further complicate the situation. A nonlinear classifier is needed to deal with the complicated situation. The speed is also an important issue for realtime performance. Great research effort has been made for constructing complex yet fast classifiers and much progress has been achieved since 1990s.

![Fig. 2.1. Face (top) and nonface (bottom) examples.](image)

Turk and Pentland [44] describe a detection system based on principal component analysis (PCA) subspace or eigenface representation. Whereas only likelihood in the PCA subspace is considered in the basic PCA method, Moghaddam and Pentland [25] also consider the likelihood in the orthogonal complement subspace; using that system, the likelihood in the image space (the union of the two subspaces) is modeled as the product of the two likelihood estimates, which provide a more accurate likelihood estimate for the detection. Sung and Poggio [41] first partition the image space into several face and nonface clusters and then further decompose each cluster into the PCA and null subspaces. The Bayesian estimation is then applied to obtain useful statistical features. The system of Rowley et al. ’s [32] uses retinally connected neural networks. Through a sliding window, the input image is examined after going through an extensive preprocessing stage. Osuna et al. [27] train a nonlinear support vector machine to classify face and nonface patterns, and Yang et al. [53] use the SNoW (Sparse Network of Winnows) learning architecture for face detection. In these systems, a bootstrap algorithm is used iteratively to collect meaningful nonface examples from images that do not contain any faces for retraining the detector.

Schneiderman and Kanade [35] use multiresolution information for different levels of wavelet transform. A nonlinear face and nonface classifier is constructed using statistics of products of histograms computed from face and nonface examples using AdaBoost learning [34]. The algorithm is computationally expensive. The system of five view detectors takes about 1 minute to detect faces for a 320×240 image over only four octaves of candidate size [35].

Viola and Jones [46, 47] built a fast, robust face detection system in which AdaBoost learning is used to construct nonlinear classifier (earlier work on the application of Adaboost for image classification and face detection can be found in [42] and [34]). AdaBoost is used to solve the following three fundamental problems: (1) learning effective features from a large feature set; (2) constructing weak classifiers, each of which is based on one of the selected features; and (3) boosting the weak classifiers to construct a strong classifier. Weak classifiers are

---

1 During the revision of this article, Schneiderman and Kanade [36] reported an improvement in the speed of their system, using a coarse-to-fine search strategy together with various heuristics (re-using Wavelet Transform coefficients, color preprocessing, etc.). The improved speed is five seconds for an image of size 240 × 256 using a Pentium II at 450MHz.
based on simple scalar Haar wavelet-like features, which are steerable filters [28]. Viola and
Jones make use of several techniques [5, 37] for effective computation of a large number of
such features under varying scale and location, which is important for realtime performance.
Moreover, the simple-to-complex cascade of classifiers makes the computation even more ef-
cient, which follows the principles of pattern rejection [3, 6] and coarse-to-fine search [2, 8].
Their system is the first realtime frontal-view face detector, and it runs at about 14 frames per
second on a 320×240 image [47].

Liu [23] presents a Bayesian Discriminating Features (BDF) method. The input image, its
one-dimensional Harr wavelet representation, and its amplitude projections are concatenated
into an expanded vector input of 768 dimensions. Assuming that these vectors follow a (sin-
gle) multivariate normal distribution for face, linear dimension reduction is performed to obtain
the PCA modes. The likelihood density is estimated using PCA and its residuals, making use
of Bayesian techniques [25]. The nonface class is modeled similarly. A classification decision
of face/nonface is made based on the two density estimates. The BDF classifier is reported to
achieve results that compare favorably with state-of-the-art face detection algorithms, such as
the Schneiderman-Kanade method. It is interesting to note that such good results are achieved
with a single Gaussian for face and one for nonface, and the BDF is trained using relatively
small data sets: 600 FERET face images and 9 natural (nonface) images; the trained classi-
fier generalizes very well to test images. However, more details are needed to understand the
underlying mechanism.

The ability to deal with nonfrontal faces is important for many real applications because
approximately 75% of the faces in home photos are nonfrontal [17]. A reasonable treatment
for the multiview face detection problem is the view-based method [29], in which several face
models are built, each describing faces in a certain view range. This way, explicit 3D face
modeling is avoided. Feraud et al. [7] adopt the view-based representation for face detection
and use an array of five detectors, with each detector responsible for one facial view. Wiskott et
al. [48] build elastic bunch graph templates for multiview face detection and recognition. Gong
et al. [11] study the trajectories of faces (as they are rotated) in linear PCA feature spaces and
use kernel support vector machines (SVMs) for multipose face detection and pose estimation
[21, 26]. Huang et al. [14] use SVMs to estimate the facial pose. The algorithm of Schneiderman
and Kanade [35] consists of an array of five face detectors in the view-based framework.

Li et al. [18, 19, 20] present a multiview face detection system, extending the work in other
articles [35, 46, 47]. A new boosting algorithm, called FloatBoost, is proposed to incorporate
Floating Search [30] into AdaBoost (RealBoost). The backtrack mechanism in the algorithm
allows deletions of weak classifiers that are ineffective in terms of the error rate, leading to a
strong classifier consisting of only a small number of weak classifiers. An extended Haar feature
set is proposed for dealing with out-of-plane (left-right) rotation. A coarse-to-fine, simple-to-
complex architecture, called a detector-pyramid, is designed for the fast detection of multiview
faces. This work leads to the first realtime multiview face detection system. It runs at 200 ms
per image (320×240 pixels) on a Pentium-III CPU of 700 MHz.

Lienhart et al. [22] use an extended set of rotated Haar features for dealing with in-plane
rotation and train a face detector using Gentle Adaboost [9] with small CART trees as base
classifiers. The results show that this combination outperforms that of Discrete Adaboost with
stumps.
In the following sections, we describe basic face-processing techniques and neural network-based and AdaBoost-based learning methods for face detection. Given that the AdaBoost learning with the Haar-like feature approach has achieved the best performance to date in terms of both accuracy and speed, our presentation focuses on the AdaBoost methods. Strategies are also described for efficient detection of multiview faces.

2 Preprocessing

2.1 Skin Color Filtering

Human skin has its own color distribution that differs from that of most of nonface objects. It can be used to filter the input image to obtain candidate regions of faces, and it may also be used to construct a stand-alone skin color-based face detector for special environments. A simple color-based face detection algorithm consists of two steps: (1) segmentation of likely face regions and (2) region merging.

![Fig. 2.2. Skin color filtering. Input image (left) and skin color-filtered map (right).](image)

A skin color likelihood model, \( p(\text{color}|\text{face}) \), can be derived from skin color samples. This may be done in the hue-saturation-value (HSV) color space or in the normalized red-green-blue (RGB) color space (see [24, 54] and Chapter 6 for comparative studies). A Gaussian mixture model for \( p(\text{color}|\text{face}) \) can lead to better skin color modeling [49, 50]. Figure 2.2 shows skin color segmentation maps. A skin-colored pixel is found if the likelihood \( p(H|\text{face}) \) is greater than a threshold (0.3), and S and V values are between some upper and lower bounds. A skin color map consists of a number of skin color regions that indicate potential candidate face regions. Refined face regions can be obtained by merging the candidate regions based on the color and spatial information. Heuristic postprocessing could be performed to remove false detection. For example, a human face contains eyes where the eyes correspond to darker regions inside the face region. A sophisticated color based face detection algorithm is presented in Hsu et al. [13].

Although a color-based face detection system may be computationally attractive, the color constraint alone is insufficient for achieving high accuracy face detection. This is due to large
facial color variation as a result of different lighting, shadow, and ethic groups. Indeed, it is the appearance, albeit colored or gray level, rather than the color that is most essential for face detection. Skin color is often combined with the motion cue to improve the reliability for face detection and tracking on video [49, 50]. However, the most successful face detection systems do not rely on color or motion information, yet achieve good performance.

2.2 Image Normalization

Appearance-based methods operate on subwindows of a fixed size. Therefore, explicit or implicit resizing (e.g., to 20×20 pixels) is necessary. Normalization of pixel intensity helps correct variations in imaging parameters in cameras as well as changes in illumination conditions. The meaning of resizing is apparent; intensity normalization operations, including mean value normalization, histogram equalization, and illumination correction, are described below.

A simple intensity normalization operation is linear stretching. A histogram equalization helps reduce extreme illumination (Figure 2.3). In another simple illumination correction operation, the subwindow \( I(x, y) \) is fitted to the best fitting plane \( I'(x, y) = a \times x + b \times y + c \), where the values of the coefficients \( a, b \) and \( c \) may be estimated using the least-squares method; and then extreme illumination is reduced in the difference image \( I''(x, y) = I(x, y) - I'(x, y) \) (Figure 2.4) [32, 41]. After normalization, the distribution of subwindow images becomes more compact and standardized, which helps reduce the complexity of the subsequent face/nonface classification. Note that these operations are “global” in the sense that all the pixels may be affected after such an operation. Intensity normalization may also be applied to local subregions, as is in the case for local Haar wavelet features [46] (See later in AdaBoost based methods).

Fig. 2.3. Effect of linear stretching and and histogram equalization. (a) Original subwindow. (b) Linearly stretched. (c) Histogram equalized.

2.3 Gaussian Mixture Modeling

The distributions of face and nonface subwindows in a high dimensional space are complex. It is believed that a single Gaussian distribution cannot explain all variations. Sung and Poggio [41] propose to deal with this complexity by partitioning the face training data into several (six) face clusters, and nonface training data into several (six) nonface clusters, where the cluster
The clustering is performed by using a modified $k$-means algorithm based on the Mahalanobis distance [41] in the image space or some another space. Figure 2.5 shows the centroids of the resulting face and nonface clusters. Each cluster can be further modeled by its principal components using the PCA technique. Based on the multi-Gaussian and PCA modeling, a parametric classifier can be formulated based on the distances of the projection points within the subspaces and from the subspaces [41]. The clustering can also be done using factor analysis and self-organizing map (SOM) [51].

It is believed that a few (e.g., six) Gaussian distributions are not enough to model the face distribution and even less sufficient to model the nonface distribution. However, it is reported in [23] that good results are achieved using a single Gaussian distribution for face and one for nonface, with a nonlinear kernel support vector machine classifier; and more interestingly, the BDF face/nonface classifier therein is trained using relatively small data sets: 600 FERET face images and 9 natural (nonface) images, and it generalizes very well to test images. The BDF work is worth more studies.
Chapter 2. Face Detection

3 Neural Networks and Kernel Based Methods

Nonlinear classification for face detection may be performed using neural networks or kernel-based methods. With the neural methods [32, 41], a classifier may be trained directly using preprocessed and normalized face and nonface training subwindows. Rowley et al. [32] use the preprocessed 20×20 subwindow as an input to a neural network. The network has retinal connections to its input layer and two levels of mapping. The first level maps blocks of pixels to the hidden units. There are 4 blocks of 10×10 pixels, 16 blocks of 5×5 pixels, and 6 overlapping horizontal stripes of 20×5 pixels. Each block is input to a fully connected neural network and mapped to the hidden units. The 26 hidden units are then mapped to the final single-valued output unit and a final decision is made to classify the 20×20 subwindow into face or nonface. Several copies of the same networks can be trained and their outputs combined by arbitration (ANDing) [32].

The input to the system of Sung and Poggio [41] is derived from the six face and six nonface clusters. More specifically, it is a vector of 2×6 = 12 distances in the PCA subspaces and 2×6 = 12 distances from the PCA subspaces. The 24 dimensional feature vector provides a good representation for classifying face and nonface patterns. In both systems, the neural networks are trained by back-propagation algorithms.

Nonlinear classification for face detection can also be done using kernel SVMs [21, 26, 27], trained using face and nonface examples. Although such methods are able to learn nonlinear boundaries, a large number of support vectors may be needed to capture a highly nonlinear boundary. For this reason, fast realtime performance has so far been a difficulty with SVM classifiers thus trained. Although these SVM-based systems have been trained using the face and nonface subwindows directly, there is no reason why they cannot be trained using some salient features derived from the subwindows.

Yang et al. [53] use the SNoW learning architecture for face detection. SNoW is a sparse network of linear functions in which Winnow update rule is applied to the learning. The SNoW algorithm is designed for learning with a large set of candidate features. It uses classification error to perform multiplicative update of the weights connecting the target nodes.

4 AdaBoost-Based Methods

For AdaBoost learning, a complex nonlinear strong classifier \( H_M(x) \) is constructed as a linear combination of \( M \) simpler, easily constructible weak classifiers in the following form [9]

\[
H_M(x) = \frac{\sum_{m=1}^{M} \alpha_m h_m(x)}{\sum_{m=1}^{M} \alpha_m}
\]

where \( x \) is a pattern to be classified, \( h_m(x) \in \{-1, +1\} \) are the \( M \) weak classifiers, \( \alpha_m \geq 0 \) are the combining coefficients in \( \mathbb{R} \), and \( \sum_{m=1}^{M} \alpha_m \) is the normalizing factor. In the discrete version, \( h_m(x) \) takes a discrete value in \( \{-1, +1\} \), whereas in the real version, the output of \( h_m(x) \) is a number in \( \mathbb{R} \). \( H_M(x) \) is real-valued, but the prediction of class label for \( x \) is obtained as \( \hat{y}(x) = \text{sign}[H_M(x)] \) and the normalized confidence score is \( |H_M(x)| \).

The AdaBoost learning procedure is aimed at learning a sequence of best weak classifiers \( h_m(x) \) and the best combining weights \( \alpha_m \). A set of \( N \) labeled training examples
\{ (x_1, y_1), \ldots, (x_N, y_N) \} is assumed available, where \( y_i \in \{ +1, -1 \} \) is the class label for the example \( x_i \in \mathbb{R}^n \). A distribution \( [w_1, \ldots, w_N] \) of the training examples, where \( w_i \) is associated with a training example \( (x_i, y_i) \), is computed and updated during the learning to represent the distribution of the training examples. After iteration \( m \), harder-to-classify examples \( (x_i, y_i) \) are given larger weights \( w_i^{(m)} \), so that at iteration \( m + 1 \), more emphasis is placed on these examples. AdaBoost assumes that a procedure is available for learning a weak classifier \( h_m(x) \) from the training examples, given the distribution \( [w_i^{(m)}] \).

In Viola and Jones’s face detection work [46, 47], a weak classifier \( h_m(x) \in \{ -1, +1 \} \) is obtained by thresholding on a scalar feature \( z_k(x) \in \mathbb{R} \) selected from an overcomplete set of Haar wavelet-like features [28, 42]. In the real versions of AdaBoost, such as RealBoost and LogitBoost, a real-valued weak classifier \( h_m(x) \in \mathbb{R} \) can also be constructed from \( z_k(x) \in \mathbb{R} \) [20, 22, 34]. The following discusses how to generate candidate weak classifiers.

### 4.1 Haar-like Features

Viola and Jones propose four basic types of scalar features for face detection [28, 47], as shown in Figure 2.6. Such a block feature is located in a subregion of a subwindow and varies in shape (aspect ratio), size, and location inside the subwindow. For a subwindow of size 20×20, there can be tens of thousands of such features for varying shapes, sizes and locations. Feature \( k \), taking a scalar value \( z_k(x) \in \mathbb{R} \), can be considered a transform from the \( n \)-dimensional space (\( n = 400 \) if a face example \( x \) is of size 20×20) to the real line. These scalar numbers form an overcomplete feature set for the intrinsically low-dimensional face pattern. Recently, extended sets of such features have been proposed for dealing with out-of-plane head rotation [20] and for in-plane head rotation [22].

![Fig. 2.6. Four types of rectangular Haar wavelet-like features. A feature is a scalar calculated by summing up the pixels in the white region and subtracting those in the dark region.](image)

These Haar-like features are interesting for two reasons: (1) powerful face/nonface classifiers can be constructed based on these features (see later); and (2) they can be computed efficiently [37] using the summed-area table [5] or integral image [46] technique.

The integral image \( II(x, y) \) at location \( x, y \) contains the sum of the pixels above and to the left of \( x, y \), defined as [46]

\[
II(x, y) = \sum_{x' \leq x, y' \leq y} I(x, y)
\]
The image can be computed in one pass over the original image using the following pair of recurrences

\[ S(x, y) = S(x, y - 1) + I(x, y) \]  \hspace{1cm} (3)

\[ II(x, y) = II(x - 1, y) + S(x, y) \]  \hspace{1cm} (4)

where \( S(x, y) \) is the cumulative row sum, \( S(x, -1) = 0 \) and \( II(-1, y) = 0 \). Using the integral image, any rectangular sum can be computed in four array references, as illustrated in Figure 2.7. The use of integral images leads to enormous savings in computation for features at varying locations and scales.

![Figure 2.7](image)

Fig. 2.7. The sum of the pixels within rectangle \( D \) can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle \( A \). The value at location 2 is \( A + B \), at location 3 is \( A + C \), and at location 4 is \( A + B + C + D \). The sum within \( D \) can be computed as \((4+1) - (2+3)\). From Viola and Jones [46], © 2001 IEEE, with permission.

With the integral images, the intensity variation within a rectangle \( D \) of any size and any location can be computed efficiently; for example

\[ V_D = \sqrt{V \star \bar{V}} \]

where \( V = (4+1) - (2+3) \) is the sum within \( D \), and a simple intensity normalization can be done by dividing all the pixel values in the subwindow by the variation.

### 4.2 Constructing Weak Classifiers

As mentioned earlier, the AdaBoost learning procedure is aimed at learning a sequence of best weak classifiers to combine \( h_m(x) \) and the combining weights \( \alpha_m \) in Eq.(1). It solves the following three fundamental problems: (1) learning effective features from a large feature set; (2) constructing weak classifiers, each of which is based on one of the selected features; and (3) boosting the weak classifiers to construct a strong classifier.

AdaBoost assumes that a “weak learner” procedure is available. The task of the procedure is to select the most significant feature from a set of candidate features, given the current strong classifier learned thus far, and then construct the best weak classifier and combine it into the existing strong classifier. Here, the “significance” is with respect to some given criterion (see below).

In the case of discrete AdaBoost, the simplest type of weak classifiers is a “stump.” A stump is a single-node decision tree. When the feature is real-valued, a stump may be constructed by thresholding the value of the selected feature at a certain threshold value; when the feature
is discrete-valued, it may be obtained according to the discrete label of the feature. A more
general decision tree (with more than one node) composed of several stumps leads to a more
sophisticated weak classifier.

For discrete AdaBoost, a stump may be constructed in the following way. Assume that we
have constructed \( M - 1 \) weak classifiers \( \{ h_m(x) | m = 1, \ldots, M - 1 \} \) and we want to construct
\( h_M(x) \). The stump \( h_M(x) \in \{-1, +1\} \) is determined by comparing the selected feature \( z_{k^*}(x) \)
with a threshold \( \tau_{k^*} \) as follows

\[
\begin{align*}
  h_M(x) &= +1 \quad \text{if} \quad z_{k^*} > \tau_{k^*} \\
  &= -1 \quad \text{otherwise}
\end{align*}
\]

In this form, \( h_M(x) \) is determined by two parameters: the type of the scalar feature \( z_{k^*} \)
and the threshold \( \tau_{k^*} \). The two may be determined by some criterion, for example, (1) the minimum
weighted classification error, or (2) the lowest false alarm rate given a certain detection rate.

Supposing we want to minimize the weighted classification error with real-valued features,
then we can choose a threshold \( \tau_k \in \mathbb{R} \) for each feature \( z_k \) to minimize the corresponding
weighted error made by the stump with this feature; we then choose the best feature \( z_{k^*} \) among
all \( k \) that achieves the lowest weighted error.

Supposing that we want to achieve the lowest false alarm rate given a certain detection rate,
we can set a threshold \( \tau_k \) for each \( z_k \) so a specified detection rate (with respect to \( w^{M-1} \)) is
achieved by \( h_M(x) \) corresponding to a pair \( (z_k, \tau_k) \). Given this, the false alarm rate (also with
respect to \( w^{M-1} \)) due to this new \( h_M(x) \) can be calculated. The best pair \( (z_{k^*}, \tau_{k^*}) \) and hence
\( h_M(x) \) is the one that minimizes the false alarm rate.

There is still another parameter that can be tuned to balance between the detection rate and
the false alarm rate: The class label prediction \( \hat{y}(x) = \text{sign}[H_M(x)] \) is obtained by thresholding
the strong classifier \( H_M(x) \) at the default threshold value 0. However, it can be done as \( \hat{y}(x) = \text{sign}[H_M(x) - T_M] \) with another value \( T_M \), which can be tuned for the balance.

The form of Eq.(6) is for Discrete AdaBoost. In the case of real versions of AdaBoost, such
as RealBoost and LogitBoost, a weak classifier should be real-valued or output the class label
with a probability value. For the real-value type, a weak classifier may be constructed as the
log-likelihood ratio computed from the histograms of the feature value for the two classes. (See
the literature for more details [18, 19, 20]). For the latter, it may be a decision stump or tree
with probability values attached to the leaves [22].

### 4.3 Boosted Strong Classifier

AdaBoost learns a sequence of weak classifiers \( h_m \) and boosts them into a strong one \( H_M \)
effectively by minimizing the upper bound on classification error achieved by \( H_M \). The bound
can be derived as the following exponential loss function [33]

\[
J(H_M) = \sum_i e^{-y_i H_M(x_i)} = \sum_i e^{-y_i \sum_{m=1}^{M} \alpha_m h_m(x)}
\]

where \( i \) is the index for training examples. AdaBoost construct \( h_m(x) (m = 1, \ldots, M) \) by
stagewise minimization of Eq.(7). Given the current \( H_{M-1}(x) = \sum_{m=1}^{M-1} \alpha_m h_m(x) \), and the
newly learned weak classifier $h_M$, the best combining coefficient $\alpha_M$ for the new strong classifier $H_M(x) = H_{M-1}(x) + \alpha_M h_M(x)$ minimizes the cost

$$\alpha_M = \arg \min_{\alpha} J(H_{M-1}(x) + \alpha_m h_M(x))$$

(8)

The minimizer is

$$\alpha_M = \log \frac{1 - \epsilon_M}{\epsilon_M}$$

(9)

where $\epsilon_M$ is the weighted error rate

$$\epsilon_M = \sum_i w_i^{(M-1)} 1[\text{sign}(H_M(x_i)) \neq y_i]$$

(10)

where $1[C]$ is 1 if $C$ is true but 0 otherwise.

Each example is reweighted after an iteration i.e., $w_i^{(M-1)}$ is updated according to the classification performance of $H_M$:

$$w_i^{(M)}(x, y) = w_i^{(M-1)}(x, y) \exp(-y \alpha_M h_M(x))$$

$$= \exp(-y H_M(x))$$

(11)

which is used for calculating the weighted error or another cost for training the weak classifier in the next round. This way, a more difficult example is associated with a larger weight so it is emphasized more in the next round of learning. The algorithm is summarized in Figure 2.8.

0. (Input)
   (1) Training examples $Z = \{(x_1, y_1), \ldots, (x_N, y_N)\}$,
       where $N = a + b$; of which $a$ examples have $y_i = +1$
       and $b$ examples have $y_i = -1$.
   (2) The number $M$ of weak classifiers to be combined.

1. (Initialization)
   $w_i^{(0)} = \frac{1}{2a}$ for those examples with $y_i = +1$ or
   $w_i^{(0)} = \frac{1}{2b}$ for those examples with $y_i = -1$.

2. (Forward inclusion)
   For $m = 1, \ldots, M$:
   (1) Choose optimal $h_m$ to minimize the weighted error.
   (2) Choose $\alpha_m$ according to Eq. (9).
   (3) Update $w_i^{(m)} \leftarrow w_i^{(m)} \exp[-y_i \alpha_m h_m(x_i)]$ and
       normalize to $\sum_i w_i^{(m)} = 1$.

3. (Output)
   Classification function: $H_M(x)$ as in Eq.(1).
   Class label prediction: $\hat{y}(x) = \text{sign}[H_M(x)]$.

Fig. 2.8. AdaBoost learning algorithm.