The Hong Kong Polytechnic University

Department of Electronic and Information Engineering

Facial Image Analysis for Video Indexing and Retrieval

Tse Siu Hong

A thesis submitted in partial fulfilment of the requirements

for the degree of Master of Philosophy

August 2008

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____ (Signed)

Tse Siu Hong_____ (Name of student)

Abstract

The aim of this research is to investigate efficient schemes for facial image analysis in video retrieval and indexing. Statistics have shown that over 95% of the primary camera's subjects in videos are humans, therefore face analysis in videos can greatly benefit on video retrieval and indexing. Our research focuses on three areas: face detection, face recognition, and indexing. Some popular techniques and recent developments of the methods for both face detection and recognition are also reviewed.

In this project, we have proposed an effective template, namely Spatially Maximum Occurrence Template (SMOT), for face detection. This template is combined with a mixture of Gaussian models to verify whether an image region is a face or not. SMOT has a high representative power for faces, and can detect faces under various conditions.

We have also proposed an efficient method for face recognition. A simplified version of the Gabor wavelets (SGWs) has been devised for feature extraction. Gabor wavelets (GWs) have commonly been used for extracting local features which are insensitive to environmental factors, but extracting these features is computationally intensive. Simplified Gabor wavelets (SGWs) are therefore devised, and an efficient algorithm for extracting the features based on an integral image is proposed. These SGW features are then applied to face recognition. Experiments show that using SGWs can achieve a performance level similar to that using GWs, and the runtime for feature extraction using SGWs is 4.39 times faster than that of GWs implemented by using the fast Fourier transform.

An efficient indexing structure for searching face images in a large database has also been investigated and proposed. This indexing structure is formed by a number of vantage objects, which are constructed using the discriminative features extracted from Gabor wavelets. The training faces in a large database are ranked in order with reference to each of the vantage objects, so a ranked list is constructed for each vantage object. A query face image will also be ranked with respect to each vantage object, and those neighboring training faces to the query face in the respective ranked lists are selected to form a much smaller database, called a condensed database. Experiments show that a condensed database whose size is 25% of the original large database can be formed with a probability of 99.3% that the matched face to the query input exists in the condensed database. Then, a more computational and accurate recognition algorithm can be adopted in the condensed database without any degradation of the recognition accuracy.

Author's Publications

The following technical papers have been published or submitted for publication based on the result generated from this work.

Journal Paper

1. Wing-Pong Choi, Siu-Hong Tse, Kwok-Wai Wong, Kin-Man Lam, "Simplified Gabor wavelets for human face recognition," Pattern Recognition, vol. 41, no. 3, pp. 1186-1199, March 2008.

Conference Papers

1. Siu-Hong Tse, Kwok-Wai Wong and Kin-Man Lam, "Face Detection Using a Novel Template with Gaussian Mixture Model," Asia-Pacific Workshop on VIP 2005, pp. 42-47, 11-13 December 2005.
2. Siu-Hong Tse and Kin-Man Lam, "Efficient Face Recognition with a Large Database," 10th International Conference on Control, Automation, Robotics and Vision, pp. 944-949, 17-20 December 2008.

Acknowledgements

I would like to take this opportunity to express my sincere gratitude to my Chief Supervisor, Dr. Kenneth, K. M. Lam, from the Department of Electronic and Information Engineering of the Hong Kong Polytechnic University. He have guided me the direction of this project, provided his professional advice and encouraged me on my research. These help me to improve myself a lot.

I would also like to say thank you to all the members in the DSP Research Laboratory, especially Professor W. C. Siu, Dr. Cheung-Ming Lai, Dr. Xudong Xie, Mr. Kwok-Wai Wong, C. Cai, King-Hong Chung, Wing-Pong Choi, Chensheng Sun, Jiwei Hu, Y. Hu, Ms. W. Jiang, Xuejuan Gao and Dr. Guoping Qiu from the School of Computer Science of the University of Nottingham. They often encourage me and give me many supportive and contributive comments. Their encouragement and supportive comments helped me to overcome many difficulties during my research studies.

I would also like to thank to all academic staffs, technical staffs, general office and secretarial support staffs in Department of Electronic and Information Engineering, especially Dr Simon Hau, Dr. E. Jelenkovic, Miss Carol Yuen, Ms. Sandy Tong, Cora Au, Annie Li, Suki Chu, Catherine Ip, Catherine Liu and Mr. S. M. Tse. They provided me plenty of computers so that I can conduct my experiments well. I am thankful to the Centre for Multimedia Signal Processing of the Department of Electronic and Information Engineering for generous support over the past four years.

Last but not least, thanks for the patience and forbearance of my family, I can concentrate on my research. I appreciate their support and understanding.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

The objective of this chapter is to introduce the different existing techniques for facial image analysis. Some face-based techniques and their applications will be introduced. The originality and organization of this thesis will be addressed at the end of this chapter.

1.1 Motivation

In this information and multimedia era, it is absolutely crucial that an efficient tool is available for managing and retrieving video files. This type of effective tool, called content-based video retrieval, aims at assisting a human to retrieve a required video sequence within a database [168]. There are three major types in video search. In the first, the user knows that the targeted video sequence is in the database, and the user is able to describe the targeted sequence precisely during the retrieval process. A system indexing the video sequence by keywords or title should be sufficient in this case. In the second type, the user does not know whether the targeted video sequence is in the database. A precise search tool should be provided so that the user can find out if the target is in the database or not. In the final type, the user simply searches for a video related to some topics or events. The hierarchical search tool should be provided to guide the user. The user is also allowed to filter the responses of the search system, too.

Traditional video retrieval techniques usually apply low-level features or information for video shot partitioning, representation, classification and retrieval. Due to the use of low-level information, their performances and capabilities are limited. Actually, most of the primary camera's subjects in videos are humans. Therefore, significant amounts of effort have been spent on

recognizing human activities. The identification of humans and their activities has huge economic potential, but also poses many technological and scientific challenges. As the human face is always the most important object in videos, adopting an object-based approach based on the human face should greatly enhance the performance of video retrieval.

Recently, in fact, many governments and businesses are paying more and more attention to face-based techniques, which play an important role for many applications in different areas (Zhao et al., 2003) [7] such as information security, law enforcement and surveillance, video retrieval and management, etc.

Because of the uniqueness of the face of a person, it can be considered a form of personal identity for a log-in system for information security. Although there are other reliable methods of biometric personal identification – such as retinal or iris scan, hand geometry scan, and fingerprint analysis – all of these methods rely on a participant's cooperation. However, with the use of facial images, the participants need not be concerned with a log-in system, or even they are being surveyed under CCTV for law enforcement and surveillance. Therefore, security issues based on facial image analysis may become more user-friendly and non-intrusive for users.

The increasing attention given to face-based techniques has attracted researchers from different disciplines, such as image processing, pattern recognition, neural networks, computer vision, computer graphics and psychology [7]. They have investigated and proposed many different face-based techniques, such as face detection [1-2], face recognition [6-11], facial expression recognition [154], face tracking [155-159], 3D face analysis, etc.

In this thesis, we mainly consider the enabling techniques for video retrieval and indexing with facial image analysis. Face detection techniques will first be investigated to locate human faces in a video. Face recognition techniques will also be researched, for the key-frame extraction based on the targeted human faces. Further, a new indexing structure for face recognition in a large face database has also been developed.

1.2 Statements of Originality

The following contributions reported in this thesis are claimed to be original.

1. An effective template, namely Spatially Maximum Occurrence Template (SMOT), is proposed for human face detection. SMOT, which has a high representative power for faces, is combined with a mixture of Gaussian models to verify whether an image is a face or not. This method is able to detect faces under various conditions.
2. A simplified version of Gabor wavelets (SGWs) and an efficient algorithm for extracting the features based on an integral image are proposed for face recognition. The runtime for feature extraction using SGWs is, at most, 4.39 times faster than that with Gabor wavelets (GWs) implemented by using the fast Fourier transform (FFT). In addition, the performance of face recognition using SGWs is similar to that using GWs.
3. An efficient indexing structure, which is formed by a number of vantage objects, is proposed for searching in a huge database for face recognition. A much smaller database, called a condensed database, will be formed for each query input with this indexing structure. A more computational and accurate recognition algorithm can then be adopted in the condensed database. Without any degradation of the recognition accuracy, the time for face

recognition can be reduced, since the condensed database is much smaller than the original database.

1.3 Outline of the Thesis

This thesis is organized into six chapters, and each chapter is outlined as follows.

Chapter 2 describes the principles of face detection and face recognition. Some well-known techniques for face detection and recognition are reviewed, including Gabor Feature Extraction, Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Neural Network, and Adaboost.

In Chapter 3, we introduce an effective template, namely Spatially Maximum Occurrence Template (SMOT), for human face detection. With a high representative power for faces, SMOT is combined with a mixture of Gaussian models to verify whether an image is a face or not. This method is able to detect faces under various conditions, such as different facial expressions, poses, and illuminations in complex backgrounds.

Chapter 4 presents a simplified version of Gabor wavelets (SGWs) and an efficient algorithm for extracting the features based on an integral image for face recognition. Gabor wavelets (GWs) can effectively extract local and discriminating features for face recognition. However, due to the intensive computational requirement for extracting features using the traditional Gabor functions, it is impractical for real-time applications. Therefore, SGWs are proposed. The runtime required for feature extraction using SGWs is, at most, 4.39 times faster than that with GWs implemented by using the fast Fourier transform (FFT). Further, the performance of face recognition using SGWs is similar to that using GWs.

In Chapter 5, we propose an efficient indexing structure, which is formed by a number of vantage objects, for searching in a huge database for face recognition. The training faces in the database are ranked either in ascending or descending order with reference to each of the vantage objects, and hence each vantage object forms one ranked list or several ranked lists. A query face is ranked with reference to each vantage object, and is positioned in each of the ranked lists accordingly. Then the neighboring training faces to the query face in the ranked lists are selected to form a much smaller database, which is called a condensed database. Since the condensed database is much smaller than the original database, the time required to search for similar faces from a very large database can be greatly reduced without any degradation of recognition accuracy.

Finally, we conclude our work in Chapter 6, and some suggestions are provided there for further development.

Chapter 2: Literature Review

In this chapter, we will first introduce the basic concepts of face detection and face recognition, and then briefly describe the steps for face detection. Finally, we will review some recent well-known techniques for face detection and face recognition.

2.1 Review on Face Detection

2.1.1 Problem Statement

A lot of research on face detection has been conducted, since human face detection is the first important step in any face processing system, such as face recognition and face tracking. Most current face recognition techniques assume the availability of frontal faces of similar sizes [1]. Similarly, the initial face location is often assumed to be known in many face-tracking algorithms [155]. The detection schemes may be classified according to a cluttered or an uncluttered background in digital images [2]. For instance, crowd surveillance is associated with a cluttered or complex background, while passport identification has an uncluttered background. Finding human faces automatically in a cluttered background is a difficult and significant problem. Different approaches have been devised for the detection of human faces in gray-level images. These include approaches that are template-based [59-63], feature-based [51-58], neural network-based [134-136], example-based [60], and most often, a combination of all of these. The computational complexity of these methods is usually too great for real-time applications.

The goal of face detection is to determine whether or not there are any faces in an image [2]. If present, the image location and extent of each face should be returned. However, there are several factors that can make face detection become difficult. These include the pose of the faces, facial expression, orientation, imaging conditions, and the presence or absence of facial features such as beards or glasses. The factors are described as follows:

1. Pose. The images of a face vary due to the relative camera-face pose, e.g. frontal, 45 degree, profile and upside down. Some facial features, such as the eyes or the nose, may become partially or wholly occluded.

2. Presence or absence of structural components. Facial features such as beards, mustaches and glasses may or may not be present. There is a great deal of variability among the shape, color and size of these components.

3. Facial expression. The appearance of faces is affected directly by different facial expressions, such as laughing, sad, and crying.

4. Occlusion. A face may be partially occluded by other persons or objects, especially in a group of people.

5. Image orientation. Face images vary with different rotations about the camera's optical axis.

6. Imaging conditions. The appearance of the face is affected by different imaging conditions, such as lighting and camera characteristics.

Many problems are closely related to face detection [2]. Face localization, a simplified detection problem, is used to determine the image position of a single face. The assumption is that an input image contains only one face. Facial feature detection is used to detect the presence and location of features, such as eyes, eyebrows, mouth, lips, nose, ears, etc. Face recognition or face

identification is used to recognize the identity of a query face image in a stored face database. Facial expression recognition is used to identify the affective states (happy, sad, disgusted, etc.) of humans. Face authentication is used to verify the claim of the identity of an individual in an input image. Face tracking continuously estimates the location and possibly the orientation of a face in an image sequence or a video in real time. Many of these face processing techniques require frontal faces of similar size as the input. Therefore, face detection is the first step in any automated system.

One of the most important problems is how to evaluate the performance of the proposed detection methods [2]. Most of the researchers compare their proposed methods with the detection rate and false alarm rate. There are also many other metrics used to evaluate face detection methods, such as the learning time, the execution time, the number of samples required in training, and the ratio between detection rates and false alarms. However, it becomes confusing if there are different definitions for detection rates and false alarms. In this thesis, the detection rate is defined as the ratio between the number of faces correctly detected and the number of faces determined by a human. An image region is identified as a face by a classifier if the image region covers more than a certain percentage of a face in the image. Often, the image region identified as a face must contain all the visible parts of the eyes and mouth. If an image region is declared to be a face, but it is not, this is called a false positive or a false alarm. If an image region is a face but the classifier fails to detect it, this is a false negative, which results in a lower detection rate.

There have been more than 150 approaches reported for face detection [2], and there are even more now. Most research treats face detection as a computer

vision research task of object recognition, especially a two-class recognition problem. An image region will be classified as being a “face” or a “nonface”. However, the “face” class contains large within-class variability, so some of the detection techniques need a large set of training images.

Face detection is also a pattern recognition problem. A raw or filtered image will be the input of a pattern classifier. A large number of pixels in training images will cause an extremely high dimension of the feature space. To deal with these high-dimensional training samples, multimodal distribution functions can be used to characterize the face and nonface classes. The decision boundaries are nonlinear, and the classifiers should also be able to extrapolate from a modest number of training samples.

2.1.2 Steps of Face Detection

There are several steps in a face detection system using supervised learning [66], as shown in Figure 2.1. They are: pre-processing, feature extraction, feature selection (optional), classification and post-processing.

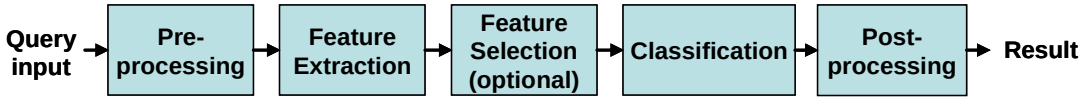


Figure 2.1 Main step involved in building a face detection system.

Pre-processing is the first step, which aims to improve the input image or standardize the image condition so that the chances for successful detection increase. For example, in order to obtain the face candidates in a query image, some detectors scan across the query image at multiple scales, orientations and locations. The scanning window is often a square. The step sizes of the scale factor, orientation factor and shift factor of the scanning window are determined by the researchers. A large step size can speed up the detection process but also

lower the accuracy level. A small step size will be more time consuming, but may result in greater accuracy. After the face candidates have been obtained, some normalization processes will be performed on them. For example, the face candidates will be normalized with respect to size and orientation. Histogram equalization will then be done to compensate for light variations. Noise reduction may also be applied.

The second step is feature extraction, where some features extracted from a face image are able to represent the main characteristics of the face, so that the classifier in the step that follows can work faster and more accurately. Some researchers use visual features, such as face color [13-19] or shape [20-21] to locate possible face candidates. Some others consider the local characteristics of faces and use receptive fields [25-26], Haar wavelets [27], Gabor wavelets [28-46] or wavelets [47-50] to extract the local features. Others treat a face as a template [59-63], or as a pattern or a vector, and transform it into another feature space [64-113], in which only a few dimensions are required to represent a face.

The third step is feature selection, where the detector will select the most representative features and neglect all the useless features. Recently, research has put more and more emphasis on feature selection methods, although not many detection methods contain the feature selection process. There have also been many approaches published for feature selection. One of the simplest ways to select the most representative features is to choose the features with a high level of mutual information [126-133]. Another effective approach is to use learning-based methods, e.g. Adaboost [144-148], to determine the highly representative features.

The next step is classification, where the classifier assigns the face candidates to its correct category, i.e. either faces or non-faces. The classifier needs to be trained before performing classification. One of the simplest ways to classify is to compute the similarity between the face candidates and the training samples, and then make a decision by thresholding. There are different distance measures available to compute the similarity, e.g. Euclidean distance, correlation or Mahalanobis distance, etc. Other more advanced methods for classification include the nearest-neighbor rule (NNR), the Bayesian approach [119-121], and neural networks [134-136].

The last step is post-processing, where algorithms to reduce the number of false positives and manage the overlapping regions are applied. Sometimes, two or more overlapped candidates in the same image are classified as a face. This means that they may be the same face. The algorithm to manage the overlapped regions is applied to avoid duplicating results [134] [136].

2.1.3 Recent Techniques for Face Detection

Former researchers have classified face detection methods into different categories. According to Hjelmas and Low [1], face detection methods can be classified into two categories: a feature-based approach and an image-based approach. Yang, Kriegman and Ahuja [2] classify face detection methods into four categories: knowledge-based methods, feature invariant approaches, template matching methods, and appearance-based methods. In this thesis, the face detection methods to be reviewed are classified into two categories: feature-based methods and appearance-based (or image-based) methods.

2.1.3.1 Feature-based methods

These methods are the combination of knowledge-based methods, feature invariant approaches and template matching methods in Yang et al. Since these three methods aim to make use of some facial features that are invariant in terms of pose, viewpoint and lighting conditions, the feature-based methods often put great emphasis on how to extract the invariant features, such as skin color, face contour, edges, texture and shape. However, some models or classifiers have to be applied with these methods before making a final judgment. The problem with these methods is that the features can be corrupted due to illumination, noise, and occlusion [2]. In this thesis, we will review some of the recent and most important feature-based methods.

2.1.3.1.1 Color

Color is a powerful fundamental feature for extracting the skin regions efficiently [13]. It is invariant to pose, viewpoint, scale and orientation of faces, and robust to cluttered backgrounds. Therefore, color-image segmentation is often the first step in the process of face detection in complex scenes. Once skin regions are extracted, other features and techniques can be applied in the skin regions to locate the face candidates.

Although skin color varies under different lighting conditions, some research has found that, under different illuminations, the chrominance components (Cb and Cr) of the facial skin are distributed with a certain range [19]. A color compensation scheme can also be applied to compensate for extreme lighting conditions [18]. Greenspan, Goldberger and Eshet [13] applied a mixture of Gaussian models to represent the face color in the normalized r-g

color model. Hsu, Abdel-Mottaleb and Jain [14] proposed a detection algorithm based on a transformed color space. A cluster of skin colors in the color space is formed, and is used to locate possible face candidates, which are then verified as true faces or not by the detection of facial features such as eyes, mouth, face boundary and the triangular relationship between the eyes and mouth.

2.1.3.1.2 Facial Features

To locate the different facial features, various methods rely on the fact that almost every face has bilateral symmetry, with the two eyebrows, two eyes, one nose and one mouth having a very similar layout. Some methods that have relied on these facial features are reviewed, as follows.

Yang and Huang [51] proposed a hierarchical knowledge-based method to construct the face detection system, which consists of three levels of rules. At levels 1 and 2, mosaic images are used to find the possible face candidates with all possible sizes and locations. At level 3, the rules based on details of facial features are applied to the possible face candidates to make a final decision. Kotropoulos and Pitas [52] extended the work of Yang and Huang [51] by using mosaic images to find the face candidates, and then located the positions of the eyes and mouth by using horizontal and vertical profiles. Lin and Fan [53] proposed a triangle-based approach for face detection. They found that the facial features such as eyes, ear holes and mouth form a triangle. Then they used a weighting-mask function for verifying a face.

Lam and Yan [55] proposed methods for locating the respective facial features, and new models for their representation. The corners of the respective facial features are detected, and are used to represent the respective features. The

positions and shapes of the features can be estimated accurately based on the locations of the different corners.

One method for face detection [15] [57] is to detect possible positions of the eyes, which exhibit as valleys in the image space. Instead of searching the whole image space for human faces, only those valley positions satisfying some features of the eyes are considered. Two possible eye candidates with similar features are grouped to form a possible face candidate, which is then further verified as a human face by measuring its corresponding symmetry and its similarity to a human face template. Wong and Lam [15] [19] applied face color detection before valley detection, in order to restrict the search space for possible eye candidates to skin color regions. This can speed up the face detection process.

2.1.3.2 Appearance-based methods

Face detection is treated as a general recognition problem. The methods often put great emphasis on feature selection and the classifier. After feature extraction on face candidates, or even just considering all the spatial pixels of face candidates as features, the features are treated as vectors or arrays. Using pattern recognition theory, the models or templates are learned from a set of training vectors, which carry the most representative variability of a face. These learned models are then used for detection. Appearance-based methods include subspace methods, statistical approaches and neural network-based methods.

2.1.3.2.1 Subspace Methods

Subspace methods consider a feature space as a linear combination of a sub-set of bases. Training or input images are projected into the subspace, where the new space removes the useless information and produces image features

which are more representative or discriminative. Subspace methods are effective and computationally efficient, and very easy to implement. The most popular subspace methods include principal component analysis (PCA) [64-66], linear discriminant analysis (LDA) [67-75], independent component analysis (ICA) [108-113], locally linear embedding (LLE) [99-100], and locality preserving projection (LPP) [101-104]. LDA, ICA, LLE and LPP are often employed for face recognition; while PCA has been used for both face detection and recognition. We will review PCA in the following section, while the other subspace methods will be reviewed in Section 2.2.

Principal Component Analysis (PCA)

PCA [64-66], also known as Karhunen-Loeve transform, has been widely used for both face detection and recognition because of its low computational complexity and high representation ability. PCA is also a good orthogonal linear transformation for dimensionality reduction and data compression. It aims to search a set of projection axes that best represent the data. The projection axes, which are orthogonal to each other, are treated as the principal components of the data. Each data sample can be decomposed as a linear combination of these principal components. The first projection axis lies in the direction such that the data projected onto this projection axis have the greatest variance. The first projection axis is considered as the most representative axis to the data. The second projection axis lies in the direction such that the data projected onto this projection axis have the second greatest variance, and so on. Those projection axes near the end are considered as the least representative, so they will often be discarded for the purpose of dimensionality reduction or data compression. If the training vectors are faces, the projection axes are then called eigenfaces. If a face

image is projected onto the face space spanned by the eigenfaces and is then reconstructed, the reconstructed image will be similar to the original face image. However, if the image is not a face, the reconstructed image will appear as very different from the original image. Therefore, the distance of an image from the face space can be computed to determine whether the image is a face or not.

Suppose there are N $H \times W$ gray-scale training images, where H and W are the height and width of the images. Each 2D pixel array is represented as a 1D face vector by concatenating the pixel values row by row in sequence from top to bottom. The face vectors are denoted as $\boldsymbol{x}_i$, where $i = 1, 2, \ldots, N$, and the dimension of each face vector $\boldsymbol{x}_i$ is $D = H \times W$. The average vector $\boldsymbol{\mu}$, the demeaned vector $\boldsymbol{a}_i$, and the covariance matrix $\boldsymbol{C}$ are defined respectively as follows:

$$\mu = \frac{1}{N}\sum_{n=1}^{N} x_i \text{ , } a_i = x_i - \mu \text{ and } C = AA^T \qquad \text{(2.1), (2.2), (2.3)}$$

where $\boldsymbol{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N]$. The eigenvectors of the covariance matrix $\boldsymbol{C}$ are then computed; these represent the principal components of the training vectors. These eigenvectors are ranked in a descending order according to their corresponding eigenvalues. However, the dimension of $\boldsymbol{C}$ is $D \times D$; determining the corresponding eigenvectors and eigenvalues from this huge matrix is an intractable task (Turk and Pentland, 1991) [64]. Turk and Pentland (1991) [64] have proposed a computationally feasible method to solve this problem. Since the number of training samples is often much smaller than the dimension of the training samples (i.e. $N << D$) and there are only N–1 meaningful eigenvectors, so the eigenvectors of the matrix A^TA are solved. Hence, the dimension of the

matrix to be considered is $N \times N$ rather than $D \times D$. Consider $\boldsymbol{v}_i$ to be the eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$, i.e.

$$A^T A v_i = \lambda_i v_i , \tag{2.4}$$

where λ_i is the eigenvalue of the eigenvector $\boldsymbol{v}_i$. Pre-multiplying both sides by $\boldsymbol{A}$, we have

$$AA^T A v_i = \lambda_i A v_i . \tag{2.5}$$

It can be observed that $\boldsymbol{A}\boldsymbol{v}_i$ are the eigenvectors of $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{A}^T$, which are the eigenfaces and denoted as $\boldsymbol{w}_i$, i.e.

$$w_i = A v_i . \tag{2.6}$$

The first M ($M < N$) eigenfaces are selected to represent the training images. A new face image $\boldsymbol{q}$ is transformed into its eigenspace by a simple transformation, as follows:

$$y = W^T (q - \mu), \tag{2.7}$$

where $\boldsymbol{W} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M]$ and $\boldsymbol{y}$ is a weight vector that describes the contribution of each eigenface in representing the input face image.

This trick is not only used in PCA, but is also widely used for computing the eigenvectors of the between-class scatter matrix in Direct LDA, the null-spaced eigenvectors of the within-class scatter matrix in Discriminant Common Vectors (DCV), and the Neighborhood Discriminant Projection (NDP). This trick can reduce the computation time significantly, thereby making the algorithm more efficient.

PCA extracts the most representative features of face images and can alleviate the variations caused by local components, such as facial expressions, occlusion, and presence or absence of beards and glasses. However, the

performance of PCA will be degraded greatly by the variations caused by global components, such as lighting, face orientations, etc.

2.1.3.2.2 Statistical Approaches

Face detection systems using statistical approaches are based on the Bayes' decision rule [119-121], a mixture of Gaussian models, probabilistic models [114-118], and information theory [126-133], etc. For the probabilistic models, an image or a feature vector derived from an image is viewed as a random variable $\boldsymbol{x}$ [2], and this random variable is characterized as a face or a non-face by the class-conditional density functions $p(x|\text{face})$ and $p(\boldsymbol{x}|\text{non-face})$. However, the dimensionality of the random variable $\boldsymbol{x}$ is often too high to implement classification directly. Therefore, the dimension of the input vectors is reduced at the beginning.

Schneiderman and Kanade [114-116] have developed two face detectors based on the Bayes' decision rule. An image is classified as a face if the likelihood ratio of $P(\text{image}|\text{object})$ to $P(\text{image}|\text{non-object})$ is greater than the likelihood ratio of $P(\text{non-object})$ to $P(\text{object})$. Moghaddam and Pentland [119] proposed an object representation system using a probabilistic framework. The representative features of faces are first extracted by PCA. With the assumption that a face space has a uniform density, maximum likelihood detectors for face detection and facial feature detection were developed. Sung and Poggio [60] proposed a distribution-based system for face detection, which uses a mixture of Gaussian models to describe the distribution of faces and non-faces. Then the normalized Mahalanobis distances between a face candidate and each cluster centroid, and the Euclidean distance between the face candidate and its projection onto each cluster, are calculated. Finally, a multilayer perceptron net

classifier, based on 12 pairs of distances, is used to verify the face candidates. Liu [121] has also presented a Bayesian Discriminating Features (BDF) method for multiple frontal face detection. The feature vectors of face candidates are a combination of the input candidate's image, its 1D Haar wavelet representation, and its amplitude projections. Then, statistical modeling is used to estimate the conditional probability density functions (PDFs) of the face and nonface classes. Finally, the Bayes classifier is applied for face detection.

2.1.3.2.3 Neural Network-based Approaches

Neural networks are a popular technique for pattern recognition. It is very feasible to train a system to capture the complex class conditional density of face patterns [2]. However, the network architecture must be extensively tuned so as to achieve an excellent performance. Neural network-based approaches include the multilayer perceptron (MLP) [60] [36], the probabilistic decision-based neural networks (PDBNN), the support vector machine (SVM) [137], the sparse network of winnows (SNoW) [138-140], and Adaboost [145-148].

Rowley et al. [134] presented a neural network-based upright frontal face detection system. After pre-processing through lighting correction and histogram equalization, face candidates with 20×20 pixels, which are the network input, will be divided into 26 receptive fields of different sizes. These 26 receptive fields represent the hidden units of the network. A single output of the network indicates whether or not the candidate is a face. The second part of this method is to eliminate the overlapped results. An arbitration strategy, such as logic operators, is used to improve the performance.

Viola and Jones [145] have proposed a popular and fast face detection method. It is a machine-learning approach for visual object detection, which is

capable of processing images extremely rapidly and can achieve high detection rates. In this approach, the “Integral Image” is first introduced, which allows the features used by the detector to be computed efficiently. The features used are a simplified form of the Haar basis functions. Even for a small image, the number of Haar-like features can be very large. A machine-learning algorithm, namely Adaboost [143-144], is used to train the classifier. It selects a small number of critical visual features from a large set of the features to form extremely efficient but weak classifiers. Finally, these weak classifiers are cascaded to form a complex and strong classifier, which allows the background regions of an image to be discarded quickly while spending more computation time on promising, object-like regions. The final classifier only uses a few hundred Haar-like features. The MIT+CMU test set is used, which contains 130 images with 507 faces. The detection rate is 93.9% when the number of false alarms is 167. In real-time applications, the detector runs at 15 frames per second, without resorting to image differencing or skin-color detection. Further extensions of this technique have been proposed by Lienhart and Maydt [146], and by Ma and Ding [147].

2.2 Review of Face Recognition

2.2.1 Problem Statement

Many face recognition approaches have been proposed during the past half decade. It is not difficult for humans to recognize a person, and in fact we perform face recognition many times every day. However, it is a challenging and interesting task to “tell” computers to perform face recognition. Face recognition has attracted many researchers from the fields of psychology, pattern recognition,

neural networks, computer vision, and computer graphics [7]. Moreover, due to a wide range of commercial and law-enforcement applications, such as logon systems and CCTV control, a fully automatic face recognition system is in great demand.

An automatic face recognition system comprises three main parts: face detection, feature extraction, and face recognition [7]. Face detection is performed first on the query image or video to obtain the locations and information about faces. The features used in face detection can be also used in face recognition, or other, more suitable features may also be considered. Finally, the target faces will be identified. In a complete face recognition system, inaccurate performance in face detection or facial-feature detection will degrade the performance of face recognition. Therefore, both face and facial-feature detection are very important in a face recognition system. In general, most researchers simply assume the availability of frontal faces of similar sizes for face recognition [1]. Actually, faces are often required to be rotated and scaled to align the centers of the eyes.

Face recognition is used to identify or verify one or more persons in a given image or video using a stored database of faces [7]. Similar to face detection, the performance of face recognition is affected by several factors, such as the pose, facial orientation, facial expression, lighting conditions, the presence or absence of beards or glasses, and occlusion. A practical face recognition technique needs to be robust to these variations.

A face recognition algorithm usually comprises the following four steps: pre-processing, feature extraction, feature selection, and classification. Different to face detection, post-processing is not required in face recognition, since face

detection needs to perform overlap elimination, but face recognition does not. The other steps are the same as the steps for face detection, as described in Chapter 2.1.2. Although the basic visual features, such as the edges and shapes, and the facial features, such as the eyes and nose, can be used in feature extraction, researchers have tended to mostly utilize the local characteristics, such as receptive fields, Haar wavelets [27], Gabor wavelets [28-46] and wavelets [47-50], over the last few years. However, the dimensionality of these features can be extremely high. Particular techniques are required to select the most discriminative features and to remove the useless features. The subspace approach may be a solution, since it can be a process combining both feature extraction and feature selection. Examples of subspace methods include PCA, LDA, ICA, LLE and LPP. Based on these methods, some advanced subspace methods have also been developed, including kernel subspace methods, matrix-based subspace methods (2DPCA, 2DLDA) [92-98], and some combined subspace methods (such as neighborhood discriminant projection, which combines LDA and LPP [91]). With the most discriminative features, even using a simple classifier, such as the nearest-neighbor classifier, can still achieve a satisfactory performance in classification. Therefore, researchers in face recognition have concentrated mainly on the techniques for feature extraction and feature selection in the last few years.

The level of accuracy and the computational complexity become challenging when the face recognition algorithms are applied to a very large database. In the training-based algorithms, the performance is greatly affected by two training parameters, which are the number of individuals and the number of training samples per individual. The intrinsic difference, which discriminates the

identity of different faces, is not obvious to the face recognition algorithm when the number of training samples per individual is small. It results in a poor performance level. This poor performance will be magnified if the number of distinct individuals is large. Moreover, more runtime is required for face recognition in a large database. Therefore, an efficient way to search for a face is essential.

2.2.2 Recent Techniques for Face Recognition

Face recognition methods can be classified into two categories: feature-based methods and appearance-based methods [7] [11]. Feature-based methods extract the local facial features such as the eyes, nose, and mouth, whose locations and local statistics are fed into a structural classifier. One very popular feature-based method is elastic bunch graph matching [31]. Appearance-based methods treat the whole face as the raw input to a recognition system. Appearance-based methods contain linear and non-linear subspace methods, as well as methods using probabilistic models and neural network. Recently, many researchers working on face recognition have been attracted by linear subspace methods and by methods using probabilistic models, especially the Bayesian method. This may be due to the easy implementation and good performance of these methods. To obtain more information on the local characteristics of faces, many researchers extract the local features of faces, such as the receptive fields, Haar wavelets, or Gabor wavelets, as training inputs for a recognition system [2] [133]. In this chapter, we will review the techniques of Gabor feature extraction, linear subspace methods, the Bayesian method, and face recognition in a large database. We will also briefly introduce LDA and its modified versions in the section on linear subspace methods.

2.2.2.1 Gabor Feature Extraction

Gabor wavelets are similar to the 2D receptive field profiles of the mammalian cortical simple cells. They exhibit the desirable characteristics of spatial localization and orientation selectivity, as well as spatial frequency characteristics [28]. The Gabor features are invariant to translation, scale and rotation [80]. Therefore, they have been widely used for feature extraction in face recognition.

In the spatial domain, a Gabor wavelet is a Gaussian function modulated by a complex exponential, which can be defined as follows:

$$\psi_{\omega,\theta}(x,y)=\frac{1}{2\pi\sigma^2}\exp\left(\frac{x_1^{\,2}+y_1^{\,2}}{2\sigma^2}\right)\left[\exp\left(i\omega x\cos\theta+i\omega y\sin\theta\right)-\exp\left(-\frac{\omega\sigma^2}{2}\right)\right], \quad (2.8)$$

where $\begin{cases} x_1 = x\cos\theta + y\sin\theta, \\ y_1 = -x\sin\theta + y\cos\theta. \end{cases}$

(x, y) denotes the pixel position in the spatial domain, ω is the radial centre frequency of the complex exponential, θ is the orientation of the Gabor wavelets, and σ is the standard deviation of the Gaussian function. The image features $Y_{\omega,\theta}(x,y)$ are extracted by convolving the image $I(x, y)$ with the filters $\psi_{\omega,\theta}$, as shown below.

$$Y_{\omega,\theta}(x,y)=I(x,y)\otimes\psi_{\omega,\theta}(x,y). \quad (2.9)$$

The convolution can be implemented efficiently by the fast Fourier transform (FFT) to reduce the computation required for feature extraction, but the computation is still intensive. Many researchers consider only the magnitude $G_{\omega,\theta}(x,y)$ of the output as Gabor representations, i.e. $G_{\omega,\theta}(x,y) = |Y_{\omega,\theta}(x,y)|$. A high-dimensional vector for face recognition G is formed by concatenating the magnitudes of the Gabor representations.

Figure 2.3 shows the magnitudes of the Gabor representations of the human face in Figure 2.2. Five different scales and eight different orientations are selected for the Gabor wavelets, i.e.

$$\theta = \frac{\pi}{8} p \text{ and } \omega = \frac{\pi}{2\left(\sqrt{2}\right)^q}, \text{ where } p = 0,\ldots,7, \text{ and } q = 0,...,4. \qquad (2.10)$$

$$G = \left[G^T_{\omega_0,\theta_0}, G^T_{\omega_0,\theta_1}, \ldots, G^T_{\omega_4,\theta_7}\right]. \qquad (2.11)$$

Hence, at each pixel position, the magnitudes of the Gabor representations are concatenated to form a high-dimensional vector of dimension 40 for face recognition. In addition, we set $\sigma = \pi/\omega$.

Figure 2.2 A human face image of size 64×64.

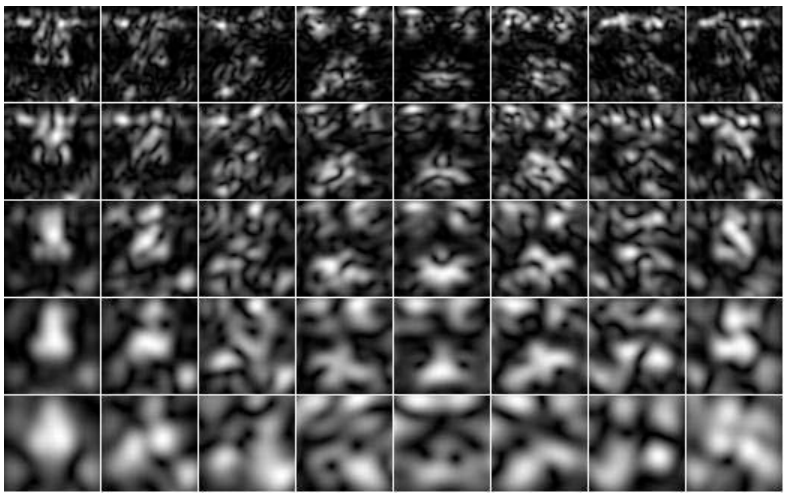


Figure 2.3 The magnitudes of the Gabor representations with 5 center frequencies and 8 orientations. The frequencies are π/2, $\sqrt{2}$ π/4, π/4, $\sqrt{2}$ π/8 and π/8 from the top to bottom row, respectively. The orientations are from 0 to 7π/8 in a step size of π/8, from the left to right column, respectively.

2.2.2.2 Linear subspace methods

Linear subspace methods include principal component analysis (PCA) [64-66], linear discriminat analysis (LDA) [67-75], independent component analysis (ICA) [108-113], locally linear embedding (LLE) [99-100], locality preserving projections (LPP) [101-104], etc. Nearly all the linear subspace methods, except ICA, assume that samples in each class are distributed in a Gaussian model.

PCA aims to search the projection axes to best represent the training samples. It is effective for dimensionality reduction and data compression, but not good for recognition, since PCA, also called eigenfaces, does not utilize the class information. Therefore, some researchers consider that PCA is just a technique for dimensionality reduction before applying some other subspace methods.

LDA utilizes the class information in training to search the projection axes to best discriminate the classes; therefore a high recognition rate can be achieved. This has been proven by many experimental results based on LDA. However, a sufficient number of samples per class is needed in training for LDA; otherwise the performance of LDA may be even worse than that of PCA [72]. Fisherfaces [70], which applies LDA in the PCA space, is an extension of LDA. Both PCA and LDA estimate the global statistics, but fail to discover the underlying local structure.

LLE [99-100] and LPP [101-104] are able to preserve the local information by incorporating the neighborhood information of a data set. A face subspace can be obtained that best detects the essential face manifold structure. Laplacianface [103], which applies LPP in the PCA space, is an extension of LPP. However, both LEE and LPP are not able to separate different classes well,

since they do not use the class information. Also, both are unsupervised methods, which are not quite efficient enough for recognition tasks.

ICA [108-113], which is considered a generalization of PCA, learns higher-order dependencies in the training data in addition to second-order correlations. This linear non-orthogonal transform makes unknown linear mixtures of multi-dimensional random variables as statistically independent as possible [109]. The distribution of the components is designed to be non-Gaussian. Two architectures – statistically independent basis images and a factorial code representation – have been proposed based on ICA for face recognition tasks [110]. Independent subspace analysis (ISA) and topographic ICA (TICA) are extensions of ICA. However, Yang et al. [112] have pointed out that the pure ICA projection seems to have little effect on the performance of face recognition. The superior performance may be due to the data centering and whitening steps, which are not included in pure ICA.

2.2.2.2.1 Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis [70-75] is a popular and effective face recognition technique, which derives a projection basis that separates different classes as much as possible, and compresses the same classes as compactly as possible. Suppose there are C distinct persons and each person has N_i face images in the training database, where $i = 1, \ldots, C$. All face images are represented in 1D face column vectors x^i_j with dimension D, where $i = 1, \ldots, C$, and $j = 1, \ldots, N_i$. A face vector x^i_j represents the j^{th} face vector in the i^{th} class. The within-class scatter matrix S_W and the between-class scatter matrix S_B are defined as follows:

$$S_W = \sum_{i=1}^{C} \sum_{j=1}^{N_i} \left(x^i_j - \mu_i\right)\left(x^i_j - \mu_i\right)^T \text{, and} \qquad (2.12)$$

$$S_B = \sum_{i=1}^{C} (\mu_i - \mu)(\mu_i - \mu)^T \ , \qquad (2.13)$$

where μ is the mean of all samples, and μ_i is the mean of samples in the i^{th} class. The optimal discriminant vector v is computed by maximizing the following criterion:

$$F(v) = \frac{v^T S_B v}{v^T S_W v}. \qquad (2.14)$$

The set of generalized eigenvectors v_k of S_B and S_W corresponds to the M largest eigenvalues λ_k, where $k = 1, \ldots, M$, i.e.

$$S_B v_k = \lambda_i S_W v_k, \quad k = 1,2,...,M \ . \qquad (2.15)$$

If S_W is non-singular, the optimal discriminant vectors can be solved with the following equation.

$$\left(S_W^{-1} S_B\right) v_k = \lambda_i v_k, \quad k = 1,2,...,M \qquad (2.16)$$

There are, at most, $C - 1$ non-zero eigenvalues, and so the upper bound of M is $C - 1$. Solving the above equation is equal to performing simultaneous diagonalization on S_W and S_B [69], which performs whitening on S_W and applies PCA on S_B using the transformed data. The whitening step aims to make S_W have equal eigenvalues for uniform gain control.

The small sample size (sss) problem occurs whenever the number of samples is smaller than the dimensionality of the samples. S_W becomes singular, and the computation of S_w^{-1} becomes complex and difficult. Simultaneous diagonalization can be used to solve this problem since it avoids computing S_w^{-1}, but the computation for whitening S_W is still so large, which it is unacceptable. Many methods have been proposed to solve the sss problem. Some researchers have proposed to modify the Fisher's criterion function as follows:

$$F(v) = \frac{v^T S_B v}{v^T S_W v + v^T S_B v}. \qquad (2.17)$$

The solution for maximizing this modified function is to have the eigenvectors corresponding to the set of the largest eigenvalues of the matrix $(S_B + S_W)^{-1}S_B$. However, the eigenvectors of $(S_B + S_W)^{-1}S_B$ are still very difficult to compute due to the singularity problem [82].

Zhao et al. [78] have proposed a subspace-based LDA, which performs LDA in the PCA space. Since the dimension of the training samples in the PCA space decreases, the matrix S_w^{-1} is no longer singular, hence the sss problem seems to be solved. Liu and Wechsler [80] have extended this work and proposed the enhanced FLD models (EFM), which perform simultaneous diagonalization in the PCA space. Some researchers suspect that some important discriminative information may be lost in the PCA process, while others have disagreed [81].

Chen et al. [82] have proposed a novel method to solve the sss problem, which is known as the "null space method". This method utilizes the eigenvectors in the null space of S_W, and ensures that the Fisher criterion function can be maximized. However, computing the null-space eigenvectors of S_W also requires a large computation. Cevikalp et al. [89] have proposed the discriminative common vectors (DCV), which can solve the problem of large computation for the null-space eigenvectors. The trick by Turk and Pentland [64] for computing eigenfaces is used to compute the eigenvectors of S_W in the non-null space, and then the discriminative common vectors of different classes are obtained by computing the residual error between the original sample and its reconstruction. This is equivalent to projecting the samples into the null space of

S_W. Experimental results show that DCV can achieve greater accuracy and requires a shorter training and testing time, with lower storage requirements, than both the eigenfaces and Fisherfaces.

Direct LDA has been proposed by Yu and Yang [86] to solve the sss problem. They first considered diagonalizing S_B, rather than S_W. With Turk and Pentland's trick [64], the process of direct LDA is extremely fast. However, Gao and Davis [87] have shown that the direct LDA is not equivalent to the traditional subspace-based LDA when dealing with the sss problem. They pointed out that direct LDA completely ignores the common covariance S_W and purely depends on the class means for classification, which is a special case of LDA. Experiments have shown that direct LDA cannot achieve a better performance than subspace-based LDA [87] [89].

Another method to solve the sss problem is the matrix-based LDA, such as 2DLDA and 2DFLD, which constructs the within-class scatter matrix and between-class scatter matrix by just using the original image samples represented in the matrix form. The within-class scatter matrix is often non-singular, and thus avoids the sss problem. However, Zheng et al. [75] have shown that matrix-based LDA actually loses the covariance information between different local geometric structures, while the traditional vector-based LDA could preserve. Experiments also showed that the performance of matrix-based LDA is not always superior to that of vector-based LDA.

2.2.2.3 Bayesian Method

The Bayesian method converts a multi-class problem into a two-class problem in face recognition. One class is the intrapersonal variation Ω_I between multiple

images of the same individual, and the other is the extrapersonal variation Ω_E for different individuals. Both classes are assumed to be Gaussian distributed. Likelihood functions $P(\Delta|\Omega_I)$ and $P(\Delta|\Omega_E)$ were estimated for a given intensity difference Δ. Two faces are considered as belonging to the same class if $P(\Delta|\Omega_I) > P(\Delta|\Omega_E)$. With the aid of PCA [119], the image-difference space is decomposed into intrapersonal principal subspace F and its orthogonal complementary subspace $\bar{F}$, where the Mahalanobis distance in F, which is the distance-in-feature-space (DIFS), and the PCA residual error in $\bar{F}$, which is the distance-from-feature-space (DFFS), can both be easily computed for recognition.

Yang et al. [148] and Shen et al. [133] have extended this idea. The image features are extracted using Gabor wavelets, and the intrapersonal and extrapersonal features are constructed by the two Gabor feature differences. These two features are fitted into Adaboost by Yang et al. [148], and are used to assist the computation of the conditional mutual information by Shen et al. [133]. Finally, a set of the most discriminative features can be computed.

Although the Bayesian method uses the class information by computing the intrapersonal and extrapersonal variations, the intrinsic difference, which discriminates different face identities, is not compacted, and spreads over F and $\bar{F}$ [73]. This causes a high computational cost for computing the DFFS. In fact, it is also obvious that the computation can be extremely high when computing the extrapersonal differences between the samples from different classes.

2.2.2.4 Face Recognition in Large Databases

Recently, there has been little research into face recognition in large databases, especially for a large number of individuals with a small number of training

samples per individual. In a database with a small number of training samples per individual, it is difficult to extract the intrinsic features for face recognition from the training samples to discriminate the different individuals. A large number of distinct individuals in the database will make the performance even worse. Moreover, the time required for searching in a large database is also greater. Therefore, the challenge is to develop an efficient face recognition system with a high accuracy level for a small number of training samples per individual and a large number of individuals.

Yang et al. (2004) [148], Yang et al. (2005) [112] and Gao et al. (2006) [74] have applied their algorithms to the large FERET database, which contains 1196 individuals in the gallery set (fa set) and 1195 individuals in the probe set (fb set). There is only one image per individual in both sets. Yang et al. (2004) have extracted intrapersonal and extrapersonal Gabor features, and obtained a strong classifier using Adaboost [148]. The FERET database is used for testing, and a recognition rate of 95.2% can be achieved with 700 features selected. Yang et al. (2005) have performed comparisons between PCA and ICA using the FERET database [112]. The PCA baseline algorithm II can achieve a recognition rate of 81.66%. Gao et al. (2006) [74] have learned the most discriminative local features (MDLF) classifier by applying Adaboost to the intrapersonal and extrapersonal Gabor features, and learned the most discriminative global features (MDGF) classifier by using LDA on Gabor features. The method can achieve a 99% recognition rate based on the FERET database. Therefore, it seems that a high recognition rate can be achieved by using algorithms that extract the intrapersonal and extrapersonal features. However, the computational time for testing has not been provided by these papers, and is believed to be very lengthy.

Some researchers have combined several small databases to form a large database. Liu et al. (2003) [150] have proposed an improved line-based face recognition algorithm, which is evaluated using a large database by combining the FERET, ORL and Jilin University Computation Intelligence Laboratory (JLCI) face databases. There are 843 individuals and 4500 images in these training databases. Although this improved version is faster than the original one, the recognition performance is worse than for PCA. Lin et al. (2003) [151] have proposed an efficient human face indexing scheme using eigenfaces. Each face in the database is ranked according to its projection onto each of the eigenfaces. In testing, the corresponding faces which are located near the query in the respective ranked lists will be selected to form a small database, namely a condensed database. Since the processing time required to generate the condensed database is very small, and the condensed database is much smaller than the original one, therefore a more advanced algorithm can be applied to those selected face images in the condensed database for recognition. There are 523 distinct subjects, with one image per distinct subject in the database, which comprises the ORL, Yale, MIT, AR, BioID, UMIST, Bern and self-captured face databases. The experiments show that the size of the condensed database is only 25% of the original database, and the average runtime for producing this condensed database is less than 1 second.

There are many other face recognition algorithms, but the databases used are not large, i.e. the number of individuals is less than 500. The databases used in some papers are large simply because the number of images per individual is large. Table 2.1 shows the different numbers of individuals, the different numbers of images per individual, and the different numbers of all images in the

training database as well as the algorithms used, which are arranged according to decreasing numbers of training individuals. The number of training individuals used in those methods listed in Table 2.1 is more than 100.

Table 2.1 Different numbers of individuals, different numbers of images per individuals, and the numbers of all images in the training databases with different methods.

Methods	Number of individuals in the training database	Number of images per individuals in the training database	Number of images in training database
Yang et al. (2004), FERET [148]	1196	1	1196
Yang et al. (2005), FERET [112]	1196	1	1196
Gao et al. (2006), FERET [74]	1196	1	1196
Liu et al. (2003), combined database with FERET, ORL, JLCI [150]	843	-	4500
Lin et al. (2003), combined database with ORL, MIT, AR, BioID, UMIST, Bern and self-captured [151]	523	1	523
Bartlett et al. (2002), FERET [110]	425	1	425
Liu et al. (2000), FERET [79]	369	2	738
Zheng et al. (2007), FERET [75]	255	2 or 3	510 or 765
Lu et al. (2002), combined databased with ORL, Bern, Yale, Harvard, UMIST and Caucasians [152]	157	2 to 8	704
Chen et al. (2000) [82]	128	2, 3 or 6	256, 384 or 768

2.3 Conclusions

This chapter has described the principles of face detection and face recognition. We have also reviewed some recent techniques for face detection. In the feature-based approaches, the color and facial features are introduced. In the appearance-based methods, we have described the subspace-based approaches, especially the PCA, statistical approaches, and neural network-based approaches. For face recognition, we have reviewed Gabor feature extraction and linear subspace methods, especially the LDA and the Bayesian method. At the end of this chapter, we have also addressed the issues of face recognition in large databases. In the

following chapters, our proposed algorithms for face detection, facial feature extraction, and face recognition in a large database will be presented.

Chapter 3: Face Detection Using a Novel Template with Gaussian Mixture Model

3.1 Introduction

Detecting human faces in an image or a video scene is the first and an important step in facial image analysis. The detection schemes may be classified according to a cluttered or an uncluttered background in the digital images. For instance, crowd surveillance is associated with a cluttered or complex background, while passport identification has an uncluttered background. Finding human faces automatically in a cluttered background is a difficult and significant problem.

The goal of face detection is to determine whether or not there are any faces in an image. If faces are present, the face detection system should return the location and size of each face. However, faces are non-rigid, complex and multi-dimensional, and they may appear in arbitrary poses, different facial expressions and different imaging conditions, and with the presence or absence of facial features such as beards or glasses. To deal with these difficulties in face detection, many different approaches have been proposed for the detection of human faces. These include template-based, feature-based, neural network-based, example-based approaches and, more often, a combination of all of these. The computational complexity of these methods is usually too high for real-time applications.

In the feature-based approach, Wong and Lam (1999) [57] and Wong, Lam and Siu (2001) [58] employed eye detection to find the possible face

candidates. Then, the Genetic algorithm was applied to verify the possible face candidates with the eigenface to form the fitness function. Lin and Fan (2001) [53] proposed a triangle-based approach for face detection, which uses a weighting mask function for verifying a face.

Kuo, Husang and Lin (2002) [61] proposed a multi-resolution template-based method. Sung and Poggio (1998) [60] presented an example-based learning method for view-based face detection, which uses a mixture of Gaussian models to describe the distribution of faces and non-faces. Then the normalized Mahalanobis distances between the face candidate and each cluster centroid, and the Euclidean distance between the face candidate and its projection onto each cluster were calculated. A multilayer perceptron net classifier based on 12 pairs of distances was used to verify the faces.

Skin color has been applied to detect human faces. Greenspan, Goldberger and Eshet (2001) [13] applied a mixture of Gaussian models to represent the face color. Hsu, Abdel-Mottaleb and Jain (2002) [14] proposed a detection algorithm based on a transformed color space. A cluster of skin colors in the color space is formed and used to locate possible face candidates, which are then verified as true faces or not by detection of facial features such as eyes, mouth, face boundary and the triangular relationship between the eyes and mouth.

3.2 Our Face Detection Algorithm

Our face detection algorithm is a template-based approach, which uses a novel template to represent human faces. This template, namely Spatially Maximum Occurrence Template (SMOT), has more than one value at each pixel position. The values of the template are extracted from the same pixel position of a set of training images. Then, a possible face candidate is projected onto the

space formed by the SMOT, and the projection is compared to a Gaussian mixture model of human faces to determine whether it is a true face or not. Projecting an image to the SMOT space refers to representing the image using the corresponding nearest values of the SMOT. Before the abovementioned procedure, a possible face candidate must first be normalized to a specified size and processed by means of histogram equalization to reduce the effect of lighting.

To speed up the face detection process, skin color detection (Greenspan, Goldberger and Eshet 2001) [13] will first be applied to identify possible face regions. Eye detection (Wong, Lam and Siu 2001) [58] is then applied to find the possible eye candidates within the face regions. If the query is of a gray-level image, eye detection will be applied directly to the image. Figure 1 shows the structure of our face detection algorithm.

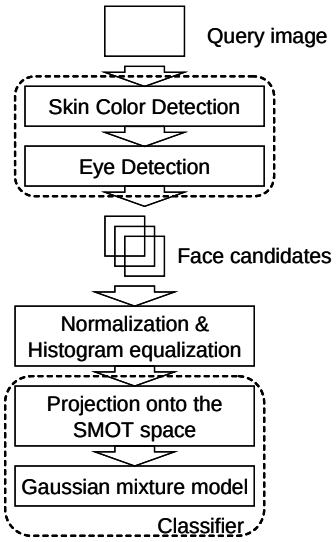


Figure 3.1 Structure of our face detection algorithm

The organization of this section is as follows. Sections 3.2.1 and 3.2.2 will describe the construction and structure of the SMOT based on a set of training face images. Section 3.2.3 will combine our proposed template with a

Gaussian mixture model for face detection. Section 3.3 will give the experimental results, and a conclusion will be drawn in Section 3.4.

3.2.1 Spatially Maximum Occurrence Template (SMOT)

It is desirable that the features to be used for face detection are simple and representative of faces so that the query images can be classified as faces or non-faces efficiently and accurately. In this section, we propose the Spatially Maximum Occurrence Template (SMOT) for representing human faces. Template matching methods have been widely used in facial image analysis because of their simplicity in terms of computational complexity.

The SMOT is generated based on a set of training faces, which is similar to the method in Sze, Lam and Qiu (2005) [160] for video shot representation. This template can form a face space such that an image can be projected onto the space to form a corresponding representation (called a SMOT representation). A distance measure is then applied to measure the difference between the candidate and its corresponding representation.

Figure 3.2 shows the construction of the SMOT based on a number of images. Suppose there are N training face images $f_{x,y}(n)$, where $n = 1, \ldots, N$, and (x, y) represent the pixel coordinates, and the image size is $W \times H$. The pixel values at the same position in the N training images can form a histogram. In every pixel position, k pixel values will be selected based on the peaks in the histogram. When compared to a query input in the testing process, the minimum distance measure is used at each pixel position. The distance measure is given as follows:

$$D_{\min} = \sum_{x=0}^{H-1} \sum_{y=0}^{W-1} \min\left(\left|q(x,y) - R(x,y,u)\right|\right), \text{ for } u = 1, \ldots, k. \qquad (3.1)$$

where $q(x, y)$ represents the pixel intensity of the query input at the position (x, y), $R(x, y, u)$ represents the pixel intensity of the u^{th} value at (x, y) in SMOT. SMOT is a very powerful representation of a series of images, because the number of different images that can be represented is k^{WH}. In addition, k special templates can be formed from the SMOT. The first peak values of all the positions can be grouped together to form the first peak template. The other k–1 peak templates are generated in the same manner.

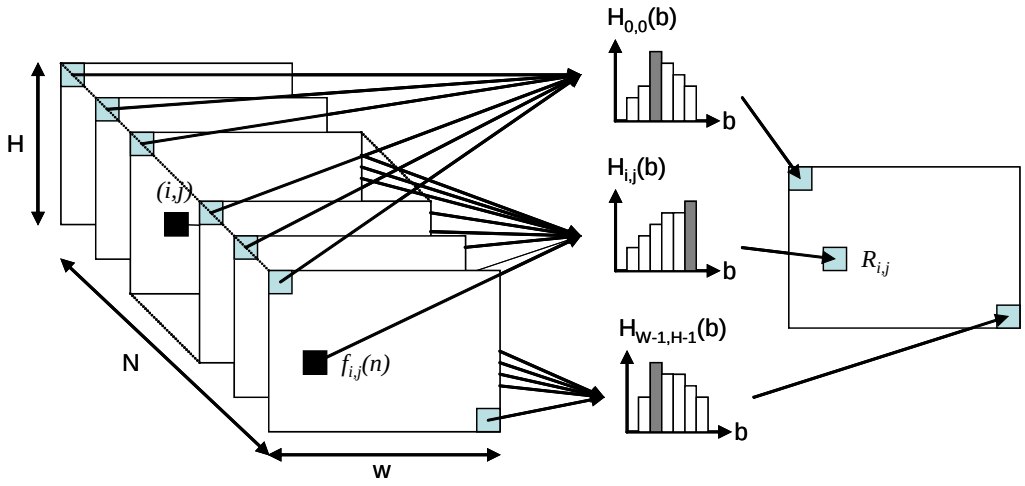


Figure 3.2 The construction of the SMOT based on a number of images

Figure 3.3 Some of the training face images

Figure 3.4 The four peak images from the SMOT

(a) (b)

Figure 3.5 (a) Query images and (b) the SMOT representations.

In Figure 3.2, a set of training face images is used to construct the SMOT, and some of these images are shown in Figure 3.3. Figure 3.4 illustrates the first four peak template represented by the SMOT. Figure 3.5(a) shows four examples of query images; two face images and two non-face images. Each pixel in a query image is represented by the nearest peak value at the same position in the SMOT. This process is called the projection of an image to the space represented by SMOT. Figure 3.5(b) shows the corresponding representations of the query images after projecting onto the SMOT, which are then compared to the corresponding query inputs.

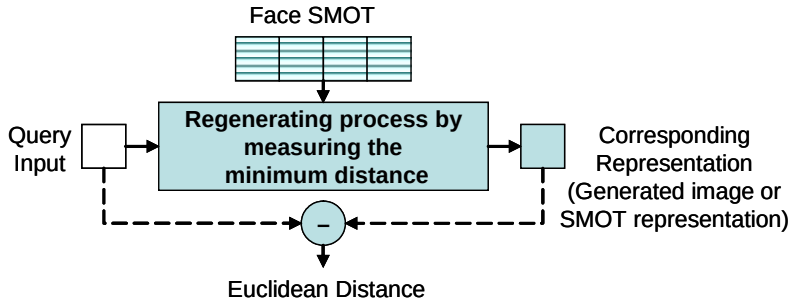


Figure 3.6 Comparison of an input image to the SMOT representation for face detection

In our algorithm, the Euclidean distance between the query image and the SMOT representation is used to verify if the input is a possible face candidate. Figure 3.6 depicts how to generate the SMOT representation and compute the difference between the input and the SMOT representation. In general, if the image is a face image, the distance between this image and the SMOT representation is small. Otherwise, the distance is large. The Euclidean distance of the query image is then compared to a threshold to determine whether the query image is a face or not. Since this verification involves pixel comparison only, its computational complexity is low.

3.2.2 The Use of Face and Non-face SMOTs

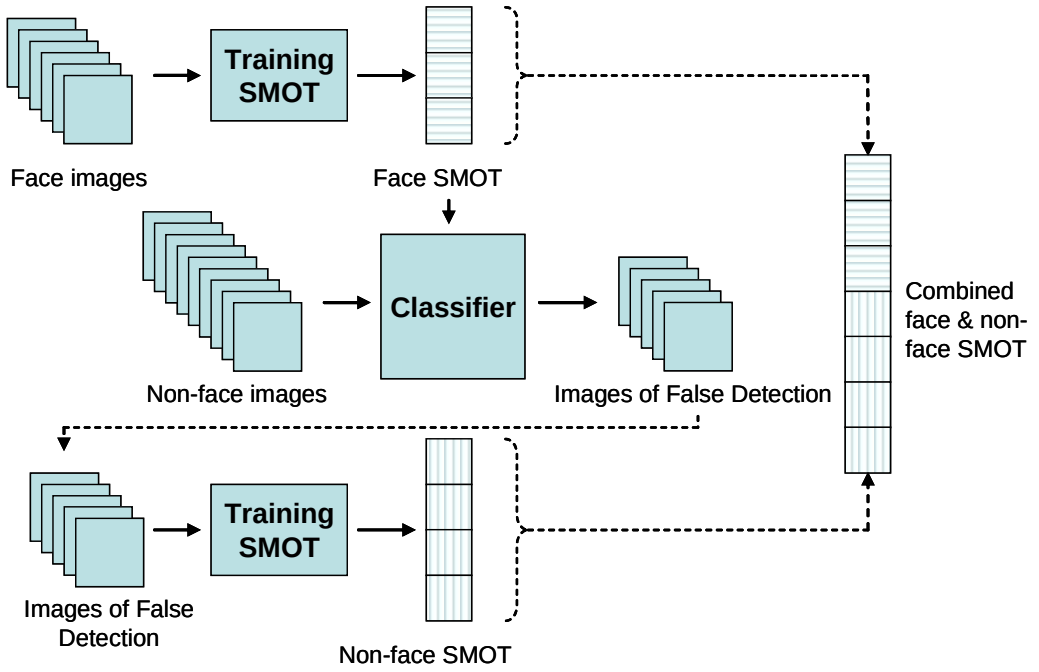


Figure 3.7 The construction of a combined face and non-face SMOT.

To improve the detection accuracy, in addition to the SMOT for face, a non-face SMOT is also constructed. The face SMOT is employed for face detection, and any false detection is collected and used to construct the non-face SMOT. These two SMOTs are then combined to form a combined SMOT for face detection. The idea behind this method is that a face candidate will be represented by those peak values from the face SMOT of the combined SMOT more faithfully, while a non-face query image will be better represented by the peaks from the non-face SMOT. Therefore, the SMOT representation, i.e. the generated image obtained by projecting onto the SMOT, of the query image will be similar to itself. Figure 3.7 illustrates the construction of the combined SMOT.

Figure 3.8 Some of the training images.

Figure 3.9 The combined SMOT with the first two images formed from the peaks of the face SMOT and the remaining five from the non-face SMOT.

(a) (b)

Figure 3.10 (a) Query images and (b) the corresponding SMOT representations.

Figure 3.8 shows some of the training face images and training non-face images. We select two peak values for the face templates and five for the non-face templates to construct a combined SMOT. The corresponding seven peak templates are shown in Figure 3.9. Four query images are shown in Figure 3.10(a); two faces and two non-faces. The query images are projected onto the SMOT, and the corresponding SMOT representations are generated and shown in Figure 3.10(b).

The Euclidean distance between the query input and the corresponding combined SMOT representation is always small irrespective of the input being face or non-face. In our algorithm, we consider the distance between the first peak template from the face SMOT and the combined SMOT representation. The reason is that the first peak template is constructed by the pixel values of the most frequently occurring faces. This face template has the most representative power to all the faces, so we use this template as a reference in measuring the distance. Figure 3.11 shows the measurement of the similarity of the query input to a true face based on the combined SMOT and the first peak template. If the input is a face, the generated image will be similar to itself, and the distance between the first face template in SMOT and the generated image is small. If the

input is a non-face, which will be mainly represented by the non-face SMOT, the generated image will be similar to a non-face and the distance between the first peak face template and the generated image will be large.

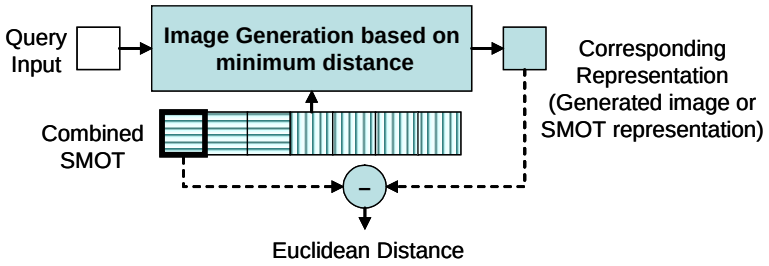


Figure 3.11 Measuring the similarity of the query input based on the combined SMOT.

3.2.3 Combined SMOT with Gaussian Mixture Model

Face detection using SMOT only considers the individual pixel positions, without including the relationship between the different pixels. This means that the global appearance of the faces is not considered in this pixel-wise method. To cope with this deficit, a probability model is employed to represent the global features of faces so as to supplement the pixel-wise operations in SMOT. We assume that the generated images of the training images using the SMOT are distributed as a mixture of Gaussian models. Therefore, the expectation maximization (EM) algorithm (Dempster, Laird, and Rubin 1997) [125] is used to generate a Gaussian Mixture Model (GMM), as described in Greenspan, Goldberger and Eshet (2001) [13], Sung and Poggio (1994) [60] and Moghaddam and Pentland (1997) [119], to represent the face and non-face clusters. In the detection, we use both the Euclidean distance based on the SMOT and the probability from the Gaussian mixture model to verify a face region.

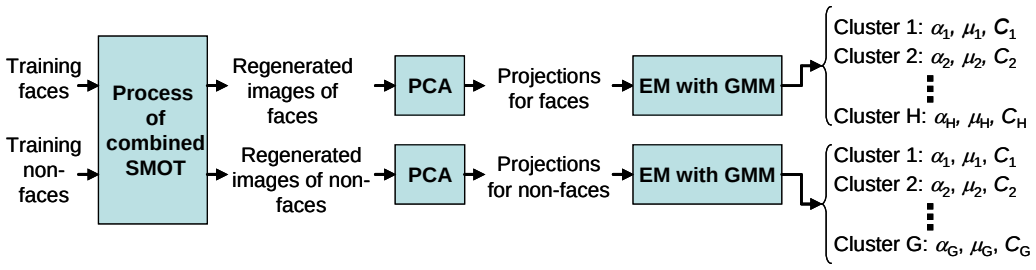


Figure 3.12 Training of the Gaussian mixture model.

Figure 3.12 shows the training process for the Gaussian mixture model, which consists of four steps. The first step is to construct the combined SMOT. The second step is to use the combined SMOT to construct the corresponding generated images for both faces and non-faces. The dimension of the generated images may be very large. To reduce the dimension of the generated images, principal component analysis (PCA), referred to Turk and Pentland (1991) [64], is applied in the third step. Therefore, a set of generated face images of low-dimension is produced. In the last step, the distribution of these low-dimensional face representations and non-face representations are described using a given number of Gaussian models (or clusters), which are constructed by the EM algorithm. In other words, a mixture of Gaussian models for faces and non-faces are produced. The prior probability α, mean μ and covariance C of each cluster can be obtained.

In the classification, a query image is projected onto the combined SMOT to generate the corresponding generated image. Then the Euclidean distance, E, between the generated image and the first peak face template of the combined SMOT is calculated. Next, the dimension of this generated image is reduced by means of PCA. This low-dimensional representation is used to calculate the maximum posterior probabilities of the face clusters, P_F, and that of the non-face

clusters, P_N, based on the Gaussian mixture model. A region is determined to be a face or non-face based on the following decision function:

$$region = \begin{cases} face, & \text{if } \lambda_1(1 - E/256) + \lambda_2 P_F > T_F, \\ non-face, & \text{otherwise.} \end{cases} \quad (3.2)$$

The input will be classified as a face if P_N is lower than the threshold T_N, and $\lambda_1(1 - E/256) + \lambda_2 P_F$ is larger than the threshold T_F, where λ_1 and λ_2 are the weights. The term $(1 - E/256)$ is normalized to a range between 0 and 1. Therefore, if the input is a face, this term should be large and the posterior probability for non-face clusters P_N should be small.

3.3 Experimental Results

We selected 4,000 frontal faces and 8,000 non-faces of size 50×50 to train the combined face and non-face SMOT. The training faces were selected from different databases including AR, FERET and ORL. Some faces are under uneven lighting, some have different facial expressions, and some have beards and glasses. We used 12 clusters to model the mixture of Gaussian distributions of faces, and 12 clusters for the non-faces.

To detect faces in an image, skin color detection is applied to the whole image to segment skin-color regions. Then eye detection is employed to identify possible eye candidates in the segmented regions, and possible face candidates are formed. Histogram equalization and size normalization are applied to these face candidates.

The testing databases used include the BERN, face95, face96, FERET, JAFFE, and Yale databases, and some images produced by our group. There are a total of 18,010 faces selected in these databases with different scales, illuminations, and facial expressions, and with complex background. Some of the

testing faces have a slight pose angle. Table 3.1 shows the characteristics of these databases.

Table 3.1 The characteristics of the different databases.

Database	Characteristics
AR	Varying luminance, different expressions
BERN	Different expressions, small pose angle
Cohn-kanade	Different expressions
Face95	Normal conditions, some varying luminance
Face96	Complex background
FERET	Different expressions, small pose angle
JAFFE	Different expressions
MIT	Normal conditions
ORL	Different expressions
Yale	Varying luminance, different expressions
Our database	Different expressions, small pose angle, complex background

Table 3.2 tabulates the detection rate and number of false detection rate based on different databases. The detection rate based on the respective databases is higher than 92.53% and the overall detection rate is 97.15%. The total number of false detection is 2189.

Our algorithm can detect the faces under different scales, facial expressions, and lighting conditions. The detected faces in the first and second rows (BERN, Cohn-kanade, JAFFE, Yale, Face95, Face96 and FERET databases) of Figure 3.13 show that our detector can detect faces with a small pose angle under different facial expressions, uneven illumination and complex backgrounds. Our system can also detect multiple faces with as shown in the third rows (our database). Figure 3.14 shows some examples of the missed faces, which are caused due to the strong reflected light from glasses. The disadvantage of this face detection method is that it is unable to detect small face images because our method is based on eye detection. If an image is very small, the eye positions may not be detected and so the face will be missed.

Table 3.2 Detection rate and number of false detections based on the different databases.

Dataset	No. of faces	No of detected faces	False detection	Detection rate (%)
BERN	300	283	4	94.33%
Cohn-kanade	8795	8707	691	99.00%
Face95	1432	1325	370	92.53%
Face96	3013	2826	650	93.79%
FERET	2466	2363	437	95.82%
JAFFE	213	213	0	100%
Yale	165	163	1	98.79%
Our database	1626	1616	36	99.38%
Total	**18010**	**17422**	**2189**	**97.15%**

The runtime required depends on the complexity of the query image. If the image is complex, many eye candidates will be detected and so many possible face candidates will be formed. On average, about one second is needed to process an image of size 176×144.

Figure 3.13 Detected faces from different databases: The first to the second rows are faces from the BERN, Cohn-kanade, JAFFE, Yale, Face95, Face96 and FERET databases, respectively. The others come from our database or other databases.

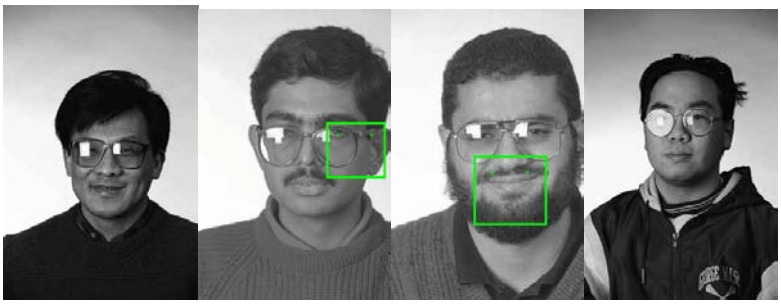


Figure 3.14 Missed faces due to the reflected light from glasses.

3.4 Conclusions

In this chapter, we have proposed an efficient face detection method using a novel template, namely Spatially Maximum Occurrence Template (SMOT), with the Gaussian Mixture Model. Our algorithm first uses skin-color detection and eye detection to find the possible face candidates. Then, the face candidates are normalized, histogram equalized and projected onto the SMOT space. The Euclidean distances between the images generated from SMOT and the SMOT first peak template are calculated. The dimension of the SMOT representation is subject to PCA to reduce its dimension. Finally, the low-dimensional SMOT projection is used to calculate the probabilities of the face clusters and non-face clusters based on a Gaussian mixture model. This face detection method can detect faces with different expressions and only a small pose angle, under varying illuminations and with complex backgrounds.

The advantage of this system is that it can detect faces under various conditions. The reason is that the SMOT has a very high representative power of face images and each cluster in the Gaussian mixture model can represent one condition of the faces, e.g. normal faces, faces with glasses, faces with open mouth, etc. However, it is impossible to collect all the conditions of the faces in the training dataset. Thus, in the detection, we use the SMOT to “normalize” the

query input, i.e. representing the query image by the corresponding nearest peak values from the SMOT, before computing the posterior probability. This classifies the faces under different conditions even though there are in sufficient face examples for training the Gaussian mixture model.

Chapter 4: Simplified Gabor Wavelets for Human Face Recognition

4.1 Introduction

The Gabor wavelet (GW) [29, 38] is well known for its effectiveness as a feature for image processing and pattern recognition. Its kernels are similar to the response of the two-dimensional receptive field profiles of the mammalian simple cortical cell [47], and exhibit the desirable characteristics of capturing salient visual properties such as spatial localization, orientation selectivity, and spatial frequency selectivity [111]. In the spatial domain, a GWis a complex exponential modulated by a Gaussian function, which is defined as follows [34]:

$$\psi_{\omega,\theta}(x,y)=\frac{1}{2\pi\sigma^2}\exp\left(-\frac{(x\cos\theta+y\sin\theta)^2+(-x\sin\theta+y\cos\theta)^2}{2\sigma^2}\right)\cdot\left[\exp(i\omega(x\cos\theta+y\sin\theta))-\exp\left(-\frac{\omega\sigma^w}{2}\right)\right] \quad (4.1)$$

where x, y denote the pixel position in the spatial domain, ω is the radial center frequency of the complex exponential, θ is the orientation of the GW, and σ is the standard deviation of the Gaussian function. By selecting different center frequencies and orientations, we can obtain a family of Gabor kernels from Equation (4.1), which can then be used to extract features from an image.

GWs can effectively abstract local and discriminating features. In textural analysis [22, 23] and image segmentation [24], GW features have achieved outstanding results, while in machine vision, they are found to be effective in object detection [40, 41], recognition [23, 41, 80] and tracking [157–159]. The most successful application of the GWs is for face recognition. In [31, 35, 42–44], GWs are employed for face recognition, and achieve very high performance

levels. As the dimension of the feature vectors using GWs is very large, linear subspace methods such as PCA and LDA are used to reduce the dimension. To further improve the performance, kernel methods are also used with the Gabor features. The improvement of both the linear methods and the kernel methods is due to the fact that the GW features are robust to illumination, rotation, and scale [38].

In spite of its superior performance, extracting GW features is highly computational. Given an image $f(x, y)$, GW features are extracted by convolving $f(x, y)$ with $\Psi_{\omega,\theta}(x, y)$ as follows:

$$Y_{\omega,\theta}(x, y) = f(x, y) * \psi_{\omega,\theta}(x, y) \qquad (4.2)$$

where * denotes the convolution operator. Usually, convolution is implemented by the fast Fourier transform (FFT) to reduce the computation required for feature extraction. However, the computation required is still very intensive; this, in turn, creates a bottleneck for real-time processing. Hence, an efficient method for extracting Gabor features is important for many practical applications.

The main contribution of this section is to propose a simplified version of Gabor wavelets, whose features can be computed efficiently and can achieve a similar performance level for face recognition. These simplified Gabor wavelets (SGWs) can be viewed as an approximation of the original Gabor wavelets (GWs). An SGW is generated by quantizing a corresponding GW into a certain number of levels. With SGWs, features can be computed efficiently using an integral image. Our proposed SGWs can replace the GWs for the purpose of real-time processing and applications. The rest of this section will describe the structure and the properties of SGWs. Fast algorithms for extracting features by using SGWs will be described, and their corresponding computational

complexity will be analyzed. Finally, we will compare the performances of the SGW features and the GW features for face recognition, and discuss the discriminative power of these features.

4.2 Simplified Gabor wavelets

In this section, we will describe the structure of our proposed SGW. This includes the shape of the SGW, the number of quantization levels, and the methods which determine the respective quantization values.

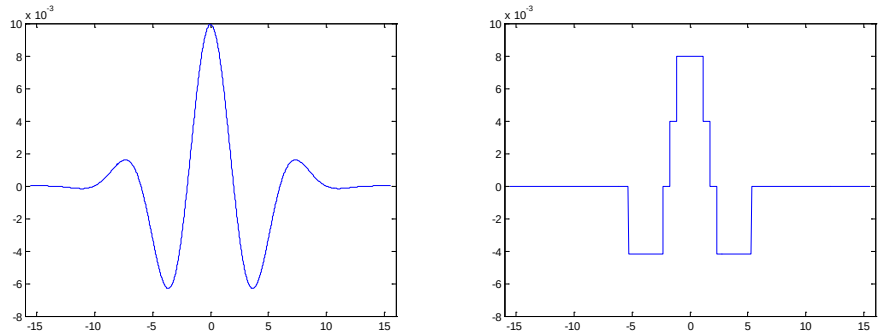
4.2.1 Shape of an SGW

To simplify our discussion, a one-dimensional GW is first considered, whose equation is shown as follows:

$$\psi_{\omega,\theta}(x) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{x^2}{2\sigma^2}\right)\exp(i\omega x) \qquad (4.3)$$

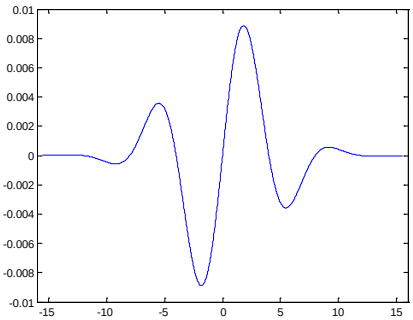
where the term $\exp(-\omega\sigma^2/2)$ in Equation (4.1) is ignored. Figure 4.1(a) shows the real part of this GW, whose values are continuous. To simplify the GW, its values are quantized to a certain number of levels. Figure 4.1(b) illustrates a quantized SGW with 2 quantization levels for the positive values and 1 quantization level for the negative values. Including a level of zero value, the wavelet is said to be quantized into 4 levels. Figure 4.1(c) and Figure 4.1(d) illustrate the corresponding imaginary part of the GW and its simplified version, respectively. The same number of quantization levels is used for the positive and the negative values of the wavelet, because their magnitudes are the same. In Figure 4.1(d), the total number of quantization levels used is 5. For two-dimensional cases, Figure 4.2(a) and Figure 4.2(d) show the real and imaginary parts of the original two-dimensional GWs with the gray-level intensities

representing the magnitudes of the wavelet. The contours of $\Psi_{\omega,\theta}(x, y)$ whose values equal those quantization levels in Figure 4.1(b) and Figure 4.1(d) are illustrated in Figure 4.2(b) and Figure 4.2(e), respectively. In SGWs, the contours are approximated by rectangles. We have derived two approximation methods for forming the rectangles, as shown in Figure 4.3(a) and Figure 4.3(b), respectively. The first method is to use a rectangle of a size just large enough to contain the corresponding contour of the quantized GW. The second method is to choose a rectangle such that the squared error between the elliptical contour of the GW and the corresponding rectangle is a minimum. To simplify the approximation, we adopt the first method in our algorithm. Figure 4.2(c) and Figure 4.2(f) illustrate the corresponding quantized GWs in Figure 4.2(b) and Figure 4.2(e), respectively, approximated by rectangles.

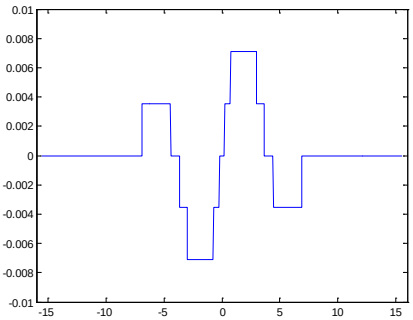


(a)

(b)



(c)



(d)

Figure 4.1 (a) The real part of a one-dimensional GW; (b) the simplified version of (a); (c) the imaginary part of the wavelet; and (d) the simplified version of (c).

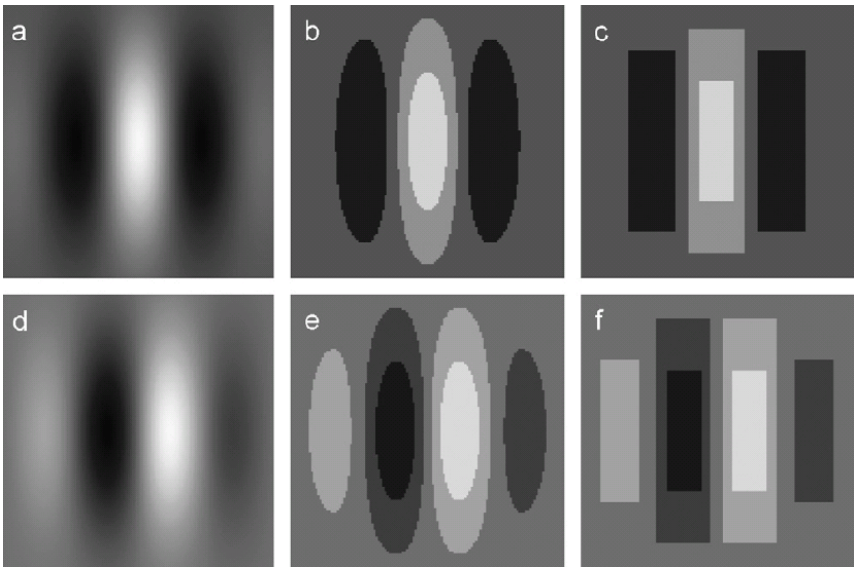


Figure 4.2 (a) The real part of a two-dimensional GW; (b) the contours of the quantized GW of (a); (c) the approximation of the contours in (b) by rectangles; (d) the imaginary part of the two-dimensional GW; (e) the contours of the quantized GW of (d); and (f) the approximation of the contours in (e) by rectangles.

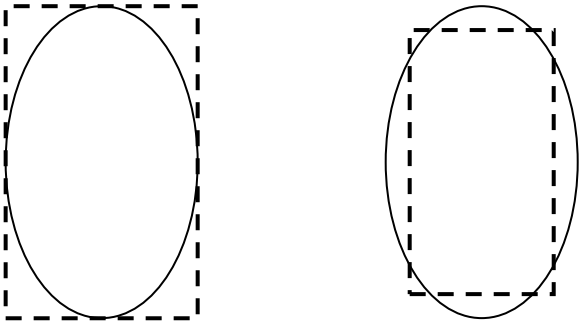


Figure 4.3 (a) Approximation of an elliptical contour using a rectangle just large enough to enclose it; and (b) approximation of the elliptical contour using a rectangle such that the squared error between the rectangle and the contour is a minimum.

4.2.2 Number of quantization levels

The number of rectangles in an SGW depends on the number of quantization levels used to quantize the GW. If more quantization levels are employed, the SGWs will be more similar to the original GW, but more

computation will then be involved for feature extraction. In other words, there is a trade-off between computation and approximation accuracy. In Section 4.4, the computational analysis of using SGWs and GWs for feature extraction will be performed, and the experiments to evaluate the relative performances of SGWs and GWs with different numbers of quantization levels for face recognition will be conducted in Section 4.5.

4.2.3 Determination of quantization levels

In this section, we describe two methods for determining the quantization levels to be used in constructing the SGWs. One of the quantization levels of the SGW is set to zero. Assume that the number of quantization levels for the positive and negative values are n_p and n_n, respectively. Then, the total number of quantization levels is $n_p + n_n + 1$.

Uniform quantization: In this method, the positive and negative parts of a GW are quantized uniformly according to the corresponding number of levels, as shown in Figure 4.4(a) and Figure 4.4(b). Suppose the most positive and negative values of a GW are A_+ and A_-, respectively, the corresponding quantization levels for positive levels $q_+(k)$ and negative levels $q_-(k)$ are as follows:

$$\begin{aligned} & q_+(k) = \frac{A_+}{2n_p + 1} \cdot 2k \quad \text{where } k = 1, \ldots, n_p, \\ & \text{and } q_-(k) = \frac{A_-}{2n_n + 1} \cdot 2k \quad \text{where } k = 1, \ldots, n_n. \end{aligned} \tag{4.4}$$

k-means clustering: As the GWs are not evenly distributed, so the k-means algorithm is used to determine the respective optimal quantization levels. The positive values and the negative values are sampled, and are then partitioned into $n_p + 1$ and $n_n + 1$ clusters, respectively. However, after each iteration, the cluster whose centroid is the closest to zero will be set at zero.

Figure 4.5 illustrates the real part and the imaginary part of a GW and their corresponding simplified versions. These SGWs are then convolved with an image to extract the SGW features at different center frequencies and orientations, which then form a simplified Gabor jet.

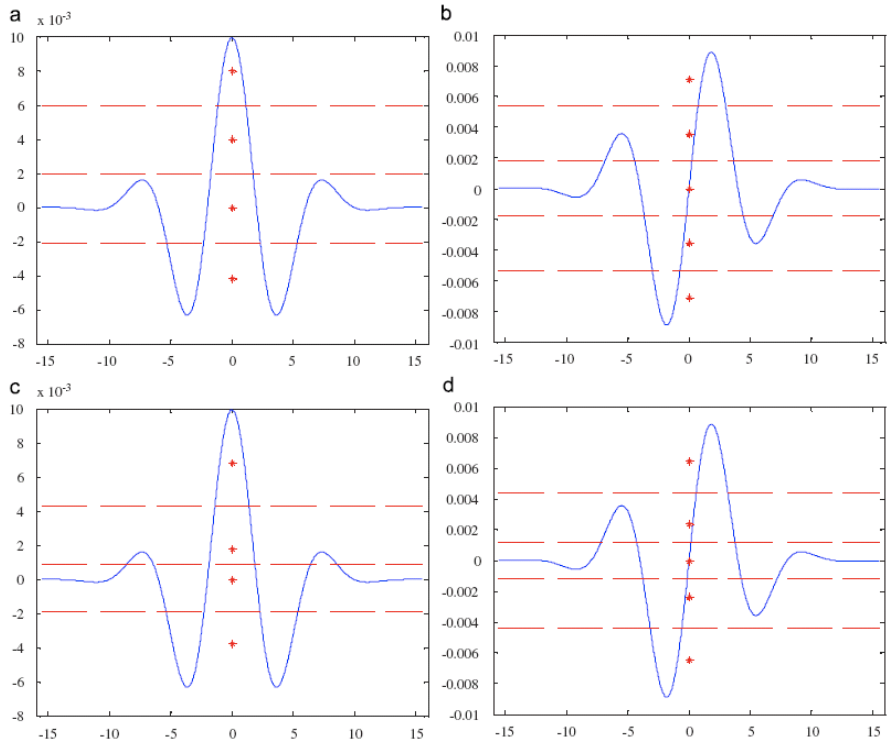


Figure 4.4 (a) The quantization levels for the real part of a GW based on uniform quantization with n_p = 2 and n_n = 1, (b) the quantization levels for the imaginary part of the GW based on uniform quantization with n_p = 2 and n_n = 2, (c) the quantization levels for the real part of the GW based on k-means clustering with n_p = 2 and n_n = 1, and (d) the quantization levels for the imaginary part of the GW based on k-means clustering with n_p = 2 and n_n = 2.

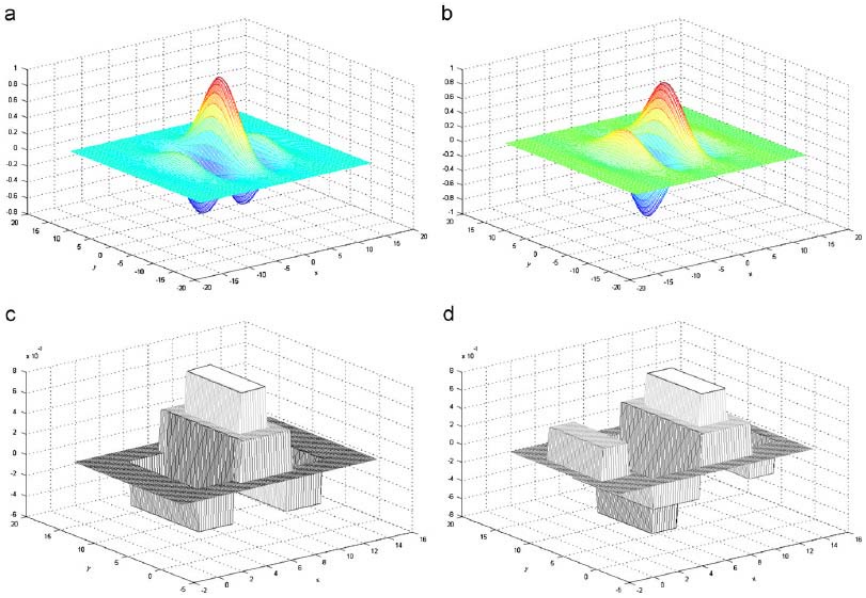


Figure 4.5 The three-dimensional structures of (a) the real part and (b) the imaginary part of a two-dimensional GW, and (c) the real part and (d) the imaginary part of the corresponding SGW.

4.2.4 Demeaned SGW (DMSGW)

The term $e^{-\frac{\omega\sigma^2}{2}}$ in Equation (4.1) makes the GW have a zero mean. An SGW formed by quantizing a GW has a non-zero mean; this makes the SGW features sensitive to the lighting conditions of an image. Hence, each of the SGWs has to be demeaned. The mean of an SGW is computed by summing all of its values, and then dividing this sum by the size of the filter. A demeaned simplified Gabor wavelet (DMSGW) is obtained by subtracting the SGW from its mean value. In the rest of this section, we will use SGW to refer to a demeaned SGW, and the mean of an SGW is denoted as q_m. The next section will describe an efficient algorithm for computing the SGW features using our proposed SGWs.

4.3 Fast algorithm for feature extraction

The feature extraction process with an SGW is far more efficient than that with a GW. This section will, firstly, describe the extraction of GW features using the FFT, and then devise the fast algorithms for extracting features using the SGWs. The computational complexities of using the GW and the proposed SGW for different orientations will be analyzed in Section 4.4, and their respective runtimes will be measured in Section 4.5. In addition to requiring less computation, the SGW features for any pixel position can be extracted. This is particularly an advantage if the SGW features are used for object tracking. To use the FFT, the size of the image must be a power of 2.

4.3.1 Feature extraction using the original GWs

By selecting different center frequencies and orientations, we can obtain a family of GW kernels from Equation (4.1), which can be used for extracting features from images. Given a gray-level image $f(x, y)$, the convolution of $f(x, y)$ and $\Psi_{\omega,\theta}(x, y)$ is given by Equation (4.2). The convolution can be computed efficiently by performing the FFT, then point-by-point multiplications, and finally the inverse FFT (IFFT). By concatenating the convolution output, we can obtain a GW feature vector $Y_{\omega,\theta}$ of dimension $N_w \cdot N_H$:

$$\begin{aligned} Y_{\omega,\theta} = [&Y_{\omega,\theta}(0,0), Y_{\omega,\theta}(0,1), \ldots, Y_{\omega,\theta}(0, N_H - 1), \\ &Y_{\omega,\theta}(1,0), \ldots, Y_{\omega,\theta}(N_W - 1, N_H - 1)]^T, \end{aligned} \tag{4.5}$$

where T represents the transpose operation, and N_W and N_H are the width and height of the image, respectively. In this section, we consider only the magnitude of the GW representations, which can provide a measure of the local properties of an image [28] and is less sensitive to the lighting conditions [32] (for

convenience, we denote it as $Y_{\omega,\theta}$). $Y_{\omega,\theta}$ is normalized to have zero mean and unit variance distribution; and then the Gabor representations with different ω and θ are concatenated to form a high-dimensional vector, as shown in Equation (4.6), and are used for face recognition,

$$Y = \left[Y_{\omega 1,\theta 1}^T Y_{\omega 1,\theta 2}^T \cdots Y_{\omega 1,\theta n}^T Y_{\omega 2,\theta 1}^T \cdots Y_{\omega 1,\theta n}^T \right]^T, \tag{4.6}$$

where l and n are the number of center frequencies and the number of orientations used, respectively. Although the FFT is employed so as to reduce the computational complexity, it is still very computationally intensive because a total of $l \times n$ GWs are involved. In addition, the size of the image must be a power of 2, so that the FFT can be used to implement the convolution for saving the computation.

4.3.2 Fast algorithms for feature extraction based on SGWs

In this section, we will present fast algorithms for feature extraction with the SGW at different orientations. Consider an SGW that is convolved with an image $f(x, y)$, and the SGW is shifted to the pixel position (x_c, y_c), as shown in Figure 4.6. The convolution output at this point is given as follows:

$$Y(x_c, y_c) = \sum_{k=1}^{NR_p} q_+(k) S_+(k) + \sum_{k=1}^{NR_n} q_-(k) S_-(k) + q_m S_F, \tag{4.7}$$

where $S_+(k)$, $S_-(k)$ and S_F are the sum of the gray-level intensities of those pixels covered by the rectangles with quantization values $q_+(k)$, $q_-(k)$, and the rectangular region of the filter, respectively. NR_p and NR_n are the numbers of rectangles with positive quantization values and negative quantization values, respectively. As an example in Figure 4.2(c), $n_p = 2$ and $n_n = 1$, then $NR_p = 2$ and $NR_n = 2$.

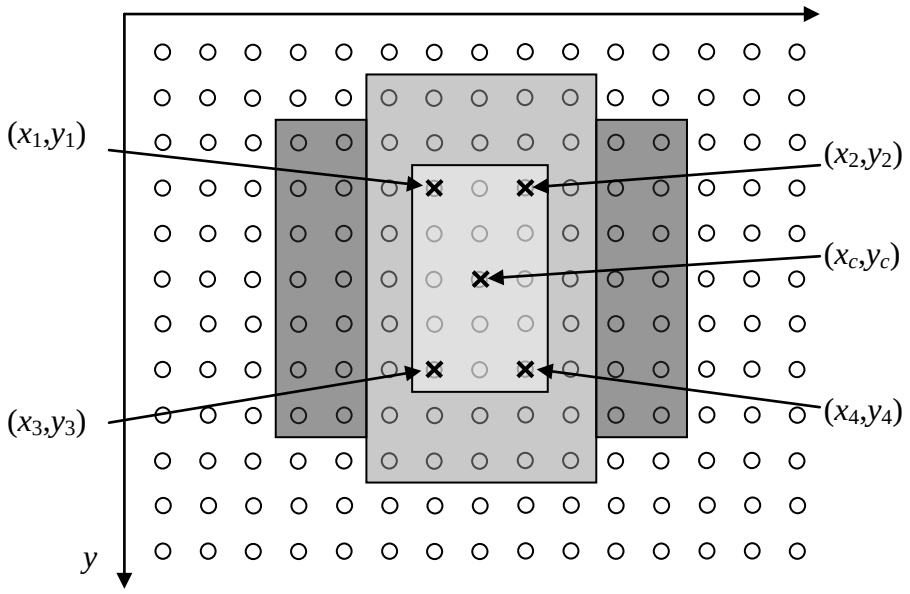


Figure 4.6 Image $f(x, y)$ is convolved with an SGW whose center is shifted to the pixel position (x_c, y_c).

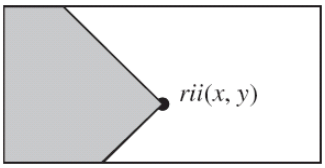


Figure 4.7 Rotated integral image $rii(x, y)$, which is equal to the sum of pixel intensities inside the shaded and rotated rectangle.

$S_+(k)$, $S_-(k)$ and S_F are computed based on the idea of an integral image [146], which can calculate the sum of pixel values within a rectangle efficiently. In addition, a fast algorithm for rectangles rotated by 45° or 135° is also available [146]. Consequently, our SGW considers four orientations only, which are 0°, 45°, 90°, and 135°. Denote $ii(x, y)$ as the integral image, and then its value at location (x, y) is the sum of the pixel values above and to the left of (x, y) inclusive, i.e.

$$ii(x, y) = \sum_{x' \le x, y' \le y} f(x', y') \tag{4.8}$$

The following pair of recursive equations is used to compute the integral image in one pass over the image:

$$s(x,y) = s(x,y-1) + f(x,y) \quad \text{and}$$
$$ii(x,y) = ii(x-1,y) + s(x,y), \tag{4.9}$$

where $s(x, -1) = ii(-1, y) = 0$. Let us denote (x_k^1, y_k^1), (x_k^2, y_k^2), (x_k^3, y_k^3), and (x_k^4, y_k^4) as the respective coordinates of the four corners of the rectangle for the k^{th} quantization level. Figure 4.6 shows the four corners for $k = n_p$. Hence, we have

$$S_+(k) = \begin{cases} ii\left(x_{n_p}^4, y_{n_p}^4\right) + ii\left(x_{n_p}^1 - 1, y_{n_p}^1 - 1\right) - ii\left(x_{n_p}^2, y_{n_p}^2 - 1\right) + ii\left(x_{n_p}^3 - 1, y_{n_p}^3\right), & k = n_p, \\ ii\left(x_k^4, y_k^4\right) + ii\left(x_k^1 - 1, y_k^1 - 1\right) - ii\left(x_k^2, y_k^2 - 1\right) + ii\left(x_k^3 - 1, y_k^3\right) - S_+(k+1), & k < n_p. \end{cases} \tag{4.10}$$

$$S_-(k) = \begin{cases} ii\left(x_{n_n}^4, y_{n_n}^4\right) + ii\left(x_{n_n}^1 - 1, y_{n_n}^1 - 1\right) - ii\left(x_{n_n}^2, y_{n_n}^2 - 1\right) + ii\left(x_{n_n}^3 - 1, y_{n_n}^3\right), & k = n_n, \\ ii\left(x_k^4, y_k^4\right) + ii\left(x_k^1 - 1, y_k^1 - 1\right) - ii\left(x_k^2, y_k^2 - 1\right) + ii\left(x_k^3 - 1, y_k^3\right) - S_-(k+1), & k < n_n. \end{cases} \tag{4.11}$$

For a rectangle at an orientation of 45°, the rotated integral image, $rii(x, y)$ at location (x, y) contains the sum of the pixel values of the rectangle rotated by 45°, with the rightmost corner at (x, y) and extended to the boundaries of the image, as shown in Figure 4.7, i.e.

$$rii(x,y) = \sum_{x' \le x, x' \le x - |y - y'|} f(x', y') \tag{4.12}$$

Two passes over an image are required to compute the rotated integral image. The first pass is performed from left to right and top to bottom as follows:

$$rii(x,y) = rii(x-1,y-1) + rii(x-1,y) + f(x,y) - rii(x-2,y-1), \tag{4.13}$$

where $rii(x, -1) = rii(-1, y) = rii(-2, y) = 0$. The second pass is performed from right to left and bottom to top as follows:

$$rii(x,y) = rii(x,y) + rii(x-1,y+1) - rii(x-2,y). \tag{4.14}$$

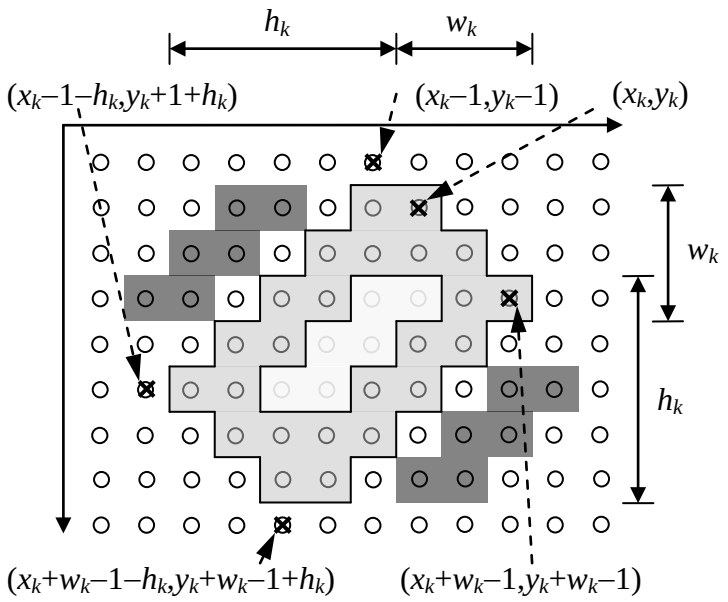


Figure 4.8 The computation scheme for a rotated rectangle.

Let us denote (x_k, y_k, w_k, h_k) as the x-coordinate, y-coordinate, width, and height, respectively, of the rotated rectangle in Figure 4.8. Then, we have

$$S_+(k)=\begin{cases} rii\left(x_{n_p}+w_{n_p}-1, y_{n_p}+w_{n_p}-1\right)+rii\left(x_{n_p}-h_{n_p}-1, y_{n_p}+h_{n_p}-1\right) \\ -rii\left(x_{n_p}-1, y_{n_p}-1\right)-rii\left(x_{n_p}+w_{n_p}-h_{n_p}-1, y_{n_p}+w_{n_p}+h_{n_p}-1\right), & k=n_p, \\ rii\left(x_k+w_k-1, y_k+w_k-1\right)+rii\left(x_k-h_k-1, y_k+h_k-1\right) \\ -rii\left(x_k-1, y_k-1\right)-rii\left(x_k+w_k-h_k-1, y_k+w_k+h_k-1\right), & k<n_p. \end{cases}$$

(4.15)

Similar formulation can be derived for the computation of $S_-(k)$, as well as for the case when a rectangle is at an orientation of 135°.

To further speed up feature extraction, let us denote $RS(k)$ as the sum of pixel intensities inside a rectangle with the coordinates of its four corners being (x_k^1, y_k^1), (x_k^2, y_k^2), (x_k^3, y_k^3), and (x_k^4, y_k^4), respectively. Thus

$$RS(k)=ii\left(x_k^4, y_k^4\right)+ii\left(x_k^1-1, y_k^1-1\right)-ii\left(x_k^2, y_k^2-1\right)-ii\left(x_k^3-1, y_k^3\right). \quad (4.16)$$

Let $RS_+(k)$, $RS_-(k)$ and RS_F be the sum of the gray-level intensities of those pixels inside the rectangles with quantization values $q_+(k)$, $q_-(k)$ and the rectangular region covered by the SGW, respectively. Figure 4.9 shows the real part of an

SGW with $n_n = n_p = 2$ or $NR_n = 4$ and $NR_p = 2$. Then, the convolution output at the pixel position (x_c, y_c) is:

$$\begin{aligned}
Y(x_c, y_c) &= \sum_{k=1}^{NR_n} [q_-(k) \cdot S_-(k)] + \sum_{k=1}^{NR_p} [q_+(k) \cdot S_+(k)] + q_m \cdot S_F \\
&= q_-(1) \cdot S_-(1) + q_-(2) \cdot S_-(2) + q_-(3) \cdot S_-(3) + q_-(4) \cdot S_-(4) \\
&\quad + q_+(1) \cdot S_+(1) + q_+(2) \cdot S_+(2) + q_m \cdot S_F \\
&= q_-(1) \cdot [RS_-(1) - RS_-(2)] + q_-(2) \cdot RS_-(2) \\
&\quad + q_-(3) \cdot [RS_-(3) - RS_-(4)] + q_-(4) \cdot RS_-(4) \\
&\quad + q_+(1) \cdot [RS_+(1) - RS_+(2)] + q_+(2) \cdot RS_+(2) \\
&\quad + q_m \cdot (RS_F - RS_-(1) - RS_-(3) - RS_+(1)) \\
&= [q_-(1) - q_m] \cdot RS_-(1) + [q_-(2) - q_-(1)] \cdot RS_-(2) \\
&\quad + [q_-(3) - q_m] \cdot RS_-(3) + [q_-(4) - q_-(3)] \cdot RS_-(4) \\
&\quad + [q_+(1) - q_m] \cdot RS_+(1) + [q_+(2) - q_+(1)] \cdot RS_+(2) + q_m \cdot RS_F \\
&= \sum_{k=1}^{NR_n} [m_-(k) \cdot RS_-(k)] + \sum_{k=1}^{NR_p} [m_+(k) \cdot RS_+(k)] + m_F RS_F
\end{aligned} \tag{4.17}$$

where

$$m_+(k) = \begin{cases} q_+(k) - q_m & k, \text{ refer to the outermost rectangles} \\ q_+(k) - q_+(k-1) & k, \text{ refer to the inner rectangles.} \end{cases}$$

$$m_-(k) = \begin{cases} q_-(k) - q_m & k, \text{ refer to the outermost rectangles} \\ q_-(k) - q_-(k-1) & k, \text{ refer to the inner rectangles.} \end{cases}$$

$m_F = q_m$.

Hence, instead of using $q(k)$ directly, the $m(k)$s are employed in the computation.

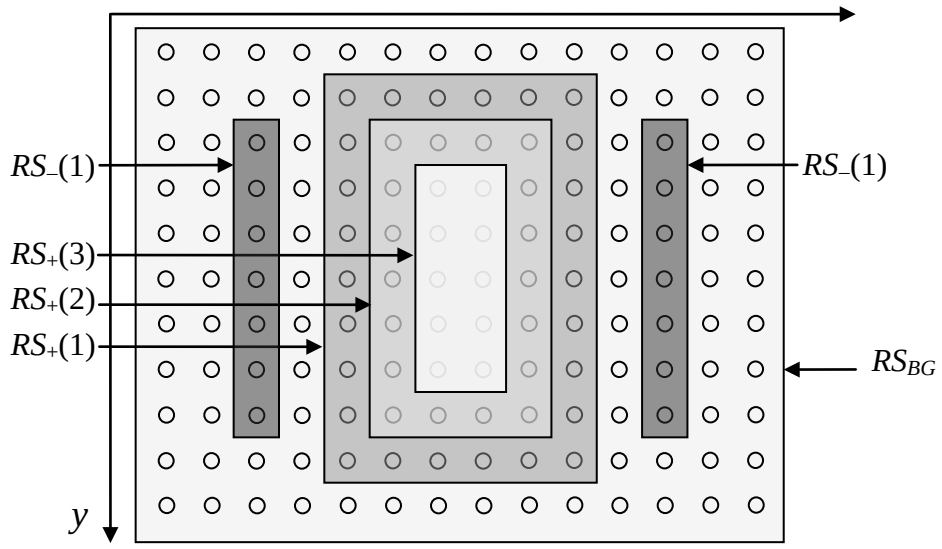


Figure 4.9 The rectangles in an SGW.

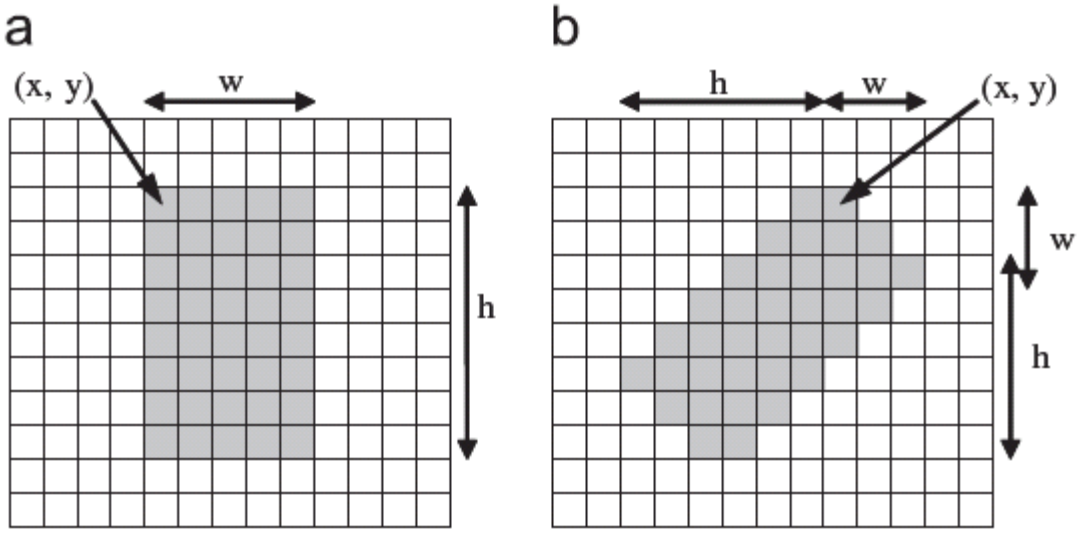
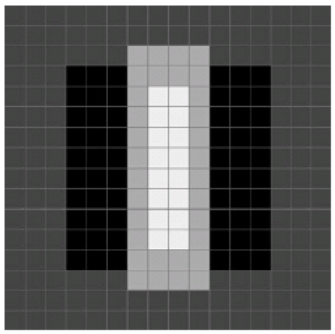


Figure 4.10 Definition of the (*x*, *y*)-coordinates, width and height of a rectangle in an SGW at an orientation of (a) 0°, and (b) 45°.

For implementation, a number of parameters are required to describe a rectangle, which govern the computation of $RS_+(k)$, $RS_-(k)$ and RS_F. These parameters include the orientation, $m_+(k)$, $m_-(k)$, m_F, (x, y) coordinates, and the width and height of each rectangle. Figure 4.10 defines the (x, y) coordinates, and the width and height of an upright rectangle and a rotated rectangle, which is similar to that in Ref. [22] and [23]. Figure 4.11(a) shows an SGW, while Figure 4.11(b) describes the parameters of this SGW.

a



b

16 16 // width & height of the SGW
0.785398 // scale of the SGW
0 // orientation of the SGW (0 = 0°, 1 = 45°, 2 = 90°, 3 = 135°)
0 // real or imaginary part (0=real, 1=imaginary)
-0.000082 // mean of SGW
2 // number of quantization levels in the positive side
1 // number of quantization levels in the negative side
-0.004159 0.000082 0.004032 0.007982 // quantization values
4 // number of rectangles

// Quantization value	$m(k)$	x	y	w	h
-0.004159	-0.004241	3	3	3	10
-0.004159	-0.004241	10	3	3	10
0.004032	0.003950	6	2	4	12
0.007982	0.003950	7	4	2	8

Figure 4.11 (a) An SGW and (b) the corresponding parameters of this wavelet

4.4 Computational analysis for feature extraction

In this section, we will analyze and compare the computations required for extracting features using GW and SGW, respectively. Within our context, computations refer to the number of real additions and real multiplications required for extracting the GW features of an image using a GW. In our analysis, we assume that the image size is a power of 2 so that the FFT can be applied when using GWs for faster feature extraction. Actually, for the use of SGW, the image may be of any size and the features at any individual pixel position can be computed efficiently.

4.4.1 Feature extraction with GW

Given an $N \times N$ image, f, and a GW, g, with an arbitrary scale and orientation, GW features can be extracted by convolution, i.e. $f * g$. The convolution is implemented by using the FFT, then point-by-point multiplications, and finally the IFFT. In our analysis, we assume that the FFTs of the GWs are pre-computed.

The FFT of an $N \times N$ image requires $N^2\log_2 N^2$ complex additions and $0.5N^2\log_2 N^2$ complex multiplications. The IFFT requires the same amount of computation as the FFT.

The point-by-point multiplications involve N^2 complex multiplications. Performing one complex addition requires 2 real additions, while one complex multiplication requires 2 real additions and 4 real multiplications. Therefore, feature extraction based on a GW requires a total of $2N^2\log_2 N^2$ complex additions and $N^2\log_2 N^2 + N^2$ complex multiplications; this is equivalent to a total of $6N^2\log_2 N^2 + 2N^2$ real additions and $4N^2\log_2 N^2 + 4N^2$ real multiplications.

4.4.2 Feature extraction with SGW

As described in Section 4.3, fast algorithms are available for extracting SGW features using SGWs at 4 different orientations. These fast algorithms are based on the use of integral images and rotated integral images, such that features at any position in an image can be computed efficiently. Our algorithm will first perform a table look-up operation to compute the sum of pixel values for the respective rectangles of the SGW. Then, each of the pixel sums is multiplied by the quantization value of the corresponding rectangle. The sum of these products is the SGW feature at a given pixel position.

The computation for extracting features using an SGW at orientation 0° or 90° (a non-rotated SGW) is different from that when using an SGW at orientation 45° or 135° (a rotated SGW). This is because, for feature extraction, the non-rotated SGW uses the integral image, while the rotated SGW uses the rotated integral image. The computations involved are different for different

orientations. Consequently, we separate our analysis into two parts: the non-rotated SGW (NR-SGW) and the rotated SGW (R-SGW).

4.4.2.1 The non-rotated SGW (NR-SGW)

Before extracting features using an NR-SGW, the integral image must be computed. From Equation (4.9), 4 real additions are required to compute an entry of the integral image. For an image of size $N \times N$, $4N^2$ real additions are required for the whole integral image. Suppose that the SGW contains a total of ${N_{rect}}^t$ rectangles. From Equation (4.16) and Equation (4.17), $3{N_{rect}}^t$ real additions are required to compute all the rectangular pixel sums, and ${N_{rect}}^t$ real multiplications and $({N_{rect}}^t - 1)$ real additions are required to compute the SGW feature for a given pixel position. The coordinates of the four corners in Equation (4.16) can be generated by a table look-up operation. Consequently, a total of $4N^2{N_{rect}}^t + 3N^2$ real additions and $N^2{N_{rect}}^t$ real multiplications are required to extract the SGW feature.

4.4.2.2 The rotated SGW (R-SGW)

The rotated integral image is computed for extracting feature with a rotated SGW. From Equation (4.13) and Equation (4.14), 9 real additions are required to compute an entry in the rotated integral image. For an image of size $N \times N$, $9N^2$ real additions are required to compute the whole rotated integral image.

Feature extraction with an R-SGW is computed in a similar way to that with the NR-SGW. The rotated pixel sums covered by the rotated rectangles of the R-SGW are computed. Suppose that the R-SGW contains ${N_{rect}}^t$ rectangles, then from Equation (4.16) and Equation (4.17), $3{N_{rect}}^t$ real additions are required

to compute all the rotated rectangular pixel sums, and $N_{rect}{}^{t}$ real multiplications and ($N_{rect}{}^{t} - 1$) real additions are required to compute the R-SGW feature at a pixel position. Therefore, a total of $4N^2N_{rect}{}^{t} + 8N^2$ real additions and $N^2N_{rect}{}^{t}$ real multiplications is required to extract the feature from the whole image. Table 4.1 shows the summarization of the computational complexities of feature extraction using GW and SGW.

To illustrate the computational advantage of using SGWs over GWs, Table 4.2 tabulates the respective numbers of arithmetic operations required for extracting GW features and SGW features, and Table 4.3 shows the respective numbers of rectangles used to represent the different level quantized SGWs. It is found that about 2.85 times and 2.44 times the arithmetic operations are saved if a 3-level quantized NR-SGW and R-SGW, respectively, are used. Moreover, the number of multiplications required for SGW feature extraction is reduced significantly when compared to that for GW. In general, the runtime required for multiplication is longer than that for addition. Furthermore, the runtime consumed by a floating point arithmetic operation is longer than that for an integer arithmetic operation. Feature extraction with SGW involves fewer floating point operations than does GW, therefore, the runtime for SGW feature extraction should in practice have a speed-up rate higher than 2.85 times.

Table 4.1 Computational complexities of feature extraction using GW and SGW

		+	×
GW	A: Compute FFT of image (floating point operations)	$3N^2 \log_2 N^2$	$2N^2 \log_2 N^2$
	B: Compute feature by multiplying FFT image and FFT GW (floating point operations)	$2N^2$	$4N^2$
	C: Compute IFFT of feature (floating point operations)	$3N^2 \log_2 N^2$	$2N^2 \log_2 N^2$
	Total	$6N^2 \log_2 N^2 + 2N^2$	$4N^2 \log_2 N^2 + 4N^2$
NR-SGW	D: Compute SAT (integer additions)	$4N^2$	0
	E: Compute rectangular pixel sums (integer additions)	$3N^2 N_{rect}{}^t$	0
	F: Compute feature by multiplying rectangular pixel sums and quantization value of rectangles (floating point multiplications)	0	$N^2 N_{rect}{}^t$
	G: Add all products in F (floating point additions)	$N^2(N_{rect}{}^t - 1)$	0
	Total	$4N^2 N_{rect}{}^t + 3N^2$	$N^2 N_{rect}{}^t$
R-SGW	H: Compute RSAT (integer additions)	$9N^2$	0
	I: Compute rotated rectangular pixel sums (integer additions)	$3N^2(N_{rect}{}^t - 1)$	0
	J: Compute SGW background pixel sums (integer additions)	$3N^2$	0
	K: Compute feature by multiplying rectangular pixel sums and quantization value of rectangles (floating point multiplications)	0	$N^2 N_{rect}{}^t$
	L: Add all products in K (floating point additions)	$N^2 (N_{rect}{}^t - 1)$	0
	Total	$4N^2 N_{rect}{}^t + 8N^2$	$N^2 N_{rect}{}^t$

[a] Image dimension $= N \times N$, where N must be to the power of 2 in order to speed up the GW feature extraction process. [b] $N_{rect}{}^t$ is the total number of rectangles in an SGW, which is listed in Table 4.3.

Table 4.2 Number of arithmetic operations required for extracting GW features from a 64 × 64 pixel image using a GW and an SGW with different numbers of quantization levels

	GW		+	×	Total
			303,104	212,992	516,096
NR-SGW	No. of quantization levels used	3 levels	147,844	32,768	180,612
		5 levels	229,764	53,248	283,012
		7 levels	344,452	81,920	426,372
R-SGW	No. of quantization levels used	3 levels	179,049	32,768	211,817
		5 levels	260,969	53,248	314,217
		7 levels	375,657	81,920	457,577

Table 4.3 The number of rectangles of an SGW with different numbers of quantization levels, where n_n and n_p are the number of negative quantization levels and the number of positive quantization levels in an SGW

Number of quantization levels ($n_n + n_p + 1$)	Number of rectangles in the real part of an SGW (N_{rect}^{r})	Number of rectangles in the imaginary part of an SGW (N_{rect}^{i})	Total number of rectangles in an SGW, including the background of SGW (N_{rect}^{t}) = (N_{rect}^{r}) + (N_{rect}^{i}) +1
3 ($n_n = 1$, $n_p = 1$)	(N_{rect}^{r}) = ($n_n \times 2 + n_p$) = 3	(N_{rect}^{i}) = (($n_n + 1$)+ ($n_p + 1$)) = 4	8
5 ($n_n = 2$, $n_p = 2$)	(N_{rect}^{r}) = ($n_n \times 2 + n_p$) = 6	(N_{rect}^{i}) = (($n_n + 1$)+ ($n_p + 1$)) = 6	13
7 ($n_n = 3$, $n_p = 3$)	(N_{rect}^{r}) = ($n_n \times 2 + (n_p + 2)$) = 11	(N_{rect}^{i}) = (($n_n + 1$)+ ($n_p + 1$)) = 8	20

Table 4.4 The number of distinct subjects, the number of images and the characteristics of the face databases

Databases	Characteristics	Number of distinct subjects	Number of images	Number of images per subject
Yale	Variations in facial expression	15	150	10
YaleB	Large variations in lighting	10	640	64
AR	Variations in facial expression	121	605	5
	Overall	146	1395	

Table 4.5 Face recognition performances of SGW1, SGW2 and GW with different scales, orientations, and quantization levels (SGW1: uniformly quantized SGWs, SGW2: *k*-means quantized SGWs, GW: Gabor wavelets)

	Different combinations of scales-orientations-quantization levels	Recognition rate		
		Yale(%)	YaleB(%)	AR(%)
SGW1	5 scales 4 orientations 3 quantization levels	82.00	90.16	92.40
	5 scales 4 orientations 5 quantization levels	84.67	92.19	92.40
	5 scales 4 orientations 7 quantization levels	82.67	92.66	92.89
	4 scales 4 orientations 3 quantization levels	82.67	93.13	92.07
	4 scales 4 orientations 5 quantization levels	82.00	94.69	91.74
	4 scales 4 orientations 7 quantization levels	82.67	94.84	92.07
	3 scales 4 orientations 3 quantization levels	82.67	92.97	92.23
	3 scales 4 orientations 5 quantization levels	82.67	93.91	92.23
	3 scales 4 orientations 7 quantization levels	83.33	94.69	92.23
SGW2	5 scales 4 orientations 3 quantization levels	82.67	91.09	91.90
	5 scales 4 orientations 5 quantization levels	82.67	92.50	92.23
	5 scales 4 orientations 7 quantization levels	82.67	92.50	92.56
	4 scales 4 orientations 3 quantization levels	82.67	93.91	91.74
	4 scales 4 orientations 5 quantization levels	82.67	95.00	91.74
	4 scales 4 orientations 7 quantization levels	83.33	95.47	91.90
	3 scales 4 orientations 3 quantization levels	82.00	93.59	92.40
	3 scales 4 orientations 5 quantization levels	82.67	95.00	91.90
	3 scales 4 orientations 7 quantization levels	83.33	94.53	92.23
GW	5 scales 4 orientation	80.00	94.69	92.73
	4 scales 4 orientation	78.00	97.50	92.23
	3 scales 4 orientation	74.00	99.22	89.92

4.5 Experimental results

In this section, we will evaluate the respective performances of the proposed SGWs with different numbers of quantization levels. The two different methods for determining the quantization values of an SGW will also be evaluated. Then, we will compare the performances of the SGW features and the GW features for face recognition. Finally, we will compare the runtimes for extracting the SGW features and the GW features.

4.5.1 Face databases and experimental set-up

The standard face databases used include the Yale database, YaleB database and AR database. The number of distinct subjects, the number of testing images and the characteristics of the databases are tabulated in Table 4.4.

For face recognition, a frontal-view image of each subject in the databases is selected as a training image, and the remaining faces are used for testing. Each face image is normalized to a size of 64×64, and is aligned based on the position of the two eyes for matching. In order to enhance the global contrast of the images and reduce the effect of uneven illuminations, histogram equalization is applied to all images. As described in Section 4.2.3, we have two different ways to determine the quantization levels of SGWs. The SGWs derived based on uniform quantization and on k-means clustering are denoted as SGW1 and SGW2, respectively. The GW and SGW adopt 3–5 center frequencies with 4 orientations. In other words, 12–20 GWs and SGWs are used for feature extraction. The extracted features with each Gabor filter are concatenated to form a feature vector, which is then normalized to have zero mean and unit variance. These Gabor jets are then used directly to compute the distance between two images, pixel position by pixel position.

4.5.2 Relative performances of SGW1 and SGW2

Table 4.5 shows the recognition rates based on SGW1 and SGW2 with different numbers of quantization levels for the different databases. For the real part of a GW, the dynamic range of the positive values is usually larger than that of the negative values. Hence, n_p should be set larger than n_n. However, for the imaginary part of the GW, the dynamic ranges of the positive values and

negative values are the same, so n_p should be equal to n_n. To simplify the experiment, we set n_p equal to n_n for both the real and imaginary parts. Consequently, including the level for zero, the numbers of quantization levels considered in the experiments are 3, 5, and 7.

From Table 4.5, the relative performances of SGW1 and SGW2 are very similar. The face recognition rate increases slightly with an increase in the number of quantization levels. If more quantization levels are used, the SGW can better approximate the GW, and its performance will then be closer to that of the GW. However, using the SGW with more quantization levels will involve more computations.

We have also investigated the effect of using more scales of the SGW with a fixed number of quantization levels. Experimental results show that using 4 scales of SGW results in the best recognition rate. Theoretically, using 5 scales should produce a better performance than using 4 scales only. However, the error in representing a GW is large when its scale is large. As discussed in Sections 4.2.1 and 4.3, in order to utilize fast algorithms to extract the features, the SGWs must be approximated with rectangles after quantizing the GWs. This constraint will alter the effective regions of the SGWs. Figure 4.12 shows a GW, a GW after quantization, and an SGW approximated by rectangles. We can observe that part of the effective regions of the quantized GW is removed or extended in order to form a rectangular shape, which will therefore introduce quantization errors. As the size of an SGW is 16 × 16 pixels only, large rectangles cannot be formed. As a result, the quantization errors in forming the rectangles are significant for those large-scale SGWs. On the contrary, for small-scale SGWs, small rectangles will be formed without requiring much of the original shape of

the quantized GW to be changed. This will introduce fewer quantization errors. For SGWs with 5 scales, the approximation of some of the large-scale GWs is not accurate. This, in turn, will degrade the overall recognition performance.

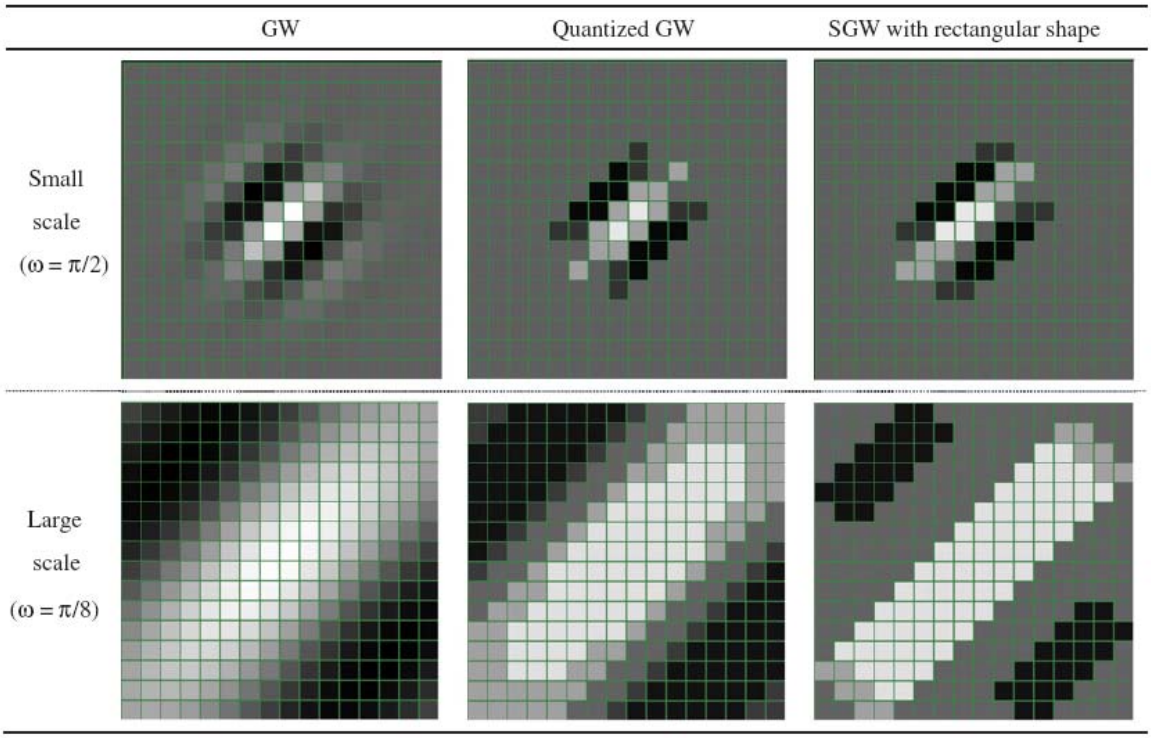


Figure 4.12 The first column is the GW, the second column is the quantized form of GW, and the third column is the SGW with a rectangular shape. The top row is the small-scale ($\omega = \pi/2$) GW being quantized and formed into a rectangular-shaped SGW. The bottom row is the large-scale ($\omega = \pi/8$) GW being quantized and formed into a rectangular-shaped SGW.

4.5.3 Performances of the SGW and the GW

The use of the SGW can save a lot of computation when compared to the GW, while maintaining a comparable performance to the GW. Table 4.5 tabulates the performances using SGW1, SGW2 and GW for face recognition with different numbers of center frequencies and orientations. The face recognition results show that, with the same number of center frequencies and orientations, the relative performances of the SGW and the GW are very similar; and in some cases, the SGW outperforms the GW. Actually, the center frequency

of an SGW should be very similar to its original GW. An SGW is a quantized version of its GW; their rates of variation should be maintained. Hence, in the frequency domain, the center frequencies of the SGW and the GW should be very close, while the shape of their spectra will differ. The features extracted by a GW and the corresponding SGW should be similar. Figure 4.13 shows the magnitudes of the GW features and the SGW features at 3 scales and 4 orientations. We can observe that the general shapes of SGW features and GW features are similar; however, SGWs introduce a directional pattern on the features, which is a drawback with quantizing GWs coefficients to a certain number of levels.

From Table 4.5, the performance of the SGW is slightly worse than that of the GW with the YaleB database, while the SGW has a very similar performance to the GW with the other databases. The reason for this is that the images in the YaleB database have a wide variation in lighting conditions. As we discussed in Section 4.2.4, an SGW is the quantized version of a GW, so the values of the SGWs are changed in step. Therefore, when two images of the same person have a significant difference in lighting conditions, the features extracted by GWs and SGWs will also differ greatly. Hence, the performance of the SGW will be degraded in this circumstance.

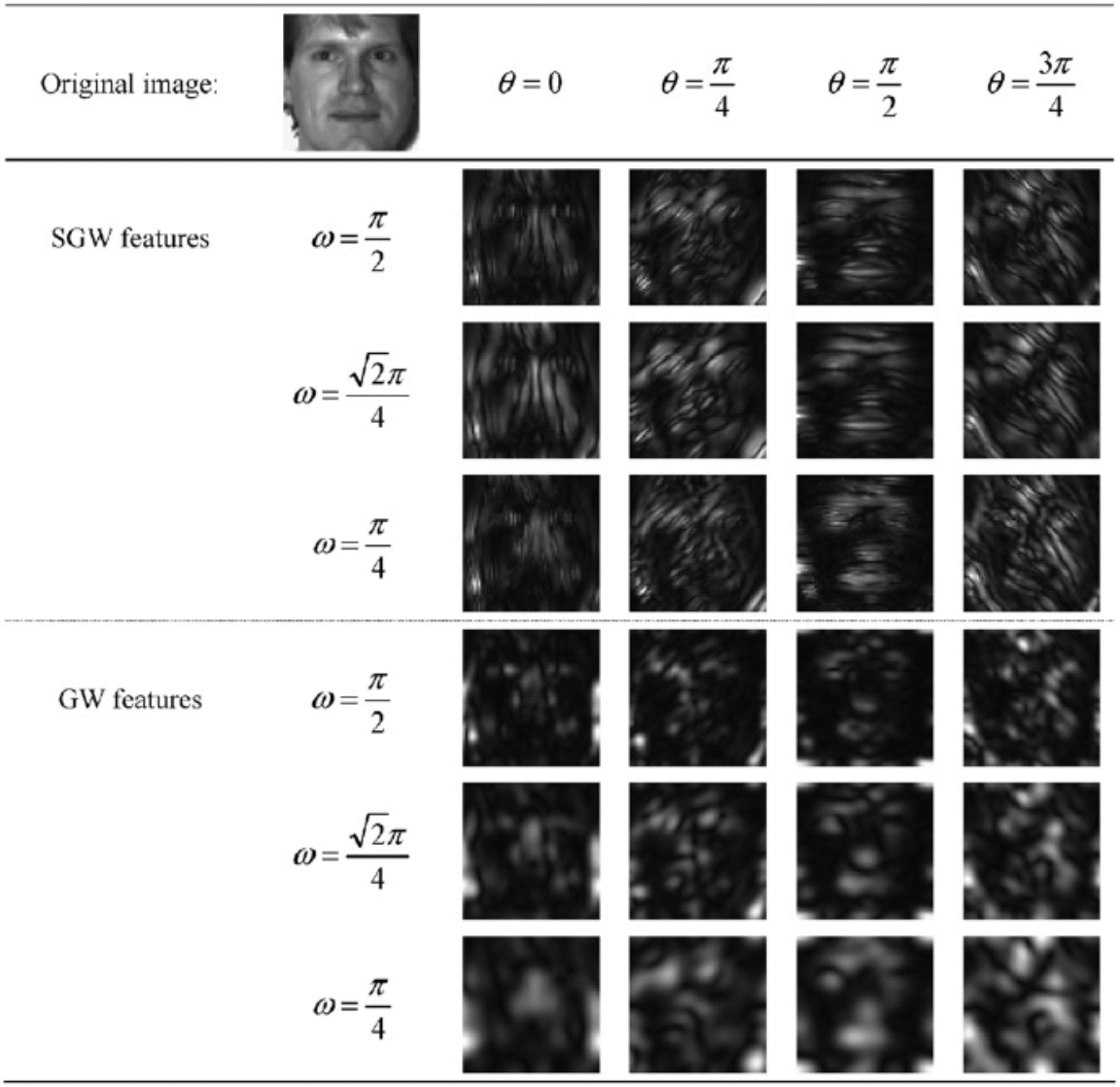


Figure 4.13 The magnitudes of SGW features and GW features at 3 scales and 4 orientations.

Table 4.6 The average runtimes for feature extraction using GW and SGW with different scales, orientations, and numbers of quantization levels

SGW	5 scales, 4 orientations			4 scales, 4 orientations			3 scales, 4 orientations		
	3-Lv	5-Lv	7-Lv	3-Lv	5-Lv	7-Lv	3-Lv	5-Lv	7-Lv
	16.09ms	27.50ms	37.97ms	12.81ms	22.50ms	30.94ms	9.37ms	17.50ms	23.44ms
GW		70.64ms			56.73ms			42.67ms	
Speed-up rate	4.39	2.57	1.86	4.43	2.52	1.83	4.55	2.44	1.82

4.5.4 Runtimes for feature extraction with the SGW and the GW

In our experiments, we also measure the runtimes required for feature extraction using the SGW and the GW. One of the images from the Yale database was used, and the size of each face region is 64×64 pixels. Feature extractions using the SGW and the GW at 5 scales and 4 orientations were performed for 100 times, and the respective total runtimes were measured. Table 4.6 tabulates the runtimes for extracting features using the SGW and the GW. With a 3-level quantized SGW, the speedup rate for feature extraction is 4.39 times that of a GW. The reduction in runtime will decrease if the SGW uses more quantization levels. For SGWs with 5 and 7 quantization levels, the runtimes for feature extraction are 27.5 and 37.97 ms, respectively, and the corresponding speed-up rates are 2.57 and 1.86, respectively.

To conclude our experiment results, the performance of the SGW is comparable to that of the GW, while the computation required by the SGW is significantly less than that for the GW. GWs can extract features which are discriminative and useful for many applications, but they are impractical for real-time applications due to their high complexity in feature extraction. Consequently, SGWs can be propelled to replace GWs for real-time applications and processing.

4.6 Conclusion

In this chapter, we have proposed a simplified version of GWs, which can achieve a performance level similar to the original GWs for face recognition. We have also described fast algorithms for feature extraction based on SGWs at different orientations. In addition, we have presented how to construct these

SGWs and their performance with different numbers of quantization levels, center frequencies and orientations. When 5 center frequencies and 4 orientations are employed, the relative performances of the SGWs and the GWs are very similar, while, at most, a speed-up rate of 4.39 times can be achieved if 3-level quantized SGWs are used. The runtimes required for feature extraction in a 64 × 64 image, based on an SGW with 3 quantization levels and a GW, are 16.09 and 70.64 ms, respectively. These results can propel SGWs to replace GWs for realizing real-time applications and processing. However, the simplified Gabor features are slightly more sensitive to lighting variations than the original Gabor features are.

Chapter 5: Face Recognition with a Large Database Using Vantage Objects

5.1 Introduction

Research in face recognition [6-8] has been conducted for several decades, and most of the face recognition algorithms can achieve a high accuracy level with a database of moderate size. However, when these face recognition algorithms are applied to a very large database, a more efficient way to search for a face is indispensable. Therefore, in this section, we propose an efficient structure for searching faces in a very large database. In our approach, based on a query image, a small subset of the large database, called a condensed database, is constructed [153]. The criterion used in extracting training faces to form the condensed database is that the selected faces should be relatively close to the query input according to a certain measurement. Since the condensed database is much smaller than the original database, the time required to search for similar faces from a very large database can be greatly reduced without any degradation of recognition accuracy.

We employ an indexing structure for image retrieval, namely vantage objects [153], as an efficient way to form a condensed database. The vantage objects can be simply some samples selected from the large database under consideration. The similarity or difference of the training samples to each vantage object is measured to form a ranked list. Similar images should be located close to each other on the ranked lists, as illustrated in Figure 5.1. For a query image, its corresponding positions in the respective ranked lists should be

close to those training images similar to it. Hence, those neighboring training samples to the query input can be selected to form a smaller condensed database.

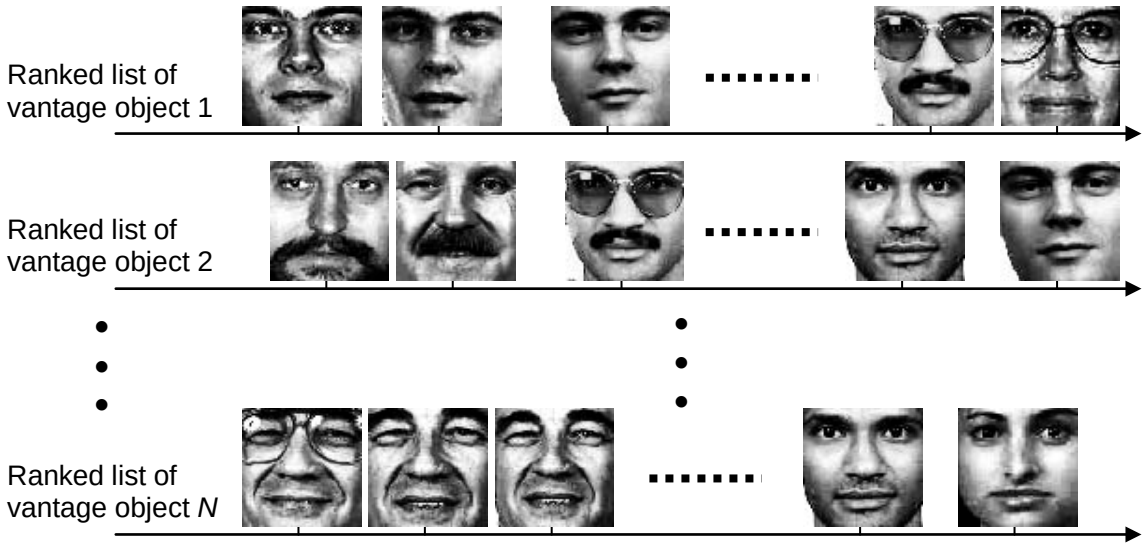


Figure 5.1 Ranked lists for different vantage objects

The vantage objects used should be able to measure different 'properties' of the objects under consideration [153]. In [151], eigenfaces were used as the vantage objects. However, for face recognition, the best performance will be achieved if those face images belonging to the same class are placed as close together as possible in the respective ranked list, while images of different classes are positioned as far apart as possible. Therefore, in our algorithm, discriminative features based on Gabor wavelets are used to extract the different 'properties' of faces [28, 29, 34, 38, 80]. It is not necessary to use all the Gabor features to form the vantage objects, but only those Gabor jets which have the greatest discriminative power. A set of Gabor jets will have the greatest discriminative power if the ratio of the between-class scatter to the within-class scatter is a maximum [68, 70, 80]. The higher this ratio is, the greater the discriminative power of the set of Gabor jets is. Then, those sets of Gabor jets

with the greatest discriminative power will be used for constructing the vantage objects.

The organization of this chapter is as follows. The basic techniques related to the proposed algorithm are described in Section 5.2. Section 5.3 will present the selection of Gabor jets to form vantage objects, and the construction of a condensed database. Experimental results for different ways of constructing the vantage objects, the use of different numbers of Gabor jets for each vantage object, and the number of vantage objects to be used will be described in Section 5.4. Section 5.5 will present the performance of our proposed index scheme with the use of linear discriminant analysis (LDA) for face recognition. Then, we will conclude this chapter in the last section.

5.2 Techniques Related to Our Proposed Scheme

Our fast searching algorithm for face databases is based on the use of vantage objects, which are constructed using the most discriminative Gabor features. Therefore, in this section, we will first describe the Gabor features used for face recognition, and then LDA will be presented.

5.2.1 Gabor Feature Extraction

Gabor features have been commonly used for face recognition [28, 29, 34, 38, 80]. The kernels of the Gabor wavelets have a similar shape to that of the 2-D receptive field profiles of the mammalian cortical simple cells [80]. The Gabor-wavelet representation can capture salient visual properties such as spatial localization, orientation selectivity, and spatial frequency characteristics [28]. In the spatial domain, a Gabor wavelet is a Gaussian function modulated by a complex exponential, which can be defined as follows:

$$\psi_{\omega,\theta}(x,y)=\frac{1}{2\pi\sigma^2}\exp\left(\frac{{x_1}^2+{y_1}^2}{2\sigma^2}\right)\left[\exp(i\omega x\cos\theta+i\omega y\sin\theta)-\exp\left(-\frac{\omega\sigma^2}{2}\right)\right] \quad (5.1)$$

where $x_1 = x\cos\theta + y\sin\theta$ and $y_1 = -x\sin\theta + y\cos\theta$. ($x$, y) denotes the pixel position in the spatial domain, ω is the radial centre frequency of the complex exponential, θ is the orientation of the Gabor wavelets, and σ is the standard deviation of the Gaussian function.

The image features are extracted by convolving the image $I(x, y)$ with the filters $\psi_{\omega,\theta}$ below. The magnitudes of the output are used as features of the image.

$$G_{\omega,\theta}(x,y)=\left\|I(x,y)\otimes\psi_{\omega,\theta}(x,y)\right\|. \quad (5.2)$$

In our algorithm, 5 different scales and 8 different orientations are selected for the Gabor wavelets, i.e. $\theta=\frac{\pi}{8}p$ and $\omega=\frac{\pi}{2\left(\sqrt{2}\right)^q}$,, where p = 0, …, 7, and q = 0, …, 4. Hence, at each pixel position, the Gabor feature or Gabor jet contains 40 coefficients. In addition, we set $\sigma = \pi/\omega$.

5.2.2 Fisher Linear Discriminant

Similar to Principal Component Analysis (PCA) [64], Fisher Linear Discriminant (FLD) or Linear Discriminant Analysis (LDA) is an efficient method for dimensionality reduction. PCA is optimal for the representation of training samples, so its performance with recognition is limited. For LDA, the projection vectors or discriminant vectors were selected in such a way that the ratio of the between-class scatter and the within-class scatter is maximized. Hence, samples of different classes can be discriminated as much as possible, and LDA has been one of the most popular projection techniques for feature extraction [82].

To compute the most discriminant projection vectors, the following Fisher's criterion is maximized:

$$F(v) = \frac{v^T S_b v}{v^T S_w v}, \tag{5.3}$$

where v is the transform matrix, S_b is the between-class scatter matrix, and S_w is the within-class scatter matrix. If the number of distinct subjects in the training set is N_c, and the number of samples in class i is n_i, then S_b and S_w are defined as follows:

$$S_w = \sum_{i}^{N_c} \sum_{j}^{n_i} \left(x_j^i - \mu_i\right)\left(x_j^i - \mu_i\right)^T \text{ and} \tag{5.4}$$

$$S_b = \sum_{i}^{N_c} N_c \left(\mu_i - \mu\right)\left(\mu_i - \mu\right)^T, \tag{5.5}$$

where μ is the mean of all the training samples, μ_i is the mean of the training samples of class i, and x_j^i represents the j^{th} sample in class i.

$F(v)$ is a ratio indicating the discriminative power of the projection vectors, which are the columns of v. The larger this ratio is, the higher the discriminative power of the projection vector is. The transform matrix that can maximize $F(v)$ can be obtained by computing the eigenvectors of the matrix $S_w^{-1}S_b$ [70]. This is equivalent to the simultaneous diagonalization of S_w and S_b [69, 73]. The latter can also cope with the small-sample-size (sss) problem. This sss problem can also be solved by the enhanced FLD models (EFM) proposed by Liu *et al*. in 2000 [79]. The dimension of the samples is first reduced by means of PCA, and then the simultaneous diagonalization of S_w and S_b is performed. Another effective method, proposed by Chen et al. [82], chooses the projection vectors in the null space of S_w. These null-space vectors can make the within-

class scattering become zero, and hence $F(v)$ becomes infinity if the corresponding between-class scattering is not zero.

5.2.3 Indexing Structure Using Vantage Objects

Using vantage objects is an indexing structure that relies on the similarity between training samples [153]. The idea is to compute the similarity of each object from some fixed objects, which are called vantage objects. For each vantage object, those samples similar to the objects are sorted to form a ranked list, as shown in Figure 5.2. It is expected that all of the similar objects should have the same type of similarity to the vantage objects, so they should be close to each other on the respective ranked lists. With a query object, its positions in the respective ranked lists are thus determined. As objects with similar features are located close together in the ranked lists, those nearest neighbors of the query object on the ranked lists should have similar features to the query, and they are therefore selected to form a condensed database for a more detailed and accurate search.

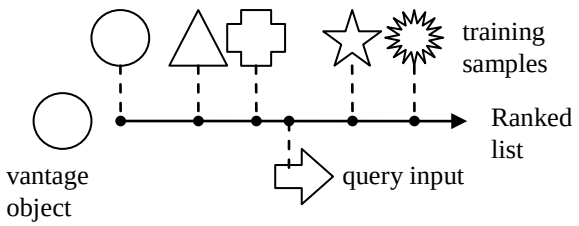


Figure 5.2 The ranked list of a vantage object with the training samples sorted according to their respective similarities to the vantage object.

The idea of using vantage objects to form an efficient indexing structure has been employed to construct condensed databases for speeding up the recognition process [151]. Eigenfaces are used as the vantage objects, and the corresponding ranked lists are constructed. The corresponding positions of a

query input on the respective ranked lists will be determined, and the corresponding nearest training faces in the different ranked lists are selected from the original large database to form a small condensed database for face recognition, instead of considering the original large database.

5.3 Construction of Vantage Objects Using Training Samples

In this section, we will present the construction of effective vantage objects, which includes feature extraction, feature selection to form the most discriminative vantage objects, and the construction of the ranked lists for different vantage objects. As described in Section 5.2.1, Gabor wavelets of 5 different scales and 8 different orientations are employed to extract facial features. The Gabor jets at different pixel positions are then combined to form different vantage objects. The vantage objects are constructed in such a way that the maximum discriminative power can be achieved, i.e. training samples of the same class will be as close together as possible on the ranked list, while training samples of different classes are placed as far apart as possible.

5.3.1 Feature Extraction

The Gabor filters can be used to extract information about local image regions effectively, and these extracted features can be invariant to translation, scale and rotation [80]. However, if a filter bank with filters of 5 different scales and 8 different orientations is applied to an image of size 64×64, a total of 163,840 complex Gabor coefficients, and hence magnitudes, will be generated. This large dimension makes the computation time required in the searching process very lengthy.

For database indexing, it is not necessary to use all the Gabor coefficients; only those with discriminative power will be chosen. The Gabor coefficients at a pixel position form a Gabor jet, and the Gabor jets can be combined to construct discriminative features for forming a vantage object. The following symbols are used throughout this section:

N_{VO}: number of vantage objects used;
N_J: number of Gabor jets selected for a vantage object;
N_V: number of projection vectors selected for a vantage object;
VO_n: the n^{th} vantage object;
$J_{(n,m)}$: the m^{th} Gabor jet of the n^{th} vantage object; and
J_i: the i^{th} Gabor jet of an image, where an image of size 64×64 should have 4,096 Gabor jets and i = 1, …, 4096.

The number of vantage objects, and hence the number of ranked lists, is N_{VO}, and the number of Gabor jets selected from each training sample for a vantage object is N_J. Then, the n^{th} vantage object VO_n will be constructed based on N_J Gabor jets from each training sample, which will make the vantage object have the highest discriminative power. The set of Gabor jets used is denoted as follows:

$$VO_n = \left\{ J_{(n,1)}, J_{(n,2)}, ..., J_{(n,N_J)} \right\},\ n = 1, 2, ..., N_{VO}, \qquad (5.6)$$

where $J_{(n,m)}$ represents the m^{th} Gabor jet selected for the n^{th} vantage object. Hence, each training sample will contribute a feature vector that is a concatenation of N_J Gabor jets and has a dimension of $40N_J$. The N_J Gabor jets are selected such that the Fisher's criterion function $F(v)$ is maximized. The corresponding projection vectors then form a vantage object, which will be used to construct a ranked list for indexing. The number of Gabor jets selected, N_J, and the number of vantage objects used, N_{VO}, have to be pre-determined. The optimum values for these variables will be determined empirically by experiments in Section 5.4. In the

following sections, we will describe how to select the Gabor jets based on FLD, and how the ranked lists are constructed. In order to keep the computation requirement low, only a few Gabor jets from each image will be considered.

5.3.2 Selection of Gabor Jets

In order to construct effective ranked lists, the Gabor jets selected to form the vantage objects should have the highest discriminative power. As described in Section 5.2.2, a set of projection vectors is derived by applying FLD to a set of Gabor jets at specific pixel positions extracted from each training sample, such that the criterion function (5.3) is maximized. The set of projection vectors comprises the columns of the transform matrix *V*. The projection vector in *V* with the largest eigenvalue is considered to possess the highest discriminative power, and so on. The Gabor jets are chosen such that the corresponding eigenvalues are a maximum.

5.3.3 Schemes for Selecting the Most Discriminative Sets of Gabor Jets

To select the *n* Gabor jets from the training samples which have the highest discriminative power, a brute-force approach can be employed by considering all the possible combinations. However, this approach is too computationally intensive to obtain the *n* most discriminative Gabor jets. For example, for a 64×64 image, there are 4,096 Gabor jets in the image, and the number of possible ways to select 3 Gabor jets is ${}_{4096}C_3 \cong$ more than 10G. Therefore, in this section, we will describe two efficient methods to select Gabor jets such that their discriminative powers are as high as possible. The first scheme will balance the discriminative power of every vantage object as much as

possible, while the second scheme will generate vantage objects with the highest discriminative power first.

In the selection of the Gabor jets, we will also consider the spatial redundancy among them. The closer two Gabor jets are, the greater the spatial redundancy between them. Therefore, to minimize the spatial redundancy between the Gabor jets for a vantage object, those Gabor jets within the neighborhood of a selected Gabor jet will no longer be chosen. Figure 5.3 illustrates a Gabor jet selected for a vantage object; then, the Gabor jets inside the square with the selected Gabor jet at the centre and a size of $R = 2r+1$ will not be selected. In our experiments, we will determine the optimal value of r to be used so as to achieve the best performance in terms of both the computational complexity for Gabor jet selection and the efficiency of the condensed database.

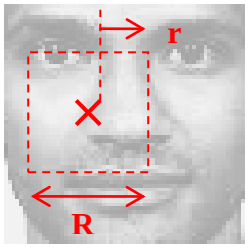


Figure 5.3 A window is set to prevent the spatial redundancy between the selected Gabor jets.

5.3.3.1 Scheme 1: Vantage Objects with Balanced Discriminative Power

In this scheme, the selection of the Gabor jets for each of the vantage objects is carried out together, and hence the discriminative power of the respective vantage objects will be similar. To reduce the required computation, we adopt a greedy algorithm in selecting the Gabor jets. We first consider feature vectors composed of one Gabor jet only. At each pixel position, the Gabor jets of the training samples are used, and their corresponding discriminative powers are measured by using (5.3). The first N_{VO} Gabor jets with the highest discriminative

powers are selected, and are assigned as the first Gabor jet $J_{(n,1)}$ of the N_{VO} vantage objects. Having selected the first Gabor jet for each vantage object, the second Gabor jets will be identified. Those Gabor jets that have already been selected will no longer be considered. In addition, as described previously, those Gabor jets within the neighborhoods of the selected Gabor jets will not be chosen. Now, each feature vector is composed of two Gabor jets: one is the first selected Gabor jet, and the other is the Gabor jet at a remaining possible pixel position. Then, (5.3) is applied again, and the combinations that result in the highest discriminative power will be selected for each of the vantage objects. This process is continued until N_J Gabor jets have been selected for each vantage object.

When selecting one more Gabor jet for a vantage object, all the previously selected Gabor jets will be considered, and those Gabor jets within their neighborhoods will not be selected. Therefore, if the window size R is set at a large value, the pre-set number of Gabor jets for a vantage object, N_J, may not be reached. When this situation occurs, N_J will be adjusted to the actual number of Gabor jets being assigned to the vantage objects.

5.3.3.2 Scheme 2: Vantage Objects with the Highest Discriminative Power First

In this scheme, the vantage objects are constructed one by one, with the first one having the highest discriminative power, and the last one having the least discriminative power. Similarly to Scheme 1, the feature vectors of one Gabor jet are considered, and the Gabor jet with the highest discriminative power is selected as the first Gabor jet $J_{(1,1)}$ for the first vantage object VO_1. Then the second Gabor jet will be selected in such a way that this Gabor jet is outside the neighborhood of the previous selected Gabor jet and, when combined with the first Gabor jet, its discriminative power is the highest. Having selected N_J Gabor

jets for the first vantage object, the construction of the second vantage object VO_2 will be started, and so on until the required number of vantage objects N_{VO} is produced.

5.3.4 Construction of Vantage Objects and the Corresponding Ranked Lists

Using the selection scheme described in Section 3.3, we can determine sets of Gabor jets at specific pixel positions that produce projection vectors having the highest discriminative powers. The dimension of the training samples is $40N_J$. Fisher linear discriminant (FLD) is applied to the training samples; N_V eigenvectors or projection vectors of dimension $40N_J$ and with the largest eigenvalues are used for each vantage object. When the corresponding feature vectors from the training samples are projected onto the N_V projection vectors of a vantage object, the dimension of the training samples is reduced from $40N_J$ to N_V, with the ratio of the between-class scatter and within-class scatter being maximized. Based on the coefficients obtained by projecting the face images in the database onto the projection vectors, these images can be ranked either in ascending or descending order to form ranked lists. Since there are N_{VO} vantage objects, and each vantage object has N_V projection vectors, so N_{VO} ranked lists can be constructed. Each of the ranked lists has a dimension of N_V. In other words, we will construct N_{VO} N_V-dimensional ranked lists. Figure 5.4 illustrates how the projection vectors are derived for the vantage objects, and the ranked lists used for the construction of condensed databases.

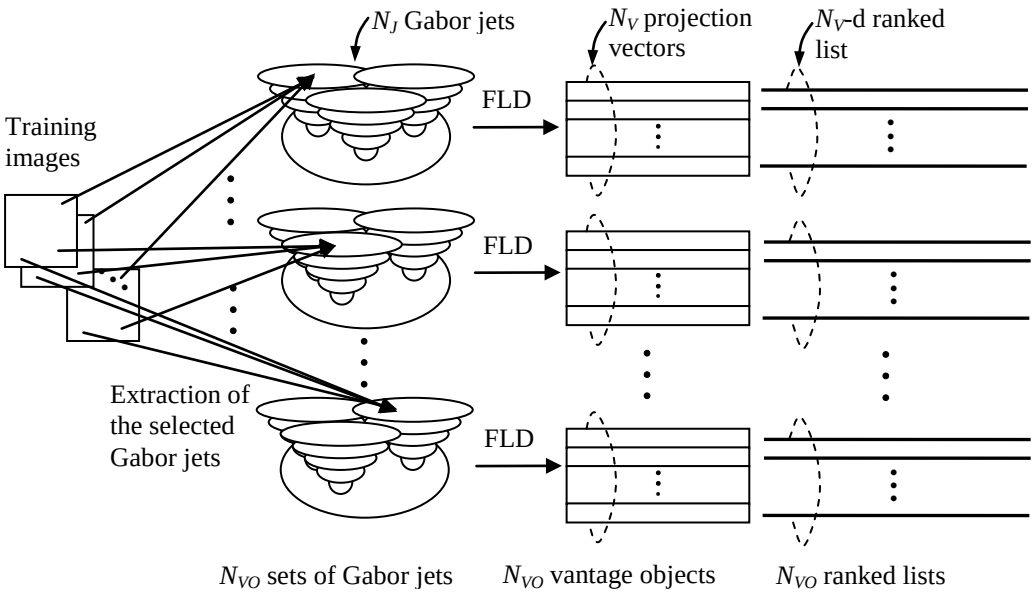


Figure 5.4 Construction of the projection vectors for the vantage objects and the ranked lists.

The length of – i.e. the number of samples in – each ranked list is equal to the number of face images, N, in the database. Each element in the ranked list contains two values; one is the projection coefficient used for sorting in the ranked list, and the other is the label of a sample in the database. For a query sample, its corresponding projection coefficients are computed, and their positions in the respective ranked lists can be searched using a binary search. For N_{VO} vantage objects and N_V projection projects for each vantage object, the computation required is about $N_{VO}N_V\lceil \log_2 N \rceil$, where $\lceil x \rceil$ represents the smallest integer larger than x. Hence, the computations required to locate the positions of a query input in the respective ranked lists are proportional to N_{VO} and N_V, and to the logarithm of N. Consequently, N_{VO} and N_V should be kept at a value as small as possible so as to reduce the computations required, while the size of the condensed database, which guarantees the face image in the database of the same class as the query input to be included, can be as small as possible.

5.3.5 Searching of Ranked lists to Construct a Condensed Database

For each query image, a corresponding small-sized condensed database will be generated by selecting similar samples to the query image from the original large database. In our algorithm, this condensed database is formed by selecting the M_{VO} nearest neighboring samples of the query image in each ranked list. With the N_{VO} ranked lists formed from the N_{VO} vantage objects, $N_{VO} \times M_{VO}$ training samples will then be extracted for the condensed database with respect to the query sample. However, some of the selected samples from the different ranked lists may be identical. For example, in Figure 5.5, where each ranked list is one-dimensional (i.e. $N_V = 1$), some samples, such as C and D, are close to the query sample in all three lists. Therefore, the samples C and D will be selected once only to form a condensed database for the query sample. Hence, the exact number of training samples in the condensed database, denoted as M_{con}, is smaller than or equal to $N_{VO} \times M_{VO}$.

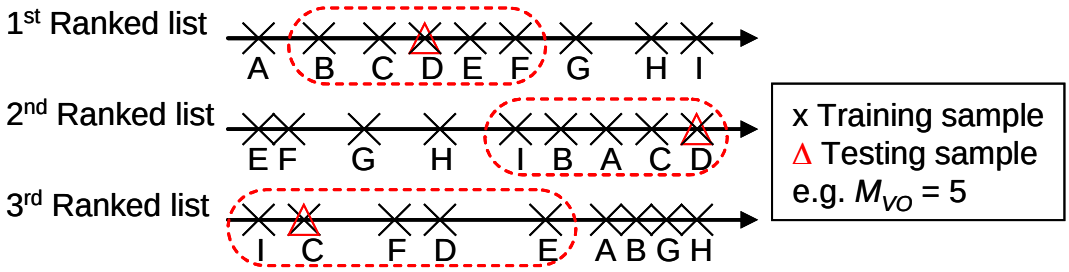


Figure 5.5 The selection of M_{VO} neighboring training samples from the ranked lists of a number of vantage objects.

When more than one projection vector is used for each vantage object, i.e. $N_V > 1$, the ranked list for each vantage object will become multi-dimensional. In other words, M_{VO} neighboring training samples will be selected from a N_V-d space, as shown in Figure 5.6, where M_{VO} is set at 6. Searching nearest samples

in the N_V-d space is time-consuming. However, an efficient search of the nearest samples can be achieved by considering each projection coefficient to form one ranked list, and these N_V ranked lists are searched simultaneously. In other words, a N_V-d ranked list can be viewed as N_V 1-d ranked lists. A neighboring sample of a query input should also be located in the vicinity of the query on all the 1-d ranked lists of a vantage object. In our scheme, each sample on the ranked lists of a vantage object is associated with a counter. Then, the nearest samples to the query on each of the 1-d ranked lists are checked one by one, and have their counters incremented by one. When the count of a sample is equal to N_V, this means that it is close to the query image, to a certain degree, in all the N_V 1-d ranked lists. This sample is then selected and placed in the condensed database. The next neighboring samples are checked in the same manner until M_{VO} training samples have been selected.

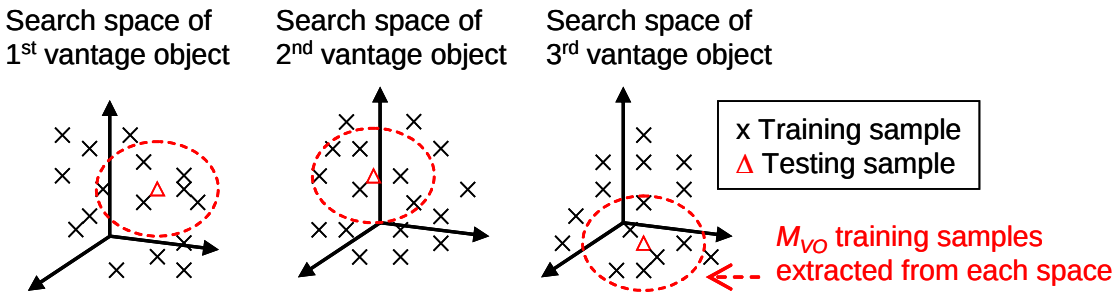


Figure 5.6 The extraction of M_{VO} neighboring training samples in the search spaces of a number of vantage objects.

5.4 Evaluation and Experiments

In this section, we will first describe the pre-processing of the face images for face recognition, and then the database used in the experiments. This will be followed by an evaluation of the performances of the different approaches for our indexing schemes, using different parameter settings.

5.4.1 Pre-processing of Training Samples

All the face images are first aligned and normalized based on the position of the two eyes [58], and are cropped to a size of 64×64. Histogram equalization is then performed on the cropped face images. Figure 5.7 shows some of the pre-processed face images. Finally, each image is normalized to zero mean and unit variance.

Figure 5.7 Some pre-processed face images.

To employ FLD, at least 2 samples are needed for each class or distinct person. If only one sample is available for a class, an additional sample is produced by flipping the available sample about the vertical axis passing through the mid-point of the two eyes. However, these mirror images will affect, to a certain extent, the positions of the most discriminative Gabor jets, even though the left and right sides of the face are similar to each other. To minimize this effect, the importance of the original sample is increased by summing up the original image with a larger weight, and the mirror image with a smaller weight, as shown in the following equation:

$$I_{Extra}(x,y) = \frac{w_1 \cdot I_{original}(x,y) + w_2 \cdot I_{Mirror}(x,y)}{w_1 + w_2}, \qquad (5.7)$$

where w_1 and w_2 ($w_1 > w_2$)are the weights of the original image and the mirror image, respectively. Some samples generated by using (5.7) are illustrated in Fig. 5.8.

Figure 5.8 The corresponding samples generated from Figure 5.7 using (5.7).

5.4.2 Face Database

We use a subset of the FERET database [167] to analyze the performance of our algorithm. There are 1,762 samples corresponding to 1,010 distinct subjects in the "fa" set, and 1,518 samples corresponding to 1,009 distinct subjects in the "fb" set. All the samples in the "fa" set were used as training samples, while those in the "fb" set were for testing. However, a few of the "fb" samples do not have any corresponding "fa" samples; these samples are excluded from the experiments. Moreover, in order to use FLD, at least two samples are required for each distinct subject. Some subjects in the "fa" set have one sample only, so an additional sample is generated by flipping. Consequently, 2,344 and 1,357 images are available for training and testing, respectively, corresponding to 1,003 distinct subjects.

5.4.3 Selection of Gabor Jets for Vantage Objects

First of all, we investigate the distribution of the discriminative powers of the Gabor jets in face images. We set N_{VO} = 4096, N_J = 1 and r = 0 in this experiment. With this setting, Scheme I and Scheme II for Gabor jet selection will produce the same results. Figure 5.9 shows the distribution of the discriminative power of the Gabor jets, whose pixel intensities represent the discriminative power of the Gabor jets at the corresponding pixel positions. It can be observed that most of the discriminative Gabor jets are located around the noses, mouths, chins and eyebrows. As the eyes are used for alignment, so the

appearances of the images at the eyes are similar, and the discriminative power there is therefore less.

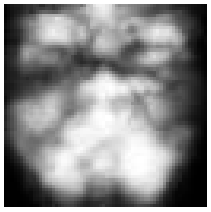


Figure 5.9 Distribution of the discriminative power of the Gabor jets.

We will first determine the optimum values of the four parameters: r, N_{VO}, N_J and N_V, so as to construct the most discriminative set of vantage objects. However, the possible combinations of these parameters are too numerous to be tested exhaustively. Therefore, in our experiments, we will first set the window parameter r at 0, 2, 4, and 6, and set the number of vantage objects used, N_{VO}, at 4, 6, 8, and 10. Note that the computation required to construct a condensed database depends on N_{VO}; the larger the N_{VO} is, the greater the computation required. Hence, the maximum number of vantage objects used in our experiment is set at 10. The maximum value of the window parameter r is set at 6. If r is set at a larger value, the number of Gabor jets available will not be sufficient to generate the number of vantage objects required. For the different settings of r and N_{VO}, we evaluate the corresponding performances of our algorithm when the number of Gabor jets N_J and the number of projection vectors N_V used for each vantage object are {4, 6, 8} and {4, 8, 12}, respectively. Then, the combination of r and N_{VO} that produces the best performance will be used in the rest of our experiments. In the next stage, we will determine the optimal values for the parameters N_J and N_V.

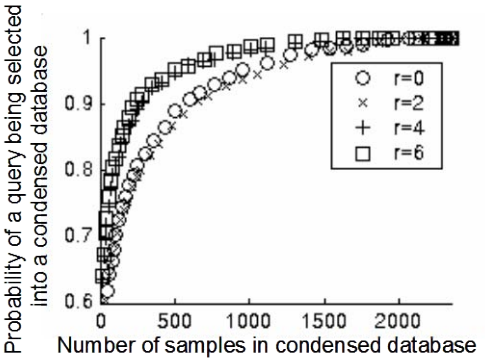
In our experiments, we will evaluate the performance of our schemes with different values of M_{VO}. These performances are measured based on the

probability of the matched training samples available to the condensed database at different M_{con} or different percentages of the original database size.

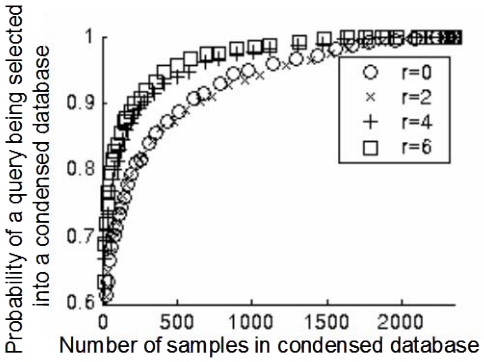
5.4.3.1 Optimal Values of r and N_{VO} in Scheme I

In this section, we will measure the relative performances of our proposed scheme using different values of r and N_{VO}. The optimal value for r is first determined by setting the number of vantage objects N_{VO} = {4, 6}, the number of Gabor jets N_J = {6, 8}, and the number of projection vectors used for each vantage object N_V = {8, 12}. Having determined the optimal value of r, we will determine the optimal value for N_{VO} when N_J = {4, 6} and N_V {4, 8, 12}. Figures 9 and 10 show the respective performances with the different settings of r and N_{VO}, which illustrate the probability of a query or testing sample being selected into a condensed database whose size is a particular percentage of the original large database. The higher the probability at a particular size of the condensed database, the better the performance of the corresponding parameter setting will be.

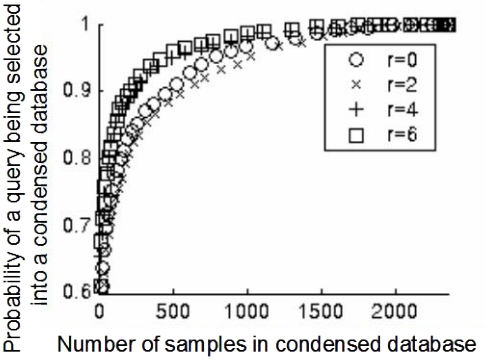
From the results, we can observe that the best performance can be achieved when r is set at 6, when N_{VO} is equal to 4 and 6. We also find that the performances are similar when N_{VO} is set at 8 and 10. Using more vantage objects, more discriminant information will be extracted to form the ranked lists. However, the computation required for constructing the condensed databases will also increase. Hence, in our subsequent experiments, we set N_{VO} at 8.



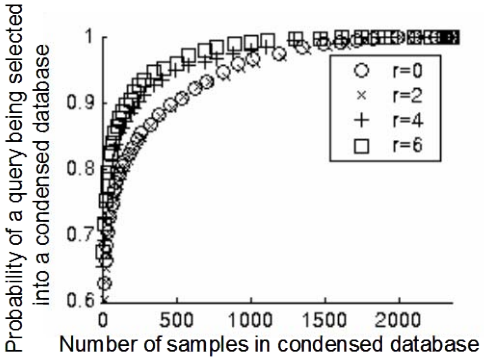
(a) N_{VO} = 4, N_J = 6, N_V = 8



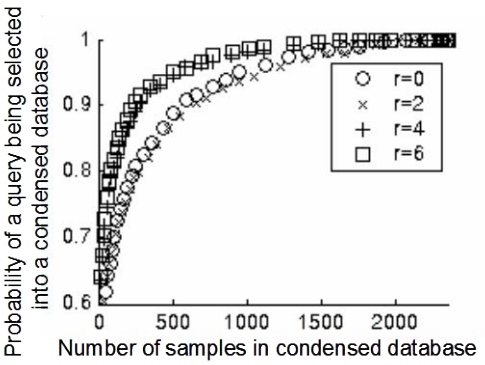
(b) N_{VO} = 4, N_J = 6, N_V = 12



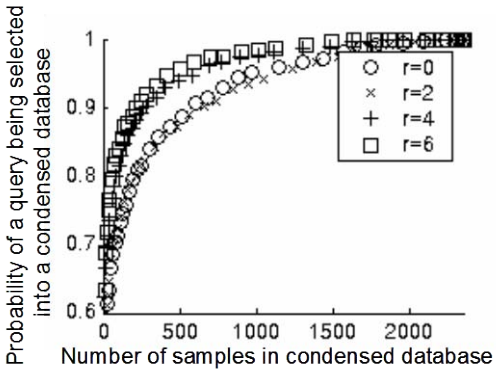
(c) N_{VO} = 4, N_J =8, N_V = 8



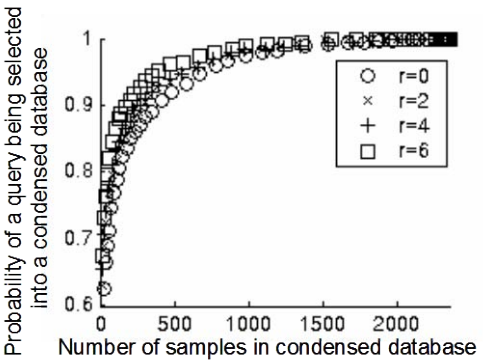
(d) N_{VO} = 4, N_J = 8, N_V = 12



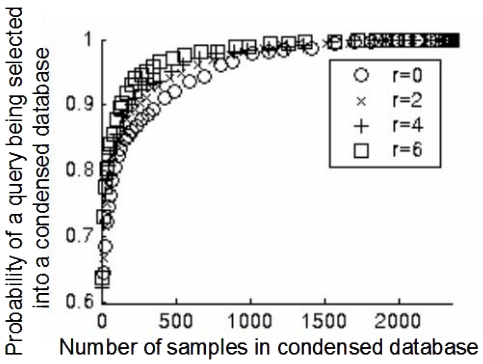
(e) N_{VO} = 6, N_J = 6, N_V = 8



(f) N_{VO} = 6, N_J = 6, N_V = 12

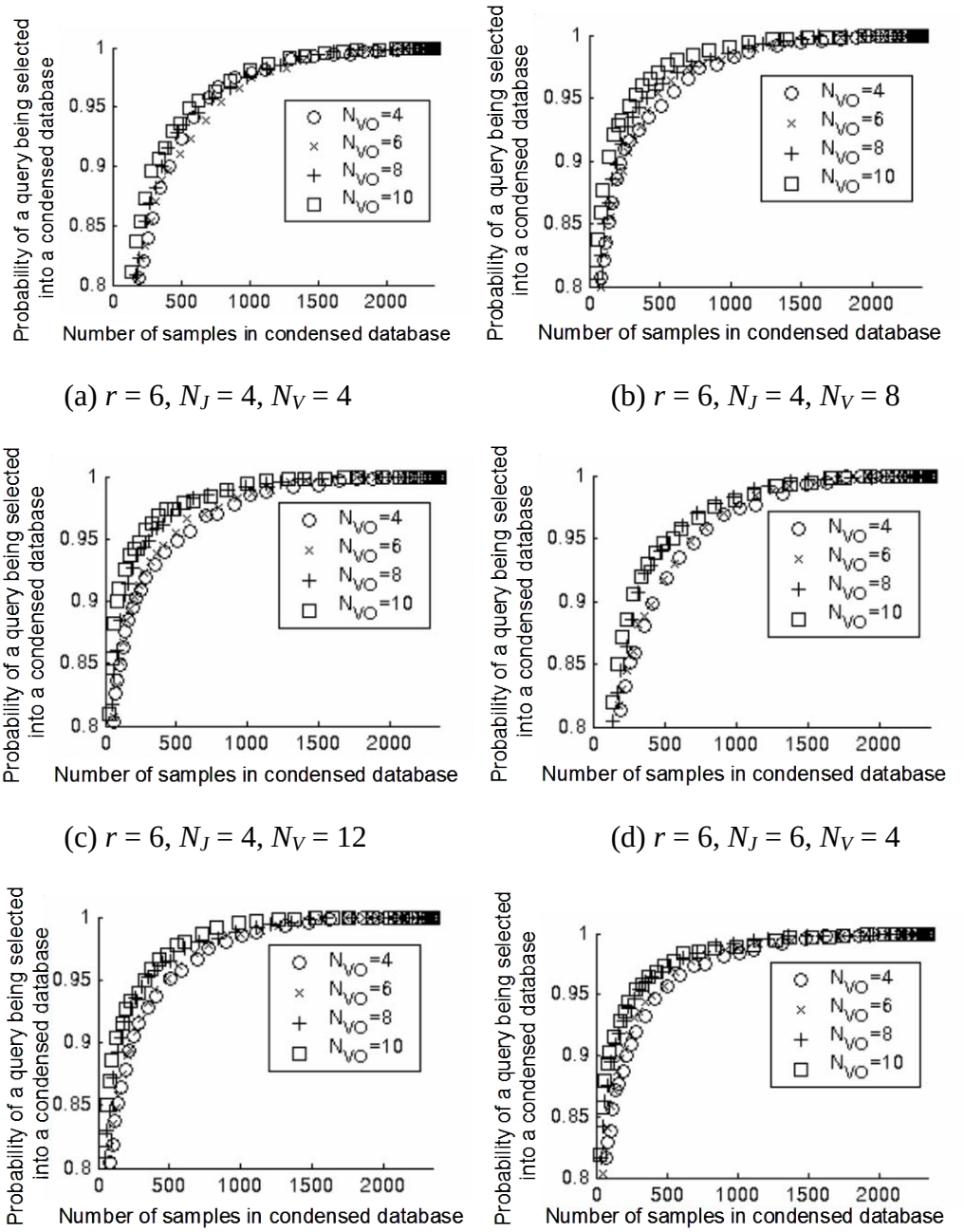


(e) N_{VO} = 6, N_J = 8, N_V = 8



(f) N_{VO} = 6, N_J = 8, N_V = 12

Figure 5.10 Performances for different settings under various *r* using Scheme I



(a) $r = 6$, $N_J = 4$, $N_V = 4$ (b) $r = 6$, $N_J = 4$, $N_V = 8$

(c) $r = 6$, $N_J = 4$, $N_V = 12$ (d) $r = 6$, $N_J = 6$, $N_V = 4$

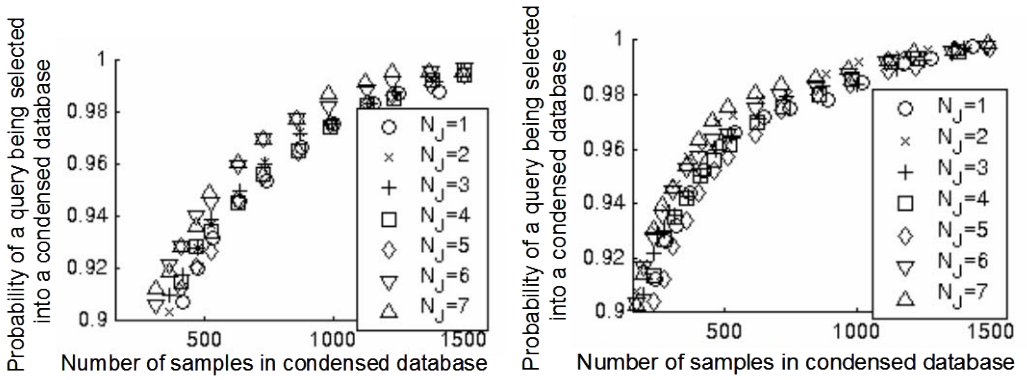
(e) $r = 6$, $N_J = 6$, $N_V = 8$ (f) $r = 6$, $N_J = 6$, $N_V = 12$

Figure 5.11 Performance for different settings of N_{VO} using Scheme I

5.4.3.2 Optimal Values of N_J and N_V in Scheme I

In this section, the relative performances of our proposed scheme using different values of N_J (the number of Gabor jets for a vantage object) and N_V

(number of project vectors for a vantage object) will be measured. Since we set r and N_{VO} at 6 and 8 in the previous section, respectively, the number of Gabor jets allowed to be selected for the vantage objects are fewer. N_J is limited to be equal to or smaller than 7. The number of optimal projection vectors used for each vantage objects is set at 4, 8, and 12. Figure 5.12 and Figure 5.13 show the respective performances for the different settings for different values of N_J and N_V, respectively.



(a) r = 6, N_{VO} = 8, N_V = 4 (b) r = 6, N_{VO} = 8, N_V = 8

Figure 5.12 Performance for different settings of N_J using Scheme I

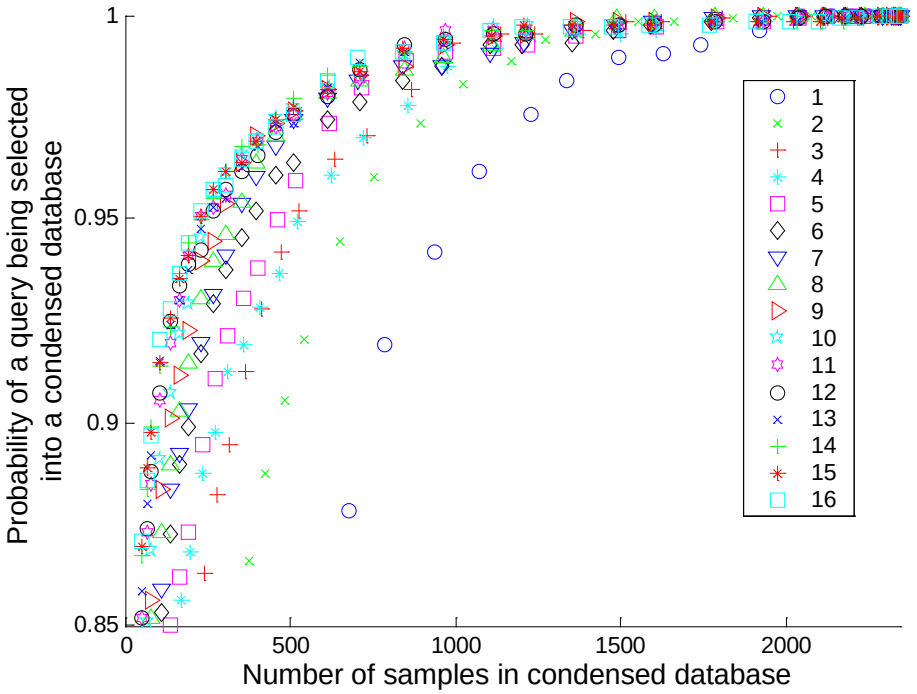


Figure 5.13 Performance for different settings of N_V using Scheme I

From the results, we can observe that the best performance can be achieved when N_J is set at the maximum value, i.e. 7, while N_V is equal to 4, 8, and 12. Selecting more Gabor jets for each vantage object will mean more information contained in the vantage objects. Although it is very computationally intensive to train up the vantage objects and determine the corresponding discriminant feature vectors for each vantage object, this can be done off-line. For testing, the computation required is dependent on N_V and N_{VO} only. If more projection vectors are used, more information about the samples is available at the expense of more computation required for constructing the condensed database. From the experimental results, it can be observed that the improvement decreases when the number of projection vectors continues to increase. Since the performance will be steady to a certain performance, we therefore set N_V to 10.

Now, for our proposed scheme, we set r, N_{VO}, N_J and N_V to 6, 8, 7 and 10, respectively. From Table 5.1, our approach can ensure that the probabilities of a

query face being selected to a condensed database of of a size 35%, 25%, 10%, and 5% of the original database are 99%, 98%, 95%, and 90%, respectively. The time for constructing the condensed database is around 95 milliseconds. The experimental results were conducted on a Pentium 4 3.2GHz computer system.

Table 5.1 The probabilities of matched training samples available in the condensed database and the corresponding size of the condensed database in term of the original database, as well as the corresponding runtimes in milliseconds required.

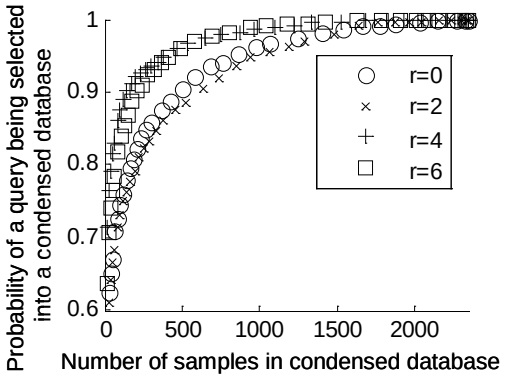
Probability of the matched training samples available in the condensed database	Percentage of size of condensed database in original training database	Time (milliseconds)
1	93.3%	101
0.99	36.1%	96
0.98	26.3%	95
0.97	19.5%	95
0.96	12.9%	95
0.95	11.4%	95
0.94	9.76%	95
0.93	9.76%	95
0.92	6.95%	95
0.91	6.95%	95
0.90	5.78%	95

5.4.3.3 Optimal Values of r and N_{VO} in Scheme II

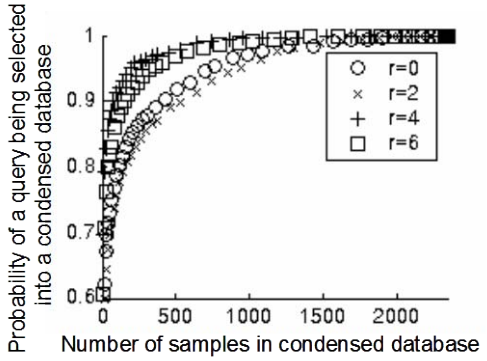
Similar to the process outlined in Section 5.4.3.1, we will measure the relative performance of Scheme II using different values of r and N_{VO}. The corresponding optimal settings of these two parameters are determined when the number of Gabor jets used for each vantage object is 6, and 8, respectively, while the number of optimal projection vectors used for each vantage object is 8, and 12, respectively. Figures 5.14 and 5.15 show the respective performances with the different setting of r and N_{VO}.

From the results, we observe that the best performance can be achieved when r is set at 4, while N_{VO} is equal to 6 or 8. The experiments also show that

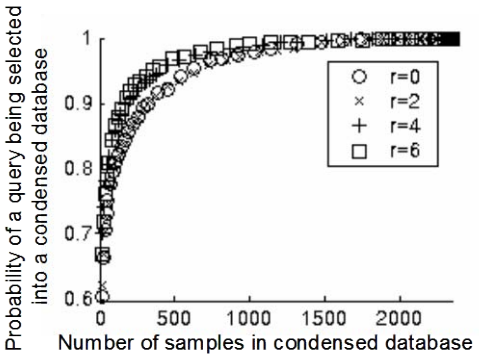
we can achieve the best performance when N_{VO} is set at 10. Therefore, we will set r = 4 and N_{VO} = 10 in the rest of our experiments when using Scheme II.



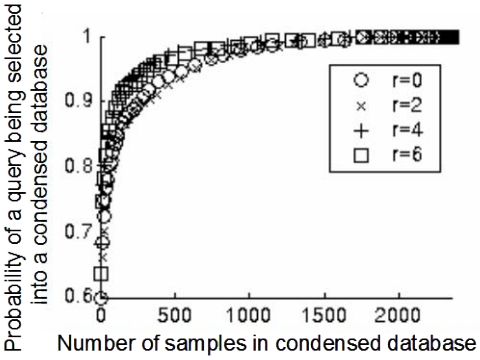
(a) N_{VO} = 6, N_J = 6, N_V = 8



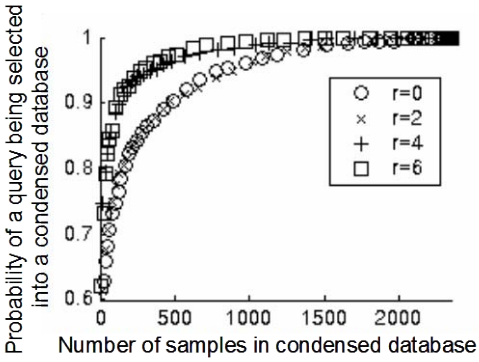
(b) N_{VO} = 6, N_J = 6, N_V = 12



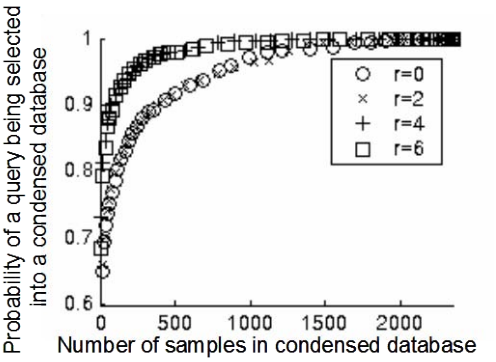
(c) N_{VO} = 6, N_J = 8, N_V = 8



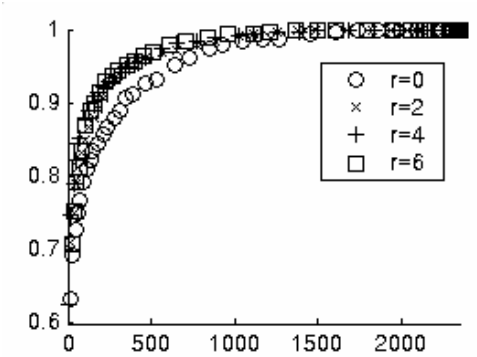
(d) N_{VO} = 6, N_J = 8, N_V = 12



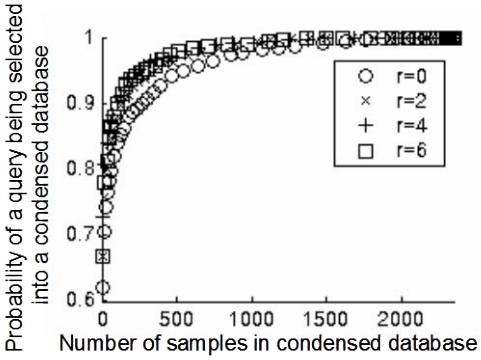
(e) N_{VO} = 8, N_J = 6, N_V = 8



(f) N_{VO} = 8, N_J = 6, N_V = 12

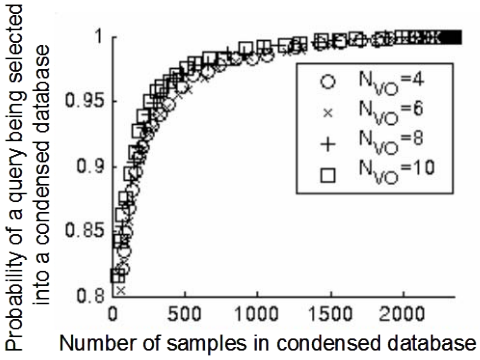


(g) N_{VO} = 8, N_J = 8, N_V = 8

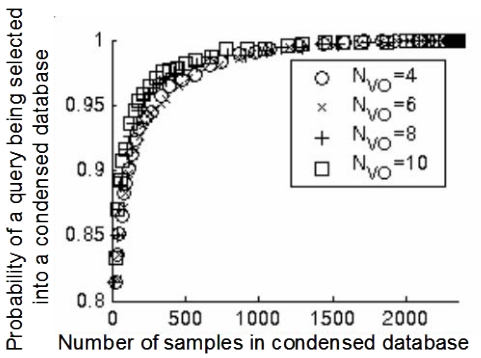


(h) N_{VO} = 8, N_J = 8, N_V = 12

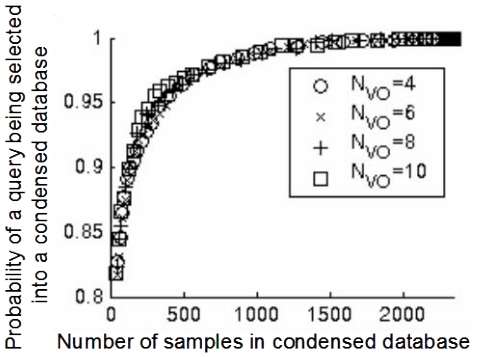
Figure 5.14 Performances for different settings of *r* using Scheme II



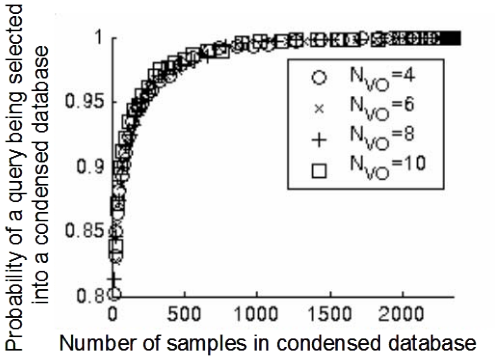
(a) r = 4, N_J = 4, N_V = 8



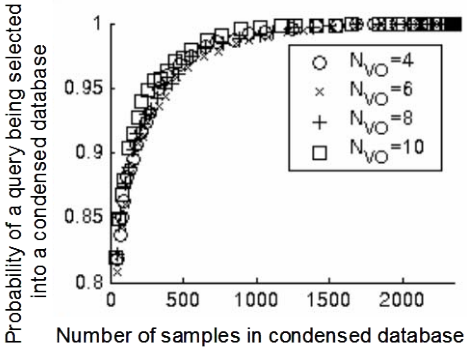
(b) r = 4, N_J = 4, N_V = 12

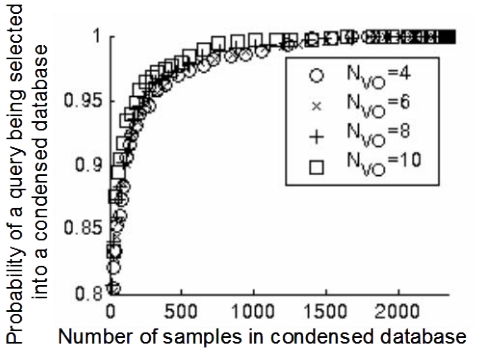


(c) r = 4, N_J = 6, N_V = 8

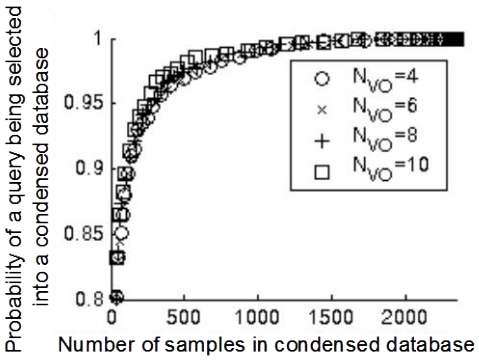


(d) r = 4, N_J = 6, N_V = 12

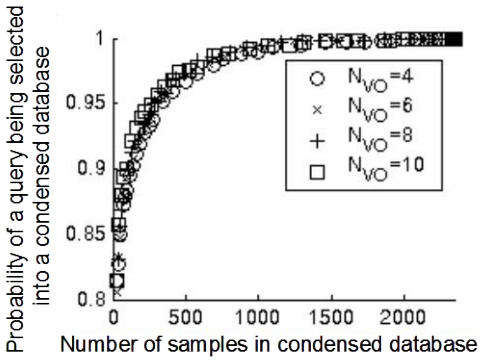




(e) r = 4, N_J = 8, N_V = 8



(f) r = 4, N_J = 8, N_V = 12



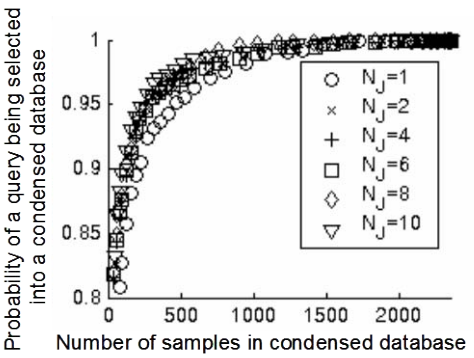
(g) r = 4, N_J = 10, N_V = 8

(h) r = 4, N_J = 10, N_V = 12

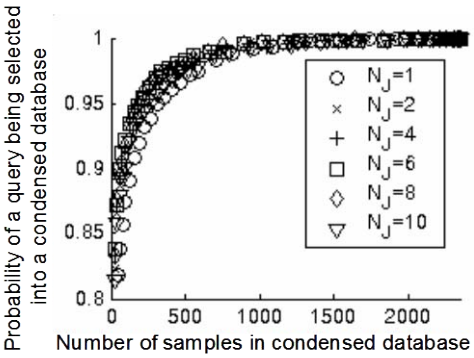
Figure 5.15 Performances for different settings of N_{VO} using Scheme II

5.4.3.4 Optimal Values of N_J and N_V in Scheme II

In this section, the relative performances of Scheme II using different values of N_J and N_V will be measured. We set r and N_{VO} at 4 and 10, respectively, as determined in the previous section. N_J will be set to 1, 2, 4, 6, 8, and 10. The number of optimal projection vectors used for each vantage objects is set at 8 and 12. Figures 5.16 and 5.17 show the respective performances for the different settings for different values of N_J and N_V, respectively.



(a) r = 4, N_{VO} = 10, N_V = 8



(b) r = 4, N_{VO} = 10, N_V = 12

Figure 5.16 Performance for different settings of N_J using Scheme II

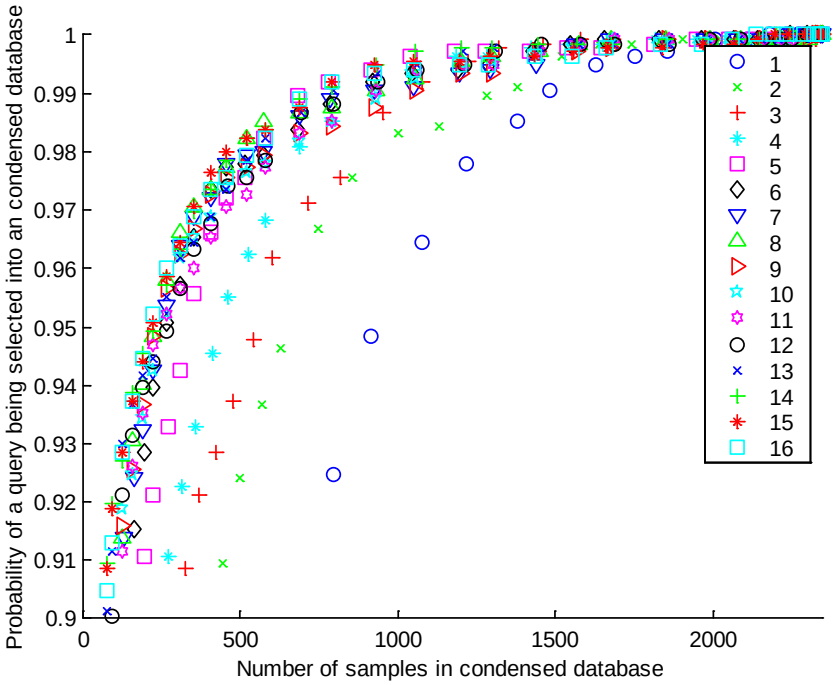


Figure 5.17 Performances for different settings of N_V using Scheme II

From the results, we observe that the best performance can be achieved when N_J is set at the maximum value, i.e. 10, while N_V is equal to 8 and 12. It can also be observed that the improvement decreases when the number of projection vectors continues to increase. Since the performance will be steady to a certain level, we therefore set N_V to 8.

In summary, for Scheme II, we set r, N_{VO}, $N_{J,}$ and N_V to 4, 10, 10, and 8, respectively. From Table 5.2, our approach can ensure that the probabilities of a query face being placed to a condensed database of size 40%, 15%, 10%, and 5% of the original database are 99%, 97%, 95%, and 90%, respectively. The time required for constructing the condensed database is around 133 milliseconds. The experimental results were conducted on a Pentium 4 3.2GHz computer system.

It can be observed that the time required for scheme II is larger than that for scheme I, and the performance of scheme II is not much better than the

performance of scheme I. Therefore, we choose to use scheme I in the following section.

Table 5.2 The probabilities of matched training samples being selected to condensed databases of different sizes, and the corresponding runtimes required, in milliseconds.

Probability of the matched training samples available in the condensed database	Sizes of the condensed databases in terms of percentages (%) of the original training database	Runtime (milliseconds)
1	99.30%	149
0.99	39.64%	139
0.98	22.13%	133
0.97	15.40%	133
0.96	13.26%	133
0.95	11.43%	133
0.94	8.23%	133
0.93	6.90%	132
0.92	6.90%	132
0.91	5.52%	132
0.90	5.52%	132

5.4.4 Performance using more projection vectors (N_V) for recognition

Having constructed the condensed database, face recognition can then be performed based on this smaller database. The projection of the query image onto the projection vectors of the respective vantage objects will form a feature vector, which can be used to represent the query image. The dimension of this feature vector is $N_V N_{VO}$. The optimal values for these two parameters are $N_V = 10$ and $N_{VO} = 8$, respectively, so the dimension of the feature vector used for face recognition in the condensed database is 80. Since the database is now of a small size, more projection vectors can also be used so as to further increase the recognition rate. Table 5.3 shows the recognition rates with different-sized

condensed databases (as percentages of the full database). The Euclidean distance is used for the distance measure. It can be observed that the performance does not improve when more N_V is used for recognition. Hence, in the next section, we will use all the selected Gabor jets to form a single feature vector for face recognition.

Table 5.3 The recognition performance using Scheme I under different-sized condensed databases.

Sizes of condensed databases as percentage of the full database	Recognition rate		
	$N_V = 10$	$N_V = 20$	$N_V = 40$
0.05	0.8047	0.8224	0.8003
0.10	0.8282	0.8298	0.8187
0.15	0.8349	0.8335	0.8305
0.25	0.8379	0.8364	0.8335

5.4.5 Performance for face recognition using LDA

Since the size of the condensed database is small, more features can be chosen for face recognition. To improve the performance, we will consider all the selected Gabor jets of the query image as a feature vector. The dimension of this feature vector is $40N_{VO}N_J$. The optimal values for these two parameters are $N_{VO} = 8$ and $N_J = 7$, respectively, so the dimension of the feature vectors is 2,240. LDA is then applied to these feature vectors for face recognition. The eigenvectors corresponding to the first N_{V2} largest eigenvalues will be used to transform the feature vector of the query image. In addition to the 56 Gabor jets selected, other Gabor jets with a high discriminative power can also be selected and added to form the final feature vector. Hence, in this experiment, we will evaluate the performance with the final feature vectors constructed using 56, 70, 75, 80, and 90 Gabor jets. Table 3 shows the recognition rates for different numbers of Gabor jets. The second column in the table shows the optimal

number of eigenvectors for the different numbers of Gabor jets used, such that the recognition rate is a maximum. It can be observed that the recognition rate using LDA is higher than that without LDA (refer to Table 5.3). Moreover, the recognition rate is highest when 75 Gabor jets are used for recognition.

Tables 5.5 and 5.6 tabulate the recognition rates and runtimes, respectively, required for searching over the large database for different sized condensed databases. The Euclidean distance and the cosine distance are used for the distance measure. The runtimes required for 1,357 queries are measured in milliseconds. It can be observed that the runtimes required for searching over the large database are 25,081 ms and 64,952 ms when the Euclidean distance and the cosine distance, respectively, are used. The Euclidean distance measure is relatively simple, so there is no advantage in using of the proposed indexing scheme in this case. Nevertheless, a significant improvement can be achieved when a more accurate but more complicated distance measure, such as the cosine distance, is used. With the cosine distance measure, when the size of the condensed database is 10% of the full database, the runtime required by our proposed method is only 28,796 ms, with a recognition rate of 91%. The runtime required by the proposed method is much smaller than that required for searching over the large full database. When the size of the condensed database is 50% of the full database, the recognition rate is 94.25%, which is even a little bit higher than that achieved by searching over the large database.

If the size of the database increases, more savings in terms of the runtime can be achieved. The reason is that, without using any indexing scheme, the required computation will increase linearly. However, with our indexing scheme (and with reference to Section 5.3.4), the rate of increase is proportional to $\log_2 N$,

where N is the size of the database. In addition, when the database size is very large, a more accurate distance measure will be indispensable.

Table 5.4 The recognition rates using Scheme I under different numbers of Gabor jets and the corresponding optimal number of eigenvectors.

No. of Gabor jets	No. of eigenvectors used (optimal)	Recognition rate
56	310	0.8828
70	398	0.8791
75	**379**	**0.8850**
80	395	0.8843
90	410	0.8762

Table 5.5 The recognition rates and runtimes required for searching over the large full database.

	Recog. rate (L2 dist.)	Time (ms)	Recog. rate (cosine dist.)	Time (ms)
Normal method	0.8850	25081	0.9418	64952

Table 5.6 The recognition rates and runtimes required by Scheme I with different sizes of the condensed database.

Percentage of condensed DB to full DB	Recog. rate (L2 dist.)	Time (ms)	Recog. rate (cosine dist.)	Time (ms)
0.052	0.8386	21996	0.8732	23713
0.103	0.8666	23463	0.9138	28796
0.150	0.8777	24787	0.9256	31673
0.204	0.8814	25079	0.9322	32227
0.252	0.8836	25225	0.9359	34889
0.292	0.8850	25306	0.9374	38290
0.302	0.8850	26777	0.9374	39193
0.352	0.8843	27171	0.9388	42209
0.403	0.8843	28575	0.9410	45537
0.450	0.8843	28717	0.9410	47059
0.504	0.8850	28705	**0.9425**	**47492**

5.5 Conclusions

In this chapter, we have proposed an efficient indexing scheme for face recognition, which constructs a small-sized condensed database for a query image from a large database. Our approach performs feature extraction by selecting the most discriminant Gabor jets to form vantage objects, and then FLD is applied to these Gabor jets to determine the N_V most discriminant projection vectors. The projection vectors of a vantage object will form a multi-dimensional ranked list, which is then used to select similar samples to the query input to form a condensed database. Experimental results show that the probabilities of a query face being selected in a condensed database of 35%, 25%, 10%, and 5% of the original database size are 99%, 98%, 95%, and 90%, respectively. The time taken to construct the condensed database is around 95 milliseconds for a database of 2,344 samples. This runtime is much lower, and the recognition rate can be maintained, as compared to searching over the entire database. It is important to note that the computation required by our indexing scheme is proportional to the logarithm of the database size. Hence, a more significant saving in runtime can be achieved when a larger database is used.

Chapter 6: Conclusions and Future Work

6.1 Conclusion on our current work

In this thesis, we have first introduced the basic concepts and steps for face detection and face recognition. Some popular techniques and the recent development of the technology for both face detection and recognition have been briefly reviewed. We have made three contributions in this thesis: an effective template-based face detection approach, an efficient feature extraction algorithm for face recognition, and an indexing structure for face recognition in a large database.

For efficient face detection, we have proposed a novel template-based method, namely Spatially Maximum Occurrence Template (SMOT), with the Gaussian mixture model. It is difficult to collect training faces under all possible conditions. With the proposed template, projecting face candidates onto the SMOT space will result in a procedure of "standardizing" the face candidates. With a limited number of face examples for training, our method can still detect faces under different conditions.

We have also proposed two methods for face recognition. One method is the use of a simplified version of Gabor wavelets (SGWs) as features for face recognition. Using these simplified features can achieve a performance level similar to that with the original Gabor wavelets (GWs). Fast algorithms for feature extraction based on SGWs at different orientations have also been described. Experimental results show that the runtime for feature extraction using SGWs is 4.39 times faster than that with GWs implemented by using the fast

Fourier transform. Therefore, SGWs can replace GWs for realizing real-time applications and processing.

The other proposed method concerns face recognition in a large database, therefore an efficient indexing scheme is devised. In our method, a condensed database, whose size is much smaller than the original large database, is constructed for a query image from a large database. Feature extraction is performed by selecting the most discriminant Gabor jets, and FLD is then applied to these Gabor jets to determine a specific number of most discriminant projection vectors. These projection vectors will form ranked lists, which are then used to select similar samples to the query input to form the condensed database. The runtime for constructing a condensed database is much lower than that required to search the complete database, therefore a more computational and accurate recognition algorithm can be adopted in the condensed database without any degradation of recognition accuracy.

6.2 Future Work

Our proposed indexing scheme for a large database can be further enhanced for face recognition. One direction is to adopt a more computational but accurate recognition algorithm to recognize those images in the condensed database. Another direction is to develop a cascade of indexing steps with different features, similar to (Viola and Jones, 2001) [145] for face detection, to form a complete face recognition system. Different indexing stages use different features to select a number of similar samples to the input query. The selected samples will form the condensed database, which will be considered at the next indexing stage. This structure of a face recognition system is suitable for a huge database, since only a small number of features will be considered at each stage,

and the size of the condensed database will become smaller after each indexing stage. Hence, the runtime for face recognition is lessened.

In our proposed indexing scheme for a large database, we treat the Gabor jets as a basic unit to form a vantage object. However, some Gabor coefficients in a Gabor jet may be redundant or irrelevant for face recognition. Therefore, to further improve the performance, we can select the Gabor coefficients as a basic unit to form a vantage object. A better performance should be achieved, since there is no redundant information in the vantage objects. Moreover, the discriminative power of the Gabor features is measured by means of the ratio of the between-class scatter matrix and the within-class scatter matrix. It is also possible to use other methods to compute the discriminative power of the features, such as mutual information (Shen and Bai, 2006) [133]. Mutual information can be applied to obtain a set of informative and non-redundant Gabor features. Two classes – the intrapersonal difference class and the extrapersonal difference class – can be introduced to convert the *N*-classes problem into a binary class problem, since mutual information is used in the binary class problem. However, the samples in the extrapersonal difference class are much more numerous than the samples in the interpersonal difference class. To achieve a balance between the numbers of training samples from these two classes, we can randomly produce a subset of extrapersonal samples. But this will involve the challenge of making the subset as representative as possible of the whole set.

We can apply the techniques proposed in this thesis, i.e. face detection, feature extraction, and indexing method, for a complete video-retrieval system

based on facial image analysis. The related topics include key-frame representation, face tracking, video-shot partitioning, etc.

References

[1] E. Hjelmas and B.K. Low, "Face Detection: A Survey," Computer Vision and Image Understanding, vol. 3, no. 3, pp. 236-274, Sept. 2001.

[2] M.H. Yang, D.J. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 1, pp. 34-58, 2002.

[3] R.C. Gonzalez and R.E. Woods, "Digital image processing," Second Edition, Prentice Hall, 2002.

[4] K.M. Lam, "Multimedia information retrieval and management: technological fundamentals and applications," Springer, Chapter 19, "Search of human faces from a face database", pp. 405-431, 2003.

[5] R.O. Duda, P.E. Hart, and D.G. Stork, "Pattern classification," 2nd Edition, Wiley-Interscience, 2000.

[6] R. Chellappa, C.L. Wilson, and S. Sirohey, "Human and machine recognition of faces: a survey," Proc. IEEE, vol. 83, no. 5, pp. 705-741, May 1995.

[7] W. Zhao, R. Chellappa, and P.J. Phillips, "Face Recognition: A Literature Survey," ACM Computing Surveys, vol. 35, no. 4, pp. 399-458, Dec. 2003.

[8] D. Voth, "Face recognition technology," IEEE Intelligent Systems, vol. 18, no. 3, pp. 4-7, 2003.

[9] P.J. Phillips, P.J. Flynn, T. Scruggs, K.W. Bowyer, Jin Chang, K. Hoffman, J. Marques, Jaesik Min, and W. Worek, "Overview of the face recognition grand challenge," IEEE Conf. Computer Vision and Pattern Recognition, vol. 1, pp. 947-954, June 2005.

[10] T. Heseltine, N. Pears, J. Austin, and Z. Chen, "Face recognition: A comparison of appearance-based approaches," Proc. VIIth Digital Image Computing: Techniques and Applications, vol. 1, pp. 59-68, 2003.

[11] X. Lu, "Image analysis for face recognition," personal notes, 36 pages, May 2003.

[12] http://www.face-rec.org/

[13] H. Greenspan, J. Goldberger, and I. Eshet, "Mixture model for face-color modeling and segmentation," Pattern Recognition Letters, vol. 22, pp. 1525-1536, 2001.

[14] R.L. Hsu, M. Abdel-Mottaleb, and A.K. Jain, "Face detection in color images," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pp. 696-706, 2002.

[15] 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, pp. 103-114, 2003.

[16] K.W. Wong, K.M. Lam, and W.C. Siu, "An efficient color compensation scheme for skin color segmentation," Proc. Int'l Symposium Circuits and Systems, vol. 2, pp. II-676 - II-679, May 2003.

[17] Z. Liu, J. Yang, and N.S. Peng, "An efficient face segmentation algorithm based on binary partition tree," Signal Processing: Image Communication, vol. 20, pp. 295-314, 2005.

[18] T.Y. Chow, K.M. Lam, and K.W. Wong, "An efficient color face detection algorithm under different lighting conditions," Journal of Electronic Imaging, vol. 15, pp. 013015-1-10, 2006.
[19] K.M. Lam, "Multimedia Information Retrieval and Management: Technological Fundamentals and Applications," Springer, Chapter 19, "Search of human faces from a face database", pp. 405-431, 2003.
[20] B. van Ginneken, A.F. Frangi, J.J. Staal, B.M. ter Haar Romeny, and M.A. Viergever, "Active shape model segmentation with optimal features," IEEE Trans. Medical Imaging, vol. 21, no. 8, pp. 924-933, 2002.
[21] B. Erol, F. Kossentini, "Shape-based retrieval of video objects," IEEE Trans. multimedia, vol. 7, no. 1, pp. 179-182, Feb. 2005.
[22] A.C. Bovik, M. Clark, and W.S. Geisler, "Multichannel texture analysis using localized spatial filters," IEEE Trans. Pattern Anal. Mach. Intell. vol. 12, no. 1, pp. 55-73, 1990.
[23] B.S. Manjunath and W.Y. Ma, "Texture feature for browsing and retrieval of image data," IEEE Trans. Pattern Anal. Mach. Intell. vol. 18, no. 8, pp. 837-842, 1996.
[24] Y. Chen and R.S. Wang, "Texture segmentation using independent component analysis of Gabor features," 18th Int'l Conf. Pattern Recognition, vol. 2, pp. 20-24, August 2006.
[25] S.C. Zhang and Z.Q. Liu, "A real-time face detector," IEEE Int'l Conf. Systems, Man and Cybernetics, vol. 3, pp. 2197-2202, Oct. 2004.
[26] B. Balas and P. Sinha, "Receptive field structures for recognition," Neural Computation, vol. 18, no. 3, pp. 497-520, 2006.
[27] C.P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," Proc. Sixth Int'l Conf. Computer Vision, pp. 555-562, 1998.
[28] M. Lades, J.C. Vorbruggen, J. Buhmann, J. Lange, C. von der Malsburg, R. P. Wrutz, and W. Konen, "Distortion invariant object recognition in the dymanic link architecture," IEEE Trans. Comput., vol. 42, pp. 300-311, 1993.
[29] T.S. Lee, "Image Representation Using 2D Gabor Wavelets," IEEE Trans. Pattern Anal. Mach. Intell. vol. 18, no. 10, pp. 959-971, 1996.
[30] R. Porter and N. Canagarajah, "Robust rotation-invariant texture classification: wavelet, Gabor filter and GMRF based schemes," IEE Proceedings - Vision, Image and Signal Processing, vol. 144, no. 3, pp. 180-188, June 1997.
[31] L. Wiskott, J.M. Fellous, N. Kruger, and C. von der Malsburg, "Face recognition by elastic bunch graph matching," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 775-779, July 1997.
[32] L. Shams, and C. Malsburg, "The role of complex cells in object recognition," Vision Res. vol. 42, no. 22, pp. 2547-2554, 2002.
[33] D.H. Liu, K.M. Lam, and L.S. Shen, "Sampling Gabor features for face recognition," Proc. Int'l Conf. Neural Networks and Signal Processing, vol. 2, pp. 924-927, Dec. 2003.
[34] D.H. Liu, K.M. Lam, and L.S. Shen, "Optimal sampling of Gabor features for face recognition," Pattern Recognition Letters, vol. 25, no. 2, pp. 267-276, Jan. 2004.

[35] C. Liu, "Gabor-based kernel PCA with fractional power polynomial models for face recognition," IEEE Trans. Pattern Anal. Mach. Intell. vol. 26, no. 5, pp. 572-781, 2004.
[36] L.L. Huang, A. Shimizu, and H. Kobatake, "Robust face detection using Gabor filter features," Pattern Recognition Letters, vol. 26, pp. 1641-1649, 2005.
[37] X. Xie and K.M. Lam, "An efficient method for facial expression recognition," Proc. Visual Communications and Image Processing, Beijing, China, pp. 786-793, 2005.
[38] J.K. Kamarainen, V. Kyrki, H. Kalviainen, "Invariance properties of Gabor filter-based features - overview and applications," IEEE Trans. on Image Process. vol. 15, no. 5, pp. 1088-1099, 2006.
[39] L. Qing, S. Shan, X. Chen, and W. Gao, "Face recognition under varying lighting based on the probabilistic model of gabor phase," Proc. 18th Int'l Conf. Pattern Recognition, vol. 3, pp. 1139-1142, 2006.
[40] H. Cheng, N. Zheng, and C. Sun, "Boosted Gabor features applied to vehicle detection," 18th Int'l Conf. Pattern Recognition, vol. 1, pp. 662-666, 2006.
[41] M. Valstar and M. Pantic, "Fully automatic facial action unit detection and temporal analysis," IEEE Conf. CVPRW, p. 149, June 2006.
[42] C. Liu, "Capitalize on dimensionality increasing techniques for improving face recognition grand challenge performance," IEEE Trans. Pattern Anal. Mach. Intell. vol. 28, no. 5, pp. 725-727, 2006.
[43] X. Xie and K.M. Lam, "Gabor-based kernel PCA with doubly nonlinear mapping for face recognition with a single face image," IEEE Trans. Image Process. vol. 15, no. 9, pp. 2481-2492, 2006.
[44] L. Shen, L. Bai, and M. Fairhurst, "Gabor wavelets and general discriminant analysis for face identification and verification," Image Vision Comput. vol. 25, no. 5, pp. 553-563, 2007.
[45] C. Caleanu, D.S. Huang, V. Gui, V. Tiponut, and V. Maranescu, "Interest operator versus Gabor filtering for facial imagery classification," Pattern Recognition Letters, vol. 28, 950-956, 2007.
[46] W.P. Choi, S.H. Tse, K.W. Wong, and K.M. Lam, "Simplified Gabor wavelets for human face recognition," Pattern Recognition, vol. 41, no. 3, pp. 1186-1199, March 2008.
[47] C.K. Chui, "An Introduction to Wavelets," Academic Press, Boston, 1992.
[48] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio, "Pedestrian detection using wavelet templates," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 193-199, 1997.
[49] Y. Zhu, S. Schwartz, and M. Orchard, "Fast face detection using subspace discriminant wavelet features," Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2000, vol. 1, pp. 636-642, June 2000.
[50] M. Vidal-Naquet and S. Ullman, "Object recognition with informative features and linear classification," Proc. Ninth IEEE Int'l Conf. Computer Vision, vol.1, pp. 281-288, 2003.
[51] G. Yang and T.S. Huang, "Human Face Detection in Complex Background," Pattern Recognition, vol. 27, no. 1, pp. 53-63, 1994.
[52] C. Kotropoulos and I. Pitas, "Rule-Based Face Detection in Frontal Views," Proc. Int'l Conf. Acoustics, Speech and Signal Processing, vol. 4, pp. 2537-2540, 1997.

[53] C. Lin and K.C. Fan, "Triangle-based approach to the detection of human face," Pattern Recognition, vol. 34, no. 6, pp. 1271-1284, 2001.
[54] P. Maragos, "Tutorial on advances in morphological image processing and analysis," Optical Engineering, vol. 26, no. 7, pp. 623-632, 1987.
[55] K.M. Lam and H. Yan, "Locating and extracting the eye in human face images," Pattern Recognition, vol. 29, no. 5, pp. 771-779, 1996.
[56] K.M. Lam and H. Yan, "An improved method for locating and extracting the eye in human face images," In Proc. IEEE ICPR'96, pp. C411-C415, August 1996.
[57] K.W. Wong and K.M. Lam, "A reliable approach for human face detection using genetic algorithm," Proc. IEEE Int'l Symposium Circuits and Systems, vol. 4, pp. 499-502, 1999.
[58] K.W. Wong, K.M. Lam, W.C. Siu, "An efficient algorithm for human face detection and facial feature extraction under different conditions," Pattern Recognition, vol. 34, pp. 1993-2004, 2001.
[59] A.K. Jain, Y. Zhong, and S. Lakshmanan, "Object matching using deformable templates," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 18, no. 3, pp. 267-278, 1996.
[60] K.K. Sung and T. Poggio, "Example-based learning for view-based human face detection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pp. 39-51, 1998.
[61] J. Kuo, R.S. Huang, and T.G. Lin, "3-D facial model estimation from single front-view facial image," IEEE Trans. Circuits and Systems for Video Technology, vol. 12, no. 3, pp. 183-192, 2002.
[62] R. Brunelli and T. Poggio, "Face recognition: features versus templates," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 15, no. 10, pp. 1042-1052, 1993.
[63] H. Schweitzer, J.W. Bell, and F. Wu, "Very fast template matching," ECCV 2002, LNCS 2353, pp. 358-372, 2002.
[64] M. Turk and A. Pentland, "Eigenfaces for recognition," J. Cognitive Neurosci. vol. 13, no. 1, pp. 71-86, 1991.
[65] A. Pentland, B. Moghaddam, and T. Starner, "View-based and modular eigenspaces for face recognition," IEEE Conf. Computer Vision and Pattern Recognition, pp. 84-91, June 1994.
[66] Z. Sun, G. Bebis, and R. Miller, "Object detection using feature subset selection," Pattern Recognition, vol. 37, no. 11, pp. 2165-2176, 2004.
[67] D.H. Foley and J.W. Sammon, "An optimal set of discriminant vectors," IEEE Trans. Computers, vol. c-24, no. 3, pp. 281-289, March 1975.
[68] D.L. Swets and J.J. Weng, "Using discriminant eigenfeatures for image retrieval," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 18, no. 8, pp. 831-836, Aug. 1996.
[69] K. Fukunaga, "Introduction to statistical pattern recognition," 2nd Edition, Academic Press, New York, 1990.
[70] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman, "Eigenfaces vs. fisherface: recognition using class specific linear projection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 711-720, 1997.
[71] W.S. Yambor, "Analysis of PCA-based and Fisher discriminant-based image recognition algorithms," Technical Report CS-00-103, Computer Science Department, Colorado State University, July 2000.

[72] A.M. Martinez and A.C. Kak, "PCA versus LDA," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 23, no. 2, pp. 228-233, 2001.
[73] X. Wang and X. Tang, "A unified framework for subspace face recognition," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 26, pp. 1222-1228, Sep. 2004.
[74] Y. Gao, Y. Wang, X. Feng, and X. Zhou, "Face recognition using most discriminative local and global features," 18th Int'l Conf. Pattern Recognition, vol. 1, pp. 351-354, 2006.
[75] W.S. Zheng, J.H. Lai, and S.Z. Li, "1D-LDA vs. 2D-LDA: When is vector-based linear discriminant analysis better than matrix-based?" Pattern Recognition, vol. 41, pp. 2156-2172, 2008.
[76] J.H. Friedman, "Regularized discriminant analysis," J. American Statistical Association, vol. 84, no. 405, pp. 165-175, 1989.
[77] D.Q. Dai, and P.C. Yuen, "Regularized discriminant analysis and its application to face recognition," Pattern Recognition, vol. 36, pp. 845-847, 2003.
[78] W. Zhao, R. Chellappa, P.J. Phillips, "Subspace linear discriminant analysis for face recognition," Center for Automation Research, University of Maryland, College Park, Technical Report CAR-TR-914, 1999.
[79] C. Liu and H. Wechsler, "Robust coding schemes for indexing and retrieval from large face databases," IEEE Trans. Image Processing, vol. 9, no. 1, pp. 132-137, 2000.
[80] C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced Fisher linear discriminant model for face recognition," IEEE Trans. Image Processing, vol. 11, no. 4, pp. 467-476, April 2002.
[81] J. Yang and J.Y. Yang, "Why can LDA be performed in PCA transformed space?" Pattern Recognition, vol. 36, pp. 563-566, 2003.
[82] L.F. Chen, H.Y.M. Liao, M.T. Ko, J.C. Lin, and G.J. Yu, "A new LDA-based face recognition system which can solve the small sample size problem," Pattern Recognition, vol. 33, no. 10, pp. 1713-1726, 2000.
[83] J. Yang, D. Zhang, and J.Y. Yang, "A generalized K-L expansion method which can deal with small sample size and high-dimensional problems," Pattern Analysis and Applications, vol. 6, pp. 47-54, 2003.
[84] R. Huang, Q. Liu, H. Lu, and S. Ma, "Solving the small sample size problem of LDA," Proc. 16th Int'l Conf. Pattern Recognition, vol. 3, pp. 29-32, Aug. 2002.
[85] X.S. Zhuang and D.Q. Dai, "Improved discriminate analysis for high-dimensional data and its application to face recognition," Pattern Recognition, vol. 40, pp. 1570-1578, 2007.
[86] H. Yu and J. Yang, "A direct LDA algorithm for high-dimensional data – with application to face recognition," Pattern Recognition, vol. 34, pp. 2067-2070, 2001.
[87] H. Gao and J.W. Davis, "Why direct LDA is not equivalent to LDA," Pattern Recognition, vol. 39, pp. 1002-1006, 2006.
[88] D.U. Cho, U.D. Chang, B.H. Kim, S.H. Lee, Y.L. J.Bae, and S.C. Ha, "2D direct LDA algorithm for face recognition," Fourth Int'l Conf. Software Engineering Research, Management and Applications, pp. 245-248, Aug. 2006.

[89] H. Cevikalp, M. Neamtu, M. Wilkes, and A. Barkana, "Discriminative common vectors for face recognition," IEEE. Trans. Pattern Analysis and Machine Intelligence, vol. 27, no. 1, pp. 4-13, January 2005.
[90] J. Liu and S. Chen, "Discriminant common vectors versus neighbourhood components analysis and Laplacianfaces: A comparative study in small sample size problem," Image and Vision Computing, vol. 24, pp. 249-262, 2006.
[91] L. Yang, W. Gong, X. Gu, W. Li, and Y. Liang, "Null space discriminant locality preserving projections for face recognition," Neurocomputing, 2008.
[92] J. Yang, D. Zhang, A.F. Frangi, and J.Y. Yang, "Two-dimensional PCA: a new approach to appearance-based face representation and recognition," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 26, no. 1, pp. 131-137, Jan. 2004.
[93] Y. Xu, D. Zhang, J. Yang, and J.Y. Yang, "An approach for directly extracting features from matrix data and its application in face recognition," Neurocomputing, 2008.
[94] J. Yang and C. Liu, "Horizontal and Vertical 2DPCA-based discriminant analysis for face verification on a large-scale database," IEEE Trans. Information Forensics and Security, vol. 2, no. 4, pp. 781-792, Dec. 2007.
[95] H. Xiong, M.N.S. Swamy, and M.O. Ahmad, "Two-dimensional FLD for face recognition," Pattern Recognition, vol. 38, pp. 1121-1124, 2005.
[96] J. Yang, D. Zhang, X. Yong, and J.Y. Yang, "Two-dimensional discriminant transform for face recognition," Pattern Recognition, vol. 38, pp. 1125-1129, 2005.
[97] X.Y. Jing, H.S. Wong, and D. Zhang, "Face recognition based on 2D Fisherface approach," Pattern Recognition, vol. 39, pp. 707-710, 2006.
[98] S. Noushath, G. Hemantha Kumar, and P. Shivakumara, "(2D) LDA: An efficient approach for face recognition," Pattern Recognition, vol. 39, pp. 1396-1400, 2006.
[99] S.T. Roweis and L.K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," Science, vol. 290, pp. 2323-2326, Dec. 2000.
[100] L.K. Saul and S. Roweis. "Think globally, fit locally: Unsupervised learning of low dimensional manifolds," J. Machine Learning Research, 4: 119-155, 2003.
[101] X. He and P. Niyogi, "Locality preserving projections," Technical Report TR-2002-09, University of Chicago Computer Science, October 2002.
[102] X. He, S. Yan, Y. Hu, and H.J. Zhang, "Learning a locality preserving subspace for visual recognition," Proc. Ninth IEEE Int'l Conf. Computer Vision, vol. 1, pp. 385-392, 2003.
[103] X. He, S. Yan, Y. Hu, P. Niyogi, and H.J. Zhang, "Face recognition using Laplacianfaces," IEEE. Trans. Pattern Analysis and Machine Intelligence, vol. 27, no. 3, March 2005.
[104] B. Niu and Q. Yang, "Two-dimensional Laplacianfaces method for face recognition," Pattern Recognition, 2008.
[105] Q. You, N. Zheng, S. Du, and Y. Wu, "Neighborhood discriminant projection for face recognition," 18th Int'l Conf. Pattern Recognition, vol. 2, pp. 532-535, 2006.

[106] Q. You, N. Zheng, S. Du, and Y. Wu, "Neighborhood discriminant projection for face recognition," Pattern Recognition Letters, vol. 28, pp. 1156-1163, 2007.
[107] H. Hu, "Orithogonal neighborhood preserving discriminant analysis for face recognition," Pattern Recognition, vol. 41, pp. 2045-2054, 2008.
[108] A. Hyvarinen and E. Oja, "Independent component analysis: algorithm and applications," Neural Networks, vol. 13, pp. 411-430, 2000.
[109] S.Z. Li, X. Lv, and H.J. Zhang, "View-subspace analysis of multi-view face patterns," Proc. IEEE ICCV Workshop Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, pp. 125-132, 2001.
[110] M.S. Bartlett, J.R. Movellan, T.J. Sejnowski, "Face recognition by independent component analysis," IEEE. Trans. Neural Networks, vol. 13, no. 6, pp. 1450-1464, Nov. 2002.
[111] C. Liu and H. Wechsler, "Independent component analysis of Gabor features for face recognition," IEEE Trans. Neural Networks, vol. 14, no. 4, pp. 919-928, 2003.
[112] J. Yang, D. Zhang, and Y.Y. Yang, "Is ICA significantly better than PCA for face recognition?" Tenth IEEE Int'l Conf. Computer Vision, vol. 1, pp. 198-203, 2005.
[113] http://www.cis.hut.fi/projects/ica/fastica/
[114] H. Schneiderman and T. Kanade, "Probabilistic modeling of local appearance and spatial relationships for object recognition," Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 45-51, 1998.
[115] H. Schneiderman and T. Kanade, "A histogram-based method for detection of faces and cars," Proc. Int'l Conf. Image Processing, vol. 3, pp. 504-507, 2000.
[116] H. Schneiderman and T. Kanade, "A statistical method for 3D object detection applied to faces and cars," Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol. 1, pp. 746-751, 2000.
[117] Z. Liu, J. Yang, and N.S. Peng, "An efficient face segmentation algorithm based on binary partition tree," Signal Processing: Image Communication, vol. 20, pp. 295-314, 2005.
[118] L. Qing, S. Shan, X. Chen, and W. Gao, "Face recognition under varying lighting based on the probabilistic model of Gabor phase," 18th Int'l Conf. Pattern Recognition, vol. 3, pp. 1139-1142, 2006.
[119] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, pp. 696-710, 1997.
[120] B. Moghaddam, W. Wahid, and A. Pentland, "Beyond eigenfaces: probabilistic matching for face recognition," Proc. Third IEEE Int'l Conf. Automatic Face and Gesture Recognition, pp. 30-35, 1998.
[121] C. Liu, "A Bayesian discriminating features method for face detection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 25, pp. 725-740, 2003.
[122] H. Demirel, T.J. Clarke, and P.Y.K. Cheung, "Adaptive automatic facial feature segmentation," Proc. Second Int'l Conf. Automatic Face and Gesture Recognition, pp. 277-282, 1996.
[123] M. Segal and E. Weinstein, "The cascade EM algorithm," Proc. IEEE, vol. 76, no. 10, pp. 1388-1390, Oct. 1988.

[124] T.K. Moon, "The expectation-maximization algorithm," IEEE Signal Processing Magazine, vol. 13, no. 6, pp. 47-60, Nov. 1996.
[125] A. Dempster, N. Laird, D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," J. Roy. Statist. Soc. series B, vol. 39, no. 1, pp. 1-38, 1997.
[126] F. Fleuret, "Binary feature selection with conditional mutual information," Rapport de recherche n4941, ISSN 0249-6399, 2003.
[127] Z.R. Yang and M. Zwolinski, "Mutual information theory for adaptive mixture models," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 23, no. 4, April 2001.
[128] L.B. Shams, M.J. Brady, and S. Schaal, "Graph matching vs mutual information maximization for object detection," Neural Networks, vol. 14, pp. 345-354, 2001.
[129] M. Vidal-Naquet and S. Ullman, "Object recognition with informative features and linear classification," Proc. Ninth IEEE Int'l Conf. Computer Vision, vol.1, pp. 281-288, 2003.
[130] H.T. Su, D.D. Feng, X.Y. Wang, and R.C. Zhao, "Face recognition using hybrid feature," Int'l Conf. Machine Learning and Cybernetics, vol. 5, pp. 3045-3049, Nov. 2003.
[131] H.T. Su, D.D. Feng, R.C. Zhao, and X.Y. Wang, "Face recognition method using mutual information and hybrid feature," Proc. Fifth Int'l Conf. Computational Intelligence and Multimedia Applications, pp. 436-440, 2003.
[132] D. Grossman and P. Domingos, "Learning bayesian network classifiers by maximizing conditional likelihood," Proc. 21 Int'l Conf. Machine Learning, 2004.
[133] L. Shen and L. Bai, "Information theory for Gabor feature selection for face recognition," EURASIP Journal on Applied Signal Processing, vol. 2006, Article ID 30274, doi:10.1155/ASP/2006/30274, 2006.
[134] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pp. 23-38, Jan. 1998.
[135] Q. Gu and S.Z. Li, "Combining feature optimization into neural network based face detection," Proc. 15th Int'l Conf. Pattern Recognition, vol. 2, pp. 814-817, 2000.
[136] C. Garcia and M. Delakis, "Convolutional face finder: a neural architecture for fast and robust face detection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 26, no. 11, pp. 1408-1423, Nov. 2004.
[137] Y. Li, S. Gong, J. Sherrah, and H. Liddell, "Support vector machine based multi-view face detection and recognition," Image and Vision Computing, vol. 22, pp. 413-427, 2004.
[138] M.H. Yang, D. Roth, and N. Ahuja, "A SNoW-based face detector," Proc. Neural Information Processing Systems, pp. 885-861, 2000.
[139] E.C. Smith, "A SNoW-based automatic facial feature detector," MPLab Technical Report TR 2001.06, 2001.
[140] Jie Chen, Xilin Chen, and Wen Gao, "Expand training set for face detection by GA re-sampling," Proc. Sixth IEEE Int'l Conf. Automatic Face and Gesture Recognition, pp. 73-78, May 2004.

[141] M. Alvira and R. Rifkin, "An empirical comparison of SNoW and SVMs for face detection," CBCL, MIT, A.I. Memo:2001-004, http://www.ai.mit.edu/projects/cbcl/software-datasets/FaceData2.html
[142] Fabrice Leroy, "Clementine's RBFN technical overview," October 1998, http://www.cs.bris.ac.uk/~cgc/METAL/Consortium/secure/RBFN_Intranet.doc
[143] Y. Freund and R.E. Schapire, "Experiments with a new boosting algorithm," Int'l Conf. on Machine Learning, pp. 148-156, 1996.
[144] Y. Freund and R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," Computer and System Sciences, vol. 55, no. 1, pp. 119-139, 1997.
[145] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proc. Conf. Computer Vision and Pattern Recognition, pp. 511-518, 2001.
[146] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," Proc. Int'l Conf. Image Processing, vol. 1, pp. I-900 - I-903, Sept. 2002.
[147] Y. Ma and X. Ding, "Robust real-time face detection based on cost-sensitive AdaBoost method," Proc. Int'l Conf. Multimedia and Expo, vol. 2, pp. II - 465-8, July 2003.
[148] P. Yang, S. Shan, W. Gao, S.Z. Li, and D. Zhang, "Face recognition using Ada-Boosted Gabor features," Proc. Sixth IEEE Int'l Conf. Automatic Face and Gesture Recognition, pp. 356-361, 2004.
[149] S.Z. Li and Z. Zhang, "FloatBoost learning and statistical face detection," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 26, no. 9, pp. 1112-1123, 2004.
[150] M. Liu, J. Duan, X.H. Liu, Y.C. Liang, and C.G. Zhou, "An improved line-based face recognition and indexing algorithm," Int'l Conf. Machine Learning and Cybernetics, vol. 5, pp. 3100-3103, Nov. 2003.
[151] K.H. Lin, K.M. Lam, X. Xie, and W.C. Siu, "An efficient human face indexing scheme using eigenfaces," Proc. IEEE Int'l Conf. Neural Networks & Signal Processing, vol. 2, pp. 920-923, December 2003.
[152] J. Lu and K.N. Plataniotis, "Boosting face recognition on a large-scale database," Pcoc. Int'l Conf. Image Processing, vol. 2, 2002.
[153] J. Vleugels and R.C. Veltkamp, "Efficient image retrieval through vantage objects," Pattern Recognition, vol. 35, pp. 69-80, 2002.
[154] X. Xie and K.M. Lam, "An efficient method for facial expression recognition," Proc. Visual Communications and Image Processing, pp. 786-793, 2005, Beijing, China.
[155] D. DeCarlo and D. Metaxas, "Optical Flow Constraints on Deformable Models with Applications to Face Tracking," Int'l J. Computer Vision, vol. 38, no. 2, pp. 99-127, July 2000.
[156] R.C. Verma, C. Schmid, K. Mikolajczyk, "Face detection and tracking in a video by propagating detection probabilities," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 25, no. 10, pp. 1215-1228, Oct. 2003.
[157] E. Painkras and C. Charoensak, "A framework for the design and implementation of a dynamic face tracking system," IEEE Region 10 TENCON, pp. 1-6, November 2005.

[158] S. Dubuisson, "An adaptive clustering for multiple object tracking in sequences in and beyond the visible spectrum," IEEE Conf. CVPRW, p. 142, June 2006.
[159] D. Tao, X. Li, S.J. Maybank, and X. Wu, "Human carrying status in visual surveillance," IEEE Computer Society Conf. CVPR, vol. 2, pp. 1670-1677, 2006.
[160] K.W. Sze, K.M. Lam, and G.P. Qiu, "A new key frame representation for video segment retrieval," IEEE Trans. Circuits and Systems for Video Technology, vol. 15, no. 9, pp. 1148-1155, Sep. 2005.
[161] AR Face Database. http://cobweb.ecn.purdue.edu/~aleix/aleix_face_DB.html
[162] Yale University Face Database. http://cvc.yale.edu/projects/yalefaces/yalefaces.html
[163] http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html
[164] Olivetti & Oracle Research Laboratory. The Olivetti & Oracle Research Laboratory Face Database of Faces, http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[165] UMIST Face Database. http://images.ee.umist.ac.uk/danny/database.html
[166] FERET database http://www.itl.nist.gov/iad/humanid/feret/
[167] P.J. Phillips, H. Moon, P.J. Rauss, and S. Rizvi, "The FERET evaluation methodology for face recognition algorithms," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 10, October 2000.
[168] http://viper.unige.ch/~marchand/CBVR/