4.843

# A Hidden Markov Model-Based Approach for Face Detection and Recognition

A THESIS

Presented to

The Academic Faculty

By

Ara Nefian

In Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy in Electrical Engineering

Georgia Institute of Technology

August, 1999

# A Hidden Markov Model-Based Approach for Face Detection and Recognition

Approved:

Monson H. Hayes III, Chairman

Mark A. Clements

Russell M. Mersereau

Date approved by Chairman _____

# Acknowledgement

First, I would like to thank my family for their love, support, and encouragements. Their continuous care has been indispensable throughout my Ph.D research.

I am most grateful to my advisor, Dr. Monson Hayes III, for his guidance, interesting discussions, and helpful suggestions. I owe a great deal of my research skills to him.

I would like to thank the other members of the reading committee, Dr. Mark Clements and Dr. Russell Mersereau, for thoroughly reading this thesis. I would also like to thank Dr. Nykil Jyant and Dr. Irfan Essa for serving on the defense committee.

I would like to thank all my fellow graduate students for creating a friendly and challenging environment. I especially thank Mahmut Ciftci, Mohammad Khan, Faramarz Fekri, Darnell Moore, Chanin Nilubol, Jeff Price, and Jorge Perez-Jacome for useful discussions and for making my stay in school more enjoyable.

I would like to thank the researchers at the Human Interface Technology Center, NCR for the professional environment during my long term internship with the Image Understanding group. Special thanks to John Ming and Dr. Mehdi Khosravi whose friendship and technical guidance made my internship a great experience.

To my family.

# Contents

# List of Tables

# List of Figures

# Glossary

DCT - Discrete Cosine Transform

FFT - Fast Fourier Transform

HMM - Hidden Markov Model

JPEG - Joint Photographic Experts Group

KLT - Karhunen Loeve Transform

LDA - Linear Discriminant Analysis

LDT - Linear Discriminant Transform

MLP - Multi Layer Perceptron

NN - Neural Network

PCA - Principal Component Analysis

SVD - Singular Value Decomposition

VFR - Volumetric Frequency Representation

# Summary

The use of hidden Markov models (HMM) for faces is motivated by their partial invariance to variations in scaling and by the structure of faces. The most significant facial features of a frontal face include the hair, forehead, eyes, nose and mouth. These features occur in a natural order, from top to bottom, even if the images undergo small rotations in the image plane, and/or rotations in the plane perpendicular to the image plane. Therefore, the image of a face may be modeled using a one-dimensional HMM by assigning each of these regions to a state. The observation vectors are obtained from the DCT or KLT coefficients.

A one-dimensional HMM may be generalized, to give it the appearance of a two- dimensional structure, by allowing each state in a one-dimensional HMM to be a HMM. In this way, the HMM consists of a set of super states, along with a set of embedded states. Therefore, this is referred to as an embedded HMM. The super states may then be used to model two-dimensional data along one direction, with the embedded HMM modeling the data along the other direction.

Both the standard HMM and the embedded HMM were tested for face recognition and detection. Compared to other methods, our proposed system offers a more flexible framework for face recognition and detection, and can be used more efficiently in scale invariant systems.

# CHAPTER 1

# Introduction

## 1.1 Motivation for the Research

Face detection and face recognition [1] from still and video images is emerging as an active research area with numerous commercial and law enforcement applications. Face detection and recognition systems can be used to allow access to an ATM machine or a computer, to control the entry of people into restricted areas, to recognize people in specific areas (banks, stores), or in a specific database (police database).

The goal of our research is to develop a face detection and recognition system that can be used in a real-time face identification system (Figure 1.1). A face identification system must operate under a variety of conditions, such as varying illuminations and backgrounds, it must be able to handle non-frontal facial images of both males and females of different ages and races, and be robust in the presence of two or more faces within a video sequence.

The overall face identification system (Figure 1.1) involves processing a video sequence to perform the following tasks. First, is to detect faces in real-time in video sequences in an uncontrolled environment. Second, is to track the detected faces over consecutive frames, and select the frame or frames that will be best for recognition. The final step is to perform face recognition on the detected faces. If it is assumed that the background is known and stationary, then the bodies can be segmented from the background, at each frame in the video sequence. If no people are detected, then the system updates the background by adding the information from the last

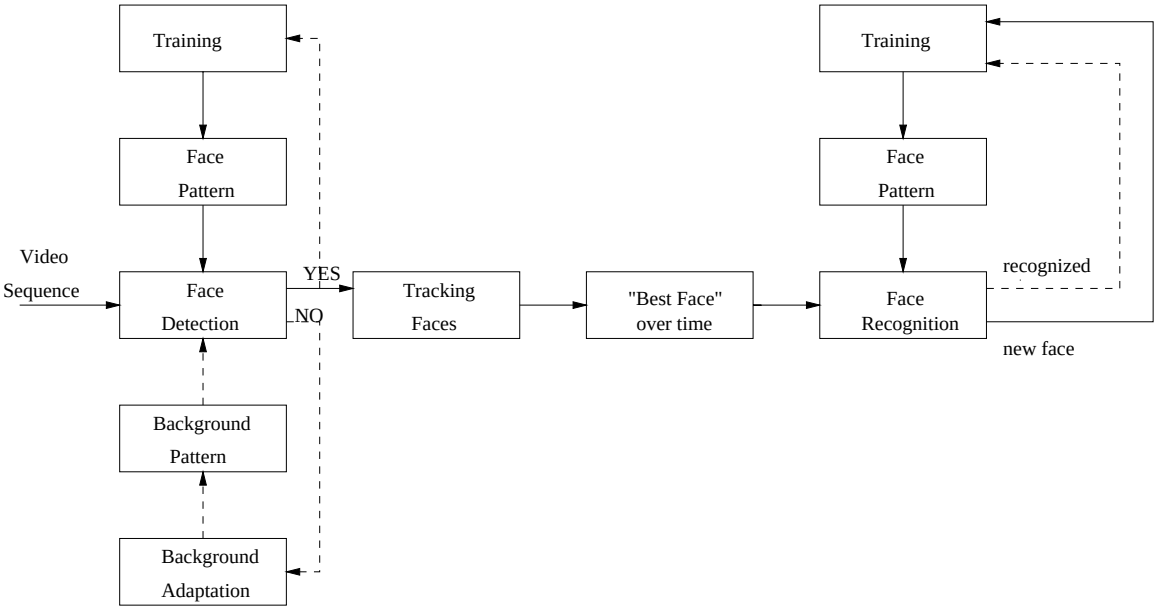



Figure 1.1: The overall face identification system

frame. Otherwise, if a face is detected, the system can re-train the face model used for detection. At the recognition stage, if a new face image is identified, then it is used for the training of the new face model. On the other hand, if a face is recognized as one of the faces stored in the database, then the corresponding face model can be re-trained.

Most of the previous attempts in face detection and recognition, which will be discussed in more detail in the next chapter, may be classified as either feature-based or template-based approaches. It has been shown [2] that the template based approaches generally perform better than feature based methods. However, these methods are very sensitive to variations in the scale of the image. To overcome this problem, templates at different sizes were considered for both detection and recognition. However, this solution dramatically increases the computational complexity of the methods.

The approach that we are proposing involves the use of Hidden Markov Models (HMM) for the face. For a frontal face, the "states" of the Markov model include hair,

forehead, eyes, nose and mouth. These states always occur in the same order, from top to bottom, even if the faces undergo small rotations in the image plane, and/or small rotations in the plane perpendicular to the image plane. Therefore, each of these facial regions will be assigned to a state, in a left-to-right one-dimensional continuous HMM. A one-dimensional HMM may be generalized, to give it the appearance of a two- dimensional structure, by allowing each state in a one-dimensional HMM to be a HMM. In this way, the HMM consists of a set of super states, along with a set of embedded states. Therefore, this is referred to as an embedded HMM. The super states may then be used to model two-dimensional data along one direction, with the embedded HMM modeling the data along the other direction. The advantage of the embedded HMM-based approach is its ability to handle variations in scale, which is a challenging problem for any face detection/recognition system, and its computational efficiency compared to other approaches.

Although the same model based approach can be used for both face detection and recognition, there are many significant differences between these two problems. While the training set for recognition is in general small, and limited by the amount of information we have about a specific person, the training set used for detection is virtually unlimited and must cover a large variety of faces of different people, from all ages, races, taken under different lighting conditions and showing all facial expressions and orientations, hair styles, facial hair, and eye wear (glasses/no glasses).

The time efficiency of the face detection system depends primarily on the image resolution, and secondarily on the number of face models used for detection. It has been shown [3] that five models corresponding to different face views are sufficient for a good face representation under a large range of orientations. The time required by the detection system is critical, since a fast face detection system will improve the robustness of the tracking system, and consequently the quality of the images sent to the recognition system. On the other hand, the time required for recognition is a function of the size of the database. No real-time constraint is imposed on the

recognition system in Figure 1.1, since recognition is not performed on each frame but only after the detection and tracking systems send a face image to the recognizer. Thus the recognition time must only be less than the time between two consecutive occurrences of people in a scene. In addition, this constraint can be further relaxed by the use of more processors.

## 1.2 Outline of the Thesis

The focus of this thesis is the modeling of human faces using hidden Markov models (HMM) and with application in face recognition and detection. Chapter 2 describes some of the most successful face recognition and detection approaches. Chapter 3 describes a real-time face detection algorithm from gray scale video sequences, that uses a deformable template based approach. Chapter 4 presents a HMM-based face detection and recognition system that uses an efficient set of observation vectors. In Chapter 5 the theoretical aspects and the key algorithms used in the training and recognition of the embedded HMM are presented. Chapter 6 describes an embedded HMM for faces and presents a face detection and recognition system that uses this model. Finally, the conclusions of this work are drawn and suggested directions for further research are given in Chapter 7.

# CHAPTER 2

# Background

This chapter overviews some of the algorithms that have been developed for face recognition and detection. These algorithms [1] may be broadly classified as either geometric feature-based methods, template-based methods or, more recently, model-based methods. For each of these approaches some of the most popular algorithms will be briefly described. We will summarize and compare these algorithms in terms of their accuracy, robustness and complexity.

## 2.1 Geometric Feature-Based Methods

The geometric feature based approaches [4], [5] are the earliest approaches to face recognition and detection. In these systems, the significant facial features are detected and the distances among them as well as other geometric characteristics are combined in a feature vector that is used to represent a face. To recognize a face, first the feature vectors of the test image and of the images in the database are obtained. Second, a similarity measure between these vectors, most often a minimum distance criterion, is used to determine the identity of the face. As pointed out by Brunelli and Poggio [2], the template-based approaches, which will be discussed in the next section, outperform the early geometric-feature based approaches.

## 2.2 Template-Based Methods

The template based approaches represent the most popular technique used to recognize and detect faces. Unlike the geometric feature-based approaches, the template-based approaches use a feature vector that represents the entire face template rather than only the most significant facial features.

### 2.2.1 Correlation-Based Methods

Correlation methods for face detection and recognition [6]are based on the computation of the normalized cross-correlation coefficient $C_N$ [7] defined by:

$$C_N = \frac{E\{I_T T\} - E\{I_T\}E\{T\}}{\sigma(I_T)\sigma(T)}, \tag{2.1}$$

where $I_T$ is the image that is to be matched to the template $T$, $I_T T$ represents the pixel-by-pixel product, $E$ is the expectation operator, and $\sigma$ is the standard deviation over the area being matched. In [2], [8], [6] the authors describe a correlation-based method for both face detection and face recognition from frontal views. The first step in these methods is to determine the location of the significant facial features such as eyes, nose or mouth. The importance of robust facial feature detection for both detection and recognition has resulted in the development of a variety of different facial feature detection algorithms [9], [10], [11], [12], [13], [14]. The facial feature detection method proposed by Brunelli and Poggio [2], [8] uses a set of templates to detect the position of the eyes in an image, by looking for the maximum absolute values of the normalized correlation coefficient of these templates at each point in the test image. To cope with scale variations, a set of templates at different scales was used. The problems associated with scale variation can be significantly reduced by using hierarchical correlation (as proposed by Burt in [15]). For face recognition, the templates corresponding to the significant facial features of the test images are compared, in turn, with the corresponding templates of all of the images in the database, returning a vector of matching scores (one per feature) computed

through normalized cross correlation. The similarity scores of different features are integrated to obtain a global score that is used for recognition. Other similar methods that use correlation [6] or higher order statistics [16], [17] revealed the accuracy of these methods but also their complexity.

In [18], Beymer extended the correlation based approach to a view-based approach for recognizing faces under varying orientations, including rotations with respect to the axis perpendicular to the image plane (rotations in image depth). To handle rotations out of the image plane, templates from different views were used. After the pose is determined, the task of recognition is reduced to the classical correlation method in which the facial feature templates are matched to the corresponding templates of the appropriate view-based models using the cross correlation coefficient. However, this approach is computationally expensive, and it is sensitive to lighting conditions.

### 2.2.2 Karhunen-Loeve Expansion - Based Methods

#### The Eigenfaces Method

The Eigenfaces method [19], [20], [21] proposed by Turk and Pentland is based on the Karhunen-Loeve Transform (KLT), and is motivated by the earlier work of Sirovitch and Kirby [22], [23] for efficiently representing face images. The eigenvectors of the covariance matrix $C$ of the ensemble of training faces are called *eigenfaces*. The space spanned by the eigenvectors $\mathbf{v}_k$, $k = 1, \ldots, K$ corresponding to the $K$ largest eigenvalues of the covariance matrix, is called the *face space*. A new face image is transformed into its eigenface components by projection onto the face space. The projections form the feature vector which describes the contribution of each eigenface in representing the input image. A test image is recognized by computing the Euclidean distance in the feature space and selecting the closest match. The effect of the lighting conditions over the KLT based method has been detailed in [24]. The eigenface method has also been used for face detection [25], [26] by measuring the

distance from each local pattern in a test image to the face space defined by the eigenfaces.

In [27], Akamatsu et. al., applied the eigenface method to the magnitude of the Fourier spectrum of the images after normalization with respect to illumination and scale. Due to the shift invariance property of the magnitude of the Fourier spectrum, and to the illumination and scale normalization, the method, called the Karhunen-Loeve Transform of Fourier Spectrum in the Affine Transformed Target Image (KL-FSAT), performed better than classical eigenfaces method under variations in head orientations and shifting.

### The "Parametric" Approach versus the "View-Based" Approach

In [28], Murase and Nyar extended the capabilities of the eigenface method to general 3D object recognition under different illumination and viewing conditions. Given $N$ object images taken under $P$ views and $L$ different illumination conditions, a set of eigenvectors was obtained by applying the eigenface method to all the available data. In this way a single "parametric space" describes the object identity as well as the viewing or illumination conditions. The eigenface decomposition of this space was used for feature extraction and classification. However, in order to ensure discrimination between different objects, the number of eigenvectors used in this method was increased compared to the classical Eigenface method.

Pentland et. al. [3] developed a "view-based" eigenspace approach for human face recognition under general viewing conditions. The "view-based" approach is essentially an extension of the eigenface technique to multiple sets of eigenvectors, one for each face orientation. First, the orientation of the test face is determined by calculating the residual description error (distance from feature space) for each view space, and selecting the space for which the distance is minimized. Once the proper view is determined, the face image is classified using the eigenface method in the corresponding space. As expected, the view-based representation has better

recognition results than the parametric approach, at a cost of a higher computational complexity.

#### Recognition Using Eigenfeatures

While the classical eigenface method uses the KLT coefficients of the template corresponding to the whole face image, in [3] Pentland et. al. introduced a face detection and recognition system that uses the KLT coefficients of the templates corresponding to the significant facial features such as eyes, nose and mouth. For each of the facial features, a feature space is built by selecting the most significant "eigenfeatures", which are the eigenvectors corresponding to the largest eigenvalues of the features correlation matrix. The significant facial features were detected using the distance from the feature space and selecting the closest match. The scores of similarity between the templates of the test image and the templates of the images in the training set were integrated in a cumulative score that measures the distance between the test image and the training images. The method was extended to the detection of features under different viewing geometries by using either a view-based eigenspace or a parametric eigenspace.

### 2.2.3 Linear Discriminant-Based Methods

In [29], [30], [31], the authors proposed a new method for face recognition using Fisher's Linear Discriminant Transform (LDT) [32]. The "Fisherface" method uses the class membership information and develops a set of feature vectors in which variations of different faces are emphasized while different instances of faces due to illumination conditions, facial expressions, and orientations, are de-emphasized. While the Karhunen-Loeve Transform performs a rotation on a set of axes along which the projection of sample vectors differ most in the autocorrelation sense, the LDT performs a rotation on a set of axes along which the projection of sample vectors show maximum discrimination. Each test image is projected onto the optimal LDT space

and the resulting set of coefficients is used to compute the Euclidean distance from the images in the training set. More recently the Fisherface method has alos been applied to face detection from color images [33].

In [27], Akamatsu et. al. applied LDT to the magnitude of the Fourier Spectrum of the intensity image. The database used in the experiments contained large variations in lighting conditions as well as variations in head orientation. The results reported by the authors showed that LDT in the Fourier domain is significantly more robust to variations in lighting than the LDT applied directly to the intensity images. However, the computational complexity of this method is significantly higher than the classical Fisherface method due to the computation of the Fourier spectrum.

### 2.2.4 Singular Value Decomposition - Based Methods

The face recognition methods presented in this section use the general result stated by the Singular Value Decomposition Theorem. In [34], Z.Hong revealed the importance of using Singular Value Decomposition method (SVD) for human face recognition by proving several important properties of the singular values (SV) vector which include: the stability of the SV vector to small perturbations caused by stochastic variation in the intensity image, the proportional variation of the SV vector with the pixel intensities, the invariance of the SV feature vector to rotation, translation and mirror transformations. The above properties of the SV vector provide the theoretical basis for using singular values as image features. In addition, it has been shown [35], [36] that compressing the original SV vector into a low dimensional space by means of various mathematical transforms leads to higher recognition performance. Among various dimensionality reducing transformations, the Linear Discriminant Transform is the most popular one. After the set of optimal discriminant vectors $\{v_1, v_2, \ldots, v_k\}$ has been extracted, the feature vectors are obtained by projecting the SV vectors onto the space spanned by $\{v_1, v_2, \ldots, v_k\}$. For each test image, the SV vector is projected onto the space spanned by $\{v_1, v_2, \ldots, v_k\}$ and classification is performed

in the feature space by measuring the Euclidean distance in this space, and assigning the test image to the class of images for which the minimum distance is achieved.

Another method to reduce the dimension of the feature space of the SV feature vectors was described by Cheng et. al. [37]. The training set used consisted of a small sample of face images of the same person. If $I_j^i$ represents the $j^{th}$ face image of person $i$, then the average image is given by $\frac{1}{N}\sum_{j=1}^{N} I_j^i$. Eigenvalues and eigenvectors are determined for this average image using SVD. The eigenvalues are thresholded to discard values that are close to zero. Average eigenvectors (called feature vectors) for all the average face images are calculated. A test image is then projected onto the space spanned by the eigenvectors. The Frobenius norm is used as a criterion to determine which person the test image should be associated with.

### 2.2.5 Matching Pursuit - Based Methods

Philips introduced a template-based face detection and recognition system [38], [39] that uses a matching pursuit filter to obtain the feature vector. The matching pursuit algorithm [40] applied to an image iteratively selects from a dictionary of basis functions the best decomposition of the image by minimizing at each iteration the residue of the image. The algorithm described by Phillips [41] constructs the best decomposition of a set of images by iteratively optimizing a cost function, which is determined from the residues of the individual images. The dictionary of basis functions used by the author consists of two-dimensional wavelets, which give a better image representation than the PCA and LDA -based techniques where images were stored as vectors. For recognition the cost function is a measure of distances between faces and is maximized at each iteration. For detection the goal is to find a filter that clusters together similar templates, therefore the cost function, which represents a measure of similarity between templates (the mean for example), is minimized at each iteration. The feature vector represents the average value of the projection of the templates on the selected basis.

### 2.2.6 Neural Networks - Based Methods

Templates have been also used as input to Neural Networks (NN)-based systems. Cotrell and Fleming [42] used in their face recognition system two backpropagation neural networks. The first is in the auto association mode and the second is in classification mode. The auto association NN automatically extracts features (as the output of a hidden layer), that are used by the classification NN. The resulting feature vector is the same as that produced by the eigenface method if the auto association net is linear. More recently, Lawrence et al. [43] proposed a hybrid NN approach that combines local image sampling, a self organizing map (SOP), and a convolutional NN. The SOP provides a a set of features that represents a more compact and robust representation of the image samples. These features are then fed into a convolutional NN. This architecture provides partial invariance to translation, rotation, scale, and face deformation. In [44], [45], the authors introduced an efficient probabilistic decision based NN (PDBNN) for face detection and recognition. The feature vector used consists of intensity and edge values obtained from a facial region of the down sampled images in the training set. The facial region contains the eyes and nose, but excludes the hair and mouth. Two PDBNNs were trained with these feature vectors and used one for face detection and the other for face recognition. Other successful NN-based approaches for face detection are reported in [46], [47], [48], [49] and more recently in [50].

### 2.2.7 The Dynamic Link Matching Approach

The above template-based matching methods use an Euclidian distance to identify a face in a gallery or to detect a face from a background. A more flexible distance measure that accounts for common facial transformations is the dynamic link matching introduced by Lades et al. [51]. In this approach, a rectangular grid is centered over all faces in the gallery. The feature vector is calculated based on Gabor-type wavelets, computed at all points of this grid. A new face is identified if the cost function, which

is a weighted sum of two terms, is minimized. The first term in the cost function is small when the distance between feature vectors is small and the second term is small when the relative distance between the grid points in the test and gallery image is preserved. It is the second term of this cost function that gives the "elasticity" of this matching measure. While the grid of the gallery image remains rectangular, the grid that is "best fit" over the test image is stretched, under certain constraints, until the minimum of the cost function is achieved. The minimum value of the cost function is used further to identify the unknown face.

Other popular template-based face recognition approaches that will only be mentioned in this work include the isodensity maps approaches [52], [53], fractal-based approaches [54], DCT feature-based [55], matched spatial filters-based [56] and natural basis functions-based [57] approaches. Successful template-based face detection methods using support vector machines and probabilistic framework are reported in [58] and [59].

## 2.3 Model-Based Methods

Recently, researchers in computer vision have started to investigate model based methods for face recognition and detection, and some very promising results have been obtained. Unlike the template based methods, the model based approaches allow for greater flexibility with respect to natural face deformations and illumination conditions. This flexibility is a result of using a mathematical model to incorporate informations from different instances of faces at different scales and orientations. In model based approaches, rather than comparing feature vectors that represent face templates to determine the identity of a face, the face model parameters are used for recognition.

### 2.3.1 The Volumetric Frequency Representation Face Model

Ben-Arie and Nandy [60] proposed a face model that incorporates both the three-dimensional (3D) face structure and its two-dimensional representation (face images). This model, which represents a volumetric (3D) frequency representation (VFR) of faces, is constructed using range image data of a human head. Making use of an extension of the Projection Slice Theorem, the Fourier transform of any face image corresponds to a slice in the face VFR. For both pose estimation and face recognition a face image is indexed in the 3D VFR based on the correlation matching in a four dimensional Fourier space, parameterized over the elevation, azimuth, rotation in the image plane and the scale of faces.

### 2.3.2 Hidden Markov Model Based Methods

Hidden Markov Models (HMM) are a set of statistical models used to characterize the statistical properties of a signal [61], [62]. HMM's have been used extensively for speech recognition, where data is naturally one-dimensional (1D) along the time axis. However, the equivalent fully-connected two-dimensional HMM would lead to a very high computational problem [63]. Attempts have been made to use multi-model representations that lead to pseudo-two-dimensional HMMs [64]. These models are currently used in character recognition [65], [66].

In [67], Samaria et. al. proposed using the 1D continuous HMM for face recognition. Assuming that each face is in an upright, frontal position, features will occur in a predictable order, i.e. forehead, eyes, nose etc. This ordering suggests the use of a top-to-bottom model, where only transitions between adjacent states in a top to bottom manner are allowed [68]. The states of the model correspond to the significant facial features such as forehead, eyes, nose, mouth and chin [69]. The observation sequence $O$ is generated from an $X \times Y$ image using an $X \times L$ sampling window with $X \times M$ pixels overlap. Each observation vector is a block of $L$ lines. There is an $M$ line overlap between successive observations [70].

Given $c$ face images for each subject of the training set, the goal of the training stage is to optimize the parameters of the HMM to "best" describe the observations in the sense of maximizing the probability of the observations given the model. Recognition is carried out by matching the test image against each of the trained models. To do this, the image is converted to an observation sequence and then model likelihoods are computed for each face model. The model with the highest likelihood reveals the identity of the unknown face.

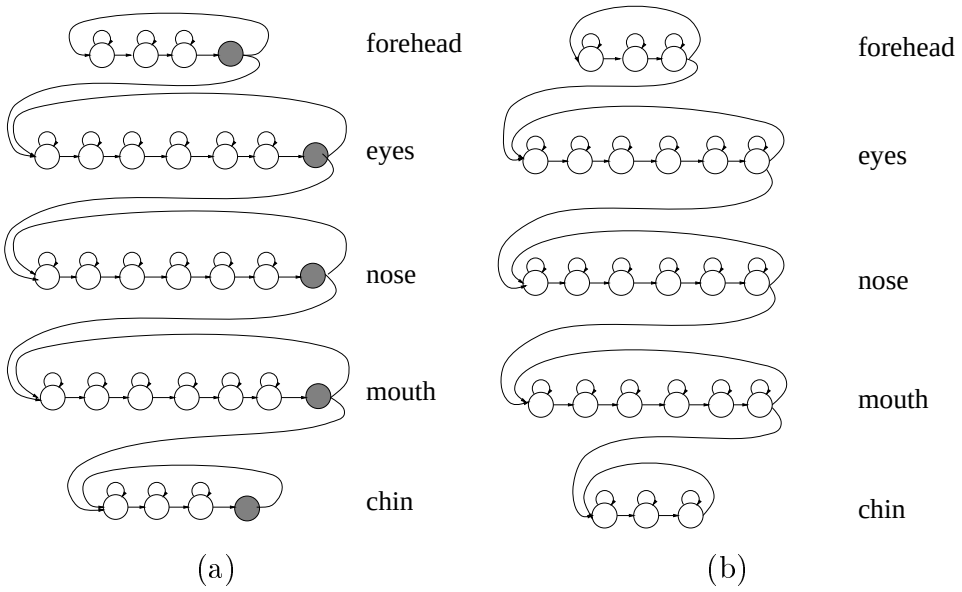



Figure 2.1: A HMM with end-of-line states (a) and an unconstrained HMM (b)

In [71], Samaria increased the number of states used to characterize each of the significant facial features. The observation sequence used with this model is obtained by sliding a rectangular window from the left to right and from top to bottom of the image and using the pixels intensities extracted from each window as observation vectors. To preserve the two-dimensional structure of the data, a marker block was added at the end of each line in the image, and an additional end-of-line state was added to the structure of each horizontal HMM (Figure 2.1-a). The end-of-line states are allowed two transitions: one to the same row of states, and one to the next row of states. It was found [71] that by setting the initial standard deviation of the

end-of-line states to be small, and the means close to the intensity of the end-of-line marker block, the state topology was preserved and the parameters of the end-of-line states were unaltered after re-estimation. In the same work, it was shown that similar recognition results were obtained for the unconstrained P2D-HMM structure (Figure 2.1 -b). However, this topology allows a transition to a state corresponding to another facial feature from a block that is not at the end of a row, and consequently does not preserve the two dimensional structure of the data. Preliminary results showed that for this structure, the face recognition results are as high as 95%. However, due to the large dimension of the observation vectors used, the system required about four minutes for a face to be recognized on Sparc 20 workstation. Recently,the HMM with end of line states was used for the segmentation and classification of hand drawn pictograms [72].

## 2.4 Summary of the Related Work

Some of the most successful approaches to face detection and recognition were analyzed. Due to the fact that these methods were tested on different databases, a quantitative comparison can not been presented. The recognition results of the approaches discussed are summarized in Tables 2.1 and 2.2. In general, the template-based methods performed at high accuracy when the size of the images was fixed. Although the scaling of the templates can overcome this problem at the cost of increasing the complexity of the system, the HMM-based approach represents a more elegant size invariant method for face recognition and detection. However, the method presented by Samaria is computational very complex, and therefore impractical for real-time face detection and recognition systems.

| **Approach** | **Training Set** | **Training Set** | **Recognition Rate** | **Complexity** |
|---|---|---|---|---|
| **Correlation** [2] | 47 subjects | 47 subjects 4 images per subject | 100% | Not specified |
| **Correlation** [18] | 62 subjects 15 images per subject | 62 images 10 images per subject | 98.7% | 10-15min on Sparc 2 |
| **Eigenface** [19] | 16 subjects one image per subject | 2500images from 16 subjects | variations in: size 64% orientation 85% lighting 96% | 350 msec Sparc 1 |
| **Eigenface Parametric** [3] | 21 subjects | 21 subjects 9 images per subject | 78%-88% | higher than the view -based approach |
| **Eigenface View based** [3] | 21 subjects | 21 subjects 9 images per subject | 83%-90% | lower than the parametric-based approach |
| **Eigenfeatures** [3] | 45 subjects one image per subject | 45 subjects one image per subject | 95%-98% | Not specified |
| **KL-FSAT** [27] | 269 subjects one image per subject | 100 images 5 images per subject | 91% | higher than eigenface method |
| **Fisherfaces** [29] | 16 subjects 9 images per subject | 16 subjects one image per subject | 99.4% | lower than eigenfaces |

Table 2.1: Comparison of some of the face recognition approaches

| Approach | Training Set | Training Set | Recognition Rate | Complexity |
|---|---|---|---|---|
| **SVD** [35], [37] | 8 subjects 3 images per subject | 40 images 5 images per subject | 100% | high due to SVD calculation |
| **Auto Association and Classification NN** [42], [73] | 40 subjects 5 images per subject | 40 subjects 5images per subject | 20% | Not specified |
| **PDBNN** [45] | 40 subjects 5 images per subject | 40 subjects 5 images per subject | 96% | 0.1 sec on SGI Indy 100 MHz |
| **Convolutional NN** [43] | 40 subjects 5 images per subject | 40 subjects 5 images per subject | 96.2% | 0.5 sec on SGI Indy 100 MHz |
| **Dynamic Link Matching** [51], [73] | 40 subjects 5 images per subject | 40 subjects 5 images per subject | 80% | Not specified |
| **VFR** [60] | 40 subjects 5 images per subject | 40 subjects 5 images per subject | 92.5% | 320 sec on Pentium 200MHz |
| **HMM** [74] | 40 subjects 5 images per subject | 40 subjects 5 images per subject | 85% | 12 sec on Sparc 2 |
| **HMM** [71] | 40 subjects 5 images per subject | 40 subjects 5 images per subject | 90-95% | 4 min on Sparc 2 |

Table 2.2: Comparison of some of the face recognition approaches (cont'd)

# CHAPTER 3

# A Deformable Template - Based Approach for Face Detection

The goal of the system described in Figure 1.1 is to efficiently detect and recognize human faces, independent of the background, lighting conditions, face size and orientation, gender, race, age, hair style, eye wear (glasses/no glasses) and facial expression. Conceptually there are two main approaches for face detection that can be used. One is to compare every location in a given image with a face template or model and to determine the face location based on some error measure between the local image and the face model or template. This method has the advantage that the detection does not depend on the background, the number of people in the scene, or their relative position in the scene. However, due to the exhaustive search at all locations in the scene, this method is generally slow and, therefore, more difficult to use in real time applications. This approach will be discussed later in more detail when we present an HMM based approach for face detection and recognition. Another approach to the face detection problem [75] is to segment the foreground from the background, locate the position of the head from the foreground, and then use the head location to decide if it corresponds to a frontal face by comparing the local image to a face template or model. This method is more dependent on the background and on the number of people in the scene than the previous method, but under certain conditions, to be discussed later, can perform with high accuracy at a higher frame rate than the first method. This chapter focuses on the second method and presents

a deformable template-based approach for real-time face detection [76] from gray scale video sequences. The goal of our system is to efficiently extract human faces, independent of their size and orientation, from a known but uncontrolled background. A deformable template based model has been used to describe the human face. An eye detection module processes the extracted faces to determine the frontal faces for further processing.

## 3.1 System Overview

This section describes the overall face detection system which is shown in Figure 3.1. It has been shown experimentally [77], [33], [78], [79] that the skin color can be

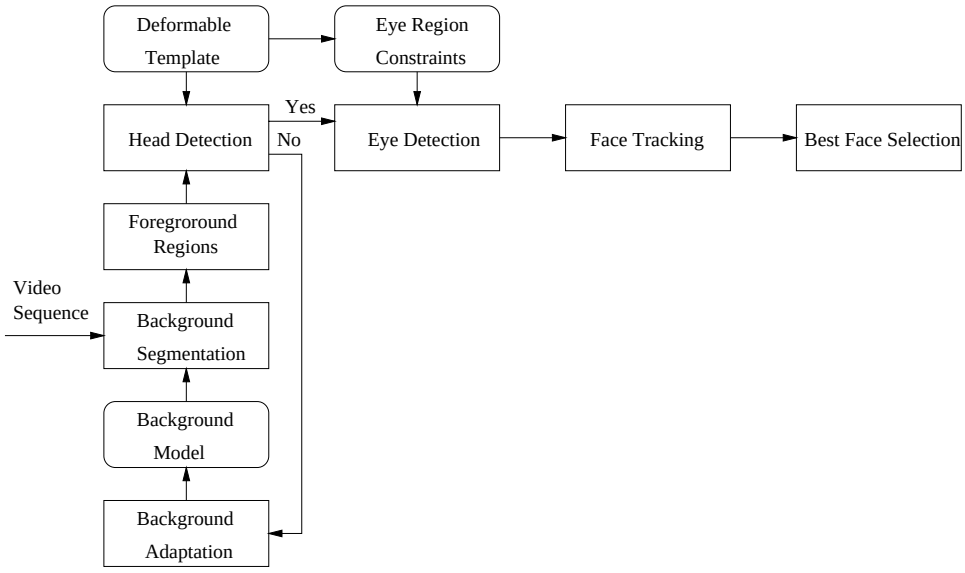



Figure 3.1: Overall deformable template - based face detection system

successfully used to detect the approximate location of faces in an image. The precise location of a face can be determined by analyzing other facial features in more detail. Unlike color images, in gray scale images the gray levels of the faces are often found in the background. Therefore the use of face gray level information disregarding any structural information of faces is not appropriate for face detection from gray scale

images.

In the first stage of our system, the pixels of each frame of the video sequence are segmented into background and foreground pixels based on a statistical model of the background pixels. The pixels with large deviations from the background model are taken to be foreground pixels. The set of connected foreground pixels determine the foreground regions. If the sizes of all the foreground regions, given by the total number of pixels inside a foreground region, fall below a fixed threshold, then the system assumes that no object is in a scene and updates the description of the background using the last frame of the video sequence. If one or more of the foreground regions exceeds the threshold, then the system assumes that these regions correspond to people in the scene and looks for the head position. The head position is determined by fitting an ellipse around the upper portions of the foreground regions. In fact, the head detection problem is reduced to finding the set of parameters $(x_0, y_0, a, b)$ that describe an elliptical deformable template. Parameters $x_0$ and $y_0$ describe the ellipse centroid coordinates, and $2a$ and $2b$ are the ellipse axis. The pose of the head is determined by running an efficient eye detection algorithm. The position of the head within each frame is tracked over consecutive frames. For each track, one or more faces that achieve the best overall scores, are tagged so that they may be used in a face recognition system. The overall face score incorporates a measure of how well data in each foreground region describes a frontal face according to the head deformable-template, and the pose of the head as determined by the location of the eyes and the size of the face. A detailed description of each of the blocks of Figure 3.1 is given in the later sections in this chapter.

## 3.2 The Segmentation of the Foreground Regions

The first stage of the system is the segmentation of foreground from background. The goal in this stage is to find all pixels in a frame of the video sequence that correspond

to objects other than those in the background. We assume that the background is known and stationary. This assumption is easily achieved if the position of the camera is fixed and the background does not change significantly over time. This scenario corresponds to indoor scenes such as stores, banks or laboratory environments. It is also assumed that several samples of the background over time are obtained during the initialization of the system.

With the above assumptions, the background scene is modeled as a texture with the intensity of each pixel given by a Gaussian density function with mean $\mu$ and variance $\sigma$, $N(\mu_b, \sigma_b)$. The values of $\mu_b$ and $\sigma_b$ are obtained from the background samples. The pixels in each frame are classified as foreground if $p(O(x,y)|N(\mu_b,\sigma_b)) < T$ and as background if $p(O(x,y)|N(\mu_b,\sigma_b)) > T$. The observation $O(x,y)$ represents the intensity of the pixels at location $(x,y)$ and $T$ is constant threshold. The connectivity analysis of the foreground pixels generates connected sets of pixels, i.e. sets of pixels that are adjacent or touching. Each of the above sets of pixels describe a foreground region. Small foreground regions are assumed to be due to shadow, camera noise and lighting variations and are removed.

The background model (the mean and variance) is updated, using the pixel intensities in frames where no people are detected. The background adaptation algorithm [80], allows for robust foreground background segmentation in the presence of illumination variation or shadowing effects.

## 3.3 The Deformable Template for Head

It is common to approximate the support of the human face by an ellipse [81], [14], [82], [83], [84] since an ellipse provides a good approximation of the head shape for a large range of views obtained from rotations in both the image plane and in a plane perpendicular to the image plane. A deformable template of elliptic shape also allows for variations in the size of the head. The deformable template is parameterized by

the set $(x_0, y_0, a, b)$ where $x_0$ and $y_0$, are the coordinates of the ellipse centroid and $a$ and $b$ are the axis of the ellipse (Figure 3.2). The set $(x_0, y_0, a, b)$ is determined

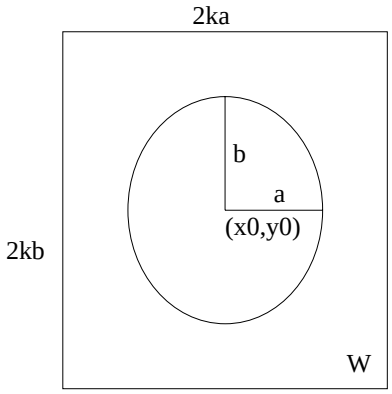



Figure 3.2: The deformable template for face

for each foreground region through an efficient ellipse fitting algorithm that will be descried later in this chapter. It is, however, very important to find a measure of how well the deformable template characterizes the available data i.e. the foreground region. For this, we define a rectangular template, $W$ in Figure 3.2, that has the same center $(x_0, y_0)$ as the ellipse template. The sides of the rectangular template are proportional to the axes of the ellipse, $2ka$ and $2kb$ in Figure 3.2, where $k$ is some constant. The head score $S_{head}$ is defined according to the following equation:

$$S_{head} = \frac{\sum_{R_e} f(O(x,y)) + \sum_{R_r} b(O(x,y))}{\sum_{R_e} b(O(x,y)) + \sum_{R_e} f(O(x,y)) + \sum_{R_r} b(O(x,y)) + \sum_{R_r} f(O(x,y))} \quad (3.1)$$

where $b(O(x,y))$ and $f(O(x,y)$ are the background and foreground functions that are defined by

$$b(O(x,y)) = \begin{cases} 1 & \text{, if } p(O(x,y)|N(\mu_b, \sigma_b) > T \\ 0 & \text{, otherwise} \end{cases}$$

and

$$f(O(x,y) = \begin{cases} 1 & \text{, if } (p(O(x,y)|N(\mu_b, \sigma_b)) < T \\ 0 & \text{, otherwise} \end{cases}$$

In equation 3.1, $R_e$ denotes the set of pixels inside the ellipse and $R_r$ denotes the set of pixels inside the rectangular template and outside the ellipse. The above score can be interpreted as the number of pixels in the rectangular template correctly classified by the elliptical template, since we expect that all pixels inside the ellipse to belong to the foreground while all the pixels outside the ellipse and inside the rectangle to belong to the background. It is also important to note that $S_{head}$ is normalized such that $0 \leq S_{head} \leq 1$. This score plays a very important role in our approach and will be used in the next stages to determine the number of people in a foreground region, and to identify the best face out of a face track.

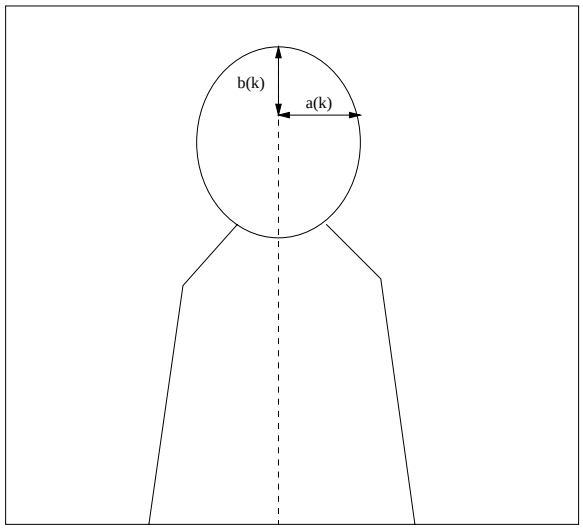



Figure 3.3: The head detection algorithm

## 3.4 The Head Detection Algorithm

In this section we will assume for simplicity, that each foreground region corresponds to only one person in the image. However, this might not always be the case. Due to body occlusions or shadowing effects, a given foreground region may contain, more

than one person, particularly in crowded scenes. This case will be discussed later in this chapter. Assuming that each foreground region corresponds to one person, the head is detected by fitting an ellipse around the upper portion of the foreground region. The ellipse parameters $x_0, y_0, a$ and $b$ are related through the following equation:

$$((x - x_0)/a)^2 + ((y - y_0)/b)^2 = 1 \tag{3.2}$$

The objective in an ellipse fitting algorithm is to find the estimated parameters $\tilde{x_0}, \tilde{y_0}, \tilde{a}$ and $\tilde{b}$ of the ellipse that best describe a given contour. A general technique for fitting an ellipse to a contour or region is the use of the Hough Transform. However, the computational complexity of the Hough Transform as well as the need for a robust edge detection algorithm make it inappropriate for real-time applications.

Our approach for ellipse fitting is an inexpensive recursive technique that reduces the search for the ellipse parameters from a four dimensional space $(\tilde{x_0}, \tilde{y_0}, \tilde{a}, \tilde{b})$ to a one dimensional space. The parameter space of the ellipse is reduced based on the following observations:

1. The centroid of the ellipse is located on the so called *vertical skeleton* of the region representing the person (Figure 3.3). The vertical skeleton is computed by taking the middle point between the left most and the right most points for each row of the region. Since the centroid of the ellipse is on the vertical skeleton, the estimated width of the ellipse $\tilde{a}$ is given by the distance between the left most and right most point of a foreground region at the line corresponding to the position of the estimated centroid of the ellipse.

2. The $\tilde{b}$ parameter of the ellipse (the height) can be calculated as the distance between the highest and the lowest point of the vertical skeleton within the head region (Figure 3.3). The highest point of the head region corresponds to the top of the skeleton since it is assumed that people are oriented vertically. However, finding the lowest point of the head region is generally very difficult

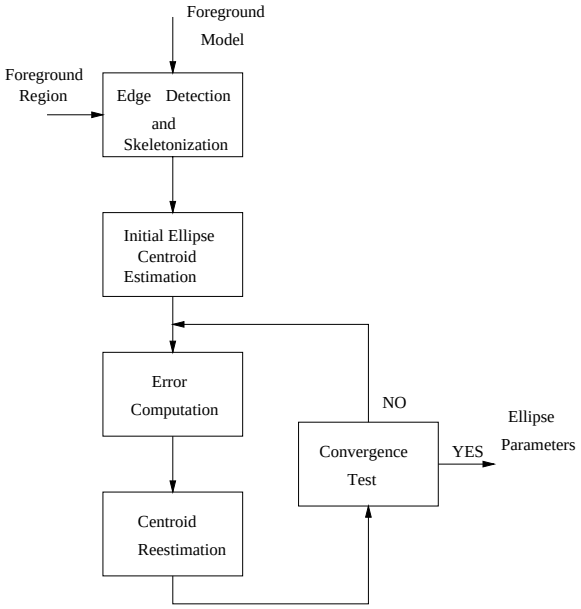



Figure 3.4: The ellipse fitting algorithm

to obtain with high accuracy due the difficulties in finding the chin line. The chin line does not represent an edge between the background and foreground region since it is entirely contained within the foreground region. Therefore a more complex edge detection technique must be used to determine the lower contour of the face. In our approach we avoid the problem of finding the chin edge by taking advantage of the fact that the aspect ratios of the ellipses ($\frac{\tilde{b}}{\tilde{a}}$) representing faces cluster in a small range. Specifically, we have found that the aspect ratio generally falls within a narrow range centered around $M = 1.4$. Therefore our deformable face template is constrained to have an aspect ratio of 1.4.

From the above observations it is clear that the problem of finding the ellipse parameters is reduced to finding the point on the vertical skeleton that "best" satisfies the constraint imposed by the aspect ratio of the ellipse. This is equivalent to finding

$\tilde{y_0}$ on the vertical skeleton that minimizes the error:

$$e = \tilde{b} - M\tilde{a} \tag{3.3}$$

The position of $\tilde{y_0}$ is iteratively computed using at each iteration $k$ a linear estimate

$$\tilde{y_0}^{(k+1)} = \tilde{y_0}^{(k)} + \mu e(k) \tag{3.4}$$

where,

$$e(k) = \tilde{b}^{(k)} - M\tilde{a}^{(k)} \tag{3.5}$$

and $\mu$ is a constant such that $0 < \mu \leq 1$. The ellipse fitting algorithm is illustrated in Figure 3.4. First the edges of the foreground regions are obtained, and the vertical skeleton is computed. The edges of the foreground region are found by choosing the left most and right most point at each line in a foreground region. The elements of the vertical skeleton are positioned on each line at the midpoint between the left and right edge of the foreground region, in the same line.

The initial estimate for the y-coordinate of the ellipse centroid, $\tilde{y_0}(0)$ must be chosen close enough to the top of the object on the vertical skeleton in order for the algorithm to perform well for all types of sequences from head-and-shoulder to full-body sequences. Specifically, if the initial centroid is chosen far from the top of the vertical skeleton, then the algorithm may converge to an ellipse that describes the whole body of the person. In our experiments, the distance from the initial centroid to the top of the vertical skeleton is chosen to represent 10 % of the height of the vertical skeleton.

The width of the ellipse at iteration $k$, $\tilde{a}^{(k)}$ is equal to the distance between the right most and left most point of the foreground region at the line corresponding to the current centroid position, $\tilde{y_0}^{(k)}$. Similarly the height of the ellipse at iteration $k$, $\tilde{b}^{(k)}$ is the distance between the current estimate of the centroid of the ellipse and the top of the vertical skeleton. The error at iteration $k$ is computed according to Equation 3.5 and the new centroid position is re-estimated according to Equation 3.4. The

iteration stops when the distance between two consecutive centroids is smaller than a threshold.

**Convergence of the Head Detection Algorithm**

Next we will show that the above algorithm converges to the correct ellipse parameters that best fits an elliptical contour. Let the parameters of the elliptical contour be $x_0, y_0, a$ and $b$. From Equation 3.2, the distance between the right most and left most points of the ellipse corresponding to $\tilde{y}_0(k)$ are determined by:

$$\tilde{a}^{(k)} = 2a\sqrt{1 - ((\tilde{y}_0^{(k)} - y_0)/Ma)^2} \tag{3.6}$$

and the distance between the top of the vertical skeleton and $y_0(k)$ is determined by:

$$\tilde{b}^{(k)} = y_0 + Ma - \tilde{y}_0^{(k)} \tag{3.7}$$

Hence, for $\mu = 1$, equation 3.4 becomes:

$$\tilde{y}_0^{(k+1)} - y_0 = Ma - Ma\sqrt{1 - ((\tilde{y}_0^{(k)} - y_0)/Ma)^2} \tag{3.8}$$

From the above equation it can be shown that:

$$|\tilde{y}_0^{(k+1)} - y_0|^2 < |\tilde{y}_0^{(k)} - y_0|^2 \tag{3.9}$$

for any $\tilde{y}_0^{(k)}$ for which $|\tilde{y}_0^{(k)} - y_0| < Ma$. This shows that the recurrence defined in equation 3.4 converges to $y_0$.

The head model can be more complex, and can include information from the eyes as well as other facial features. Using only a simple model for the human face may lead to the detection of objects that do not correspond to faces. These objects are removed in the next stage of our system by running an efficient eye detection algorithm, which will be described later in this chapter.

## 3.5 Region Splitting

The presence in a scene of people that are partially occluding each other leads to the generation of foreground regions that correspond to more than one person, and

therefore results in a failure of the head detection algorithm presented above. Based on the assumption that people in a scene are oriented vertically (not lying down or sanding on their heads) and that their faces are not occluded (although their bodies might be), there is a vector $\mathbf{x}_K = \{x_1, x_2, \ldots, x_{K+1}\}$ that represents the horizontal coordinates of the vertical segments that separate a foreground region of $K$ people into $K$ disjoint regions, each corresponding to one person. The overall score $H_K$ of a foreground region computed under the hypothesis that it represents $K$ people, combines the individual scores of each person, such as a weighted sum of the individual scores $S_i(x_i, x_{i+1})$.

$$H_K(\mathbf{x}_K) = \sum_{i=1}^{K} S_i(x_i, x_{i+1}) w_i \tag{3.10}$$

The individual scores $S_i(x_i, x_{i+1})$ are obtained from equation 3.1 computed in a region bordered by the vertical lines of horizontal coordinates $x_i, x_{i+1}$. This score can also incorporate information about the size of the ellipse templates. The score of the foreground region under the hypothesis that it represents $K$ people is given by:

$$H_K = \max_{all\ \mathbf{x}_K} H_K(\mathbf{x}_K) \tag{3.11}$$

The score of the foreground region is obtained as the best score of the foreground region under all hypothesis:

$$H = \max_{all\ K} \max_{all\ \mathbf{x}_K} H_K(\mathbf{x}_K) \tag{3.12}$$

As shown from the above equation in order to determine the foreground score, there are two optimization problems to be addressed. First is to find the number of people that best represent a foreground region, and second is to determine the set of vertical lines that best separate among people in the foreground region. To solve this double optimization problem, we first generate a set of hypothesis, each describing the number of people in each foreground region. For each hypothesis, the vector $\mathbf{x}_K^{opt}$ is found as the vector that maximizes the score of the foreground region $H_K$.

$$\mathbf{x}_K^{opt} = arg \max_{all\ \mathbf{x}_K} H_K(\mathbf{x}_K) \tag{3.13}$$

The best score $H_K$ for each hypothesis represents the score of the foreground region under the hypothesis that the foreground region corresponds $K$ people. Second the number of people that best represent the foreground region, $N^{opt}$ is obtained by choosing the best score of the foreground regions among all hypothesis:

$$N^{opt} = arg \max_{allK} H_K \tag{3.14}$$

In general, finding the vector $\mathbf{x}$ is difficult and becomes less accurate as the number of people in a foreground region increases since the head model considered here is very simple. However, in face recognition applications, it is not necessary to consider very crowded scenes since the faces in a crowded scene will typically be too small to be recognized. Under the assumption that no more than two people are in a foreground region, the system performed well on a large variety of sequences. Figure 3.5 shows a synthetic example of correct segmentation of a foreground region corresponding to one and two people. Figure 3.6 show the same example with an erroneous segmentation. The dashed ellipse shows the actual position of the ellipse found using our ellipse fitting algorithm. Unlike the score obtained in Figure 3.5 - a, the overall score in Figure 3.6-a computed using Equations 3.10 and 3.1 is small since both detected ellipses contain a large number of foreground pixels outside their contour. Therefore, the system will choose Figure 3.5-a as the correct foreground region segmentation. Similarly Figure 3.5-b and 3.6-b show two possible segmentations for a foreground region containing two people. The foreground region score computed according to Equation 3.10 is lower in Figure 3.6-b than in Figure 3.5-b because the segmentation there are a large number of background pixels inside the contour of the detected ellipse in Figure 3.6-b.

## 3.6 The Eye Detection Algorithm

A detected ellipse is a potential region of support for a human face. After the detection of these regions, a more refined model for the face is required in order to determine

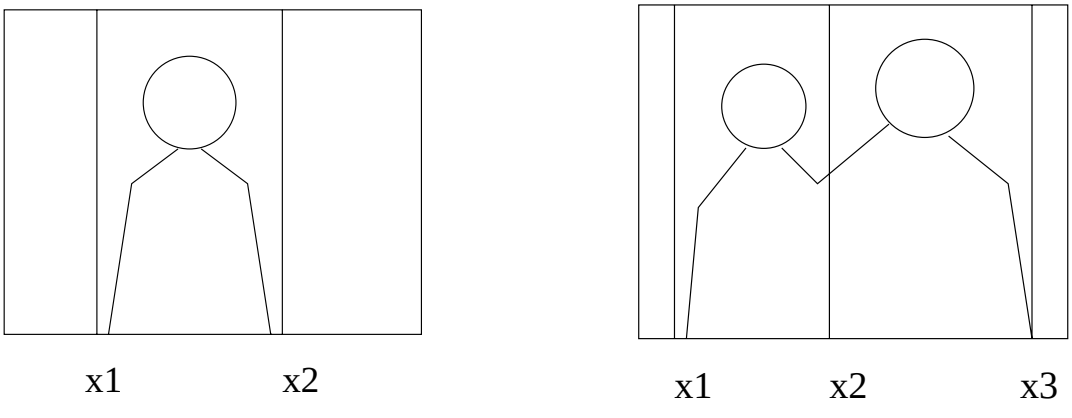



Figure 3.5: An example of correct segmentation of a foreground region representing (a) one person (b) two persons

which of the detected regions in previous stages correspond to valid faces. The use of an eye detection algorithm in conjunction with the head detection module improves the accuracy of the head model and discards regions corresponding to back views of faces or other regions that do not correspond to a face. The results of the eye detection algorithm are used to estimate the face pose and to determine the image containing the most frontal pose among a sequence of images. This result may then be used in recognition and classification systems.

There are many approaches in the computer vision literature to eye detection from known poses, more often frontal views [12], [2], [9]. However, the assumption of dealing with frontal faces is not valid for real world applications. Pentland et.al [3], described a template based approach for eye detection from unknown poses. This method requires to build one feature space for all poses of interest and also implies an exhaustive search for the eye position at all locations in a test image. In addition this method is inflexible to variations in scale of the faces and eyes. In this section we describe an efficient, scale invariant algorithm for eye detection and pose estimation. The algorithm makes use of the head location and the a priori knowledge of the size

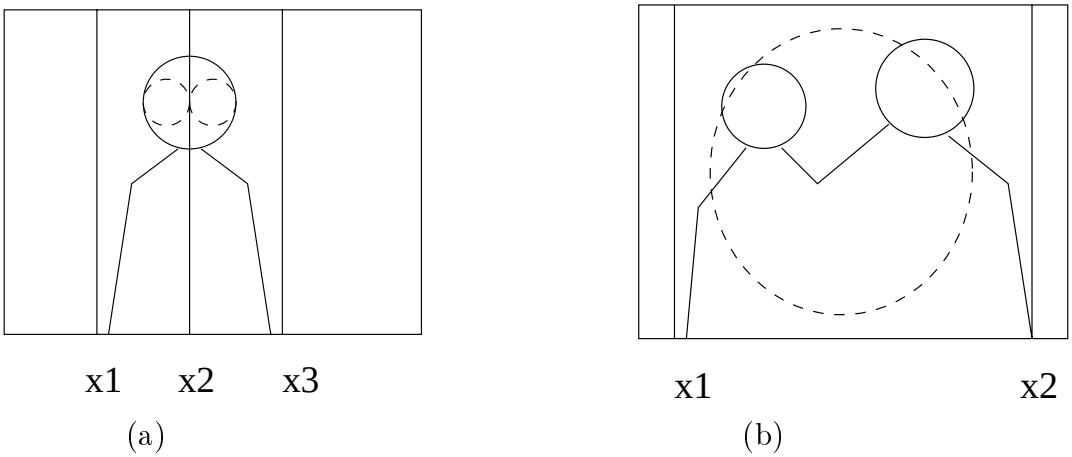



Figure 3.6: An example of incorrect segmentation of a foreground region representing (a) one person (b) two persons

and position of the eyes given the size of the face. Figure 3.7 shows the steps of the eye detection approach. In the first stage, the pixel intensities inside the face regions are modeled using a Gaussian density function with mean $\mu_f$ and variance $\sigma_f$. Since most of the pixels in a face region have the skin gray level, the statistics $\mu_f, \sigma_f$ of the pixels in this region will be close to the statistics of the pixels that represent the face skin. Pixels that are far from the face statistics i.e. $P(O(x,y)|\mu_f, \sigma_f) < T_f$ where $O(x,y)$ is the pixel intensity at $(x,y)$ and $T_f$ is a threshold, are assigned to facial features other than the skin.

A connectivity analysis of the extracted pixels generates connected sets of pixels, i.e. sets of pixels that are adjacent or touching. Each of these connected sets of pixels describe a region of the face that is different from the statistics of the skin gray level, such as hair, nose or eyes. These regions are further analyzed based on their size, i.e. the number of pixels in a region, and the relative position of their centroids, to determine which of them can represent the eyes in a face.

Given a frontal view of a face, one may expect to find the eyes in a rectangular window, or eye band, centered around the centroid of the ellipse such as $W_{eye}$ in

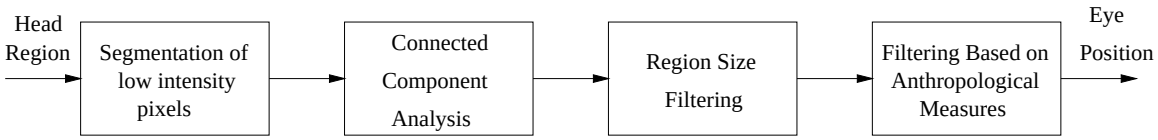



Figure 3.7: The eye detection block diagram

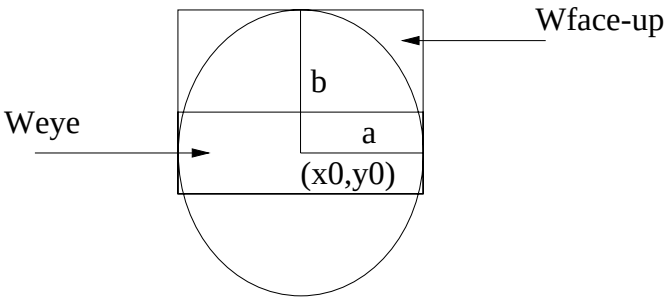



Figure 3.8: The regions of interest for eye detection

Figure 3.8. If frontal views are assumed, the size of the eye band can be chosen to include only non face regions that correspond to eyes. However, as mentioned before, the assumption of frontal faces is not valid in a real world environment. If non frontal views are considered, some regions inside $W_{eye}$ may correspond to hair. Furthermore, these regions are in general close in size and intensity to the eye regions. Therefore, under the assumption of varying poses, the simple inspection of distances between the centroids of the regions and their relative positions inside the eye band cannot indicate which regions correspond to the eyes. Alternative approaches that use template or edge information may be used to detect the eyes in a more robust way, but these methods are in general very complex and sensitive to scaling variations.

In this section we present a simple method to discriminate eye and hair region by analyzing the size of the non-face regions in a larger window around the upper portion of the face such as $W_{face-up}$ in Figure 3.8. First, the regions having a small number of pixels due to camera noise or shadows are removed. Second, large regions can not represent eyes and correspond in general to hair. The size of the regions

selected at this stage is in the interval $[\theta_m, \theta_M]$ where $\theta_m$ is the minimum and $\theta_M$ is the maximum number of pixels allowed by our system to describe a valid eye region. Threshold values $\theta_m$ and $\theta_M$ are relative to the size of the ellipse that characterizes the head region.

In the last stage of our approach, the selected regions inside the eye band $W_eye$ are further analyzed based on the relative distances between their centroids. The eye regions are identified by analyzing the minimum and maximum distance in both horizontal and vertical directions between the regions inside this band.

## 3.7 Results and Conclusion

The approach presented in this chapter has been implemented on a PC platform with the aid of simple commercial devices such as an NTSC video camera and a monochrome frame grabber. The approach has been tested in a variety of scenarios from head-and-shoulder to full-body sequences, including scenes that contain one or more people and where human bodies (but not heads) occlude each other. This large set of sequences was obtained by running the system in a company environment for over four months.

Figures 3.9- 3.12 show some of the results obtained by running the system in a laboratory environment. These figures illustrate four different scenarios generated to demonstrate the performances under different conditions such as non-frontal poses, multiple occluding people, back views, and faces with glasses. In Figure 3.9 the face of a single person is detected. In this figure the ellipse is properly fitted around the face and the eyes are detected even with glasses. Figure 3.10 shows the back view of a single person in the scene. In this figure, the ellipse is fitted around the head, but no eye is detected indicating the robustness of the eye detection module. Figure 3.11 and 3.12 show two scenarios in which two people are present in the scene. In both figures, the body of one person is covering part of the body of the other person. In

Figure 3.9: An example of face detection in a head-and-shoulder scenario

Figure 3.10: An example of face detection for a back view of a head

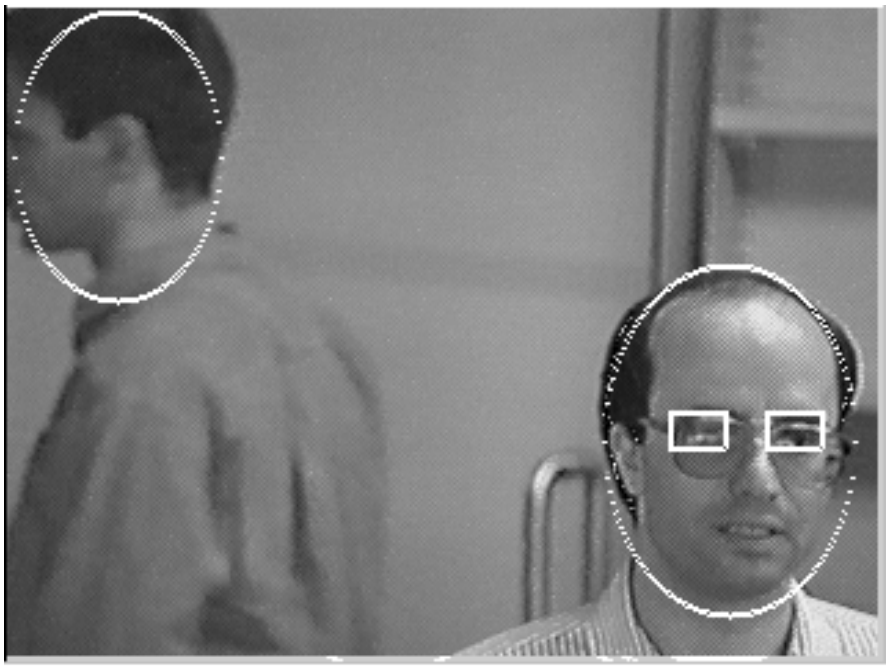


Figure 3.11: An example of face detection in the presence of two people

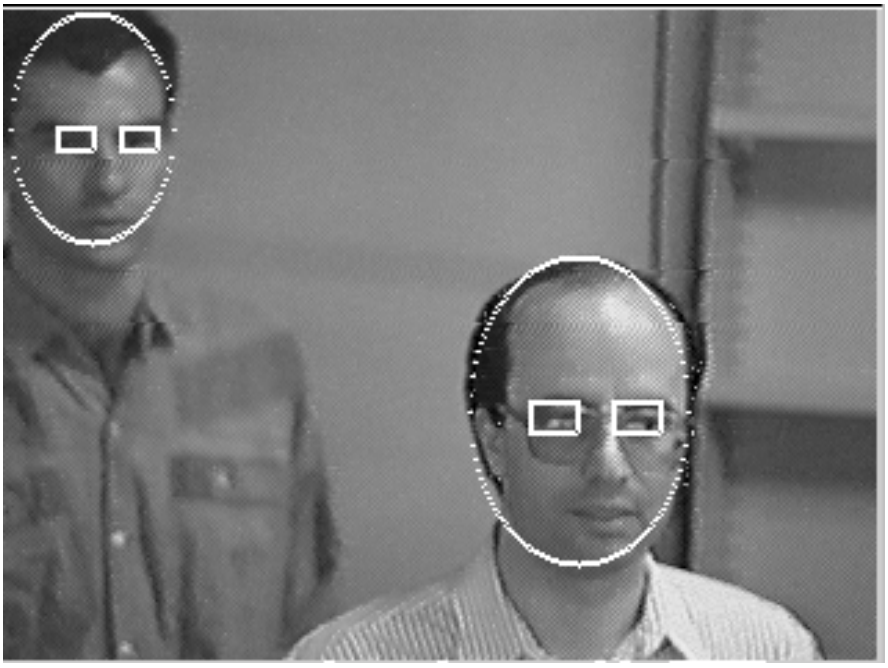


Figure 3.12: An example of face detection in the presence of two people in the presence of partial body occlusion

both cases the face and eyes are detected. The fact that the system is able to extract the faces and eyes in situations with body occlusion is a strong merit of the proposed approach. In Figure 3.12 the face of the person in the back has a non-frontal position. Also due to different distances from the camera the size of the two faces are different. The faces of both persons are detected indicating the robustness of the system to variations in parameters such as size and position of the face.

# CHAPTER 4

# A Hidden Markov Model for Face Detection and Recognition

Hidden Markov Models (HMM) have been successfully used for speech recognition and more recently in gesture [85] and action recognition [86], lipreading [87] or video indexing [88] where data is one dimensional over time. Successful HMM based methods were reported in recent years also for character recognition [66], [89], word spotting [90] and face recognition [71]. The use of an HMM for face recognition is motivated by their partial invariance to variations in scale and by the structure of faces. This chapter describes an efficient HMM based face recognition approach [91], and presents a new face detection method [92] that uses an HMM for faces. Unlike previous HMM based approaches for face recognition which use pixel intensities as observation vectors, in our approaches for both face detection and recognition the observation vectors are obtained from either the two-dimensional Discrete Cosine Transform (DCT) coefficients or from the Karhunen Loeve Transform (KLT) coefficients. The advantage of using image transform coefficients instead of pixel intensities include reduced sensitivity to noise, and the possibility of reducing the dimension of the observation vector by eliminating coefficients that are negligible in magnitude.

## 4.1 The Hidden Markov Model

A hidden Markov model consists of a Markov chain with a finite number of states, a state transition probability matrix, and an initial state probability distribution. Although the states are hidden (not directly observable), each state generates observations that are drawn according to some probability distribution (either discrete or continuous) [62]. The elements of an HMM are:

1. A set of $N$ states, $S = \{S_1, S_2, ..., S_N\}$, with the state at time $t$ denoted by $q_t \in S$.

2. The initial state probability distribution, $\mathbf{\Pi} = \{\pi_i\}$, where

$$\pi_i = P[q_1 = S_i] \ , 1 \leq i \leq N \tag{4.1}$$

3. The state transition probability matrix, $\mathbf{A} = \{a_{ij}\}$, where

$$a_{ij} = P[q_t = S_j | q_{t-1} = S_i] \ , 1 \leq i, j \leq N, \tag{4.2}$$

with $0 \leq a_{i,j} \leq 1$ and the constraint that

$$\sum_{j=1}^{N} a_{ij} = 1 \ , 1 \leq i \leq N$$

4. The probability distribution matrix for the observations, $\mathbf{B} = \{b_j(\mathbf{O}_t)\}$, where $b_j(\mathbf{O}_t)$ is the probability of observation $\mathbf{O}_t$ at time $t$ given that the state is $q_t = S_j$,

$$b_j(\mathbf{O}_t) = P(\mathbf{O}_t | q_t = S_j)$$

In a *discrete* HMM, the observation symbol probability is defined as:

$$b_j(k) = P[\mathbf{O}_t = v_k | q_t = S_j], \ 1 \leq j \leq N \tag{4.3}$$

where $V = \{v_1, v_2, ..., v_M\}$ is the set of all possible observation symbols (also called the *codebook* of the model), and $M$ is the number of different observation symbols.

In a *continuous density* HMM, the observations are characterized by continuous probability density functions. A general representation for the probability density function is a finite mixture of the form

$$b_i(\mathbf{O}_t) = \sum_{k=1}^{M_i} c_{ik} N(\mathbf{O}_t, \mu_{ik}, \mathbf{U}_{ik}) \ , 1 \leq i \leq N \tag{4.4}$$

where $c_{ik}$ is the mixture coefficient for the $k$th mixture in state $i$. Usually, $N(\mathbf{O}_t, \mu_{ik}, \mathbf{U}_{ik})$ is a Gaussian density with $\mu_{ik}$ the mean vector, and $\mathbf{U}_{ik}$ the covariance matrix.

A *tied-mixture* HMM is one in which all Gaussian components are stored in a pool and all state output distributions share this pool such that the output distribution for state $i$ is defined as:

$$b_i(\mathbf{O}_t) = \sum_{k=1}^{M} c_{ik} N(\mathbf{O}_t, \mu_k, U_k), \ 1 \leq i \leq N \tag{4.5}$$

The above equation differs from Equation 4.4 in that the Gaussian component parameters and the number of mixture components are state independent.

Using a shorthand notation, a HMM is defined as the triplet

$$\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi}). \tag{4.6}$$

The use of an HMM for face modeling is motivated by their partial invariance to scaling and by the structure of the face. For frontal face images, the significant facial regions (hair, forehead, eyes, nose, mouth) come in a natural order from top to bottom, even if the images are taken under small rotations in the image plane and/or rotations in the plane perpendicular to the image plane. Each of these facial regions is assigned to a state in a left to right 1D continuous HMM with diagonal covariance matrix. Before a hidden Markov model may be used for face recognition, we must define the structure of the HMM, which includes the number of states, the set of allowable transitions, and the observations that are produced by each state. The model that we have used is a 6-state left-to-right HMM, where the states are

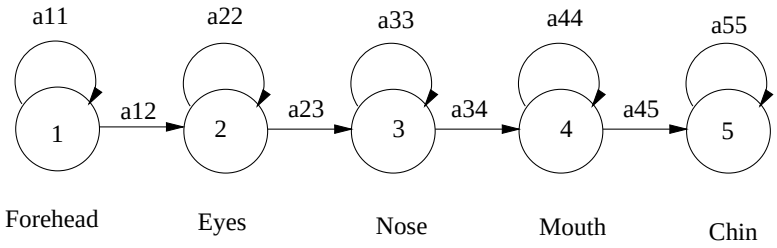



Figure 4.1: A five state face HMM

used to represent hair, forehead, eyes, nose, mouth, and chin (a 5-state version of this model is shown in Figure 4.1). Although HMM's are able to model one dimensional data such as speech signals over time, images are two dimensional. In next section we will describe how to obtain the observation sequence from the face images.

## 4.2 The Observation Vectors

Given an image $W$ pixels wide by $H$ pixels high that contains a face, image blocks of $L$ rows are extracted and used to form the observations (Figure 4.2). Adjacent blocks are allowed to overlap by $P$ rows. This overlapping allows the features to be captured in a manner that is independent of vertical position, while a disjoint partitioning of the image could result in the truncation of features occurring across blocks boundaries. Using a small value for $L$ can bring insufficient discriminant information to the observation vector, while using a large value for $L$ increases the probability of cutting across the features. However, the system recognition rate is not very sensitive to variations in $L$, as long as $P$ is large ($P \leq L - 1$) and $L \approx H/10$ . In [74] the observation vectors consist of all the pixel values from each of the blocks, and therefore the dimension of the observation vector is $L \times W$ ($L = 10$ and $W = 92$). The use of the pixel values as observation vectors has two important disadvantages: First, pixel values do not represent robust features, since they tend to be very sensitive to image noise as well as image rotation, shift, or changes in illumination. Second,

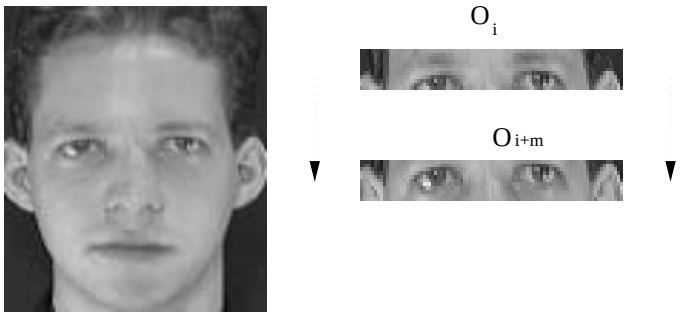



Figure 4.2: Face image parameterization and block extraction

the large dimension of the observation vector leads to high computational complexity of the training and detection/recognition system, and therefore increases the processing time required for both face detection and recognition. This can be critical for a face recognition system that operates on a large database, or when the detection/recognition system is used for real time applications. Instead of using the pixel intensities within each $L \times W$ image block, we form an observation vector from the coefficients from either the Karhunen Loeve Transform (KLT) or the two-dimensional Discrete Cosine Transform (2D - DCT) of each image block.

The optimal compression properties of the KLT as well as its decorrelation properties make it an attractive transform to use to extract the observation vectors. The KLT is an adaptive transform i.e. the basis functions of this transform depend on the signal to which it is applied. The basis functions of the KLT which are the eigenvectors of the covariance matrix of all input samples, are obtained as follows:

1. The image blocks extracted from all the images in the training set are arranged as vectors by concatenating the columns. Let $\mathbf{x}_i^{(r)}$ be the vector obtained from the $i$th block of the $r$th face image.

2. The sample mean of the vectors is computed according to:

$$\mu = \frac{1}{\sum_{r=1}^{R} T^{(r)}} \sum_{r=1}^{R} \sum_{i=1}^{T^{(r)}} \mathbf{x}_i^{(r)} \tag{4.7}$$

where $R$ is the number of face images in the training set and $T^{(r)}$ is the number of blocks extracted from the $r$th image. A new set of vectors is obtained by subtracting the mean from all $\mathbf{x}_i^{(r)}$:

$$\tilde{\mathbf{x}}_i^{(r)} = \mathbf{x}_i^{(r)} - \mu \tag{4.8}$$

Since the vectors $\tilde{\mathbf{x}}_i^{(r)}$ are characterized by zero statistical means, they can be used to determine the set of basis functions for the KLT.

3. The covariance matrix of the vectors $\tilde{\mathbf{x}}_i^{(r)}$ is computed according to:

$$\mathbf{C} = \frac{1}{\sum_{r=1}^{R} T^{(r)}} \sum_{r=1}^{R} \sum_{i=1}^{T^{(r)}} \tilde{\mathbf{x}}_i^{(r)} (\tilde{\mathbf{x}}_i^{(r)})^T \tag{4.9}$$

and the normalized eigenvectors $\mathbf{u}_k$ of the covariance matrix are determined by $\mathbf{C}\mathbf{u}_k = \lambda_k \mathbf{u}_k$ with $\lambda_k$ the eigenvalue corresponding to the eigenvectors of the covariance matrix $\mathbf{C}$.

Figure 4.3 shows the mean and the first five eigenvectors corresponding to the five largest eigenvalues. The mean vector models the structure of the set of face blocks. Note that, the values of the middle elements of the mean vector represent the average face gray level, while the first and last set of elements describe the average values of the hair or background which, in our database, is darker than the face. Looking at the eigenvectors, we see that they capture facial significant features, which are evident in Figure 4.3. It is also important to notice that all eigenvectors tend to be symmetric. This is not unexpected, since all faces in the database correspond to frontal views, and therefore the faces approximately symmetric with respect to an vertical axis that goes through the center of the face. Once the basis functions were obtained, the observation vectors were formed from the image blocks in both the training and testing sets using the following procedure:

1. The image blocks are arranged columnwise to form vectors.

2. The mean computed in 4.7 is subtracted from each vector.

3. The resulting vectors are projected onto the eigenvectors of the covariance matrix corresponding to the largest ten eigenvalues $\lambda_k$. The coefficients of the projections form the observation vector.

It should be noted that only the first few coefficients have significant values and that they tend to decrease with the index. Coefficients of index ten or larger are ignored since their values are negligible. Figure 4.4 shows some typical observation vectors for blocks corresponding to some significant facial regions are illustrated.

The use of KLT to obtain the observation vectors has however the disadvantage of requiring the computation of the basis functions from a large number of face images. An alternative is to use the two-dimensional DCT (2D DCT) to form the observation vectors. There are several reasons for using 2D DCT coefficients as observation vectors. First, many of the DCT coefficients tend to be small, and those that are large are generally concentrated around the low frequencies. For example, shown in Figure 4.5 are five sets of DCT coefficients corresponding to image blocks that contain specific facial features, e.g., eyes or nose. What is clear is that only a small number of the coefficients may be considered to be significant. Therefore, keeping only those coefficients that are large will result in a reduction in the dimension of the observation vectors, which leads, in turn, to a decrease in the complexity of the recognition system. The second advantage in using DCT coefficients instead of pixel intensities is that the DCT coefficients tend to be less sensitive to noise, image rotations and shifts, and changes in illumination.

## 4.3 Training The Face HMM

The process of training an HMM for face detection and for face recognition is similar. The only difference lies in the images that are used for training. For face detection, the images in the training set represent frontal faces of different people taken under different illumination conditions. All of the face images in the training set are used to

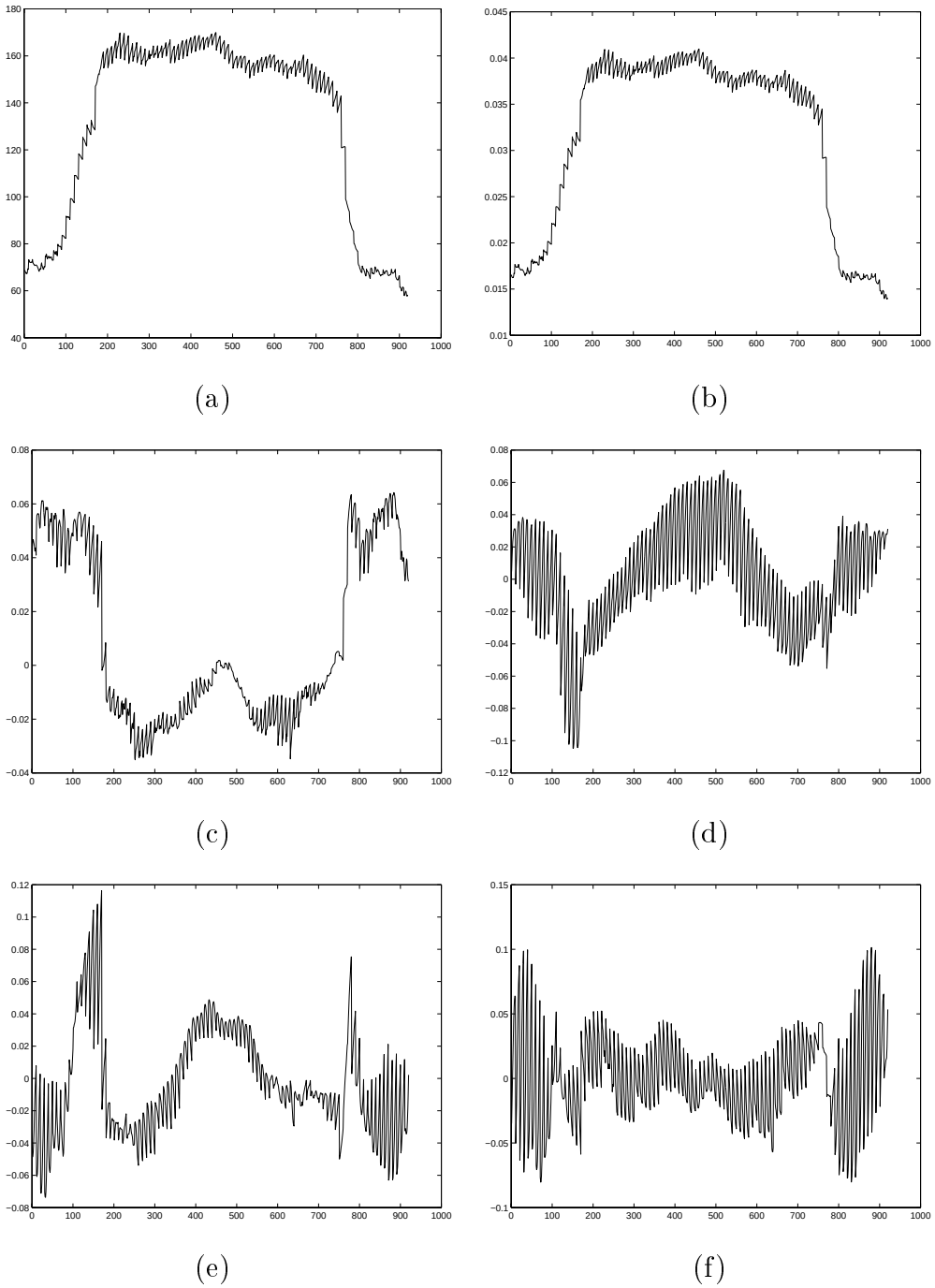



Figure 4.3: The average (a) and the first five eigenvectors (b-f) of the covariance matrix for the face blocks.

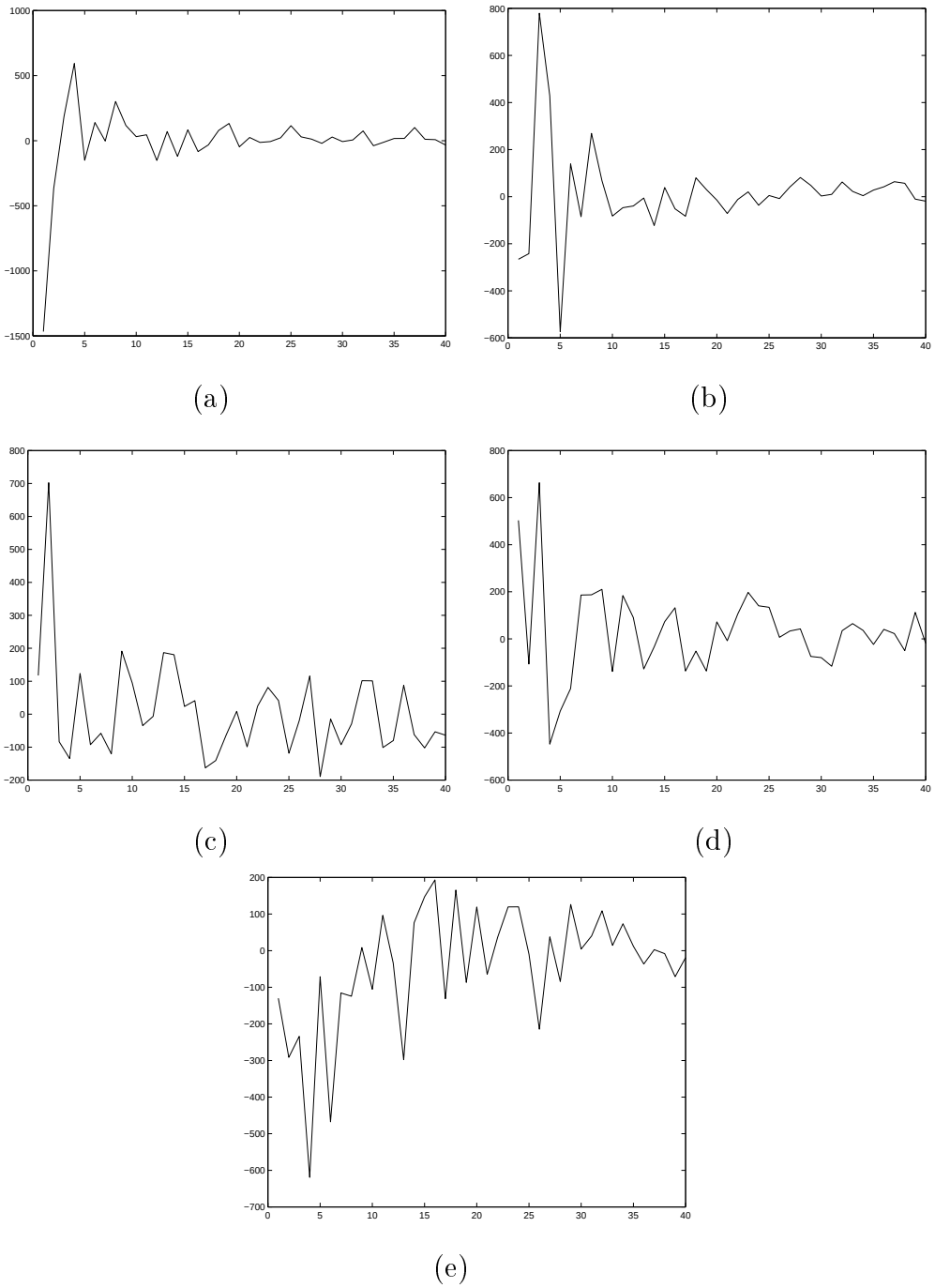



Figure 4.4: An example of KLT coefficients (first 40) extracted from face blocks corresponding to (a) hair, (b) forehead, (c) eyes, (d) nose and (e) mouth.

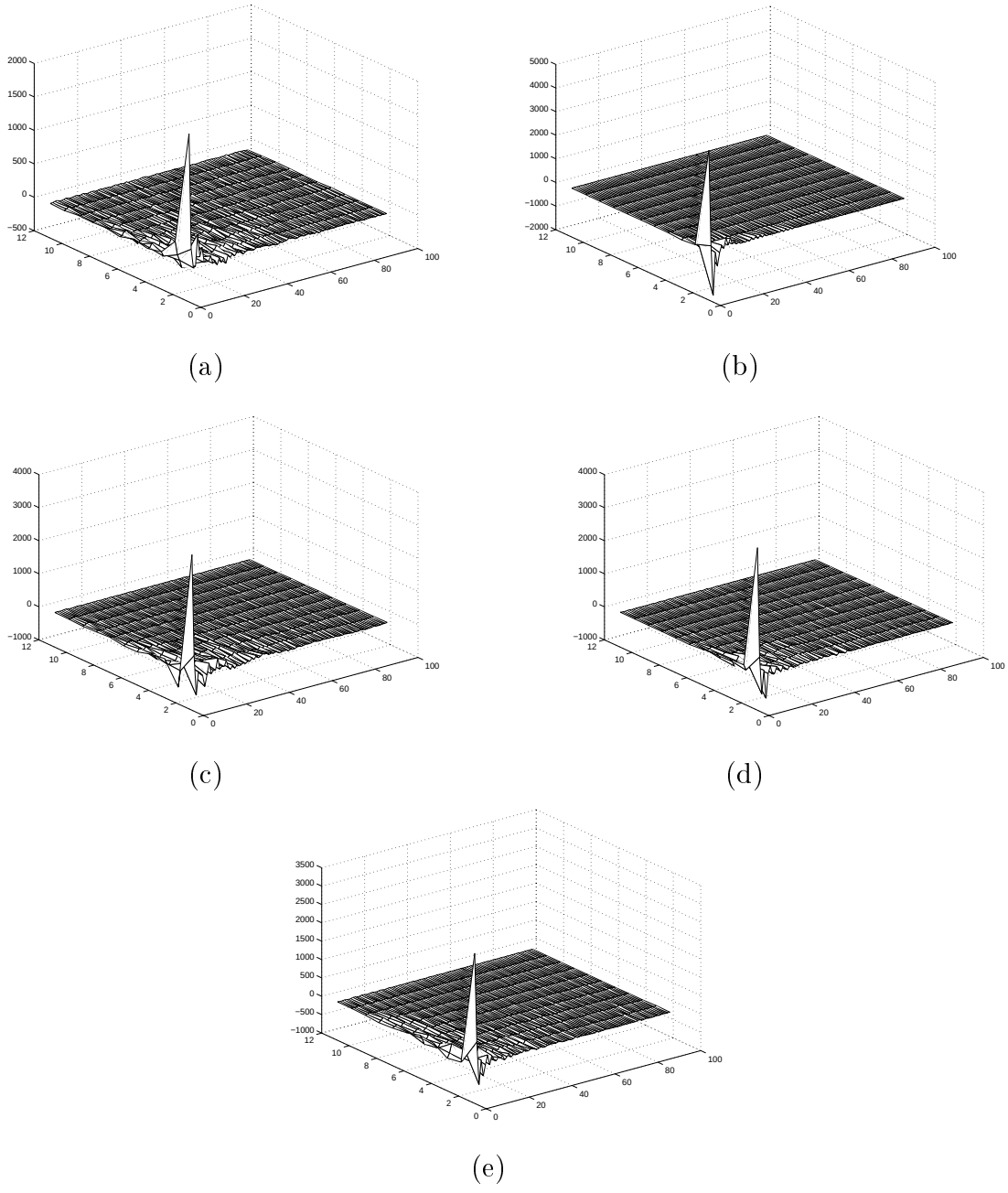



Figure 4.5: An example of 2D-DCT coefficients extracted from face blocks corresponding to (a) hair, (b) forehead, (c) eyes, (d) nose and (e) mouth.

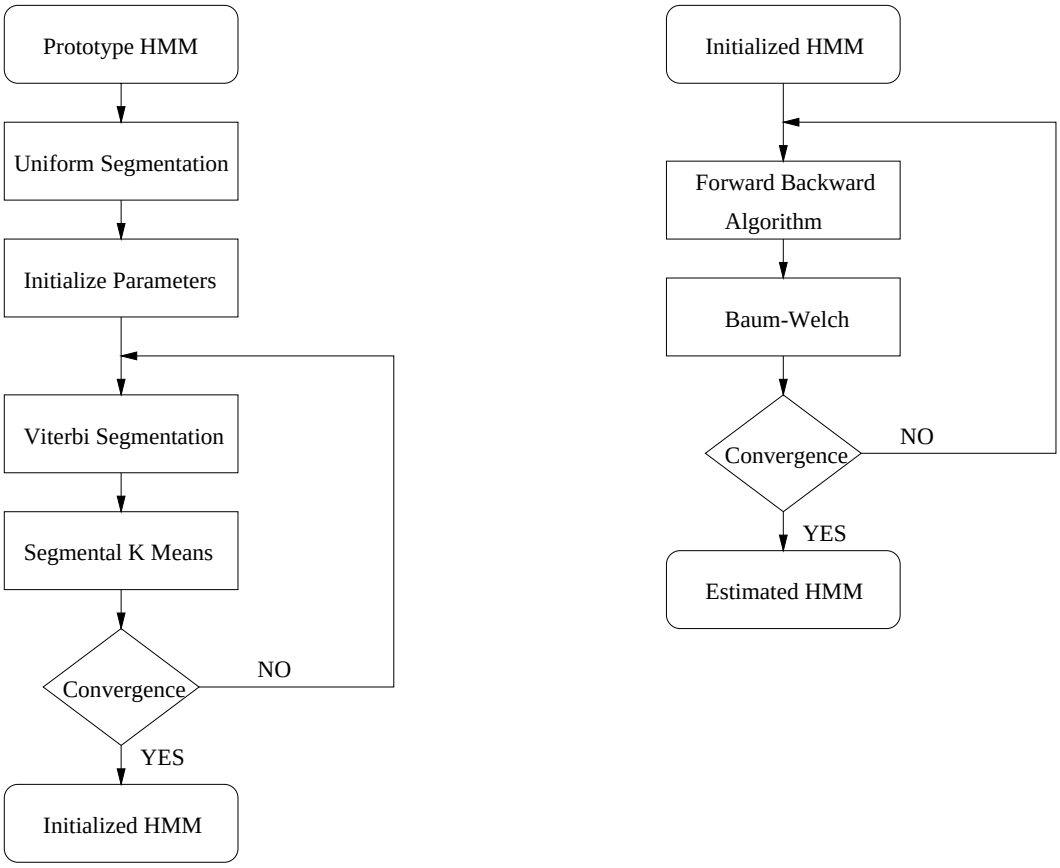



Figure 4.6: The training scheme for HMM

Figure 4.7: The initial face segmentation for the face HMM

train one HMM. For face recognition, each individual in the database is represented by an HMM face model. A set of images representing different instances of the same person are used to train each HMM.

After extracting the blocks from each image in the training set, the observation vectors (DCT or KLT coefficients) are obtained as explained in the previous section, and used to train each of the HMMs. In our system, we used $L = 10$ rows of length $W = 92$ (for a total of $L \times W = 920$ pixels per block), with an overlap of $P = 8$ rows between blocks (Figure 4.2). We used as observation vectors either 39 DCT coefficients (a $3 \times 13$ array of low-frequency coefficients), or ten KLT coefficients (corresponding to the largest ten eigenvalues). The use of these observation vectors results in a dimensionality reduction by a factor of approximately twenty-three for the DCT based observation vectors and ninety two for the KLT based observation vectors. Given a set of faces that form the HMM training set, the procedure used to train the hidden Markov model is as follows.

1. First, the training data is uniformly segmented (Figure 4.7), from top to bottom, according to the number of states of the HMM, and the observation vectors associated with each state are generated and used to obtain initial estimates of the observation probability matrix $\mathbf{B}$. The goal of this stage is to find a good es-

timate for the observation model probability $\mathbf{B}$. In [62], it has been shown that good initial estimates of the parameters are essential for rapid and proper convergence (to the global maximum of the likelihood function) of the re-estimation formulas. The initial values for $\mathbf{A}$ were set so that they are consistent with the top to bottom structure of the model, i.e., $a_{ij} = 0$ for $j < i$ and $j > i+1$, which means that we can only make a state transition from state $i$ back to itself, or into the next state $j = i + 1$. For the initial state probability distribution, we set $\pi_1 = 1$ and $\pi_i = 0$ for $i \neq 1$, i.e., the HMM begins in the first state.

2. At the next iteration, the uniform segmentation is replaced by the Viterbi segmentation. The result of segmenting each of the training sequences is, for each of $N$ states, a maximum likelihood estimate of the set of observations that occur within each state according to the current model. The iterations continue until the Viterbi segmentation likelihood at consecutive iterations is less than some threshold. At this point the parameters of the HMM are initialized.

3. Following the model initialization, the model parameters are re-estimated using the Baum-Welch re-estimation procedure [93]. This procedure adjusts the model parameters so as to maximize the probability of observing the training data, given each model.

4. The resulting model is then compared to the previous model (by computing a distance score that reflects the statistical similarity of the HMMs). If the model distance score exceeds a threshold, then the old model $\lambda$ is replaced by the new model $\tilde{\lambda}$, and the overall training loop is repeated. If the model distance score falls below the threshold, then model convergence is assumed and the final parameters are saved.

## 4.4 Face Recognition Using HMM

To use an HMM for face recognition, we begin with an image of a face that is to be recognized. First, the observation sequence is generated by scanning the image, from top to bottom, forming image blocks of $L$ rows, and extracting the observation vectors for each block. Then, the probability of the observation sequence given the HMM model for each face is found using the Viterbi algorithm. The model with the highest likelihood is selected, and the model reveals the identity of the unknown face (Figure 4.8). This system was tested on the Olivetti Research Ltd. (ORL) database [94],

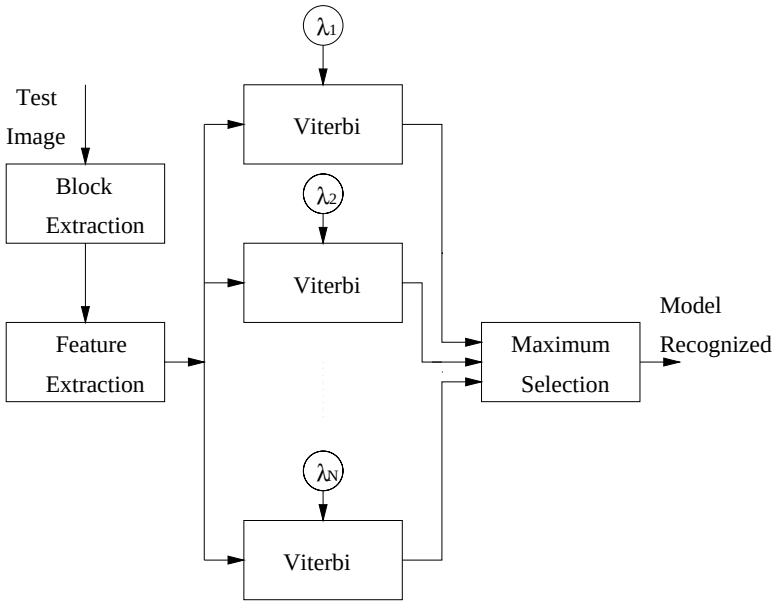



Figure 4.8: Face recognition using HMM

which consists of 400 images of 40 individuals (10 faces per individual) where each image is $92 \times 112$ pixels. Within this database are faces of males and females of different ages, with different facial expressions, hair styles, and eye wear (glasses/no glasses). Using half of the images to train the HMM, and the other half for testing, the recognition rate for Samaria's one-dimensional HMM [74] was 84%, whereas for our system it is slightly higher at 86%. However, it should be pointed out that this slight increase in the recognition rate is obtained along with a significant decrease

Figure 4.9: Face recognition results using HMM

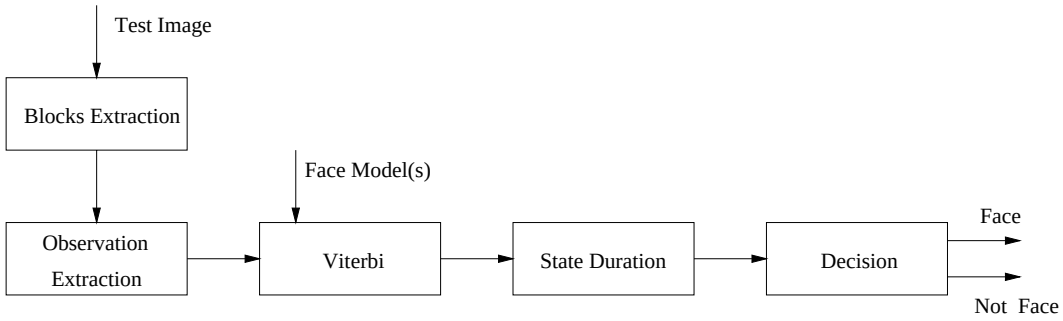



Figure 4.10: Face detection using HMM

in computational complexity that comes from the reduction in the dimension of the observation vectors from 920 down to 39, for the DCT based observation vectors or from 920 to 10 for the KLT based observation vectors.

## 4.5 Face Detection Using HMM

In this section, we investigate the performance of HMM to the problem of face detection. The goal of a face detection system is to locate the position of all faces within an image. If such a system is to be robust, then it must be able to detect faces of males and females of different races, independent of their appearance (facial hair, glasses/no glasses), and it should be insensitive to the size of the face within the image, the orientation of the head, and the background surrounding the face. One of the advantages in using an HMM for face detection compared to other approaches, such as template-based methods, is that it is partially invariant to deformations (scale and orientation) in the vertical direction.

The overall face detection system is illustrated in Figure 4.10. Since in our approach no assumptions are made with regard to the background, the face likelihood is computed for all rectangular patterns in the face image. The face likelihood for each pattern is given by the Viterbi likelihood weighed by a state duration correction factor. The rectangular patterns for which the face likelihood increases a fixed threshold are

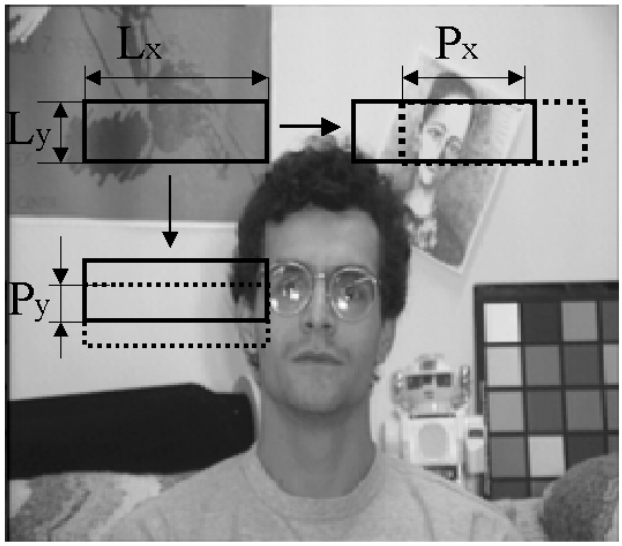



Figure 4.11: Block extraction for face detection using HMM

taken as valid faces.

Before we describe the face detection algorithm in more detail, it is important to notice that in this approach we can exploit the large overlap of the rectangular patterns and decrease the complexity of the face detection. In template based methods, the template features are extracted individually from each rectangular patterns and no computation can be saved even for the rectangular patterns that have large overlaps. By taking advantage of the vertical overlap of the rectangular patterns, in our approach the computation of the state probabilities for the observation vectors corresponding to overlapping regions can be saved. Based on this observation, the first step in our face detection approach is to extract the image blocks from a test image, followed by the computation of the observation vectors and the corresponding state probabilities. The image blocks are extracted by scanning the test image from left to right and from top to bottom. The blocks have the same width, height, and vertical overlap as those used in training. The horizontal overlap between blocks can

be chosen arbitrarily. An increase in the horizontal overlap, results in an increase in the precision of the detection system at the cost of increasing the complexity. Given an image of size $W \times H$, let $W_f$ be the width and $H_m$ and $H_M$ be the largest and smallest height of the face that we wish to find in the image (Figure 4.12). To detect a face of width $W_f$, and any height $H_f$, $H_m \leq H_f \leq H_M$, the number of blocks that are extracted in the horizontal direction, $T_x$, and the vertical direction, $T_y$, is

$$T_x = \frac{W - L_x}{L_x - P_x} + 1 \tag{4.10}$$

$$T_y = \frac{H - L_y}{L_y - P_y} + 1 \tag{4.11}$$

where $P_x$ and $P_y$ represent the amount of overlap between successive blocks in the horizontal and vertical direction, respectively, and $L_y$ and $L_x$ are the height and respectively the width of the blocks (Figure 4.11). Note that with a HMM for faces, the width of the block images is constrained to equal the width of the face that we wish to detect $W_f$ to form a one dimensional observation sequence. Therefore, as seen from Equations 4.10 - 4.11, the number of observations in the horizontal direction varies with the width of the face to be detected, while the number of observations in the vertical direction is constant. For each of the blocks extracted from the image, the observation vectors are obtained using the same technique as that used for training.

Following the extraction of the observation vectors from each test image and the computation of the state probabilities, the face likelihood is computed within each rectangular pattern of height $H_M$ and width $W_f$ via the Viterbi algorithm. One important advantage of our approach is that it can efficiently compute the face likelihood for all rectangular patterns having the same top left corner and the same width. This is a result of the fact that when the Viterbi algorithm is computed for a pattern of width $W_f$ , height $H_M$ and having the left top corner at $(x, y)$, it generates the partial likelihood of all the patterns with the same left top corner and width and of any height less than or equal to $H_M$. Given the constraints of our

system, $H_m \leq H_f \leq H_M$, only the face likelihood for pattern of length larger that $H_m$ are retained. The accuracy of the detection was improved by including the state duration modeling [95], [96], [97]. The duration $d_i$ of state $i$ is modeled using a Poisson distribution [97]. Since the duration modeling increases significantly the detection time, the state duration was introduced in the face likelihood score in the form of a correction factor that requires a very small increase in the computational complexity. Hence, following the computation of the Viterbi score, a new likelihood score is computed as:

$$\log \tilde{P}(\mathbf{O}, \mathbf{q}|\lambda) = \log P(\mathbf{O}, \mathbf{q}|\lambda) + \alpha \sum_{i=1}^{N_0} \log\ p_i(d_i)$$

where $\alpha$ is a constant, $p_i(d)$ is the Poisson distribution, and $d_i$ is the duration of the $i$th state, measured from the state segmentation. The parameter of the state duration probability $p_i(d)$ was measured in the training stage, from the segmented observation sequences. To deal with patterns of different heights, the partial log likelihood score is averaged over the length of the observation sequence. The last stage of the detection system is a decision block that selects the rectangular patterns that are taken to represent faces. The observation sequences that have a face model likelihood higher than a threshold are selected as possible face candidates. In order to remove the false alarms in rectangular patterns close to the actual position of a face, only the sequences with the maximum face likelihood over a vicinity around the current face location are selected as faces. Figure 4.13 shows the log likelihood surface for a test image in the MIT database. Each point of the surface represents the maximum face likelihood score over all rectangular patterns with the same left top corner. It can be seen that the largest values of the surface are obtained for rectangular patterns that are close to the actual location of the face. As expected the scores of the rectangular patterns that are shifted vertically from the actual face location have large likelihood scores, due to their face like appearance. However, faces that are shifted horizontally are rejected due to their low scores. Again this corresponds to our intuition since a horizontal shift in a face image deteriorates the view of the face image more than a

shift in the vertical direction. Rectangular patterns in a position close to the actual face location, which have a large likelihood score, are discarded since their scores fall below the score of the rectangular pattern that correspond to the actual face location.

The above face detection system has been tested in three experiments. First, a set of manually segmented images, from the MIT database were used to train one face model. The detection system was tested on images of the same database, that contain frontal faces of the same width as those used in training, but show significant variations in illumination. The detection rate for this experiment is 100%. Note that the use of the state duration model incresed the performance of the system by 10%, compared to our results reported in [91].

In the second and third experiment, the face images in the ORL database (resolution $92 \times 112$) were used to train eight face models. The images used to train each of the face models have been chosen to show similar facial characteristics. In the second experiment, the detection was performed on the same images from the MIT database as in the first experiment. In the last experiment, the test set consists of a larger number of images from the MIT database, that allowed for variations in face orientation (rotations in the image plane) and lighting conditions. The detection results (detection rate and false alarms) for the above experiments, when KLT or 2D-DCT -based observation vectors were used, are shown in Table 4.1. The false alarm rate is reported to the total number of rectangular patterns obtained from all the test images in each experiment.

Compared with the template-based methods for face detection our approach has the following advantages:

1. It is more flexible with respect to variations in scale, natural deformations in the vertical direction, and variations in illumination conditions.

2. It allows for a faster implementation of the face detection algorithm due to the breaking of the face templates into short image blocks, which are processed to obtain the observation vectors.

| | Experiment 1 | | Experiment 2 | | Experiment 3 | |
|---|---|---|---|---|---|---|
| | DR | FA | DR | FA | DR | FA |
| KLT | 100% | $\frac{0}{84,188,160}$ | 81.3% | $\frac{9}{84,188,160}$ | 72.6% | $\frac{39}{252,564,480}$ |
| 2D-DCT | 100% | $\frac{0}{84,188,160}$ | 79.2% | $\frac{10}{84,188,160}$ | 68.3% | $\frac{45}{252,564,480}$ |

Table 4.1: Comparison of the face detection rate (DR) and false alarms (FA) in different experiments using the face HMM

However both flexibility and fast implementation of the HMM based face detection presented in this section refer to the vertical direction, but not to the horizontal direction. This is determined by the fact that the HMM are one dimensional model while images are two dimensional.

If the height of the image changes, the size of the observation sequence will only increase, while a change of the image width will require either a rescaling of the width of the test image, or the use of a new training set where all faces have the new width. This effect is in general important in both face detection and recognition applications since the aspect ratio of faces tend to cluster in a small range and changes in height of faces determines changes in the widths of faces as well.

Although more efficient than the template-based approaches, the HMM based face detection cannot make use of the large overlap of the rectangular patterns in the horizontal direction. The observation vectors from rectangular patterns even with large overlap in horizontal direction must be recalculated, and no computation can be saved in the calculation of the observation vectors.

The two problems related to the face HMM and its one dimensional structure represent the reason for looking to a more complex HMM that is able to model two dimensional data better. Next chapter will discuss the theory of an embedded HMM and its applications for face detection and recognition.

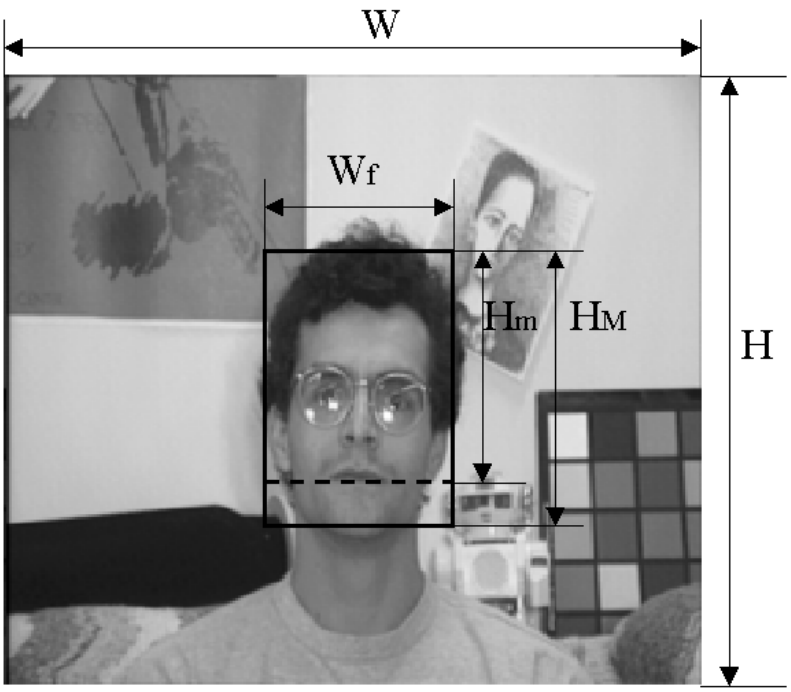
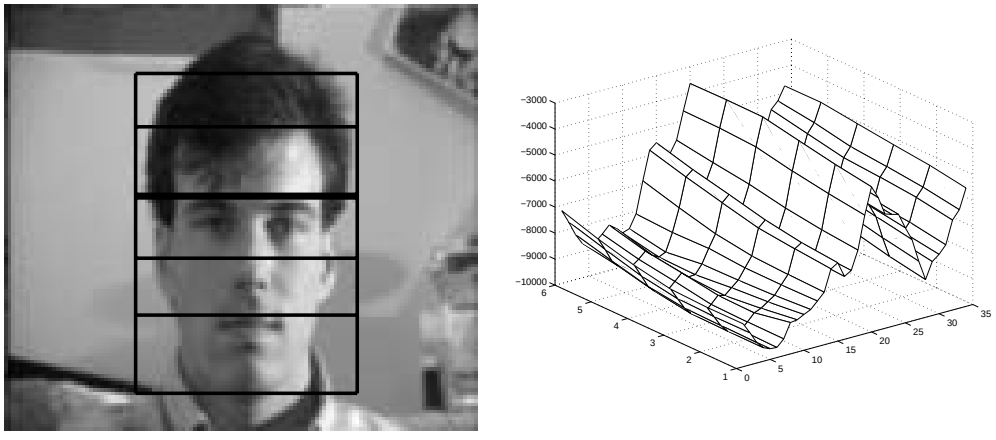



Figure 4.12: Image parameterization for face detection using HMM




(a) (b)

Figure 4.13: An example of face detection results (a) and the associated log likelihood surface (b)

Figure 4.14: Face detection results using HMM

# CHAPTER 5

# The Embedded Hidden Markov Model

Although successfully used in speech recognition where data is essentially one dimensional over time, a one-dimensional (standard) HMM is only moderately successful in modeling two-dimensional data, such as images. This fact has been discussed in the previous chapter and the experimental results for face detection and recognition confirmed our intuition. A fully connected two-dimensional HMM (Figure 5.1) as introduced by Levin and Pierraccini [63] can better characterize images. However it has been shown that the complexity of the training and recognition algorithms for the two-dimensional HMM [63], make this model inefficient in practical applications. However, under certain assumptions the complexity of this structure can be reduced, as shown in [98], where the authors used a two dimensional HMM for image classification. Another approach for modeling two dimensional data is to use an embedded HMM as introduced by Kuo and Agazzi [65] for character recognition. An embedded HMM (Figure 5.2) is a generalization of a HMM where each state in a one-dimensional HMM is itself an HMM. Thus, an embedded HMM consists of a set of *super states* along with a set of *embedded* states. The super states model the two-dimensional data along one direction, while the embedded HMMs model the data along the other direction. Note that a doubly embedded HMM is not a true two-dimensional HMM since transitions between the states in different super states are not allowed. In this chapter we will give a formal definition of the embedded HMM and we will describe the three key algorithms required for the training and recognition of the embedded HMMs: the decoding algorithm, the evaluation algorithm and the

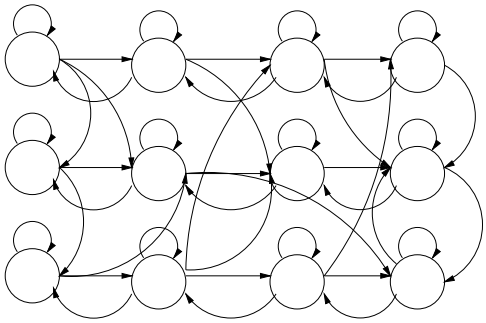


Figure 5.1: A fully connected two dimensional HMM

estimation algorithm. Finally we will discuss some implementation issues and we will describe the complexity of the algorithms presented in this chapter.

## 5.1 Embedded HMM Structure

In Chapter 4 we presented the structure of a one-dimensional HMM that we have used for face recognition. An embedded HMM shares many of the same features. Specifically, the elements of an embedded HMM are:

1. A set of $N_0$ super states, $\mathbf{S}_0 = \{S_{0i},\ i = 1, 2, \ldots N_0\}$.

2. The initial super state probability distribution, $\mathbf{\Pi}_0 = \{\pi_{0,i}\}$, where $\pi_{0,i}$ is the probability of being in super state $i$ at time zero.

3. The state transition matrix between the super states, $\mathbf{A}_0 = \{a_{0,ij}\}$, where $a_{0,ij}$ is the probability of making a transition from super state $i$ to super state $j$.

4. In an embedded HMM, each super state is itself an HMM, and the structure of these embedded HMMs is the same as that for a one-dimensional HMM. However, unlike a one-dimensional HMM, the number of states, the initial state probabilities, and the state transition matrix will, in general, depend on what super state the HMM is in. Therefore, some additional bookkeeping is necessary. What we have for each embedded HMM is the following:

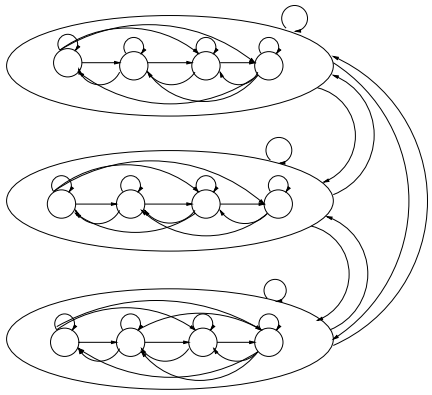


Figure 5.2: An embedded HMM with 3 super states

(a) The number of embedded states in the $k^{th}$ super state, $N_1^k$, and the set of embedded states, $\mathbf{S}_1^k = \{S_{1,i}^k,\ i = 1, 2, \ldots, N_1^k\}$.

(b) The initial state probability distribution of the embedded states, $\mathbf{\Pi}_1^k = \{\pi_{1,i}^k\}$, where $\pi_{1,i}^k$ is the probability of being in state $i$ of super state $k$ at time zero.

(c) The state transition matrix for the embedded states, $\mathbf{A}_1^k = \{a_{1,ij}^k\}$, where $a_{1,ij}^k$ specifies the probability of making a transition from state $i$ to state $j$ within super state $k$.

(d) The probability distribution matrix for the observations, $\mathbf{B}^k = \{b_j^k(\mathbf{O}_{t_0,t_1})\}$, where $b_j^k(\mathbf{O}_{t_0,t_1})$ is the probability of the observation $\mathbf{O}_{t_0,t_1}$, given that we are in embedded state $j$ in super state $k$. Note that for the observation vector $\mathbf{O}_{t_0,t_1}$ we have two subscripts, $t_0$ and $t_1$.

For a *discrete* embedded HMM it is assumed that $\mathbf{O}_{t_0,t_1}$ can take a finite number of observation symbols. Let $P$ be the number of different observation symbols and let $V$ be the set of all possible observation symbols (also called the codebook of the model), $V = v_1, \ldots, v_P$. In this case, $\mathbf{B}$ is the observation symbol probability matrix, i.e. $\mathbf{B}^k = \{b_j^k(p)\}$, where,

$$b_i^k(p) = P(\mathbf{O}_{t_0,t_1} = v_p | S_{0,k}, S_{1,i}^k, \lambda) \tag{5.1}$$

With a *continuous density* embedded HMM, the observations are characterized by a continuous probability density function which, as for a one-dimensional HMM, are taken to be finite Gaussian mixtures of the form,

$$b_i^k(\mathbf{O}_{t_0,t_1}) = \sum_{m=1}^{M_i^k} c_{im}^k N(\mathbf{O}_{t_0,t_1}, \mu_{im}^k, \mathbf{U}_{im}^k) \tag{5.2}$$

for $1 \leq i \leq N_1^k$, where $c_{im}^k$ is the mixture coefficient for the $m$th mixture in state $i$ of super state $k$, and $N(\mathbf{O}_{t_0,t_1}, \mu_{im}^k, \mathbf{U}_{im}^k)$ is a Gaussian density with a mean vector $\mu_{im}^k$ and covariance matrix $\mathbf{U}_{im}^k$.

A *tied-mixture* embedded HMM is one in which all Gaussian components are stored in a pool and all state output distributions share this pool such that the output distribution for state $i$ is defined as:

$$b_i^k(\mathbf{O}_{t_0,t_1}) = \sum_{m=1}^{M} c_{im}^k N(\mathbf{O}_{t_0,t_1}, \mu_m^k, \mathbf{U}_m^k). \tag{5.3}$$

The above equation differs from Equation 5.2 in that the Gaussian component parameters and the number of mixture components are state independent.

If we let $\mathbf{\Lambda}^k = \{\mathbf{\Pi}_1^k, \mathbf{A}_1^k, \mathbf{B}^k\}$ be the set of parameters that define the $k^{th}$ super state, then the embedded HMM is defined by the triplet

$$\lambda = (\mathbf{\Pi_0}, \mathbf{A_0}, \mathbf{\Lambda}). \tag{5.4}$$

where $\mathbf{\Lambda} = \{\mathbf{\Lambda}^1, \mathbf{\Lambda}^2, \ldots, \mathbf{\Lambda}^{N_0}\}$. Finally we introduce some notation that will be used extensively in this chapter. Let $\mathbf{O}_{t_0}$, $1 \leq t_0 \leq T_0$ denote the sequence $\mathbf{O}_{t_0,1}, \mathbf{O}_{t_0,2}, \ldots, \mathbf{O}_{t_0,T_1}$, and let $\mathbf{O}$ denote the sequence $\mathbf{O}_{t_0}, \ldots, \mathbf{O}_{T_0}$. Let $q_{t_0,t_1}$, $1 \leq t_1 \leq T_1$ denote the state of the observation $\mathbf{O}_{t_0,t_1}$. If $q_{t_0,t_1}^0$ is the super state of $\mathbf{O}_{t_0,t_1}$ and $q_{t_0,t_1}^1$ is the embedded state of $O_{t_0,t_1}$, then $q_{t_0,t_1} = (q_{t_0,t_1}^0, q_{t_0,t_1}^1)$. Let $\mathbf{q}_{t_0}^1 = q_{t_0,1}^1, \ldots, q_{t_0,T_1}^1$ be a sequence of embedded states and $\mathbf{q}^0 = q_0^0, \ldots, q_{T_0}^0$ be the sequence of super states. Let $\mathbf{q}_{t_0}$ represent the sequence $q_{t_0,t_1}, \ldots, q_{t_0,T_1}$ and let $\mathbf{q} = \mathbf{q}_0, \ldots, \mathbf{q}_{T_0}$.

## 5.2 The Decoding Algorithm

Given the observation sequence $\mathbf{O}$, the goal of the decoding algorithm is to recover the most likely set of states $\mathbf{q}$ that maximizes $P(\mathbf{O}, \mathbf{q}|\lambda)$. In order to describe an efficient algorithm for recovering the set of states it is useful to define the following variable:

$$\delta_{t_0}(i) = \max_{\mathbf{q}^1_{t_0}} \max_{\mathbf{q}_1,\ldots,\mathbf{q}_{t_0-1}} P(\mathbf{q}_1, \ldots, \mathbf{q}_{t_0-1}, q^0_{t_0} = i, \mathbf{O}_1, \ldots, \mathbf{O}_{t_0}|\lambda) \tag{5.5}$$

As shown by the above equation, $\delta_{t_0}(i)$ represents the best probability of observations and state sequence along a single state path that ends at $\mathbf{O}_{t_0}$ in super state $i$. From equation 5.5, it is clear that $P(\mathbf{O}, \mathbf{q}|\lambda) = \max_i \delta_{T_0}(i)$. The decoding algorithm can be efficiently implemented by making use of the efficient calculation of variable $\delta_{t_0+1}(i)$ from its previous values $\delta_{t_0}(j)$. From equation 5.5, it follows that

$$\begin{aligned}\delta_{t_0+1}(i) = \max_j [\max_{\mathbf{q}^1_{t_0+1}} \max_{\mathbf{q}_1,\ldots,\mathbf{q}_{t_0}} \quad & P(\mathbf{q}_1, \ldots, \mathbf{q}_{t_0-1}, q^0_{t_0} = j, \mathbf{O}_1, \ldots, \mathbf{O}_{t_0}|\lambda) a_{0,ji} \\ & P(\mathbf{q}^1_{t_0+1}, \mathbf{O}_{t_0+1}|q^0_{t_0+1} = i, \lambda)]\end{aligned} \tag{5.6}$$

The above equation can be written as

$$\begin{aligned}\delta_{t_0+1}(i) \quad = \quad & \max_j [\max_{\mathbf{q}^1_{t_0}} \max_{\mathbf{q}_1,\ldots,\mathbf{q}_{t_0-1}} [P(\mathbf{q}_1, \ldots, \mathbf{q}_{t_0-1}, q^0_{t_0} = j, \mathbf{O}_1, \ldots, \mathbf{O}_{t_0}|\lambda) a_{0,ji}]] \\ & \max_{\mathbf{q}^1_{t_0+1}} [P(\mathbf{q}^1_{t_0+1}, \mathbf{O}_{t_0+1}|q^0_{t_0+1} = i, \lambda)]\end{aligned} \tag{5.7}$$

It follows that

$$\delta_{t_0+1}(i) = \max_{all\ j} [\delta_{t_0}(j) a_{0,ji}] \max_{\mathbf{q}^1_{t_0+1}} P(\mathbf{q}_{t_0+1}, \mathbf{O}_{t_0+1}|q^0_{t_0+1} = i, \lambda) \tag{5.8}$$

It is clear now, from the above equation, that in order to compute $\delta_{t_0}(i)$ it is necessary to use the Viterbi algorithm [62] for the sequence $\mathbf{O}_{t_0}$ and recover the best sequence of embedded states corresponding to the super state $i$. Then, a Viterbi type of algorithm is performed for the sequence $\mathbf{O}_1, \ldots, \mathbf{O}_{T_0}$. The following steps formally characterize the decoding algorithm (Figure 5.3)for the embedded HMM:

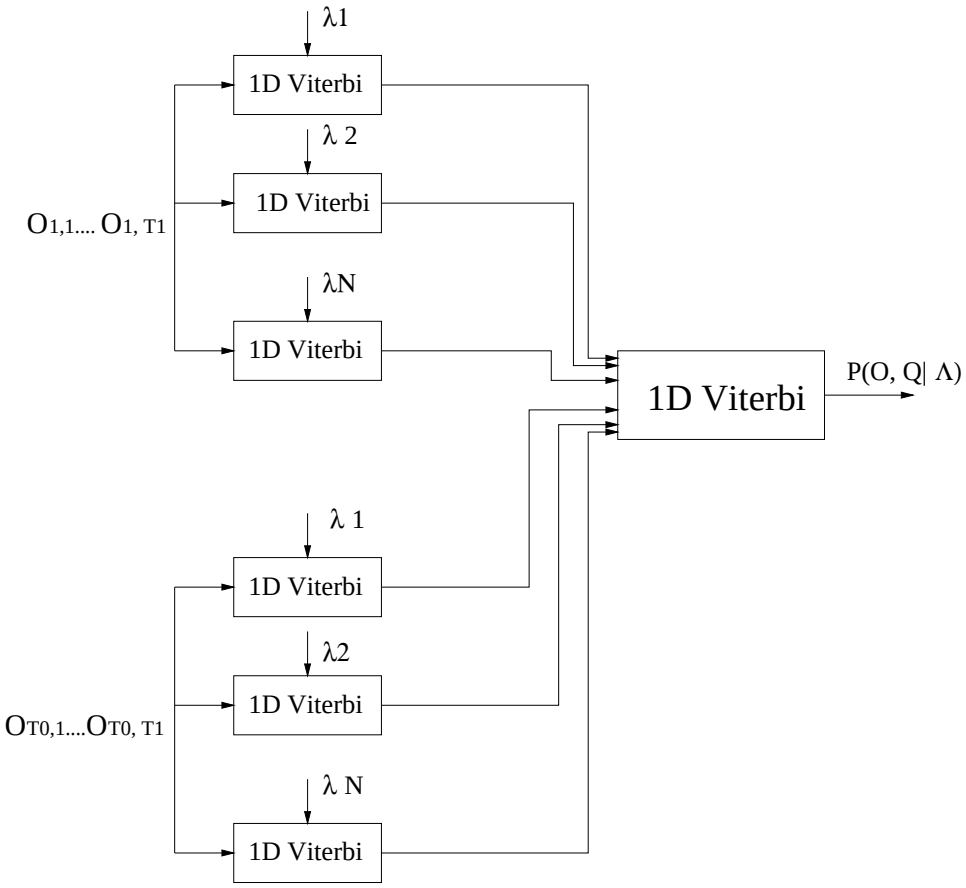



Figure 5.3: The decoding algorithm for the embedded HMM

1. Step 1:

   Compute $P_{t_0}^i = \max_{\mathbf{q}_{t_0}^1} P(\mathbf{O}_{t_0}, \mathbf{q}_{t_0}^1 | q_{t_0}^0 = i, \lambda)$ using the Viterbi algorithm for each super state and each observation sequence $\mathbf{O}_{t_0}$.

2. Step 2:

   (a) Initialization

   $$\begin{aligned} \delta_1(i) &= \pi_{0,1} P_0^i \\ \psi_1(i) &= 0 \end{aligned}$$

   The array $\psi_{t_0}(i)$ is used to keep track of the argument that maximize $\delta_{t_0}(i)$.

   (b) Recursion

   $$\begin{aligned} \delta_1(i) &= \max_{j \in [1, N_0]} [\delta_{t_0 - 1}(j) a_{0,ji}] P_{t_0}^i \\ \psi_1(i) &= \arg \max_{j \in [1, N_0]} [\delta_{t_0 - 1}(j) a_{0,ji}] \end{aligned}$$

   (c) Termination

   $$\begin{aligned} P^* &= \max_{j \in [1, N_0]} \delta_{T_0}(j) \\ (q_{T_0}^0)^* &= \arg \max_{j \in [1, N_0]} [\delta_{T_0}(j)] \end{aligned}$$

   $P^*$ in the above equation represents $\max_{all\ \mathbf{q}} P(\mathbf{O}, \mathbf{q}|\lambda)$ and $(q_{T_0}^0)^*$ is the super state at $T_0$ of the path $\mathbf{q}$ that maximizes $P(\mathbf{O}, \mathbf{q}|\lambda)$.

   (d) Backtracking

   The best path $\mathbf{q}$, which maximizes $P(\mathbf{O}, \mathbf{q}|\lambda)$ is obtained from the array $\psi_{t_0}(i)$ as follows:

   $$(q_{t_0}^0)^* = \psi_{t_0+1}((q_{t_0+1}^0)^*)$$

In order to reduce the computational complexity of the above algorithm and to avoid the underflow problems that often occurs in calculations that involve a large number of probability multiplications, all terms of the above algorithm can be changed

to their logarithmic form. In this representation all probabilities are replaced by their log values and the multiplications are transformed into additions. Due to the monotonicity of the logarithmic function, the maximum will be obtained for the same arguments, but of course in their logarithmic form.

In addition, the time required by the decoding algorithm can be significantly reduced when a parallel architecture is used. As shown in Figure 5.3, all the Viterbi scores obtained in the first step of the algorithm can be computed independently from each other and therefore all these scores can be computed at the same time using parallel implementation. Hence the total delay introduced by the decoding algorithm can be reduced to approximatively twice the delay introduced by the calculation of a standard Viterbi algorithm.

## 5.3 The Evaluation Algorithms

In this section we describe two algorithms for the computation of the probability $P(\mathbf{O}|\lambda)$ of the observation sequence $\mathbf{O}$ given our model $\lambda$. Because of their similarity with the algorithms for the one dimensional HMM, we will refer to the algorithms described in this section as the forward and backward algorithms.

### 5.3.1 The forward algorithm for the embedded HMM

Let us denote the forward variable for the sequence $\mathbf{O}_{t_0}$ as:

$$\alpha_{t_0,t_1}(i,j) = P(\mathbf{O}_{t_0,0},\ldots,\mathbf{O}_{t_0,t_1}, q^1_{t_0,t_1} = j | q^0_{t_0} = i, \lambda) \tag{5.9}$$

Similar to equation 5.9, the backward variable for the sequence $\mathbf{O}_{t_0}$ is given by:

$$\beta_{t_0,t_1}(i,j) = P(\mathbf{O}_{t_0,t_1+1},\ldots,\mathbf{O}_{t_0,T_1} | q^1_{t_0,t_1} = j, q^0_{t_0} = i, \lambda) \tag{5.10}$$

The forward and backward variables are computed iteratively using the forward and backward algorithm for a one-dimensional HMM [62]. The probability of the obser-

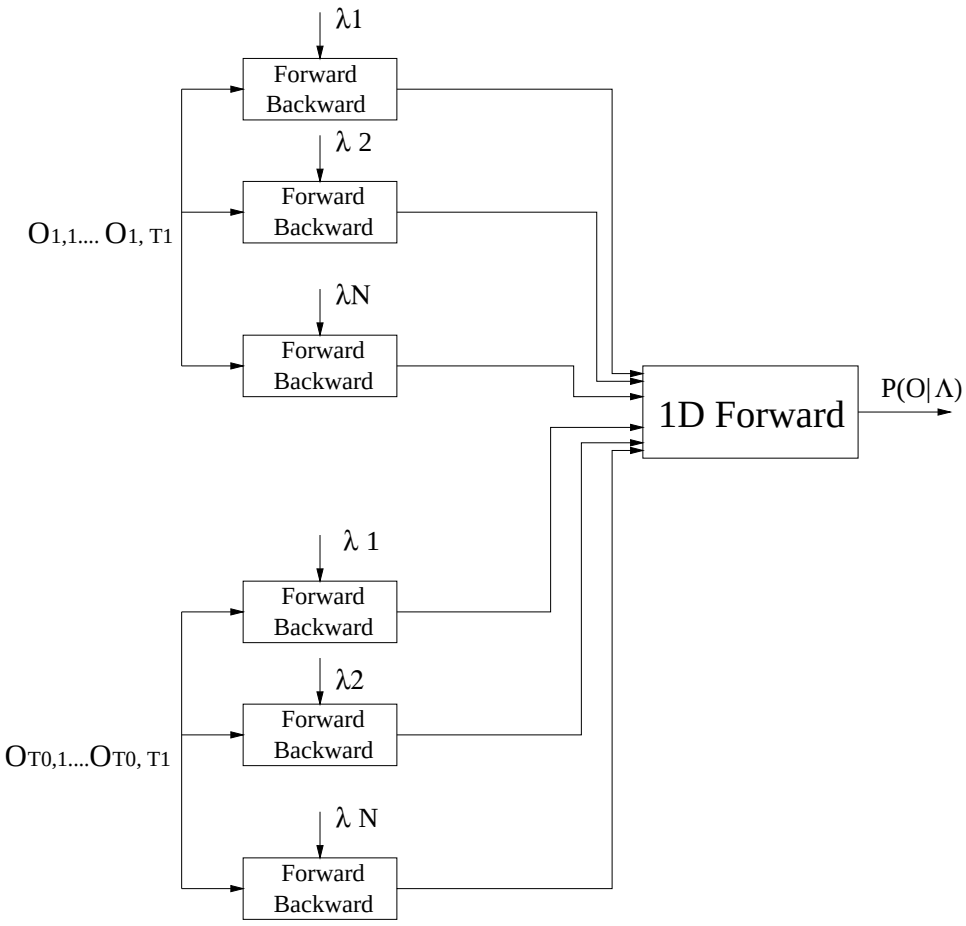



Figure 5.4: The forward algorithm for the embedded HMM

vation sequence $\mathbf{O}_{t_0}$, given the one dimensional HMM corresponding to super state $i$, is computed from both $\alpha_{t_0,t_1}(i,j)$ and $\beta_{t_0,t_1}(i,j)$ as follows:

$$P(\mathbf{O}_{t_0}|q^0_{t_0} = i, \lambda) = \sum_{t_1} \alpha_{t_0,t_1}(i,j)\beta_{t_0,t_1}(i,j) \tag{5.11}$$

An efficient algorithm for the computation of $P(\mathbf{O}|\lambda)$ is obtained if the forward variable for the sequence $\mathbf{O}_0, \mathbf{O}_1, \ldots, \mathbf{O}_{T_0}$ is defined as:

$$\alpha_{t_0}(i) = P(\mathbf{O}_1, \mathbf{O}_1, \ldots, \mathbf{O}_{t_0}, q^0_{t_0} = i|\lambda) \tag{5.12}$$

The forward variable $\alpha_{t_0}(i)$ describes the probability of the partial sequence $\mathbf{O}_0, \mathbf{O}_1, \ldots, \mathbf{O}_{T_0}$ and super state $i$ given the model $\lambda$. Therefore, $P(\mathbf{O}|\lambda)$ can be computed as:

$$P(\mathbf{O}|\lambda) = \sum_{all\ i} \alpha_{t_0}(i) \tag{5.13}$$

The forward variable $\alpha_{t_0}(i)$ is computed iteratively from its previous values and the probability of $\mathbf{O}_{t_0}$ given the super state $i$, $P(\mathbf{O}_{t_0}|q^0_{t_0} = i, \lambda)$:

$$\alpha_{t_0+1}(i) = [\sum_i \alpha_{t_0}(i)a_{0,ij}]P(\mathbf{O}_{t_0}|q^0_{t_0} = i, \lambda) \tag{5.14}$$

From the above discussion, the forward algorithm for the embedded HMM can be formally described by the following steps:

1. Step 1:

   Compute $P^i_{t_0} = P(\mathbf{O}_{t_0,t_1}|q^0_{t_0} = i, \lambda)$ for all $t_0$ and and super states $i$
   using the one dimensional HMM forward backward algorithms using 5.11.

2. Step 2:

   (a) Initialization

   $$\alpha_0(i) = \pi_{0,i}P^i_0$$

(b) Recursion

$$\alpha_{t_0+1}(i) = [\sum_i \alpha_{t_0}(i) a_{0,ij}] P^i_{t_0}$$

(c) Termination

$$P(\mathbf{O}_1, \mathbf{O}_1, \ldots, \mathbf{O}_{T_0}|\lambda) = \sum_i \alpha_{T_0}(i)$$

### 5.3.2 The Backward Algorithm

In a manner similar to the forward variable defined in 5.12, a backward variable can be defined for the sequence $\mathbf{O}_{t_0}$:

$$\beta_{t_0}(i) = P(\mathbf{O}_{t_0+1}, \ldots, \mathbf{O}_{T_0}|q^i_{t_0} = i, \lambda) \tag{5.15}$$

The backward variable $\beta_{t_0}(i)$ defines the probability of the partial observation sequence $\mathbf{O}_{t_0}, \ldots, \mathbf{O}_{T_0}$ given that $\mathbf{O}_{t_0-1}$ is in super state $i$. The variables can be computed iteratively from their future values $\beta_{t_0+1}(i)$ and $P(\mathbf{O}_{t_0}|q^0_{t_0} = i, \lambda)$.

$$\beta_{t_0}(i) = \sum_j a_{0,ij} P(\mathbf{O}_{t_0+1}|q_{0,t_0+1} = j, \lambda) \beta_{t_0+1}(j) \tag{5.16}$$

Therefore the backward algorithm for the embedded HMM consists of the following steps:

1. Step 1:

   Compute $P^i_{t_0} = P(\mathbf{O}_{t_0,t_1}|q^0_{t_0} = i, \lambda)$ for all $t_0$ and and super states $i$
   using the one dimensional HMM forward-backward algorithms using 5.11.

2. Step 2:

   (a) Initialization

   $$\beta_{T_0}(i) = 1$$

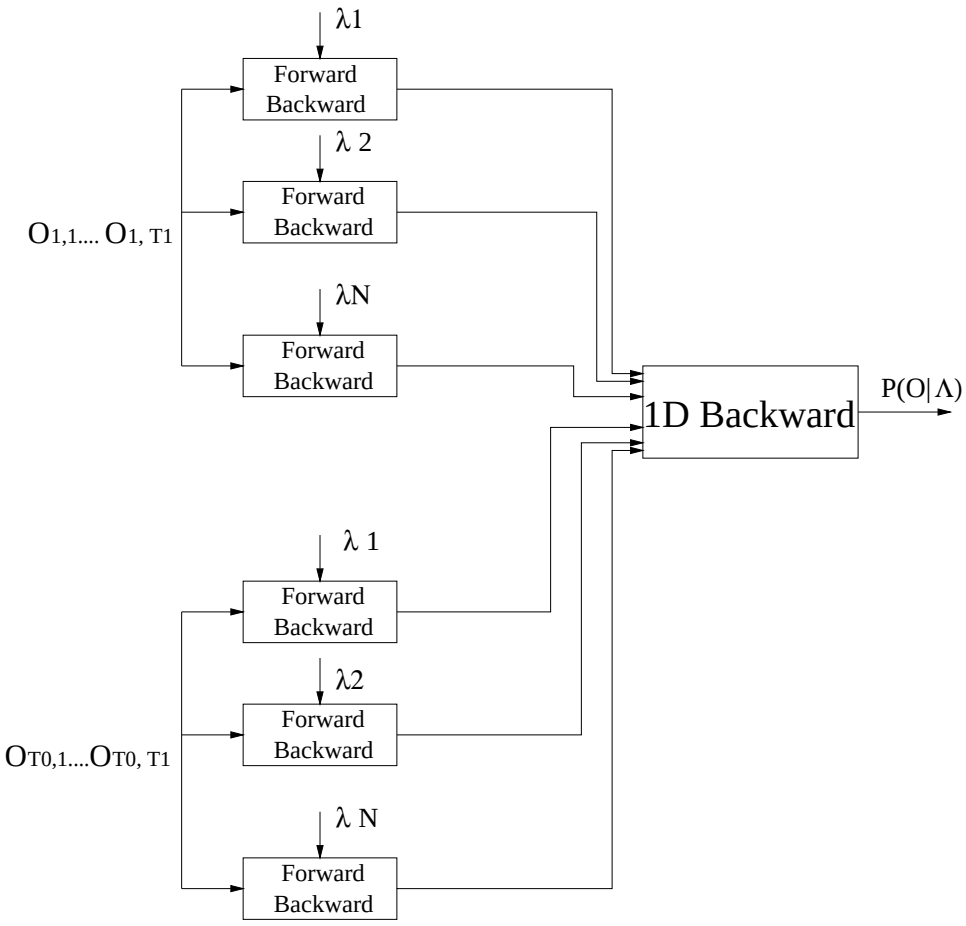



Figure 5.5: The backward algorithm for the embedded HMM

(b) Recursion

$$\beta_{t_0}(i) = \sum_j a_{0,ij} P^i_{t_0} \beta_{t_0+1}(j)$$

It is important to notice that, similarly to the decoding algorithm, the time required for the computation of the evaluation algorithms can be highly reduced if a parallel architecture is used.

## 5.4 The Estimation Algorithm

The estimation of the parameters of the embedded HMM is used to iteratively adjust the parameters of this model with respect to a certain optimization criterion. The algorithm presented in this section is similar to the Baum-Welsh algorithm derived for the one-dimensional HMM. The derivation of the estimation algorithm is described for both the discrete and continuous mixture embedded HMM.

### 5.4.1 Discrete embedded HMM

The objective of the re-estimation algorithm is to estimate the new set of parameters of the embedded HMM that maximize $P(\mathbf{O}|\lambda)$. This is equivalent to maximizing the auxiliary function defined as follows:

$$Q(\lambda, \tilde{\lambda}) = \frac{1}{P(\mathbf{O}|\lambda)} [\sum_{\mathbf{q}} P(\mathbf{O}, \mathbf{q}|\lambda) \log P(\mathbf{O}, \mathbf{q}|\tilde{\lambda}] \tag{5.17}$$

with respect to $\lambda$. It results from the structure of the embedded HMM that the probability of the observation sequence $\mathbf{O}$ and a state sequence $\mathbf{q}$ is given by:

$$P(\mathbf{O}, \mathbf{q}|\lambda) = \pi_{0,q^0_1} \prod_{t_0} a_{0,q^0_{t_0-1},q^0_{t_0}} P(\mathbf{O}_{t_0}|q^0_{t_0}, \lambda) \tag{5.18}$$

where,

$$P(\mathbf{O}_{t_0}|q^0_{t_0}, \lambda) = \pi^{q^0_{t_0,1}}_{1,q^1_{t_0,1}} \prod_{t_1} a^{q^0_{t_0}}_{1,q^1_{t_0,t_1-1},q^1_{t_0,t_1}} b^{q^0_{t_0,t_1}}_{q^1_{t_0,t_1}}(\mathbf{O}_{t_0,t_1}) \tag{5.19}$$

Therefore, by substituting equation 5.19 in 5.18 and computing the logarithm it results that:

$$\begin{aligned}\log P(\mathbf{O},\mathbf{q}|\lambda) &= \log \pi_{0,q_1^0} + \sum_{t_0} \log a_{0,q_{t_0-1}^0,q_{t_0}^0} + \sum_{t_0} \log \pi_{1,q_{t_0}^1}^{q_{t_0,1}^0} \\ &\quad \sum_{t_0}\sum_{t_1} \log a_{1,q_{t_0,t_1-1}^1,q_{t_0,t_1}^1}^{q_{t_0,t_1}^0} + \sum_{t_0}\sum_{t_1} b_{q_{t_0,t_1}^1}^{q_{t_0,t_1}^0}(\mathbf{O}_{t_0,t_1}) \end{aligned} \tag{5.20}$$

By substituting equations 5.18 and 5.20 in equation 5.17, the auxiliary function $Q(\lambda,\tilde{\lambda})$ becomes:

$$\begin{aligned} Q(\lambda,\tilde{\lambda}) &= \sum_{\mathbf{q}} P(\mathbf{O},\mathbf{q}|\tilde{\lambda}) \log \tilde{\pi}_{0,q_1^0} + \sum_{\mathbf{q}} P(\mathbf{O},\mathbf{q}|\tilde{\lambda}) \sum_{t_0} \log \tilde{a}_{0,q_{t_0-1}^0,q_{t_0}^0} \\ &+ \sum_{\mathbf{q}} P(\mathbf{O},\mathbf{q}|\tilde{\lambda}) \sum_{t_0} \log \tilde{\pi}_{1,q_{t_0,t_1}^1}^{q_{t_0,t_1}^0} + \sum_{\mathbf{q}} P(\mathbf{O},\mathbf{q}|\tilde{\lambda}) \sum_{t_0}\sum_{t_1} \log \tilde{a}_{1,q_{t_0,t_1-1}^1,q_{t_0,t_1}^1} \\ &+ \sum_{\mathbf{q}} P(\mathbf{O},\mathbf{q}|\tilde{\lambda}) \sum_{t_0}\sum_{t_1} \log \tilde{b}_{q_{t_0,t_1}^1}^{q_{t_0,t_1}^0}(\mathbf{O}_{t_0,t_1}) \end{aligned} \tag{5.21}$$

The auxiliary function can be rewritten in the form:

$$Q(\lambda,\tilde{\lambda}) = Q_{\pi_0} + \sum_{i=1}^{N_0} Q_{a_0} + \sum_{i=1}^{N_0} Q_{\pi_1} + \sum_{i=1}^{N_0}\sum_{j=1}^{N_1} Q_{a_1} + \sum_{i=1}^{N_0}\sum_{j=1}^{N_1} Q_b \tag{5.22}$$

where,

$$Q_{\pi_0}(\lambda,\tilde{\pi}_{0,i}) = \sum_{i=1}^{N_0} \frac{P(\mathbf{O},q_0^0=i|\lambda)}{P(\mathbf{O}|\lambda)} \log \pi_{0,i} \tag{5.23}$$

$$Q_{a_0}(\lambda,\tilde{a}_{0,ij}) = \sum_{j=1}^{N_0}\sum_{t_0=1}^{T_0} \frac{P(\mathbf{O},q_{t_0-1}^0=i,q_{t_0}^0=j|\lambda)}{P(\mathbf{O}|\lambda)} \log(\tilde{a}_{0,ij}) \tag{5.24}$$

$$Q_{\pi_1}(\lambda,\tilde{\pi}_{1,j}^i) = \sum_{j=1}^{N_1}\sum_{t_0=1}^{T_0} \frac{P(\mathbf{O},q_{t_0,0}^1=j,q_{t_0,0}^0=i|\lambda)}{P(\mathbf{O}|\lambda)} \log(\tilde{\pi}_{1,j}^i) \tag{5.25}$$

$$Q_{a_1}(\lambda,\tilde{a}_{1,jl}^i) = \sum_{l=1}^{N_1}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} \frac{P(\mathbf{O},q_{t_0}^0=i,q_{t_0,t_1-1}^1=j,q_{t_0,t_1}^1=l|\lambda)}{P(\mathbf{O}|\lambda)} \log(\tilde{a}_{1,jl}^i) \tag{5.26}$$

$$Q_b(\lambda,\tilde{b_{ij}}) = \sum_{t_0=1}^{N_0}\sum_{t_1=1}^{T_1} \frac{P(\mathbf{O},q_{t_0,t_1}^1=j,q_{t_0,t_1}^0=i|\lambda')}{P(\mathbf{O}|\lambda)} \log(\tilde{b}_j^i(\mathbf{O}_{t_0,t_1}=v_k)) \tag{5.27}$$

In order to maximize 5.21, each term of the equation has to be maximized individually based on the following stochastic constraints:

$$\sum_{i=1}^{N_0} \pi_{0,i} = 1 \tag{5.28}$$

$$\sum_{j=1}^{N_0} a_{0,ij} = 1 \tag{5.29}$$

$$\sum_{i=1}^{N_1} \pi_{1,j}^{i} = 1 \tag{5.30}$$

$$\sum_{j=1}^{N_1} a_{1,jl}^{i} = 1 \tag{5.31}$$

$$\sum_{p=1}^{P} b_j^i(p) = 1 \tag{5.32}$$

The re-estimated parameters of the embedded HMM are derived using a variant of the EM algorithm to minimize the auxiliary function in Equation 5.17. All the functions in equations 5.23- 5.27 are of the form

$$\sum_{j=1}^{N} w_j \log y_j \tag{5.33}$$

and are subject to the constraints in equations 5.28- 5.32 that are of the form $\sum_{j=1}^{N} y_j = 1$. Using the same technique as in the standard Baum-Welsh algorithm the re-estimation equations for the embedded HMM are given by:

$$\pi_{0,i} = \frac{\frac{P(\mathbf{O}, q_1^0 = i|\lambda)}{P(\mathbf{O}|\lambda)}}{\sum_{i=1}^{N_0} \frac{P(\mathbf{O}, q_1^0 = i|\lambda)}{P(\mathbf{O}|\lambda)}} \tag{5.34}$$

$$a_{0,ij} = \frac{\sum_{t_0=1}^{T_0} \frac{P(\mathbf{O}, q_{t_0-1}^0 = i, q_{t_0}^0 = j|\lambda)}{P(\mathbf{O}|\lambda)}}{\sum_{t_0=1}^{T_0} \frac{P(\mathbf{O}, q_{t_0-1}^0 = i|\lambda)}{P(\mathbf{O}|\lambda)}} \tag{5.35}$$

$$\pi^i_{1,j} = \frac{\sum_{t_0=1}^{T_0} \frac{P(\mathbf{O}, q^1_{t_0,1}=j, q^0_{t_0,1}=i|\lambda)}{P(\mathbf{O}|\lambda)}}{\sum_{t_0=1}^{T_0} \frac{P(\mathbf{O}, q^1_{t_0,1}=i|\lambda)}{P(\mathbf{O}|\lambda)}} \tag{5.36}$$

$$a^i_{j,l} = \frac{\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} \frac{P(\mathbf{O}, q^1_{t_0 t_1-1}=j, q^1_{t_0,t_1}=l, q^0_{t_0}=i|\lambda)}{P(\mathbf{O}|\lambda)}}{\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} \frac{P(\mathbf{O}, q^0_{t_0}=i, q^1_{t_0,t_1-1}=j|\lambda)}{P(\mathbf{O}|\lambda)}} \tag{5.37}$$

$$b^i_j(k) = \frac{\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} \frac{P(\mathbf{O}, \mathbf{q}^1_{t_0,t_1}=j, q^0_{t_0,t_1}=i|\lambda)\delta(\mathbf{O}_{t_0,t_1}, v_k)}{P(\mathbf{O}|\lambda)}}{\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} \frac{P(\mathbf{O}, q^1_{t_0,t_1}=i, q^0_{t_0,t_1}=j|\lambda)}{P(\mathbf{O}|\lambda)}} \tag{5.38}$$

Although the above re-estimation equations are obtained from one observation sequence $\mathbf{O}$, in general the re-estimation of the embedded HMM parameters requires multiple observation sequence, that can be assumed independent. Let $\mathbf{O}^1, \ldots, \mathbf{O}^r, \ldots, \mathbf{O}^R$, be a set of $R$ independent sequences of observations. The re-estimation formulae in terms of the a-posteriori probabilities, for multiple observations sequences is described by the following equations:

$$\pi_{0,i} = \frac{\sum_{r=1}^{R} P(q^0_1 = i|\mathbf{O}^r, \lambda)}{\sum_{r=1}^{R}\sum_{i=1}^{N_0} P(q^0_1 = i|\mathbf{O}^r, \lambda)} \tag{5.39}$$

$$a_{0,ij} = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0} P(q^0_{t_0-1} = i, q^0_{t_0} = j|\mathbf{O}^r, \lambda)}{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0} P(q^0_{t_0-1} = i|\mathbf{O}^r, \lambda)} \tag{5.40}$$

$$\pi^i_{1,j} = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0} P(q^1_{t_0,1} = j, q^0_{t_0} = i|\mathbf{O}^r, \lambda)}{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0} P(q^0_{t_0} = i|\mathbf{O}^r, \lambda)} \tag{5.41}$$

$$a^i_{j,l} = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} P(q^1_{t_0,t_1-1} = j, q^1_{t_0,t_1} = l, q^0_{t_0} = i|\mathbf{O}^r, \lambda)}{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} P(q^0_{t_0} = i, q^1_{t_0,t_1-1} = j|\mathbf{O}^r, \lambda)} \tag{5.42}$$

$$b^i_j(k) = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} P(q^1_{t_0,t_1} = j, q^0_{t_0,t_1} = i|\mathbf{O}^r, \lambda)\delta(\mathbf{O}^r_{t_0,t_1}, v_k)}{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} P(q^0_{t_0,t_1} = i, q^1_{t_0,t_1} = j|\mathbf{O}^r, \lambda)} \tag{5.43}$$

The complexity of the equations will be greatly reduced by the use of the following notations:

$$\gamma'^{\,(r)}_{t_0}(i) = P(q^0_{t_0} = i|\mathbf{O}^r_0, \ldots, \mathbf{O}^r_{t_0}, \lambda) \tag{5.44}$$

$$\gamma'_{t_0}{}^{(r)}(i,j) = P(q^0_{t_0-1} = i, q^0_{t_0} = j | \mathbf{O}^r_0, \ldots, \mathbf{O}^r_{t_0}, \lambda) \tag{5.45}$$

$$\gamma''_{t_0,t_1}{}^{(r)}(i,j) = P(q^1_{t_0,t_1} = j | \mathbf{O}^r_{t_0}, q^0_{t_0} = i, \lambda) \tag{5.46}$$

$$\gamma''_{t_0,t_1}{}^{(r)}(i,j,l) = P(q^1_{t_0,t_1-1} = j, q^1_{t_0,t_1} = l | \mathbf{O}^r_{t_0}, q^0_{t_0} = i, \lambda) \tag{5.47}$$

$$\gamma^{(r)}_{t_0,t_1}(i,j) = P(q^0_{t_0} = i, q^1_{t_0,t_1} = j | \mathbf{O}^r, \lambda) \tag{5.48}$$

$$\gamma^{(r)}_{t_0,t_1}(i,j,l) = P(q^0_{t_0} = i, q^1_{t_0,t_1-1} = j, q^1_{t_0,t_1} = l | \mathbf{O}^r, \lambda) \tag{5.49}$$

Next we will provide an efficient way of computing the above equations using the forward and backward variables. We will denote by $\alpha^r_{t_0}(i), \alpha^r_{t_0,t_1}(i,j), \beta^r_{t_0}(i), \beta^r_{t_0,t_1}(i,j)$ the forward and backward variables, as defined in equations 5.9- 5.15, and computed for the observation sequence $\mathbf{O}^r$. The above a-posteriori probabilities can be calculated from the forward backward variables as follows

$$\gamma'_{t_0}{}^{(r)}(i) = \frac{\alpha^r_{t_0}(i)\beta^r_{t_0}(i)}{\sum_i \alpha^r_{t_0}(i)\beta^r_{t_0}(i)} \tag{5.50}$$

$$\gamma'_{t_0}{}^{(r)}(i,j) = \frac{\alpha^r_{t_0-1}(i)a_{0,ij}P(\mathbf{O}^r_{t_0}|q^0_{t_0} = j, \lambda)\beta^r_{t_0}(j)}{\sum_i \alpha^r_{t_0}(i)\beta^r_{t_0}(i)} \tag{5.51}$$

Equations 5.50 and 5.51 are similar to the equations derived for the corresponding a-posteriori probabilities of the one dimensional HMMs. However, since the above equations refer to the structure of super states, the probability of observations given the state is replaced by the probability of $\mathbf{O}^r_{t_0}$ given the super state and the embedded HMM $P(\mathbf{O}^r_{t_0}|q^0_{t_0} = j, \lambda)$. $P(\mathbf{O}^r_{t_0}|q_{0,t_0} = j, \lambda)$ is computed according to equation 5.11. Since equations 5.52 and 5.53 calculate a-posteriori probabilities defined inside each of the embedded HMM, the calculation of the $\gamma''_{t_0,t_1}{}^{(r)}(i,j)$ and $\gamma''_{t_0,t_1}{}^{(r)}(i,j,l)$ is straightforward:

$$\gamma''_{t_0,t_1}{}^{(r)}(i,j) = \frac{\alpha^r_{t_0,t_1}(i,j)\beta^r_{t_0,t_1}(i,j)}{\sum_i \alpha^r_{t_0,t_1}(i,j)\beta^r_{t_0,t_1}(i,j)} \tag{5.52}$$

$$\gamma''_{t_0,t_1}{}^{(r)}(i,j,l) = \frac{\alpha^r_{t_0,t_1}(i,j)a^i_{1,jl}b^i_l(\mathbf{O}^r_{t_0,t_1})\beta^r_{t_0,t_1}(i,l)}{\sum_i \alpha^r_{t_0,t_1}(i,j)\beta^r_{t_0,t_1}(i,j)} \tag{5.53}$$

Because it is more difficult to express the a-posteriori probabilities $\gamma^{(r)}_{t_0,t_1}(i,j)$ and $\gamma^{(r)}_{t_0,t_1}(i,j,l)$ in terms of the forward and backward variable we will discuss this part in more detail. We will start by decomposing $P(\mathbf{O}, q^0_{t_0,t_1} = i, q^1_{t_0,t_1} = j|\lambda)$ as:

$$\begin{aligned} P(\mathbf{O}, q^0_{t_0,t_1} = i, q^1_{t_0,t_1} = j|\lambda) &= P(\mathbf{O}_0, \ldots, \mathbf{O}_{t_0-1}, q^0_{t_0,t_1} = i|\lambda)P(\mathbf{O}_{t_0}, q^1_{t_0,t_1}|q^0_{t_0} = i, \lambda) \\ & \quad P(\mathbf{O}_{t_0+1}, \ldots, \mathbf{O}_{T_0}|q_{t_0} = i, \lambda) \end{aligned} \tag{5.54}$$

The first term of the right hand side, which represents the probability of the observations ending with $\mathbf{O}_{t_0}$ and super state $i$ can be expressed as:

$$P(\mathbf{O}_1, \ldots, \mathbf{O}_{t_0-1}, q^0_{t_0} = i|\lambda) = \sum_j P(\mathbf{O}_0, \ldots, \mathbf{O}_{t_0-1}|q^0_{t_0-1} = j)a_{0,ij} \tag{5.55}$$

Given the embedded structure described here, the following two equalities hold:

$$\begin{aligned} & P(\mathbf{O}_0, \ldots, \mathbf{O}_{t_0-1}, q^0_{t_0} = i|\lambda_i) = \\ & (\sum_j P(\mathbf{O}_0, \ldots, \mathbf{O}_{t_0-1}|q^0_{t_0-1} = j)a_{0,ij})P(\mathbf{O}_{t_0}|q^0_{t_0} = i, \lambda) \end{aligned} \tag{5.56}$$

$$\begin{aligned} & P(\mathbf{O}_1, \ldots, \mathbf{O}_{T_0}|\lambda_i) = \\ & P(\mathbf{O}_1, \ldots, \mathbf{O}_{t_0-1}, q^0_{t_0} = i|\lambda_i)P(\mathbf{O}_{t_0+1}, \ldots, \mathbf{O}_{T_0}|q^0_{t_0} = i, \lambda_i) \end{aligned} \tag{5.57}$$

In addition, the a-posteriori probability $P(q^{t_0}_{1,t_1} = j, \mathbf{O}_{t_0}|q^0_{t_0} = i, \lambda)$ is calculated from the following equation:

$$P(q^1_{t_0,t_1} = j, \mathbf{O}_{t_0}|q^0_{t_0} = i, \lambda) = \frac{P(\mathbf{O}_{t_0}, q^1_{t_0,t_1} = j|q^0_{t_0} = i, \lambda)}{P(\mathbf{O}_{t_0}, |q^0_{t_0} = i, \lambda)} \tag{5.58}$$

Substituting equations 5.55, 5.56, 5.57 and, 5.58 in equation 5.54, after some simple math work it results:

$$P(\mathbf{O}, q^0_{t_0} = i, q^1_{t_0,t_1} = j|\lambda) = P(\mathbf{O}, q^0_{t_0} = i|\lambda)P(q^{t_0}_{1,t_1} = j|\mathbf{O}_{t_0}, q^0_{t_0} = i, \lambda)$$

After dividing both sides of the equation by $P(\mathbf{O}|\lambda)$

$$P(q^0_{t_0} = i, q^1_{t_0,t_1} = j|\mathbf{O}, \lambda) = P(q^0_{t_0} = i|\mathbf{O}, \lambda)P(q^1_{t_0,t_1} = j|\mathbf{O}_{t_0}, q^0_{t_0} = i, \lambda) \tag{5.59}$$

In a similar way it can be shown that

$$P(q_{t_0}^0 = i, q_{t_0,t_1-1}^1 = j, q_{t_0,t_1}^1 = l|\mathbf{O}, \lambda) =$$
$$P(q_{t_0}^0 = i|\mathbf{O}, \lambda)P(q_{t_0,t_1-1}^1 = j, q_{t_0,t_1}^1 = l|\mathbf{O}_{t_0}, q_{t_0}^0 = i, \lambda) \tag{5.60}$$

Using the notations in 5.48 and 5.49, the equations 5.59 and 5.60 become:

$$\gamma_{t_0,t_1}^{(r)}(i,j) = {\gamma'_{t_0}}^{(r)}(i){\gamma''_{t_0,t_1}}^{(r)}(i,j) \tag{5.61}$$

$$\gamma_{t_0,t_1}^{(r)}(i,j,l) = {\gamma'_{t_0}}^{(r)}(i){\gamma''_{t_0,t_1}}^{(r)}(i,j,l) \tag{5.62}$$

The re-estimated parameters are given by the following equations:

$$\pi_{0,i} = \frac{\sum_r {\gamma'}_0^{(r)}(i)}{\sum_r \sum_i {\gamma'}_0^{(r)}(i)} \tag{5.63}$$

$$a_{0,ij} = \frac{\sum_r \sum_{t_0} {\gamma'_{t_0}}^{(r)}(i,j)}{\sum_r \sum_{t_0} {\gamma'_{t_0}}^{(r)}(i)} \tag{5.64}$$

$$\pi_{1,j}^i = \frac{\sum_r \sum_{t_0} \gamma_{t_0,0}^{(r)}(i,j)}{\sum_r \sum_{t_0} {\gamma'_{t_0}}^{(r)}(i)} \tag{5.65}$$

$$a_{1,jl}^i = \frac{\sum_r \gamma_{t_0,t_1}^{(r)}(i,j,l)}{\sum_r \sum_{t_0} \sum_{t_1} \gamma_{t_0,t_1}^{(r)}(i,j)} \tag{5.66}$$

$$b_j^i(p) = \frac{\sum_r \sum_{(t_0,t_1)\in \mathbf{O}_{t_0,t_1}=v_p} \gamma_{t_0,t_1}^{(r)}(i,j)}{\sum_r \sum_{t_0} \sum_{t_1} \gamma_{t_0,t_1}^{(r)}(i,j)} \tag{5.67}$$

### 5.4.2 Continuous embedded HMM

In this section we will present the algorithm for the re-estimation of the parameters for the continuous embedded HMM. The algorithm follows the same general steps as the algorithm for discrete embedded HMM described in the previous section. However before we prove the re-estimation equations, some notations and discussions are

needed. Equation 5.2, which defines the probability matrix $\mathbf{B}$ of the continuous mixture embedded HMM, can be written as:

$$b_j^i(O_{t_0,t_1}) = \sum c_{jm}^i b_{jm}^i(\mathbf{O}_{t_0,t_1}) \tag{5.68}$$

where $b_{jm}^i(\mathbf{O}_{t_0,t_1})$ is the Gaussian density function of the $m$th mixture in state $j$ of super states $i$. Let $k_{t_0,t_1}$ represent a mixture component of the embedded state, and let $\mathbf{k}_{t_0} = (k_{t_0,0}, k_{t_0,1}, \ldots, k_{t_0,T_1})$ and $\mathbf{k} = (\mathbf{k}_0, \mathbf{k}_1, \ldots, \mathbf{k}_{T_0})$. The elements of $\mathbf{k}_{t_0}$ represent a sequence of mixtures corresponding to the embedded states $\mathbf{q}_{t_0}^1$, while the elements of $\mathbf{k}$ are all mixture sequences $\mathbf{k}_{t_0}$. It is clear now that with the continuous mixture embedded HMM, the sequence of states is more refined than in the discrete case and includes both the state sequence and the array of mixtures $\mathbf{k}$ corresponding to theses states. Therefore in a similar way to equation 5.17, the auxiliary function $Q$ for the continuous mixture embedded HMM is defined as

$$Q(\lambda, \tilde{\lambda}) = \frac{1}{P(\mathbf{O}|\lambda)}[\sum_{\mathbf{q}} \sum_{\mathbf{k}} P(\mathbf{O}, \mathbf{q}, \mathbf{k}|\lambda) \log P(\mathbf{O}, \mathbf{q}, \mathbf{k}|\tilde{\lambda})] \tag{5.69}$$

Let's recall that the goal of there-estimation equation is to maximize the auxiliary function $Q$ with respect to $\lambda$. The probability of observation $\mathbf{O}$, a single state path $\mathbf{q}$, and a sequence of mixtures $\mathbf{k}$ corresponding to $\mathbf{q}$ is given by

$$P(\mathbf{O}, \mathbf{q}, \mathbf{k}|\lambda) = \pi_{0,q_0^0} \prod_{t_0} a_{0,q_{t_0-1}^0,q_{t_0}^0} P(\mathbf{O}_{t_0}, \mathbf{q}_{t_0}, \mathbf{k}_{t_0}|\lambda) \tag{5.70}$$

where $P(\mathbf{O}, \mathbf{q}_{t_0}, \mathbf{k}_{t_0}|\lambda)$ can be decomposed as:

$$P(\mathbf{O}_{t_0}, \mathbf{q}_{t_0}, \mathbf{k}_{t_0}|\lambda) = \pi_{1,q_{t_0,1}^1}^{q_{t_0,1}^0} \prod_{t_1} a_{1,q_{t_0,t_1-1}^1 q_{t_0,t_1}^1}^{q_{t_0}^0} b_{q_{t_0,t_1}^1,k_{t_0,t_1}}^{q_{t_0,t_1}^0}(\mathbf{O}_{t_0,t_1}) c_{q_{t_0,t_1}^1,k_{t_0,t_1}}^{q_{t_0,t_1}^0} \tag{5.71}$$

By substituting equation 5.71 in 5.70 and and computing the logarithm it results:

$$\begin{aligned} \log P(\mathbf{O}, \mathbf{q}, \mathbf{k}|\lambda) \;=\; & \log \pi_{0,q_{t_0}^0} + \sum_{t_0} \log a_{0,q_{t_0-1}^0,q_{t_0}^0} + \sum_{t_0} \log \pi_{1,q_{t_0,1}^1}^{q_{t_0,1}^0} + \\ & \sum_{t_0} \sum_{t_1} \log a_{1,q_{t_0,t_1-1}^1,q_{t_0,t_1}^1}^{q_{t_0}^0} + \sum_{t_0} \sum_{t_1} \log b_{q_{t_0,t_1}^1}^{q_{t_0,t_1}^0}(\mathbf{O}_{t_0,t_1}) + \\ & \sum_{t_0} \sum_{t_1} \log c_{q_{t_0,t_1}^1,k_{t_0,t_1}}^{q_{t_0,t_1}^0} \end{aligned} \tag{5.72}$$

Using equations 5.70, 5.71 and 5.72 in 5.69, the auxiliary function $Q(\lambda, \lambda')$ can be decomposed in the form of the following sum:

$$\begin{aligned} Q(\lambda,\lambda') &= \frac{1}{P(\mathbf{O}|\lambda)}[\sum_{\mathbf{q}}\sum_{\mathbf{k}} P(\mathbf{O},\mathbf{q},\mathbf{k}|\lambda)\log\pi_{0,q^0_{t_0}} + \sum_{\mathbf{q}}\sum_{\mathbf{k}} P(\mathbf{O},\mathbf{q},\mathbf{k}|\lambda)\sum_{t_0}\log\tilde{a}_{0,q^0_{t_0-1},q^0_{t_0}} \\ &+ \sum_{\mathbf{q}}\sum_{\mathbf{k}} P(\mathbf{O},\mathbf{q},\mathbf{k}|\lambda)\sum_{t_0}\log\tilde{\pi}^{q^0_{t_0,t_1}}_{1,q^1_{t_0,t_1}} \\ &+ \sum_{\mathbf{q}}\sum_{\mathbf{k}} P(\mathbf{O},\mathbf{q},\mathbf{k}|\lambda)\sum_{t_0}\sum_{t_1}\log\tilde{a}^{q^0_{t_0}}_{1,q^1_{t_0,t_1-1},q^1_{t_0,t_1}} \\ &+ \sum_{\mathbf{q}}\sum_{\mathbf{k}} P(\mathbf{O},\mathbf{q},\mathbf{k}|\lambda)\sum_{t_0}\sum_{t_1}\log\tilde{b}^{q^0_{t_0,t_1}}_{q^1_{t_0,t_1}k_{t_0,t_1}}(\mathbf{O}_{t_0,t_1}) \\ &+ \sum_{\mathbf{q}}\sum_{\mathbf{k}} P(\mathbf{O},\mathbf{q},\mathbf{k}|\lambda)\sum_{t_0}\sum_{t_1}\log\tilde{c}^{q^0_{t_0}}_{q^1_{t_0,t_1}k_{t_0,t_1}}] \end{aligned} \tag{5.73}$$

and it can be rewritten in a more compact way:

$$Q(\lambda,\tilde{\lambda}) = Q_{\pi_0} + \sum_{i=1}^{N_0} Q_{a_0} + \sum_{i=1}^{N_0} Q_{\pi_1} + \sum_{i=1}^{N_0}\sum_{j=1}^{N_1} Q_{a_1} + \sum_{i=1}^{N_0}\sum_{j=1}^{N_1}\sum_{k} Q_b + \sum_{i=1}^{N_0}\sum_{j=1}^{N_1} Q_c \tag{5.74}$$

where,

$$Q_{\pi_0}(\lambda,\tilde{pi}_{0,i}) = \sum_{i=1}^{N_0}\frac{P(\mathbf{O},\mathbf{k},q^0_0=i)}{P(\mathbf{O}|\lambda)}\log\tilde{\pi}_0 \tag{5.75}$$

$$Q_{a_0}(\lambda,\tilde{a}_{0,ij}) = \sum_{j=1}^{N_0}\sum_{t_0=1}^{T_0}\frac{P(\mathbf{O},\mathbf{k},q^0_{t_0}=i,q^0_{t_0-1}=j|\lambda)}{P(\mathbf{O}|\lambda)}\log(\tilde{a}_{0,ij}) \tag{5.76}$$

$$Q_{\pi_1}(\lambda,\tilde{\pi}^i_{1,j}) = \sum_{j=1}^{N_1}\sum_{t_0=1}^{T_0}\frac{P(\mathbf{O},\mathbf{k},q^0_{t_0,0}=i,q^1_{t_0,0}=j|\lambda)}{P(\mathbf{O}|\lambda)}\log(\tilde{\pi}^q_{0,i0},t_0) \tag{5.77}$$

$$Q_{a_1}(\lambda,\tilde{a}^i_{1,jl}) = \sum_{l=1}^{N_1}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1}\frac{P(\mathbf{O},\mathbf{k},q^0_{t_0}=i,q^1_{t_0,t_1-1}=j,q^1_{t_0,t_1}=l|\lambda)}{P(\mathbf{O}|\lambda)}\log(\tilde{a}^i_{1,jl}) \tag{5.78}$$

$$Q_b(\lambda,\tilde{b}^i_{jm}) = \sum_{t_0=1}^{N_0}\sum_{t_1=1}^{T_1}\frac{P(\mathbf{O},\mathbf{k},q^0_{t_0,t_1}=i,q^1_{t_0,t_1}=jk_{t_0,t_1}=m|\lambda)}{P(\mathbf{O}|\lambda)}\log(\tilde{b}^i_{jm}(\mathbf{O}_{t_0,t_1})) \tag{5.79}$$

$$Q_c(\lambda,\tilde{c}^i_{jm}) = \sum_{t_0=1}^{N_0}\sum_{t_1=1}^{T_1}\frac{P(\mathbf{O},\mathbf{k},q^0_{t_0}=i,q^1_{t_0,t_1}=j,k_{t_0,t_1}=m|\lambda)}{P(\mathbf{O}|\lambda)}\log(\tilde{c}^i_{jm}) \tag{5.80}$$

Due to the decomposition of the auxiliary function, a global maximum of $Q$ is obtained by the maximization of the individual auxiliary functions. The maximization of the individual auxiliary functions $Q_{\pi_0}, Q_{a_0}, Q_{\pi_1}$ and $Q^i_{a_1}$ subject to the stochastic constraints in equations 5.28- 5.31 is obtained in a similar manner as discussed in the previous section. The re-estimated parameters obtained from multiple independent observation sequences and expressed in terms of the a-posteriori probabilities are given by the following equations:

$$\pi_{0,i} = \frac{\sum_{r=1}^{R} \sum_{\mathbf{k}} P(q_1^0 = i, \mathbf{k} | \mathbf{O}^r, \lambda)}{\sum_{r=1}^{R} \sum_{i=1}^{N_0} \sum_{\mathbf{k}} P(q_1^0, \mathbf{k} | \mathbf{O}^r, \lambda)} = \frac{\sum_{r=1}^{R} P(q_1^0 = i | \mathbf{O}^r, \lambda)}{\sum_{r=1}^{R} \sum_{i=1}^{N_0} P(q_1^0 | \mathbf{O}^r, \lambda)} \tag{5.81}$$

$$\begin{aligned} a_{0,ij} &= \frac{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} \sum_{\mathbf{k}} P(q_{t_0-1}^0 = i, q_{t_0}^0 = j, \mathbf{k} | \mathbf{O}^r, \lambda)}{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} \sum_{\mathbf{k}} P(q_{t_0-1}^0 = i, \mathbf{k} | \mathbf{O}^r, \lambda)} = \\ & \frac{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} P(q_{t_0-1}^0 = i, q_{t_0}^0 = j | \mathbf{O}^r, \lambda)}{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} P(q_{t_0-1}^0 = i | \mathbf{O}^r, \lambda)} \end{aligned} \tag{5.82}$$

$$\begin{aligned} \pi^i_{1,j} &= \frac{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} \sum_{\mathbf{k}} P(q_{t_0,1}^1 = j, q_{t_0,1}^0 = i, \mathbf{k}_{t_0} | \mathbf{O}^r, \lambda)}{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} \sum_{\mathbf{k}\, t_0} P(q_{t_0,1}^0 = i, \mathbf{k}_{t_0} | \mathbf{O}^r, \lambda)} = \\ & \frac{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} P(q_{t_0,1}^1 = j, q_{t_0,1}^0 = i | \mathbf{O}^r, \lambda)}{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} P(q_{t_0,1}^0 = i | \mathbf{O}^r, \lambda)} \end{aligned} \tag{5.83}$$

$$\begin{aligned} a^i_{j,l} &= \frac{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} \sum_{t_1=1}^{T_1} \sum_{\mathbf{k}\, t_0} P(q_{t_0,t_1-1}^1 = j, q_{t_0,t_1}^1 = l, q_{t_0}^0 = i, \mathbf{k}_{t_0} | \mathbf{O}^r, \lambda)}{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} \sum_{t_1=1}^{T_1} \sum_{\mathbf{k}\, t_0} P(q_{t_0}^0 = i, q_{t_0,t_1-1}^1 = j, \mathbf{k}_{t_0} | \mathbf{O}^r, \lambda)} = \\ & \frac{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} \sum_{t_1=1}^{T_1} P(q_{t_0,t_1-1}^1 = j, q_{t_0,t_1}^1 = l, q_{t_0}^0 = i | \mathbf{O}^r, \lambda)}{\sum_{r=1}^{R} \sum_{t_0=1}^{T_0} \sum_{t_1=1}^{T_1} P(q_{t_0}^0 = i, q_{t_0,t_1-1}^1 = j | \mathbf{O}^r, \lambda)} \end{aligned} \tag{5.84}$$

Furthermore it can be seen from equations 5.81- 5.84 that the re-estimation equations for $\pi_0, a_{0,i}, \pi^i_{1,j}$ and $a^i_{1,jl}$ have the same form as in the discrete case and therefore they will be computed accordingly to equations 5.63- 5.66. The only difference from the computation of the parameters corresponding to a discrete HMM is that in all computation the discrete probability function 5.1 is replaced by the continuous mixture density 5.2. From the form of the individual auxiliary function $Q_c$ and the constraint:

$$\sum_{m=1}^{M} c^i_{jm} = 1 \tag{5.85}$$

it results, that the estimated value of the $m^{th}$ mixture weight in the embedded states $j$ of super states $i$ is given by:

$$c^i_{jm} = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} P(q^1_{t_0,t_1}=j, q^0_{t_0,t_1}=i, k_{t_0,t_1}=m|\mathbf{O}^r,\lambda)}{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} P(q^0_{t_0,t_1}=i, q^1_{t_0,t_1}=j|\mathbf{O}^r,\lambda)} \tag{5.86}$$

The maximization of $Q_b$ with respect to $b^i_{jm}$ is obtained through differentiation, and by using the partial derivatives of $b^i_{jm}$ with respect to $\mu^i_{jm}$ and $(\mathbf{U}^i_{jm})^{-1}$ . The details of this derivation are not given here since this maximization procedure is very similar to the one used for one dimensional HMM [99]. It can be shown that the values $\mu^i_{jm}$ and $\mathbf{U}^i_{jm}$ that maximize $Q_{b^i_{jm}}$, and consequently represent the re-estimation equation of $\mu^i_{jm}$ and $\mathbf{U}^i_{jm}$, are given by:

$$\mu^i_{jm} = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} P(q^1_{t_0,t_1}=j, q^0_{t_0,t_1}=i, k_{t_0,t_1}=m|\mathbf{O}^r,\lambda)\mathbf{O}^r_{t_0,t_1}}{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1} P(q^0_{t_0,t_1}=i, q^1_{t_0,t_1}=j, k_{t_0,t_1}=m|\mathbf{O}^r,\lambda)} \tag{5.87}$$

$$\mathbf{U}^i_{jm} = \frac{\sum_r\sum_{t_0}\sum_{t_1} P(q^1_{t_0,t_1}=j, q^0_{t_0,t_1}=i, k_{t_0,t_1}=m|\mathbf{O}^r,\lambda)(\mathbf{O}^r_{t_0,t_1}-\mu^j_i)(\mathbf{O}^r_{t_0,t_1}-\mu^i_{jm})^T}{\sum_r\sum_{t_0}\sum_{t_1} P(q^0_{t_0,t_1}=i, q^1_{t_0,t_1}=j, k_{t_0,t_1}=m|\mathbf{O}^r,\lambda)} \tag{5.88}$$

To give a more compact representation of equation 5.86 and 5.88 it is useful to make the following notations:

$$\zeta''^{(r)}_{t_0,t_1}(i,j,m) = P(q^1_{t_0,t_1}=j, k_{t_0,t_1}=m|\mathbf{O}^r_{t_0}, q^0_{t_0}=i,\lambda) \tag{5.89}$$

$$\zeta^{(r)}_{t_0,t_1}(i,j,m) = P(q_{t_0,t_1}=i, q^1_{t_0,t_1}=j, k_{t_0,t_1}=m|\mathbf{O}^r,\lambda) \tag{5.90}$$

$\zeta''^{(r)}_{t_0,t_1}(j,m)$ is the a-posteriori probability of state $j$ and mixture $m$ given the observation sequence $\mathbf{O}_{t_0}$, the super state $i$ and model $\lambda$. It is important to notice that $\zeta''^{(r)}_{t_0,t_1}(i,j,m)$ is practically independent of model $\lambda$. The only reason that it appears in equation 5.89 is to allow for the definition of $\mathbf{\Lambda}^i$ as one of its super states. Unlike $\zeta''^{(r)}_{t_0,t_1}(i,j,m)$, $\zeta^{(r)}_{t_0,t_1}(i,j,m)$ depends on the context of the structure of super states. It denotes the probability of super state $i$, embedded state $j$ and mixture $m$, given

the entire sequence $\mathbf{O}$ and model $\lambda$. Since ${\zeta''_{t_0,t_1}}^{(r)}(i,j,m)$ is practically independent of model $\lambda$, it can be computed as in a one-dimensional HMM. Hence,

$${\zeta''_{t_0,t_1}}^{(r)}(i,j,m) = \frac{\alpha^r_{t_0,t_1}(i,j)\beta^r_{t_0,t_1}(i,j)}{\sum_i \alpha^r_{t_0,t_1}(i,j)\beta^r_{t_0,t_1}(i,j)} \frac{c^i_{jm} N(\mathbf{O}_{t_0,t_1}, \mu^i_{jm}, \mathbf{U}^i_{jm})}{\sum_m c^i_{jm} N(\mathbf{O}_{t_0,t_1}, \mu^i_{jm}, \mathbf{U}^i_{jm})} \tag{5.91}$$

In a similar way to equation 5.59, $\zeta^{(r)}_{t_0,t_1}(i,j,m)$, can be related to ${\zeta''_{t_0,t_1}}^{(r)}(j,m)$:

$$\zeta^{(r)}_{t_0,t_1}(i,j,m) = {\gamma'_{t_0}}^{(r)}(i){\zeta''_{t_0,t_1}}^{(r)}(i,j,m) \tag{5.92}$$

In the above equation, ${\gamma'_{t_0}}^{(r)}(i)$ is related to the structure of super states of our model while ${\zeta''_{t_0,t_1}}^{(r)}(j,m)$ depends only on the HMM inside each super state. From this point of view, equation 5.92 gives another relation in terms of the a-posteriori probability that confirms our intuition of the separability of the embedded HMM structure. Substituting equation 5.89 and 5.90 in 5.86- 5.88 the re-estimation equations for $c^i_{jm}$, $\mu^i_{jm}$ and $\Sigma^i_{jm}$ become:

$$c^i_{jm} = \frac{\sum_r \sum_{t_0} \sum_{t_1} \zeta^{(r)}_{t_0,t_1}(i,j,m)}{\sum_r \sum_{t_0} \sum_{t_1} \gamma^{(r)}_{t_0,t_1}(i,j)} \tag{5.93}$$

$$\mu^i_{jm} = \frac{\sum_r \sum_{t_0} \sum_{t_1} \zeta^{(r)}_{t_0,t_1}(i,j,m)\mathbf{O}^r_{t_0,t_1}}{\sum_r \sum_{t_0} \sum_{t_1} \zeta^{(r)}_{t_0,t_1}(i,j,m)} \tag{5.94}$$

$$\mathbf{U}^i_{jm} = \frac{\sum_r \sum_{t_0} \sum_{t_1} \zeta^{(r)}_{t_0,t_1}(i,j,m)(\mathbf{O}^r_{t_0,t_1} - \mu_{i,j})(\mathbf{O}^r_{t_0,t_1} - \mu_{i,j})^T}{\sum_r \sum_{t_0} \sum_{t_1} \zeta^{(r)}_{t_0,t_1}(i,j,m)} \tag{5.95}$$

The interpretation of the re-estimation equations is straightforward. The estimated value $c^i_{jm}$ can be seen as the ratio between the expected number of occurrences of super state $i$, embedded state $j$ using $m$th mixture component and the expected number of times the system is in super state $i$ and embedded state $j$. A similar interpretation can be given for all re-estimation equations for the discrete 5.63- 5.67 and continuous embedded HMM 5.93- 5.95. From the definitions in 5.90 and 5.45, it can be seen that in the case of simple mixture, the term $\zeta^{(r)}_{t_0,t_1}(i,j,m)$ reduces to

${\gamma'_{t_0,t_1}}^{(r)}(i,j)$ Therefore the re-estimation equations for the parameters of the uni modal Gaussian density $\mu^i_j$ and $\mathbf{U}^i_j$ become:

$$\mu^i_j = \frac{\sum_r \sum_{t_0} \sum_{t_1} \gamma^{(r)}_{t_0,t_1}(i,j)\mathbf{O}^r_{t_0,t_1}}{\sum_r \sum_{t_0} \sum_{t_1} \gamma^{(r)}_{t_0,t_1}(i,j)} \tag{5.96}$$

$$\mathbf{U}^i_j = \frac{\sum_r \sum_{t_0} \sum_{t_1} \gamma^{(r)}_{t_0,t_1}(i,j)(\mathbf{O}^r_{t_0,t_1} - \mu_{i,j})(\mathbf{O}^r_{t_0,t_1} - \mu_{i,j})^T}{\sum_r \sum_{t_0} \sum_{t_1} \gamma^{(r)}_{t_0,t_1}(i,j)} \tag{5.97}$$

## 5.5 Implementation Issues

In this section we will discuss some of the implementation issues of the algorithms described in the previous sections for the embedded HMM structure. As expected, because of the similarity of the algorithms of the embedded structure and one dimensional HMMs, some of the implementation issues are common to both models. Hence we will discuss in more detail here the issues that are specific to the embedded HMM and shortly refer to the problems addressed by the one-dimensional HMM such as scaling, log representation and thresholding. While the computation of the forward backward variables $\alpha$ and $\beta$ for the HMM corresponding to each super state follows the scaling procedures described in [62], the calculation of 5.14, requires a closer look. The probability of $\mathbf{O}_{t_0}$ given the super state $j$ and model $\lambda$, $P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$ in equation 5.14 is in general very small and often leads to underflow problems. This value is retrieved from the scaled forward algorithm [62] applied to the HMM corresponding to super state $j$ in the logarithmic form:

$$\log P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda) = \sum_{t_1=0}^{T_1} \log s_{t_0,t_1} \tag{5.98}$$

where $s_{t_0,t_1}$ are the scaling coefficients corresponding to the observation $\mathbf{O}_{t_0,t_1}$. Thus although $\log P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$ can be computed, $P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$ is out of the dynamic range of the machine. This underflow will propagate to the computation of the a-posteriori probabilities as denoted by equation 5.44- 5.49, 5.89 and 5.90 and will further make

the implementation of all the algorithms described in this chapter impossible. Next we will describe a simple way to avoid this problem. Let $F$ be a constant such that:

$$\max_{t_0,j} \log P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda) + \log F = 0. \tag{5.99}$$

The addition of the value $\log F$ to all $P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$ leads to:

$$\log \tilde{P}(\mathbf{O}_{t_0}|q^0_{t_0},\lambda) = \log P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda) + \log F. \tag{5.100}$$

which in fact represents the multiplication of each $P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$ by a constant such that

$$\max_{t_0,j} \tilde{P}(\mathbf{O}_{t_0}|q^0_{t_0},\lambda) = 1. \tag{5.101}$$

Values of $\log \tilde{P}(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$ that fall under a minimum value $V_{min}$ are set to $\log \tilde{P}(\mathbf{O}_{t_0}|q^0_{t_0},\lambda) = V_{min}$. $V_{min}$ is chosen such that $\tilde{P}(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$ for all $t_0$ and $j$ remains in the dynamic range of the machine. By replacing $P(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$ by the scaled values $\tilde{P}(\mathbf{O}_{t_0}|q^0_{t_0},\lambda)$, the values of the a-posteriori probabilities in 5.44 - 5.48 will be multiplied by the same factor $F$. Therefore by substituting the new a-posteriori probabilities in the re-estimation equations 5.63- 5.67 or 5.93- 5.95 the factor $F$ in the numerator and denominator will cancel and the values of the re-estimated parameters will remain unchanged. Finally it is important to notice that the scaled overall log probability $\log \tilde{P}(\mathbf{O}|\lambda)$, obtained by using the scaled probability $\tilde{P}(\mathbf{O}|q^0_{t_0},\lambda)$, becomes:

$$\log \tilde{P}(\mathbf{O}|\lambda) = \log P(\mathbf{O}|\lambda) + \log F \tag{5.102}$$

Another way of avoiding the problems related to the limitations of the dynamic range of the machine is the use of the logarithmic form for all probabilities used in the algorithms described in this chapter [99]. However, the complexity of this method is higher than the method proposed in this section.

## 5.6 Computational Complexity

In this section we will discuss the computational complexity of the decoding and evaluation algorithms for our embedded HMM structure described in this chapter .

Because the computation of the above algorithms, as it have been shown earlier in this chapter, uses the basic algorithms used for the one dimensional HMM, it is useful to consider first the complexity of these algorithms. The number of additions required by the logarithmic implementation of the Viterbi decoder [62] is $N^2T$, where $N$ is the number of states and $T$ is the number of observations. Similarly the number of calculations (additions and multiplications) for the forward and backward algorithm is of the order of $N^2T$, $O(N^2T)$. As described earlier in this chapter, the decoding and the evaluation algorithm consist in two stages. First the classical algorithms the decoding (Viterbi) and evaluation (forward backward) algorithms for one dimensional HMM are computed for all $\mathbf{O}_{t_0}$ and super states $j$ to obtain $P(\mathbf{O}_{t_0}|q_{t_0}^0 = j, \lambda)$ and respectively $\max_{q_{t_0}^1} P(\mathbf{O}_{t_0}, q_{t_0}^1|q_{t_0}^0 = j, \lambda)$. Therefore the total number of calculations required by the first stage of the evaluation and decoding algorithms for the embedded HMM is of the order $O(\sum_{k=1}^{N_0} \left(N_1^{(k)}\right)^2 T_1 T_0)$. In the second stage, the same algorithms derived for the one dimensional HMM are applied, but this time for the overall HMM defined by the structure of super states. Hence the number of calculations required in this stage is on the order of $O(N_0^2 T_0)$. Adding the computation complexity for both stages of the algorithms it results that the overall complexity, $NumCalc$, for each of these algorithms is:

$$NumCalc = (\sum_{k=1}^{N_0} \left(N_1^{(k)}\right)^2 T_1) T_0 + {N_0}^2 T_0 \tag{5.103}$$

For the doubly embedded Viterbi algorithm all, if the logarithmic representation is used, all calculations are reduced to additions. In the case of the forward-backward algorithms the above $NumCalc$ refers to the number of additions and multiplications since the logarithmic implementation is not possible in this case. Equation 5.103 shows that although the complexity of the basic algorithms for the embedded HMM is higher than for the one dimensional HMM, it is lower than the for the fully connected two dimensional HMM. This fact allows for considering this model for modeling two dimensional data such as images. A more detailed description of the use of this model for face analysis, recognition and detection will be given in the next chapter.

# CHAPTER 6

# An Embedded Hidden Markov Model for Face Detection and Recognition

Although one-dimensional HMMs have been used with moderate success for face recognition and detection, a one-dimensional HMM is not well-matched to the two-dimensional structure of images. Therefore, in this chapter we consider using an embedded HMM, as described in the previous chapter, for face modeling, detection and recognition. The notation regarding the parameters of the embedded HMM used in this chapter are consistent with the notation used in Chapter 5.

## 6.1 The Embedded HMM for Face

An example of an embedded HMM for faces is shown in Figure 6.1 where each of the five states in the one-dimensional HMM in Figure 4.1 becomes a "superstate" that contains an embedded HMM. The embedded HMM for face follows the same structure illustrated in Figure 5.1, where all transitions backwards are removed. The superstates model the image along the vertical direction, while the embedded HMMs model the data along the horizontal direction.

Before we can use an embedded HMM for face recognition [100] and detection [101] it is necessary to select the number of superstates in the top-to-bottom HMM, the number of states in each embedded HMM, the state transition probabilities, and the observations that are produced by the HMM. The structure that we have used

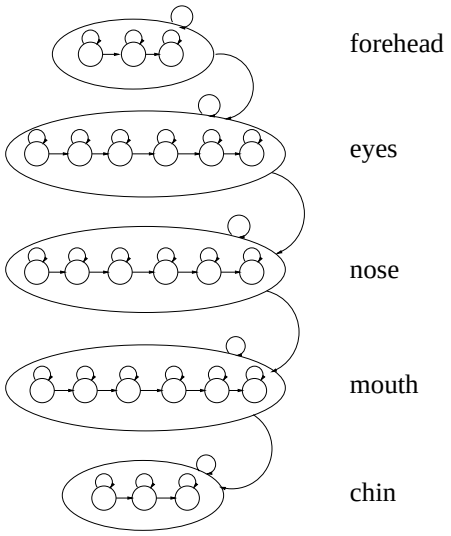



Figure 6.1: An embedded HMM for face detection and recognition

is shown in Figure 6.1. Specifically, we have $N_0 = 5$ superstates that are used to represent the five primary facial regions as an image is scanned from top to bottom, i.e., forehead, eyes, nose, mouth, and chin. The first and last superstates have three embedded states, whereas the middle three superstates have five. The number of embedded states was selected based on the number of regions we expect to find in a face as it is scanned from left to right within a given superstate. For example, in the second superstate (eyes) we have one state for each eye, one state for each of the temple regions (the regions between the eyes and the edges of the face) and two states for the region between eyes. We also considered using six superstates, using the sixth state for hair above the forehead, but this extra state did not significantly improve the recognition results, and increased the computational complexity of the model. In our experiments, each state was modeled by a mixture of three Gaussian densities and the covariance matrix was chosen to be diagonal.

## 6.2 The Observation Vectors

The observation sequence for a face image is formed from image blocks of size $L_x \times L_y$ that are extracted by scanning the image from left-to-right and top-to-bottom as illustrated in Figure 6.2. Adjacent image blocks overlap by $P_y$ rows in the vertical direction, and $P_x$ columns in the horizontal direction. Specifically in our experiments

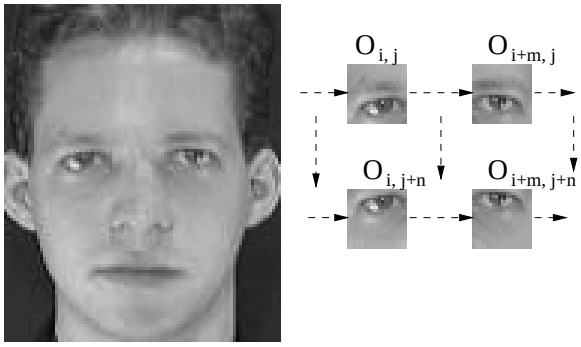



Figure 6.2: Face image parameterization and blocks extraction

the following values were used for parameters $L_x, L_y, P_x, P_y$: $L_x = 8, L_y = 10, P_x = 6$ and $P_y = 8$. The observation vectors are formed from either six two-dimensional DCT coefficients (a $3 \times 2$ low-frequency array) or four KLT coefficients corresponding to the largest eigenvalues. The technique used to extract the KLT coefficients is analogous to that described in Chapter 4. In the hidden Markov models developed by Samaria [71], the observation vectors are formed from the pixel values within each block, so the dimension of the observation vectors were $L_x \times L_y = 80$. Therefore, retaining only six low-frequency DCT coefficients or four KLT coefficients reduces the dimension of the observation vectors by a factor of about thirteen to twenty. As discussed in Chapter 4, in addition to reducing the dimensionality of the observation vectors, using DCT coefficients also tends to reduce the sensitivity of the HMM to noise, image rotations or shifts, and changes in image illumination. Changes in illumination, which is a challenging problem in any face recognition system, can be considered to be negligible over small blocks relative to the image size such as those used in our

approach (of $8 \times 10$ pixels). Therefore variations in illumination will affect primarily only the DC component of the 2D-DCT and leave the rest of coefficients unchanged. If pixel intensities are used as observation vectors, the variations in illumination affect all elements of these vectors, and consequently, decrease the robustness of the system. Finally, using DCT coefficients for the observations allows one to perform face detection and face recognition on images that are in compressed form, e.g., JPEG.

## 6.3 Training the Embedded HMM

The training of the embedded HMM for face detection and recognition uses the maximum likelihood criterion. The training follows the same steps as those used for the training of the standard HMM explained in Chapter 4, except that the Viterbi segmentation, the forward-backward procedure, and the Baum-Welch re-estimation algorithm are replaced by the corresponding decoding (doubly embedded Viterbi), evaluation and estimation algorithms described in Chapter 5. The same training procedure, which will be explained later in this section, is used for both face detection and recognition. The only difference between training for face recognition and training for face detection is in the images that are used in the training set. For face detection, the training images represent faces of different people taken under different illumination conditions and small deviations from a frontal view. The images also consist of different facial expressions, hair styles and represent both males and females from different ages and races. All of the face images in the training set are used to train a single face model. For face recognition, on the other hand, each individual in the database is represented by an HMM face model. A set of images representing different instances of the same person, showing different expressions, hair styles, or eye wear, are used to train each HMM.

To train an embedded HMM we proceed as follows. First, the observation vectors are extracted for each individual in the training set. Let us assume that we have $T_1$

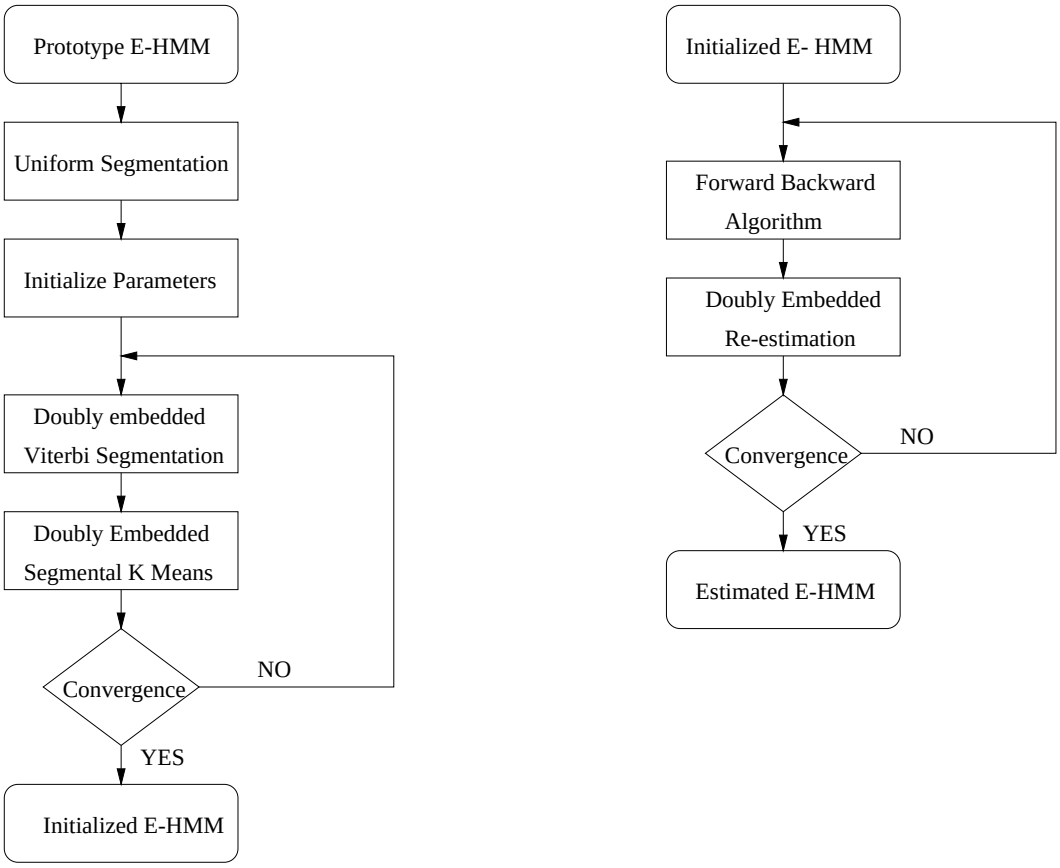



Figure 6.3: Training scheme for the embedded HMM

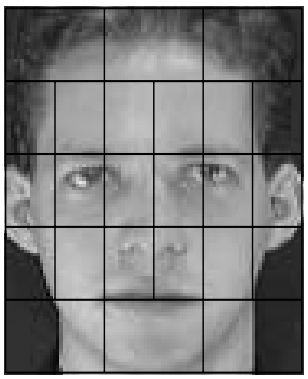


Figure 6.4: Initial face segmentation for the embedded HMM

observations from left to right, and $T_0$ observations from top to bottom. In other words, we have an array of $T_0 T_1$ observations. The values for $T_0$ and $T_1$ are determined, of course, on the image size, the size of the image blocks ($L_x$ and $L_y$), and the amount of overlap ($P_x$ and $P_y$). These observations are then used for training as follows (See Figure 6.3):

1. At the first iteration, the observation vectors are uniformly segmented into $N_0$ vertical super states, and the vectors within each super state $k$ are uniformly segmented from left-to-right into $N_1^k$ states (Figure 6.4).

2. At the next iteration, the uniform segmentation is replaced by a doubly embedded Viterbi segmentation, which is illustrated in Figure 5.3.

   The model parameters are then estimated using an extension of the segmental k-means algorithm [61] to two dimensions. Specifically, the estimated transition probabilities between super states $\tilde{a}_{0,ij}$ and the estimated transition probabilities between the embedded states $\tilde{a}^i_{1,jl}$ are obtained as follows,

$$\tilde{a}^i_{1,jl} = \frac{\text{number of transitions from } S^i_{1,j} \text{ to } S^i_{1,l}}{\text{total number of transitions from } S^i_{1,j}}$$

$$\tilde{a}_{0,ij} = \frac{\text{number of transitions from } S_{0,i} \text{ to } S_{0,j}}{\text{total number of transitions from } S_{0,i}}$$

In addition, the estimated mean $\tilde{\mu}^i_{jm}$, the covariance $\tilde{\mathbf{U}}^i_{jm}$ of the Gaussian mixture, and the mixture coefficients $\tilde{c}^i_{jm}$ for mixture $m$ of state $j$ in super state $i$ are obtained as follows:

$$\tilde{\mu}^i_{jm} = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1}\psi^{i,r}_{jm}(t_0,t_1)\mathbf{O}(t_0,t_1)}{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1}\psi^{i,r}_{jm}(t_0,t_1)}$$

$$\tilde{\mathbf{U}}^i_{jm} = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1}^{T_1}\psi^{i,r}_{jm}(t_0,t_1)(\mathbf{O}(t_0,t_1)-\tilde{\mu}^i_{jm})(\mathbf{O}(t_0,t_1)-\tilde{\mu}^i_{jm})^T}{\sum_{r=1}^{R}\sum_{t_0}^{T_0}\sum_{t_1}^{T_1}\psi^{i,r}_{jm}(t_0,t_1)}$$

$$\tilde{c}^i_{jm} = \frac{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1}\psi^{i,r}_{jm}(t_0,t_1)}{\sum_{r=1}^{R}\sum_{t_0=1}^{T_0}\sum_{t_1=1}^{T_1}\sum_{m=1}^{M}\psi^{i,r}_{jm}(t_0,t_1)}$$

where $\psi^{i,r}_{jm}(t_0,t_1)$ is equal to one if $\mathbf{O}(t_0,t_1)$ is associated with mixture component $m$ in state $j$ of super state $i$, and it is zero otherwise.

3. The iteration stops, and the embedded HMM is initialized, when the doubly embedded Viterbi score at consecutive iterations is smaller than a specified threshold.

4. The parameters of the embedded HMM are further re-estimated using the forward backward procedure, and a re-estimation procedure to maximize the probability of the observations given the face model. The forward and backward procedure together with the re-estimation equations for the embedded HMM were derived and explained in detail in Chapter 5.

5. The iteration stops, and the parameters of the embedded HMM is trained, when the probability of the observations given the model at consecutive iterations is smaller than a specified threshold.

## 6.4 Face Recognition Using the Embedded HMM

Once an embedded HMM has been trained on a set of face images, it may be used for face recognition. The procedure is as follows. First, the observation vectors are extracted from the unknown face. Then, the probability of the observation sequence given the embedded HMM face model of each person is computed using the doubly embedded Viterbi algorithm. Finally, the model with the highest likelihood is selected, and the selected model reveals the identity of the unknown face (Figure 6.5). We have tested the embedded HMM face recognition system on the Olivetti Research

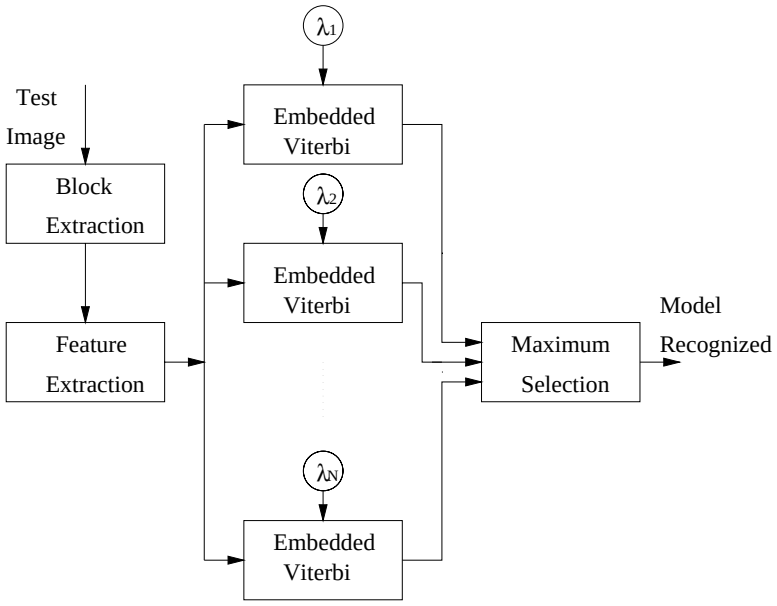



Figure 6.5: Face recognition scheme using the embedded HMM

Ltd. (ORL) database. For both DCT and KLT-based observation vectors we have achieved a recognition rate of 98%, when five images were used for training and each state was modeled by a Gaussian density function. If the number of images used for training is increased to seven or more, or the number of Gaussian mixtures used to model each state is increased to three or more, our system achieves perfect (100%) recognition for both KLT and DCT-based observation vectors.

Table 6.1 compares some of the HMM based face recognition approaches, that

| | **Recognition Rate** | **Complexity (additions)** |
|---|---|---|
| HMM [74] | 85% | $N_0^2 T_0$ |
| HMM [71] | 90-95% | $(\sum_{k=1}^{N_0} N_1^{(k)})^2 T_0 T_1$ |
| Embedded HMM | 98% | $(\sum_{k=1}^{N_0} (N_1^{(k)})^2 T_1) T_0 + {N_0}^2 T_0$ |

Table 6.1: Comparison of the face recognition rate and numerical complexity for different HMM-based approaches

have been tested on the ORL database, in terms of recognition rate and computational complexity. In all experiments, five images were used for the training set and each state was modeled by a Gaussian density function. From Table 6.1, it is clear that the standard HMM provides the most efficient implementation but has the smallest recognition rate. Although the complexity of the embedded HMM is larger than for the standard HMM, its recognition rate is significantly higher. The embedded HMM has a recognition rate that is from 3% to 8% better than the HMM with end of line states (Figure 2.1 -a) along with a significant decrease in the amount of computation compared to the same model. In addition, unlike the HMM with end of line states, the embedded HMM allows for a parallel implementation of the decoding algorithm, i.e. the doubly embedded Viterbi algorithm. This can dramatically reduce the recognition time. Furthermore, the use of the 2D-DCT or KLT coefficients rather than the pixel intensities decreases significantly the size of the observation vectors and, consequently, decreases the recognition complexity compared to the face recognition system proposed by Samaria [71].

Since face recognition systems are often tested on different databases, it is difficult to compare the performance of one system to another. However, in Table 6.4 we present some of the more successful face recognition systems that have been tested on the ORL database. It should be noted that methods 1-3 were tested using one

| | **Approach** | **Recognition Rate** |
|---|---|---|
| 1 | Auto Association and Classification NN [42] | 20% |
| 2 | Dynamic Link Matching [51] | 80% |
| 3 | Eigenface [19] | 80% |
| 4 | HMM [74] | 85% |
| 5 | VFR model [60] | 92.5% |
| 6 | HMM [71] | 90-95% |
| 7 | PDBNN [45] | 96% |
| 8 | Convolutional NN [43] | 96.2% |
| 9 | Embedded HMM | 98-100% |

Table 6.2: Comparison of the face recognition rate for different approaches tested on the ORL database

Figure 6.6: Face recognition results using the embedded HMM

image per person for training (these systems were designed to recognize a face given only one training example) [73]. In methods 4-9, on the other hand, both the testing and training sets contained five images of each person, and there was no duplication of images between the testing and training sets. It should also be pointed out that the recognition rate for the "eigenfaces" method depends on the number of eigenfaces that are used, and that the rate varies from about 73% when less than 5 eigenfaces are used to about 90% when there are 175-200 eigenfaces [71]. The results shown in the table were obtained using 39 eigenfaces [73]. In [60], the authors reported that the performance of the VFR model increases with the number of images used in the training set, with perfect recognition (100%) when eight images per person were used. Figure 6.6 shows some recognition results when five images were used to train each of the face embedded HMMs. The crossed images represent incorrect classifications, while the rest of images are examples of correct classification.

## 6.5 Face Detection Using the Embedded HMM

The embedded HMM structure allows for an efficient implementation of a face detection system using the doubly embedded Viterbi algorithm. A face detection system should be able to locate all faces in an image of both males and females independent of their appearance, race, age, scale, orientation, background or illumination. As it will be shown later in this section, an embedded HMM allows for more flexibility with scale deformations in both horizontal and vertical direction, than the standard HMM or the template based approaches. Let's recall here that while the HMM based approach discussed in Chapter 4 allows only for variations in the height of the faces, the template based approaches are rather inflexible to any scaling variations. This represents one important problem of these approaches since variations in the size of faces should be expected in most of face detection applications. The overall face detection system is illustrated in Figure 6.7. Since in our approach no assumptions

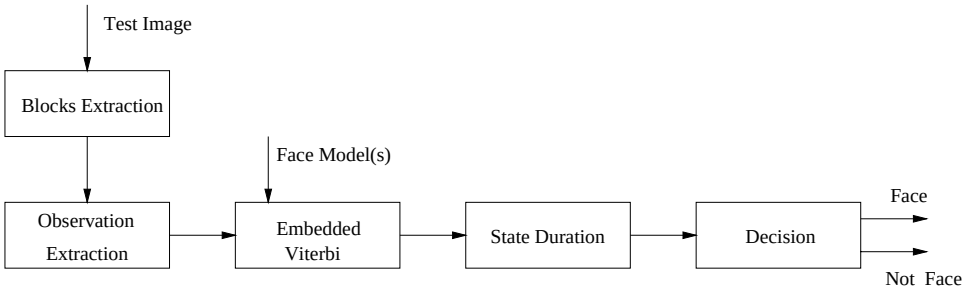



Figure 6.7: Face detection using the embedded HMM

are made with regard to the background, the face likelihood is computed for all rectangular patterns in the face image. The face likelihood for each rectangular pattern is given by the doubly embedded Viterbi likelihood, computed for the observation vectors within the rectangular pattern, weighed by a state duration correction factor. The rectangular patterns for which the face likelihood increases a fixed threshold are taken as valid faces. Before we describe the steps of the face detection algorithm in more detail, it is important to notice that by taking advantage of the large horizontal and vertical overlap of the rectangular patterns, the computation of the state probabilities of the observation vectors within the rectangular patterns can be highly decreased. This is due to the fact that the overlapping patterns have in common a large number of observation vectors. Based on the above observation, the first step in our face detection approach is to extract the image blocks from a test image, followed by the computation of the observation vectors and the corresponding state probabilities. The image blocks are extracted by sliding a rectangular window from left to right and from top to bottom across the test image (Figure 6.8). The blocks have the same height $L_y$, width $L_x$, horizontal and vertical overlap $P_x, P_y$ as those used for training. Given an image of size $W \times H$, the pairs $(W_m, H_m)$ and $(W_M, H_M)$ represent the width and height of the smallest and largest face, respectively, that we wish to find in an image (Figure 6.9). To detect a face of width $W_f$, and height $H_f$, where $W_m \leq W_f \leq W_M$ and $H_m \leq H_f \leq H_M$, the number of blocks that are extracted in

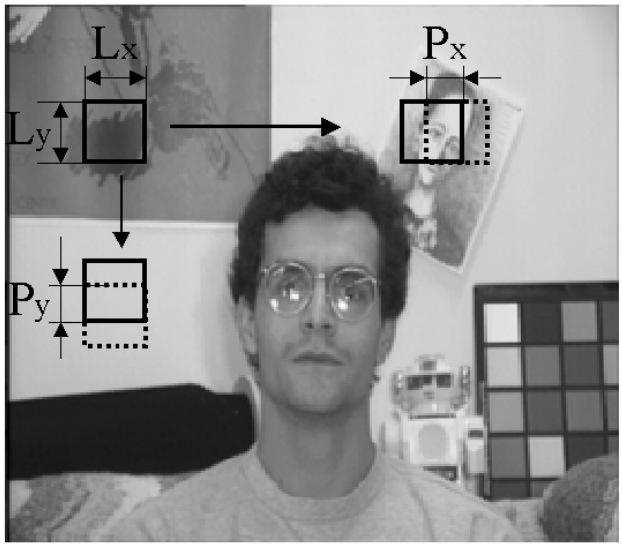



Figure 6.8: Block extraction for face detection using embedded HMM

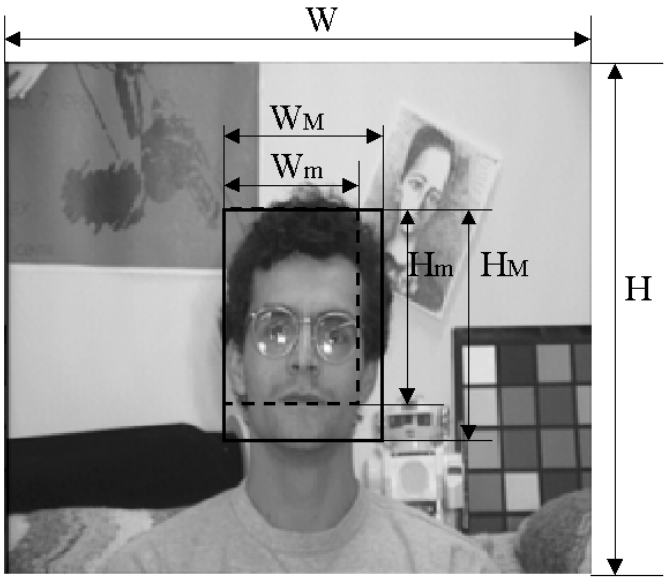



Figure 6.9: Image parameterization for face detection using embedded HMM.

the horizontal direction, $T_x$, and the vertical direction, $T_y$, is

$$T_x = \frac{W - L_x}{L_x - P_x} + 1 \tag{6.1}$$

$$T_y = \frac{H - L_y}{L_y - P_y} + 1 \tag{6.2}$$

Note that, unlike the standard HMM based face detection system, the number of horizontal and vertical blocks given by Equation 6.1- 6.2, is independent of the face size to be detected. The observation vectors are obtained from either 2D-DCT coefficients or KLT coefficients extracted from the image blocks using the same technique used for training.

Next stage of the face detection algorithm is to compute the face likelihood of each rectangular pattern using the doubly embedded Viterbi algorithm. Before the face likelihood of each rectangular pattern is computed, it is important to notice that by taking advantage of the large vertical overlap of the rectangular patterns, the complexity of the face likelihood computation can be significantly reduced. Let's recall that, the calculation of the embedded Viterbi algorithm requires the calculation of the Viterbi score of each row of a rectangular pattern given all the superstates of the model, followed by the calculation of the Viterbi score for the overall top to bottom structure. Hence, by taking advantage of the vertical overlap of the rectangular patterns, the computation of the Viterbi algorithm for the horizontal sequences extracted from the overlapping region can be saved. Furthermore, when the Viterbi score is computed for an observation sequence corresponding to $W_M$, given all the super states of the model, the partial likelihood is obtained for all the observation sequences corresponding to $1 \leq W_f \leq W_M$. Since the minimum width of the faces we wish to find is $W_m$, only scores for horizontal sequences corresponding to $W_f$, $W_m \leq W_f \leq W_M$ are retained. The above observation is very important, because it describes an efficient way to compute the likelihood of an horizontal observation sequence extracted from a rectangular pattern of any width $W_f$, $W_m \leq W_f \leq W_M$,

given the super state of a face model. This allows, as will be explained in the next paragraph, to obtain the face likelihood for a pattern of any size of interest, using the same face model.

Using the super state probabilities of the horizontal observation sequences obtained in the previous stage, the doubly embedded Viterbi score for the rectangular patterns of any width $W_f$, $W_m \leq W_f \leq W_M$ and height $H_M$ is obtained by running the Viterbi segmentation algorithm for the overall top-to-bottom structure of super states. When the doubly embedded Viterbi score is computed for a rectangular pattern of width $W_f$, height $H_M$, and having the left top corner at $(x, y)$, the likelihood scores given the face models are implicitly computed for all rectangular patterns with the same left top corner, width, and of any height $H_f$, $1 \leq H_f \leq H_M$. Given the constraints of the system related to the height of the faces we wish to detect, only patterns of height $H_f$, $H_m \leq H_f \leq H_M$ and $W_m \leq W_f \leq W_M$ are further considered. The computation of the face likelihood for images from different scales using the same face model is one very important advantage of this approach over the template based and standard HMM based approaches. In template based approaches for each scale of a rectangular pattern a new face template, at the same scale, has to be used to determine the face likelihood score of the pattern. Although more flexible than the template based approach, in standard HMM based approach for each width of the rectangular pattern a new face model has to be trained. Unlike these methods, in our approach, one embedded HMM can be used to compute the face likelihood of patterns with significant scale variations. Considering the above discussions, the total number of additions required to compute the doubly embedded Viterbi score for all rectangular patterns in the test image is:

$$\begin{aligned} Num\ Adds &= T_y(T_x - T_{1,m})T_{1,M}\sum_{k=1}^{N_0}(N_1^{(k)})^2) \\ &+ (T_x - T_{1,m})(T_y - T_{1,m})N_0^2 T_{0,M}(T_{1,M} - T_{1,m}), \end{aligned} \tag{6.3}$$

where $T_{0,m}, T_{1,m}$ and $T_{0,M}, T_{1,M}$ represent the size of the observation array correspond-

ing to the smallest $H_m \times W_m$ and respectively largest $H_M \times W_M$ face of interest. The above complexity does not include the calculations required to obtain the observation vectors. The detection time can be further decreased, by taking advantage of a parallel implementation for the doubly embedded Viterbi algorithm, as discussed in Chapter 5.

The accuracy of detection may be improved by including the state duration into the overall top to bottom HMM. The duration $d_i$ of superstate $i$ is modeled using a Poisson distribution [97] of parameter $l_i$. It has been shown that the inclusion of the states duration increases significantly the complexity of the system [62]. However, a very simple and efficient method is to compute the face likelihood as follows,

$$\log \tilde{P}(\mathbf{O}, \mathbf{q}|\lambda) = \log P(\mathbf{O}, \mathbf{q}|\lambda) + \alpha \sum_{i=1}^{N_0} \log\ p_i(d_i) \tag{6.4}$$

where $\alpha$ is a constant that was set equal to 1000 in our experiments. The parameter of the Poisson distribution for the super state duration is obtained in the training part from the segmented data using

$$l_i = \frac{\text{number of observations in superstate } i}{\text{size of the observation sequence along vertical direction}} \tag{6.5}$$

To deal with different scales of the images in the training and test sets, the Poisson parameter is normalized to the integer value $\tilde{l_i}$ computed as follows

$$\tilde{l_i} = \text{Int}[l_i \frac{\text{length of test sequence}}{\text{average length of training sequence}}] \tag{6.6}$$

The likelihood scores obtained for the observation sequences corresponding to each rectangular pattern are normalized by averaging the likelihood score over the size of the array of the observation sequence. Next, the face likelihoods obtained for each rectangular pattern in the image are compared to a threshold, and the patterns that have likelihoods that exceed this threshold are candidate faces. It is natural to expect that similar patterns will have similar likelihoods and, therefore, several patterns around the actual face location will be declared to be candidate faces. In

order to remove these "false alarms", a candidate face represents a valid face location if its likelihood is larger than the likelihoods of all face candidates within the vicinity.

This embedded HMM face detection system using both KLT and DCT based observation vectors was tested on the MIT database in three experiments. The experiments are similar to those described for the face detection using the standard HMM. For the first experiment, a set of nine manually segmented images, from the MIT database were used to train one face model. The detection system was tested on images of the same database, that contain frontal faces of the same width as those used in training, but show significant variations in illumination. In the second and third experiment, eight face models were trained using face images, with no background, from the ORL database. The images used to train each embedded HMM are the same as used by the 1D-HMM based face detection system. The images tested in the second experiment contain frontal faces of different sizes and different illumination conditions. In the third experiment 432 images were tested. For this experiment the test images include faces of different orientations (rotations in the image plane), sizes (varying from $60 \times 90$ to $120 \times 180$) and taken under different illumination conditions. The detection results (detection rate and false alarms) in these three experiments are summarized in Table 6.3. The false alarms are reported to the total number of rectangular patterns obtained from all the test images. Figure 6.10 shows some of the detection results. As in the standard HMM based face detection system, the use of KLT observation vectors instead of DCT observation vectors increases slightly the detection rate. This is a result of the fact that the KLT basis functions are adapted to face images while the 2D DCT basis function are general.

Compared with the template-based or standard HMM-based methods for face detection our approach has the following advantages:

1. It is more flexible with respect to variations in scale, and natural deformations in both vertical and horizontal direction.

2. It allows for an efficient implementation of the face detection algorithm due to

| | Experiment 1 | | Experiment 2 | | Experiment 3 | |
|---|---|---|---|---|---|---|
| | DR | FA | DR | FA | DR | FA |
| 2D-DCT | 100% | $\frac{0}{426,187,008}$ | 91.7% | $\frac{16}{426,187,008}$ | 91.2% | $\frac{56}{1,278,561,024}$ |
| KLT | 100% | $\frac{0}{426,187,008}$ | 96.3% | $\frac{7}{426,187,008}$ | 91.5% | $\frac{43}{1,278,561,024}$ |

Table 6.3: Comparison of the detection rate (DR) and false alarms (FA) in different experiments obtained using the embedded HMM

Figure 6.10: Face detection results using the embedded HMM

the breaking of the face templates or horizontal face bands (for the standard HMM approach) into image blocks that are processed to obtain the observation vectors. Unlike the standard HMM based approach in Chapter 4, in this approach the width of the blocks is not constrained to equal the width of the faces. Therefore the face likelihood score for rectangular patterns with variations in scale in both vertical and horizontal direction can be computed efficiently.

Furthermore compared to the HMM with end of line states, the embedded HMM for face modeling proposed here has the following advantages:

1. It allows for better initial estimates for the training stage. The initial parameters can be obtained by uniformly segmenting the data according to the structure of states and super states of the embedded HMM,

2. It has better recognition results (the detection performance of the HMM with end of line states were not tested),

3. It allows for a faster implementation for both detection and recognition due to the reduced complexity of the doubly embedded Viterbi algorithm as shown in Chapter 5,

4. It can be implemented using a parallel architecture, which will dramatically reduce the complexity of both detection and recognition,

5. It preserves the two dimensional structure of the data without using end-of-line states and blocks.

# CHAPTER 7

# Conclusions, Contributions of Thesis and Topics for Further Research

This chapter presents the conclusions of this thesis work, highlights the contributions of this thesis, and gives general direction for further research.

## 7.1 Conclusions

The primary goal of this work is to consider the use of an HMM and an embedded HMM for face detection and recognition. The use of an HMM for face modeling is justified by both the flexibility of the HMM and by the structure of faces. As a result of the theoretical aspects discussed in this work, and based on our experimental results the following conclusions are drawn:

- Compared with the template-based methods for face detection and recognition, a hidden Markov model for faces:
  - Is more flexible with respect to variations in scale, natural deformations in the vertical direction.
  - Allows for a faster implementation of the face detection algorithm due to the breaking of the face templates in short image blocks which are processed to obtain the observation vectors.

  - Although moderately successful for both detection and recognition, is outperformed by some of the template based approaches. More training data may help to improve the accuracy of this approach.

- The flexibility to scale and natural deformations of the face HMM refers only to the vertical direction, but not to the horizontal direction. The embedded HMM represents a more appropriate model for face images than the standard HMM or templates.

  - For face recognition the continuous mixture embedded HMM achieves the best recognition rate reported on the ORL database, and shows perfect recognition rate (100%).
  - The embedded HMM is more flexible with respect to variations in scale and natural deformations in both vertical and horizontal direction than both template based and standard HMM based approaches.
  - The embedded HMM allows for a faster implementation of the face detection algorithm due to the breaking of the face templates or horizontal face bands (for the standard HMM approach) in image blocks which are processed to obtain the observation vectors.

- Compared to the HMM with end-of-line states (Figure 2.1-a), the embedded HMM for faces has the following advantages:

  - Allows for better initial estimates for the training stage. The initial parameters can be obtained by uniformly segmenting the data according to the structure of states and super states of the embedded HMM,
  - Has better recognition results,
  - Allows for parallel implementation of the training, recognition or detection,
  - Allows for a faster implementation for both detection and recognition,

- Allows for parallel implementation of the decoding, evaluation and estimation algorithms,
- Preserves the two dimensional structure of the data without using end-of-line states and blocks.

## 7.2 Contributions of Thesis

A list of the original contributions of this thesis is listed below:

- A near real-time system for detection of human heads from a known background has been developed. The system is able to segment the human heads and eyes from gray scale sequences using some natural rules and a deformable template of elliptical shape, track and select the best instance of the face over a sequence of frames.
- The capabilities of a continuous one dimensional HMM for human face recognition have been investigated. The size of the observation vectors has been significantly reduced from the previous HMM-based system by using either the DCT or KLT coefficients. The recognition rate was slightly improved while the complexity was dramatically decreased.
- A new HMM-based face detection method from unknown background has been developed. The face HMM used for detection uses the same efficient observation vectors as used for recognition. The state duration model has been incorporated in the HMM structure as a correction factor that is added to the face likelihood score. The state duration modeling increased significantly the detection rate.
- The decoding, evaluation (forward and backward), and estimation algorithms for the embedded HMM (Figure 5.2) have been studied. The evaluation and estimation algorithms have been derived for both the discrete and continuous

mixture embedded HMM. The complexity of the decoding and evaluation algorithms have been determined. Finally the implementation issues related to all these algorithms have been discussed.

- A new face recognition approach using the continuous and continuous mixture embedded HMM has been developed. The observation vectors use an efficient set of two-dimensional DCT coefficients. The training scheme includes the decoding, evaluation, estimation algorithms as well as a variant of the segmental K-means algorithm derived for the continuous mixture embedded HMM.

- A new face detection approach for face detection using the continuous and continuous mixture embedded HMM has been developed. The embedded HMM for detection uses the same observation vectors as used for recognition. The state duration modeling incorporated as an correction factor in the face likelihood score has significantly improved the detection score.

## 7.3 Recommended Topics for Further Research

Based on the experiments and results presented in this work, we present some general directions for future work in modeling faces using the embedded HMM.

- Increasing the number of images used in the training set will allow for more robust estimation of the model parameters and eventually increase the recognition and detection rates of the system. Experiments on a larger face recognition and detection database will provide results that are statistically more significant.

- It has been shown [61] that the initial parameters are very important for the proper convergence of the training algorithm. As shown in Figure 6.4, a uniform segmentation does not always provide initial parameters that correspond to significant facial features. A more sophisticated face segmentation algorithm

that may use adaptive segmentation algorithms [102], [103], [104] may provide better initial estimates for the embedded HMM.

- The use of discriminant observation vectors can provide not only a good representation of all the blocks in an image but also a better discrimination among blocks corresponding to different states of the model. These observation vectors will provide a more robust estimation of the parameters of the model.

- The use of color information, specially from the HSV space, has been shown [33], [105], [77] to be effective in face detection. The inclusion of the color information in the observation vectors should significantly improve both the recognition and the detection systems.

- The minimum recognition error criteria [106] for training provides better discrimination between models to be trained. Unlike the maximum likelihood training for the embedded HMM described in Chapter 5, the minimum recognition error training maximizes the "discrimination" among models. This training technique is expected to improve both the detection rate, as an effect of better separating face models from background models, and the recognition rate, as a result of better discrimination among face models.

- The inclusion of the background model will provide a more efficient face detection algorithm. It might also improve the accuracy of the face detection system, especially if discriminant training or discriminant observation vectors are used. The background model can be integrated with the face model in the form of a background super state. For the embedded HMM, a Passing Token algorithm [107], similar to those applied in continuous speech recognition can be applied from the top to the bottom on the overall super state structure. However, the background varies, in general, from image to image and, therefore, it is impossible to find a general background model. Thus the incorporation of the

background model for face detection is appropriate only if it can be assumed that the statistics of the background in an image are known.

- Eventually the embedded HMM based face detection and recognition systems will be used in the overall face identification system described in Figure 1.1 and the performance of the system will be tested on video sequences.

## 7.4 Published Work

- Ara V. Nefian , and Monson H. Hayes III , "Hidden Markov Models for face detection and recognition," submitted to *IEEE Transactions on Pattern Analysis and Machine Intelligence*, April 1999.

- Ara V. Nefian , Monson H. Hayes III,"An embedded HMM based approach for face detection and recognition," *IEEE International Conference on Acoustic, Speech and Signal Processing 99*, March 1999, p.3553-3556, vol VI.

- Ara V. Nefian , Monson H. Hayes III,"Face Recognition using an embedded HMM," *IEEE International Conference Audio Video Biometric based Person Authentication 99*, March 1999, p. 19-24.

- Ara V. Nefian , Monson H. Hayes III,"Hidden Markov Models For Face Detection and Recognition," *IEEE International Conference on Image Processing 98*, October 1998, vol. 1, p. 141-145.

- Ara V. Nefian , Monson H. Hayes III,"Hidden Markov Models For Face Recognition," *IEEE International Conference on Acoustic, Speech and Signal Processing 98*, May 1998, vol.5, p. 2721-2724.

- Ara V. Nefian, M. Khosravi, M.H. Hayes III, "System and Method for Detecting Human Face in Uncontrolled Environments," US/European Patent.

- Ara V. Nefian, Mehdi Khosravi, Monson H. Hayes III, "Real Time Human Face Detection From Uncontrolled Environments," *Visual Communications on Image Processing*, p. 211-219, February 1997.

# Bibliography

[1] R. Chellappa, C. Wilson, and S. Sirohey, "Human and machine recognition of faces: A survey," *Proceedings of IEEE*, vol. 83, May 1995.

[2] R. Brunelli and T. Poggio, "Face recognition:features versus templates," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 15, October 1993.

[3] A. Pentland, B. Moghaddam, and T. Starner, "View based and modular eigenspaces for face recognition," in *Proceedings on IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 84–91, 1994.

[4] L. D. Harmon, M. K. Khan, R. Lasch, and P. F. Ramig, "Machine identification of human faces," *Pattern Recognition*, pp. 97–110, 1981.

[5] A. J. Goldstein, L. Harmon, and A. Lesk, "Identification of human faces," *Proceedings of the IEEE*, pp. 748–760, 1971.

[6] R. Brunelli and S. Messelodi, "Robust estimation of correlation with applications to computer vision," *Pattern Recognition*, vol. 28, no. 6, pp. 833–841, 1995.

[7] M. Bichsel and A. Pentland, "Human face recognition and face image set's topology," *CVGIP:Image Understanding*, vol. 59, pp. 254–261, March 1994.

[8] R. Brunelli and D. Falavigna, "Person identification using multiple cues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 955–966, October 1995.

[9] L.Yuille, P. Hallinan, and D. Cohen, "Feature extraction from faces using deformable templates," *International Journal of Computer Vision*, vol. 8, no. 2, pp. 99–111, 1992.

[10] B. Manjunath, R. Chellappa, and C. d. Malsburg, "A feature based approach to face recognition," *Proceeding of IEEE Computer Society. Conference on Computer Vision and pattern Recognition*, pp. 373–378, 1992.

[11] G. Chow and X. Li, "Towards a system for automatic facial feature detection," *Pattern Recognition*, vol. 26, no. 12, pp. 1739–1755, 1993.

[12] L. Stringa, "Eyes detection for face recognition," *Applied Artificial Intelligence*, vol. 7, pp. 365–382, Oct-Dec 1993.

[13] M.Michaelis, R. Herpers, L. Witta, and G. Sommer, "Hierarchical filtering scheme for the detection of facial keypoints," in *International Conference on Acoustics, Speech and Signal Processing*, vol. 4, pp. 2541–2544, 1997.

[14] A. M. Alattar and S. A. Rajala, "Facial features localization in front view head and shoulders images," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 6, pp. 3557–3560, 1999.

[15] P. Burt, "Smart sensing within a pyramid vision machine," *Proceedings of the IEEE*, vol. 76, pp. 1006–1015, August 1988.

[16] M. Kreutz, B. Volpel, and H. Janssen, "Scale invariant image recognition based on higher order autocorrelation features," *Pattern Recognition*, vol. 29, no. 1, pp. 19–26, 1996.

[17] K. Matsuda, T. Kageyama, and T. Aibara, "Application of the seconfd order statistics for the recognition of human front faces," in *Proceedings of the International Conference on Image Processing*, pp. 543–546, 1992.

[18] D. Beymer, "Face recognition under varying pose," in *Proceedings of 23rd Image Understanding Workshop*, vol. 2, pp. 837–842, 1994.

[19] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proceedings of International Conference on Pattern Recognition*, pp. 586 – 591, 1991.

[20] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, pp. 71–86, March 1991.

[21] A. Pentland, R. Picard, and S. Scarloff, "Photobook:content-based manipulation of image databases," *International Journal of Computer Vision*, vol. 18, pp. 233–254, June 1996.

[22] L. Sirovitch and M.Kirby, "Low-dimensional procedure for the characterization of human faces," *Journal of the Optical Society of America*, vol. 4, pp. 519–524, March 1987.

[23] M. Kirby and L. Sirovitch, "Application of the Karhunen-Loeve procedure for the characterization of human faces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 103–108, January 1990.

[24] R. Epstein, P. Hallina, and A. Yuille, "5+/- eigenimages suffices: An empirical investigation of low-dimensional lighting models," in *Proceedings of the Workshop on Physics-based Modeling in Computer Vision*, pp. 108–116, 1995.

[25] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Transactions On Pattern Analysis and Machine Intelligence*, vol. 19, pp. 676–710, July 1997.

[26] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," in *The 5th International conference on Computer Vision, Cambridge MA*, pp. 786–793, June 1995.

[27] S. Akamatsu, T. Sasaki, H. Fukumachi, and Y. Suenuga, "A robust face identification scheme - KL expansion of an invariant feature space," in *SPIE Proceedings::Intelligent Robots and Computer Vision X:Algorithms and Techniques*, vol. 1607, pp. 71–84, 1991.

[28] H. Murase and S. Nayar, "Visual learning and recognition of 3-d objects from appearance," *International Journal of Computer Vision*, vol. 14, pp. 5–24, 1995.

[29] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs Fisherfaces: Recognition using class specific linear projection," in *Proceedings of Fourth European Conference on Computer Vision, ECCV'96*, pp. 45–58, April 1996.

[30] P. Belhumeur, J. Hespanha, and D. J. Kriegmand, "Eigenfaces vs fisherfaces:recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 711–720, July 1997.

[31] K. Etemad and R. Chellappa, "Face recognition using discriminant eigenvectors," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, pp. 2148–2151, 1996.

[32] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* Academic Press, 1990.

[33] A. W. Senior, "Face and feature finding for a face recognition system," in *Second International Conference on Audio- and Video-Based Biometric Person Authentication*, pp. 154–159, 1999.

[34] Z. Hong, "Algebraic feature extraction of image for recognition," *Pattern Recognition*, vol. 24, no. 3, pp. 211–219, 1991.

[35] Y. Cheng, K. Liu, J. Yang, Y. Zhang, and N. Gu, "Human face recognition method based on the statistical model of small sample size," in *SPIE Proceedings:Intelligent Robots and Computer Vision X:Alg. and Techn*, vol. 1607, pp. 85–95, 1991.

[36] Q. Tian, "Comparaision of statistical pattern-recognition algorithms for hybrid processing, ii: eigenvector-based algorithms," *Journal of the Optical Society of America*, vol. 5, pp. 1670–1672, 1988.

[37] Y. Cheng, K. Liu, J. Yang, and H. Wang, "A robust algebraic method for human face recognition," in *Proceedings of 11th International Conference on Pattern Recognition*, pp. 221–224, 1992.

[38] P. Phillips, "Matching pursuit filter design," in *12th International Conference on pattern recognition*, pp. 57–61, October 1994.

[39] P. Phillips, "Matching pursuit filters design for face identification," in *SPIE*, vol. 2277, pp. 2–9, 1994.

[40] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequncy dictionaires," *IEEE Transactions on Signal Processing*, vol. 41, pp. 3397–3415, December 1993.

[41] J. Phillips, "Matching pursuit filters applied to face identification," *IEEE Transactions on Image Processing*, vol. 7, pp. 1150–1164, August 1998.

[42] G. Cotrell and M. Fleming, "Face recognition using unsupervised feature extraction," in *Proceedings International neural Network Conference*, pp. 322–325, 1990.

[43] A. Lawrence, C. Giles, A. Tsoi, and A. Back, "Face recognition : A convolutional neural network approach," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 98–113, 1997.

[44] S.-H. Lin, Y. Chan, and S. Kung, "A probabilistic decision-based neural network for locationg deformable objects and its applications to surveillance system and video browsing," in *International Conference on Acoustics, Speech and Signal Processing*, vol. 6, pp. 3553–3556, 1996.

[45] S.-H. Lin, S.-Y. Kung, and L.-J. Lin, "Face recognition/detection by probabilistic decision-based neural network," *IEEE Transactions on Neural Network*, vol. 8, pp. 114–132, January 1997.

[46] R. Vaillant, C. Monrocq, and Y. L. Cun, "Original approach for the localization of objects in images," *IEE Proceedings on Vision, Image, and Signal Processing*, vol. 141, pp. 245–250, August 1994.

[47] K. Sung and T. Poggio, "Exampled-based learning for viewbased human face," in *Image Understanding Workshop*, vol. 2, pp. 843–850, November 1994.

[48] K. Sung and T. Poggio, "Exampled-based learning for viewbased human face," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 39–51, January 1998.

[49] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 23–38, January 1998.

[50] S. Ben-Yacoub, B.Fasel, and J. Luttin, "Fast face detection using MLP and FFT," in *Second International Conference on Audio- and Video-Based Biometric Person Authentication*, pp. 31–35, 1999.

[51] M. Lades, J.Vorbruggen, J. Buhmann, J. Lange, C. Malsburg, and R. Wurtz, "Distorsion invariant object recognition in the dynamic link architecture," *IEEE Transcations on Computers*, vol. 42, no. 3, pp. 300–311, 1993.

[52] O. Nakamura, S. Mathuri, and T. Minami, "Identification of human faces based on isodensity maps," *Pattern Recognition*, vol. 24, no. 3, pp. 263–272, 1991.

[53] C. Nastar and A. Pentland, "Matching and recognition using deformable intensity surfaces," in *Proceedings on International Symposium on Computer Vision*, pp. 221–228, November 1995.

[54] T. Tan and H. Yan, "Face recognition by fractal transformations," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 6, pp. 3537–3540, 1999.

[55] C. Podilchuk and X. Zhang, "Face recognition using dct-based feature vectors," in *International Conference on Acoustics and Specch and Signal Processing*, vol. 4, pp. 2144–2147, 1996.

[56] R. Brunelli and T. Poggio, "Template matching:matched spatial filters and beyond," *Pattern Recognition*, vol. 30, pp. 751–768, May 1997.

[57] R. P. Rao and D. Ballard, "Natural basis functions and topographic memory for face recognition," in *International Joint Conference on Artificial Intelligence*, vol. 1, pp. 10–17, 1995.

[58] E. Osuna, R. Freud, and F. Girosi, "Training support vector machines:an application to face detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 130–136, 1997.

[59] A. J. Colmenarez and T. Huang, "Face detection with information-based maximum discrimination," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 782–787, 1997.

[60] J. Ben-Arie and D. Nandy, "A volumetric/iconic frequency domain representation for objects with application for pose invariant face recognition," *IEEE*

*Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 449–457, May 1998.

[61] L. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proceedings of IEEE*, vol. 77, pp. 257–286, February 1989.

[62] L. Rabiner and B. Huang, *Fundamentals of Speech Recognition.* Englewood Cliffs, NJ: Prentice-Hall, 1993.

[63] E.Levin and R. Pieraccini, "Dynamic planar warping for optical character recognition," in *International Conference on Acoustics, Speech and Signal Processing*, pp. 149–152, 1992.

[64] O. Agazzi, S. Kuo, E. Levin, and R. Pieraccini, "Connected and degraded text recognition using planar HMM," in *International Conference on Acoustics, Speech and Signal Processing '93*, vol. 5, pp. 113–116, 1993.

[65] S. Kuo and O. Agazzi, "Keyword spotting in poorly printed documents using pseudo 2-D Hidden Markov Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 842–848, August 1994.

[66] O. Agazzi and S. Kuo, "Hidden Markov Models based optical character recognition in presence of deterministic transformations," *Pattern Recognition*, vol. 26, no. 12, pp. 1813–1826, 1993.

[67] F. Samaria, "Face segmentation for identification using hidden Markov models," in *British Machine Vision Conference*, 1993.

[68] F. Samaria and F. Fallside, "Face identification and feature extraction using hidden Markov models," *Image Processing: Theory and Applications*, 1993.

[69] F. Samaria and F. Fallside, "Automated face identification using hidden Markov models," in *Proceedings of the International Conference on Advanced Mechatronics*, pp. 1–9, 1993.

[70] F. Samaria and A. Harter, "Parametrisation of stochastic model for human face identification," in *Proceedings of the Second IEEE Workshop on Application of Computer Vision*, pp. 138–142, 1994.

[71] F. Samaria, *Face Recognition Using Hidden Markov Models*. PhD thesis, University of Cambridge, 1994.

[72] S. Mueller and S. Eickeler, "Segmentation and classification of hand-drawn pictograms in cluttered scenes-an integrated approach," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 6, pp. 3489–3451, 1999.

[73] J. Zhang, Y. Yan, and M. Lades, "Face recognition: Eigenface, elastic matching and neural nets," *Proceedings of the IEEE*, vol. 85, pp. 1423–1435, September 1997.

[74] F. Samaria and S. Young, "HMM based arhitecture for face identification," *Image and Vision Computing*, vol. 12, pp. 537–543, October 1994.

[75] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real time tracking of the human body," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 780–785, July 1997.

[76] A. V. Nefian, M. Khosravi, and M. H. Hayes, "A real time face dection system from uncontrolled environments," in *Visual Communications on Image Processing*, pp. 211–219, 1997.

[77] R. Herpers, G. Verghese, L. Chang, K.Darcourt, K. Derpanis, R. Enenkel, J. Kaufman, M. Jenkin, E. Milos, A. Jepson, and J. Tsotsos, "An active stereo vision system for recognition of faces and related hand gestures," in *Second International Conference on Audio- and Video-Based Biometric Person Authentication*, pp. 217–223, 1999.

[78] J.-C. Terrillon, M. David, and S. Akamatsu, "Automatic detection of human faces in natural scene images by use of a skin color model and of invariant moments," in *International Conference on Face and Gesture Recognition*, pp. 112–117, 1998.

[79] T. Darrell, G. Gordon, J. Woodfill, and M. . Harville, "A virtual mirror interface using real-time robust face tracking," in *International Conference on Face and Gesture Recognition*, pp. 616–621, 1998.

[80] "System for analyzing movement patterns in a localized zone," August 1996.

[81] R. Rao and R. Mersereau, "Merging Markov Models with deformable templates," in *Proceedings on International Conference on Image Processing*, 1995.

[82] H. Nugroho, S. Takahashi, Y. Ooi, and S. Ozawa, "Detectin human face from monocular image sequence by genetic algorithms," in *IEEE International Conference on Acoustics Speech and signal Processing*, vol. 4, pp. 2533–2536, April 1997.

[83] R. Uhl, N. d Vitoria Lobe, and Y. Kwon, "Recognizing a facial image from a police sketch," in *Second International Workshop on Applications of Computer Vision*, pp. 129–137, December 1994.

[84] X. Jia and M. S. Nixon, "Extending the feature vector for automatic face recognition," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 1167–1176, December 1995.

[85] S. Eickeler, A. Kosmala, G. Rigoll, A. Jain, S. Venkatesh, and b.C. Lovell, "Hidden Markov model based continuous online gesture recognition," in *Fouteenth International Conference on Pattern Recognition*, vol. 2, pp. 1206–1208, 1998.

[86] J. Yang, Y. Xu, and C. Chen, "Human action learning via hidden Markov model," *IEEE Transactions on Systems, Man and Cybernetics, Part A(Systems and Humans)*, vol. 27, pp. 34–44, January 1997.

[87] G. Potamianos, H. P. Graf, and E. Cosatto, "An image transform approach for HMM based automatic lipreading," in *Proceedings of the International Conference on Image Processing*, vol. 3, pp. 173–177, 1998.

[88] S. Eickeler and S. Mueller, "Content-based video indexing of tv broadcast using hidden Markov models," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 6, pp. 2997–3000, 1999.

[89] Y. He, M. Chen, and A. Kundu, "Handwritten word recognition using hidden Markov models with adaptive length viterbi algorithm," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 3, pp. 153–156, 1992.

[90] F. R. Chen, L. D. Wilcox, and D. S. Bloomberg, "Word spotting in scanned images using hidden Markov models," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 5, pp. 1–4, 1993.

[91] A. V. Nefian and M. H. Hayes, "A Hidden Markov Model for face recognition," in *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing*, vol. 5, pp. 2721–2724, 1998.

[92] A. V. Nefian and M. H. Hayes, "Face detection and recognition using Hidden Markov Models," in *International Conference on Image Processing*, vol. 1, pp. 141–145, October 1998.

[93] C. Wu, "On the convergence properties of the em algorithm," *Annals of statistics*, vol. 11, no. 1, pp. 95–103, 1983.

[94] "http://www.cam-orl.co.uk/facedatabase.html."

[95] S. Levinson, "Continuously variable duration hidden Markov models for speech analysis," in *International Conference on Acoustics, Speech and Signal Processing 86*, pp. 1241–1244, 1986.

[96] B. Juang, L. Rabiner, S. Levinson, and M. Sondhi, "Recent developments in the application of hidden Markov models to speaker independent isolated word recognition," in *International Conference on Acoustics, Speech and Signal Processing 85*, 1985.

[97] M. Russell and R. Moore, "Explicit modelling of state occupancy in Hidden Markov Models for automatic speech recognition," in *International Conference on Acoustics, Speech and Signal Processing 85*, pp. 5–8, 1985.

[98] J.Li, A. Najmi, and R. Gray, "Image classification by a two dimensional hidden Markov model," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 6, pp. 3313–3316, 1999.

[99] X. Huang, Y. Ariki, and M. Jack, *Hidden Markov Models for speech recognition.* Edinburgh University Press, 1990.

[100] A. V. Nefian and M. H. Hayes, "Face recognition using an embedded HMM," in *Proceedings of the IEEE Conference on Audio and Video-based Biometric Person Authentication*, pp. 19–24, March 1999.

[101] A. V. Nefian and M. H. Hayes, "An embedded HMM approach for face detection and recognition," in *Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing*, vol. 6, pp. 3553–3556, March 1999.

[102] R. R. Coifman and M. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Transactions on Information Theory*, vol. 38, pp. 713–718, March 1992.

[103] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate distorsion sense," *IEEE Transactions on Image Processing*, vol. 2, April 1993.

[104] M. Goodwin, "Multiresolution sinusoidal modeling using adaptive segmentation," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 3, pp. 1525–1528, 1998.

[105] S. Marchand-Maillet and B. Merialdo, "Pseudo two-dimensional hodden Markov models for face detection in colour images," in *Second International Conference on Audio- and Video-Based Biometric Person Authentication*, pp. 13–18, 1999.

[106] L. Bahl, P. Brown, P. deSouza, and L. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 49–52, 1986.

[107] S. Young, N. Russell, and J. Thornton, "Token passing: a simple conceptual model for connected speech recognition systems," Tech. Rep. 38, Cambridge University, 1989.

# Vita

I grew up in Bucharest, Romania. In 1994, I graduated from "Politehnica" University of Bucharest the Department of Electronics and Telecommunications. The graduating thesis, "A comparative analysis, between the Fast Hartley Transform and the Discrete Cosine Transform" served as a basis for my future research with Dr. Ciochina at the "Politehnica" University. In 1995, after working one year with the Electric Research Institute, the Telecommunication Department, in Bucharest, I decided to go back to school this time at Georgia Tech Lorraine , the European platform of Georgia Tech. I obtained the Master of Science in Electrical Engineering a year later and I moved to Georgia Tech in Atlanta, My Ph. D. research at the Center for Signal and Image Processing was conducted by advisor Dr. Monson Hayes. Since 1995, I joined NCR , the Human Interface Technology Center as a part-time/full-time intern being involved in several image and video analysis projects. Currently my research interests are in the general areas of digital signal and image processing, pattern recognition and statistical machine learning, statistical signal modeling, and computer vision and multimedia applications.

# A Hidden Markov Model-Based Approach for Face Detection and Recognition

Ara Nefian

128 pages
Directed by Dr. Monson H. Hayes III

The use of hidden Markov models (HMM) for faces is motivated by their partial invariance to variations in scaling and by the structure of faces. The most significant facial features of a frontal face include the hair, forehead, eyes, nose and mouth. These features occur in a natural order, from top to bottom, even if the images undergo small rotations in the image plane, and/or rotations in the plane perpendicular to the image plane. Therefore, the image of a face may be modeled using a one-dimensional HMM by assigning each of these regions to a state. The observation vectors are obtained from the DCT or KLT coefficients.

A one-dimensional HMM may be generalized, to give it the appearance of a two- dimensional structure, by allowing each state in a one-dimensional HMM to be a HMM. In this way, the HMM consists of a set of super states, along with a set of embedded states. Therefore, this is referred to as an embedded HMM. The super states may then be used to model two-dimensional data along one direction, with the embedded HMM modeling the data along the other direction.

Both the standard HMM and the embedded HMM were tested for face recognition and detection. Compared to other methods, our proposed system offers a more flexible framework for face recognition and detection, and can be used more efficiently in scale invariant systems.