FACE RECOGNITION AND VIDEO FACE TRACKING

Deepa Seetharamaiah
M.Sc. Electronics, Bangalore University, India, 2001

PROJECT

Submitted in partial satisfaction of the requirements for the degree in

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

at

CALIFORNIA STATE UNIVERSITY, SACRAMENTO

SPRING
2011
FACE RECOGNITION AND VIDEO FACE TRACKING

A Project

by

Deepa Seetharamaiah

Approved by:

__________________________________, Committee Chair
Jing Pang, Ph.D.

__________________________________, Second Reader
Suresh Vadhva, Ph.D.

Date
Student: Deepa Seetharamaiah

I certify that this student has met the requirements for format contained in the university format manual and that this project is suitable for shelving in the library and credits to be rewarded for the Project.

__________________________, Graduate Coordinator
Suresh Vadhva, Ph.D.  Date

Department of Computer Engineering
Abstract

of

FACE RECOGNITION AND VIDEO FACE TRACKING

by

Deepa Seetharamaiah

This project is about human face recognition in image files and human face detection in video streaming. Face recognition involves matching a given image with the database of images and identifying the image that it resembles the most. Face detection is a computer technology that determines the locations and sizes of human faces in digital images.

In this project, face recognition is done using two methods: (a) Eigen faces and (b) applying Principal Component Analysis (PCA) on wavelet sub-band image.

In the first face recognition method, Eigen faces technique is used. Development is done in Java.

The second method for face recognition here is to apply Principal Component Analysis (PCA) on wavelet sub-band image. Instead of using the entire image for creating the test database and sample test image, the images are decomposed into frequency sub-bands. This provides greater computational speed when the test database is huge because low resolution decomposed images are used instead of high resolution images. This is also done in Java using standard Java Graphical User Interface packages for Wavelet Transform.
For face detection, Haar-like feature based method is used here. Haar-like features are digital image features used in object recognition that encode information about the object to be detected. Java Media Framework (JMF) library is used to show the video captured from the web camera in a Java application. Java imaging libraries are used to develop the algorithm for the face detection.

The aim of this project is to successfully demonstrate the human face recognition using Eigen faces and Wavelet sub-band methods and also to detect human face in a video streaming.

__________________________, Committee Chair
Jing Pang, Ph.D.

__________________________
Date
ACKNOWLEDGMENTS

I would like to take this opportunity to thank the people behind successful completion of this project. First and foremost, I would like to thank my professor and mentor for this project, Dr. Jing Pang for her tremendous support and guidance throughout the project. I would like to thank her for always being there to help me in developing better understanding of the topic and encouraging me to give my best. Also, I am extremely grateful to my Graduate Coordinator, Dr. Suresh Vadhva for his consistent support in my academic years at the School guiding me through the entire process, right from enrolling for a class until submission of my master’s project. I would also like to thank all my professors throughout my Master’s course for giving me this opportunity to learn and grow towards attainment of this degree. Lastly, I would like to express heartfelt gratitude to my family in India who have constantly supported me and blessed me. I am extremely thankful to my husband Gururaj and my son Skanda, who helped me and supported me to make my dreams come true.
# TABLE OF CONTENTS

Acknowledgments........................................................................................................ vi

List of Tables .................................................................................................................. ix

List of Figures .................................................................................................................. x

Software Specifications .................................................................................................. xi

Chapter

1. INTRODUCTION ....................................................................................................... 1

2. FACE RECOGNITION, DETECTION AND VIDEO TRACKING TECHNIQUES. 4

3. FACE RECOGNITION USING EIGENFACE............................................................ 9
   3.1 Introduction ............................................................................................................. 9
   3.2 Design/Algorithm ................................................................................................... 10
   3.3 Program Flow ......................................................................................................... 14
   3.4 Running the Program ............................................................................................ 15

4. FACE RECOGNITION USING WAVELET SUB BANDS ...................................... 18
   4.1 Introduction ............................................................................................................. 18
   4.2 Design/Algorithm ................................................................................................... 19
   4.3 Program Flow and Running the Program............................................................... 24

5. VIDEO FACE DETECTION USING HAAR-LIKE FEATURES ......................... 28
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1 Introduction</td>
<td>28</td>
</tr>
<tr>
<td>5.2 Design/Algorithm</td>
<td>29</td>
</tr>
<tr>
<td>5.3 Implementation</td>
<td>31</td>
</tr>
<tr>
<td>5.3 Program Flow</td>
<td>32</td>
</tr>
<tr>
<td>6. CONCLUSION</td>
<td>34</td>
</tr>
<tr>
<td>References</td>
<td>36</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 1 – Face recognition results comparison................................................................. 34
LIST OF FIGURES

Figure 1 - Flowchart of Face Recognition using Eigen Faces ................................................. 12
Figure 2 – Input test image for Eigen Faces ............................................................................. 16
Figure 3 – Successful test result 1.......................................................................................... 17
Figure 4 – Successful test result 2.......................................................................................... 17
Figure 5- Original Image of resolution n X n ......................................................................... 20
Figure 6- First Level of Decomposition .................................................................................. 20
Figure 7 - Second Level of Decomposition ............................................................................. 20
Figure 8- Third Level of Decomposition ................................................................................ 21
Figure 9 – Flowchart of Face Recognition using Wavelet sub-bands ................................. 23
Figure 10 – Test image for Face Recognition using Wavelet sub-bands ............................ 25
Figure 11 – Wavelet Decomposition ....................................................................................... 26
Figure 12 – Successful test result of Face Recognition method using Wavelet sub-bands .................................................. 26
Figure 13 – Flowchart of Video Face Tracking using Haar Cascade Profile ................... 30
Figure 14 – Successful test result of Video Face Tracking method ....................................... 33
SOFTWARE SPECIFICATIONS

Different Software required to run the program are listed below.

1. Operating System - Windows XP and Windows 7

2. Java(TM) SE Development Kit 6 (JDK1.6.0) - Oracle. This is required for development.

3. Java SE 6 JRE - The Java(TM) Platform, Standard Edition Runtime Environment (JRE(TM)), Java (TM) SE Runtime Environment (build 1.6.0_24-b07). This is required for runtime.

4. Java Media Framework (JMF) 2.1.1e. This API enables audio, video and other time-based media to be added to applications and applets built on Java technology. This package, which can capture, playback, stream, and transcode multiple media formats, extends the Java 2 Platform, Standard Edition (J2SE) for multimedia developers by providing a powerful toolkit to develop scalable, cross-platform technology.

5. JAMA package 1.0.2. This is required for development. The APIs provided in the jar files are used for linear algebra computation.

Chapter 1
INTRODUCTION

Human face plays an important role in conveying individual’s identity. Despite large variations in the viewing conditions such as lighting, expressions, differences in hair style and so on, the human ability to recognize faces is remarkable [1]. Similarly, a face recognition system is a computer application for automatically identifying a person’s face from a set of digital images. Face recognition is a technique that is performed on the facial image. Comparison of the given image is done with the set of stored images to check if the face really matches.

Since face has complex features, developing a face recognition computer system is a very difficult task. Automated system has to address all the difficult steps involved in face recognition and detection. This difficulty is due to the fact that the faces must be represented in a way that best utilizes the available face information to distinguish a particular face from all other face [2]. Faces pose a particularly difficult problem in this respect because all faces are similar to one another in that they contain the same set of features such as eyes, nose, mouth arranged in roughly the same manner.

Face Detection is technique is finding a face/ the faces in a given digital image. It detects the size and position of the face. There are several applications of face-detection and a few of them are as follows - in biometrics, Human-computer interaction and Medical imaging.
Face recognition system is an important requirement in many applications such as the security systems, criminal identification and credit card verification, identity access management and access control, prevention of fraud voters, prevention of obtaining fake identification cards, online photo management sites such as Picasa and so on.

Video tracking is the process of locating a moving object/ multiple objects with respect to time. Face tracking is the process of face-detection when applied to a video image. Face tracking computer system has to track the face continuously in a moving video with respect to time.

There are many applications of Video tracking system. It is mainly used in human-computer interaction, security and surveillance, video communication and compression, augmented reality, traffic control, medical imaging and video editing.

The organizations of different Chapters are as follows. Chapter 2 describes about different techniques and methods used for face detection, face recognition and face tracking. It also briefly describes about different algorithms used for face detection, recognition and tracking.

Chapter 3 is dedicated to face recognition method using Eigen faces and PCA. It explains how the algorithm is implemented, design of the system and program flow. It also shows the results of the face recognition system using Eigen faces.

Chapter 4 explains how face recognition system using wavelet sub-band method is implemented. It explains algorithm, design, program flow and results of the face recognition system using wavelet sub-bands. It also compares face recognition using
Eigen faces and face recognition using wavelet sub-band methods.

Chapter 5 is about detecting and tracking a face in the video. This explains the algorithm and design and program flow of the video tracking system and also shows the result of the video tracking system.

Chapter 6 has the conclusion of the entire work. It also talks about future implementations.

In Chapter 1, we learnt brief introduction about Face detection, Face recognition and Video tracking. Let us learn about different techniques and algorithms in the following Chapter.
Chapter 2

FACE RECOGNITION, DETECTION AND VIDEO TRACKING TECHNIQUES

This Chapter is about different techniques that are used in recognizing the face, detecting the face and tracking the video object in general. It also briefly describes the algorithms available in the market for the above.

Several algorithms are used for face recognition. Some of the popular methods are discussed here. Face recognition by feature matching is one such method [8]. We have to locate points in the face image with high information content. We don’t have to consider the face contour or the hair. We have to concentrate on the center of the face area, as most stable and informative features are found there. The high informative points in the face are considered around eyes, nose and mouth. To enforce this we apply Gaussian weighting to the center of the face.

Linear Discriminate Analysis algorithm is also used in face recognition. Originally developed in 1936 by R.A. Fisher, Discriminant Analysis is a classic method of classification that has stood the test of time [5]. Discriminant analysis often produces models whose accuracy approaches (and occasionally exceeds) more complex modern methods [5]. LDA is similar to Principal Component Analysis (PCA). In both PCA and LDA we look for the linear combinations of the variables which explain the given data well. LDA explicitly attempts to model the difference between the classes of data [8]. PCA on the other hand does not take into account any difference in class but builds the feature combinations based on differences rather than similarities.
In a computer based face recognition system, each face is represented by a large number of pixel values. Linear discriminant analysis is primarily used here to reduce the number of features to a more manageable number before classification. Each of the new dimensions is a linear combination of pixel values, which form a template. The linear combinations obtained using Fisher's linear discriminant are called Fisher Faces [5].

Principal Component Analysis with Eigen faces is one of the popular algorithms for face recognition. Principal component analysis is a mathematical procedure that uses orthogonal co-ordinates to convert a set of observations of possibly correlated variables into a set of values of uncorrelated variables called principal components [2]. PCA is then applied on the Eigen faces. This method is explained in details in Chapter 3.

Wavelet sub-bands with PCA algorithm uses wavelets. This method decomposes the image into several smaller images. This is faster as well as accurate compared to other methods. This is studied in detail in Chapter 3. Our focus here is on two methods of face recognition that is Eigen face recognition using Principal Component Analysis (PCA) and Face recognition using wavelet sub-bands.

Face detection is a computer technology that determines the locations and sizes of human faces in digital images. Face tracking is an extension of face-detection when applied to a video image. Some of the face-detection techniques are Feature-based methods, using skin color to find face segments, detecting a blinking eye pattern in a video image and template-based matching. In Feature based technique, the content of a given region of an image is transformed into features, after which a classifier trained on example faces decides whether that particular region of the image is a face, or not.
Video tracking is the process of locating a moving object/multiple objects with respect to time. Video tracking requires gathering of large amount of data since it is dealing with the video itself. ‘Object recognition’ techniques are one of the suitable methods for tracking.

The objective of the video tracking is to relate the target objects in the sequence of video frames. Relating video frame can be difficult when the frame has higher oscillation (that is number of frames per second). Another situation that would make video tracking difficult is the change in position/direction of the object. If the object changes its position/direction too frequently, it adds complexity to video tracking. For these reasons, Video object tracking uses a motion model (like a protocol) which anticipates the change in the image of the object for a particular motion of the object. (How the image of the original target object changes when the object moves from one position to other).

Some examples of simple motion models are discussed here. Firstly, Affine Transformation: This is a 2D Transformation motion model used to track planar objects. Initial frames are all in 2D in this method. Second method is 3D transformation model. This is used when the tracking target is a 3D object [7]. The motion model defines its image depending on the 3D position and direction. One more popular method is Disruption key frame motion model. This is used in video compression method. Key frames are converted into macro blocks. Macro block is then translated by a motion vector derived from motion parameters.

Video object tracking algorithms first takes all the sequential video frames and
outputs the images of the moving target between the frames. There are number of algorithms available for video object tracking having both pros and cons. We have to consider the right algorithm depending upon the application that we intend. The important components of the video tracking systems are, Target Representation and Localization and Filtering and Data Association.

Target Representation and Localization is a bottom up process. This method gives a lot of tools to identify the moving object. Location and tracking of an object absolutely depends on the algorithms. The computational complexity of these Target Representation and Localization algorithms are low. Some of the target representation and localization algorithms are, Blob Tracking and Kernel-based tracking [7].

Since the human’s profile changes dynamically with respect to time, blob tracking is used to identify human movement in a moving video. It uses block based correlation/blob detection (visual modules that detects regions/areas that are darker than the surrounding). Kernel-based tracking is also called Mean-shift tracking. This measures the similarity of two discrete or continuous probability distributions. The probability distribution describes the range of possible values that a random variable can attain and the probability that the value of the random variable is within any measurable subset of that range. It is closely related to the Bhattacharyya coefficient which is a measure of the amount of overlap between two statistical samples or populations.

Contour tracking/ Condensation algorithm is a computer vision algorithm. The principal application is to detect and track the contour of objects moving in a cluttered environment. Object tracking is one of the more basic and a difficult aspect of computer
vision and is generally a prerequisite to object recognition. Being able to identify which pixels in an image make up the contour of an object is a non-trivial problem. Condensation is a randomized algorithm that attempts to solve this problem. It detects the object boundary.

Filtering and Data Association is a top-down process. It utilizes prior information about the scene or object, dealing with object dynamics, and evaluation of different hypotheses. It can track complex objects such as objects moving behind obstructions. Hence the computational complexity for these algorithms is higher. Some of the common Filtering algorithms are Kalman filter and Particle filter [7]. Kalman filter is an optimal recursive Bayesian filter for linear functions subjected to Gaussian noise; Particle filter is useful for sampling the underlying state-space distribution of non-linear and non-Gaussian processes.

Some of the above techniques are prone to errors due to variation in skin-colors, effect of lighting and complexity in detecting blinking eye pattern among others. In this report, the focus is on feature-based method for face detection. Haar-like features technique is discussed in chapter 5. The key advantage of a Haar-like feature over most other features is its calculation speed.

Following chapter describes Face recognition method using PCA on Eigen faces in detail.
Chapter 3

FACE RECOGNITION USING EIGENFACE

3.1 Introduction

This approach was first developed by Sirovich and Kirby (1987) and used later by Matthew Turk and Alex Pentland [1].

A dataset, such as a digital image, consists of a large number of inter-related variables. Using Principal Component Analysis, the dimensionality of the dataset is reduced while retaining as much as variation in the dataset as possible. The datasets are transformed to a new set of uncorrelated variables called the principal components. These principal components are ordered in such a way that the first few retain most of the variation present in all of the original variables.

PCA method is applied in face recognition by discriminating the image data into several classes. There will be a lot of noise in the image caused by differing lighting conditions, pose and so on. Despite these noises, there are patterns that can be observed in the image such as the presence of eyes, mouth or nose in a face and the relative distances between these objects. PCA’s aim is to extract these patterns or the features from the image.

In the domain of face recognition, the principal components (features) are referred to as Eigen faces. The original image can be reconstructed by summing up all the Eigen faces in the right proportion by adding more weights to the real features of the face.
Certain Eigen faces that do not contribute to the important face features are omitted. This is necessary because of performance issues while doing large computations. This idea is applied in our approach where Eigen faces are prepared from a set of training images and stored first. Then Eigen face is prepared for the input test image and compare with the training images. The matching image is the image having similar weights in the test database.

3.2 Design/Algorithm

Face recognition using Eigen faces approach was initially developed by Sirovich and Kirby and later used by Matthew Turk and Alex Pentland [1]. They showed that a collection of face images can be approximately reconstructed by storing a small collection of weights for each face and a small set of standard pictures [1].

Using Principal Component Analysis on a set of human face images, a set of Eigen faces can be generated. Eigen faces are a set of eigenvectors used mostly in human face recognition. Eigenvectors are a set of features that characterize the variation between face images. These eigenvectors are derived from the covariance matrix of the probability distribution of the high-dimensional vector space of faces of human beings. The main idea here is to use only the best Eigen faces that account for the major variance within the set of face images. By using lesser Eigen faces, computational efficiency and speed is achieved. The Eigen faces are the basis vectors of the Eigen face decomposition.

Below are the steps of face recognition process:

- A training set of same resolution digital images is initially prepared.
- The images are stored as a matrix with each row corresponding to an image.
Each image is represented as a vector with (r X c) elements where “r” and “c” are the number of rows and the number of columns respectively.

An average image is calculated from the individual training set images.

For each image, the deviation from the average image is then calculated and stored.

The Eigen vectors and Eigen values are then calculated. These represent the directions in which the training set images differ from the average image.

A new image is then subtracted from the average image and projected into Eigen face space.

This is compared with the projection vectors of training faces and the matching image is determined.

A face image is represented by a two dimensional N by N array of intensity values or a vector of dimension N^2. If there is an image of size 128 by 128, then that can be said as a vector of dimension 16384. Or, this is equivalent to one point in a 16384-dimensional space. A group of images then maps to a collection of points in this image space. The training images chosen are all of same dimensions. We need to find the vectors that best represent the distribution of face images within this image space and these vectors that define the sub-space of face images are termed as face space. Each vector of length N^2 represents an image of dimension N by N and is a linear combination of the original face images. These vectors are termed as Eigen faces because these are the vectors of the covariance matrix corresponding to the original face images and they have face-like appearance.
Flow chart of the Algorithm is shown in Fig.1.

Figure 1 - Flowchart of Face Recognition using Eigen Faces
Step 1: Prepare the test data

Choose “M” training face images and prepare the training set images $T_i$. The training set of face images are represented as $T_1, T_2, T_3...T_M$.

Step 2: Calculate the average of the matrix

Average face of the set $\Psi = \frac{1}{M} (T_1 + T_2 + T_3 + ... + T_M)$. [1]

$$\Psi = \frac{1}{M} \sum_{n=1}^{M} T_n$$

Step 3: Subtract the average

For each face, the difference with the average is $\Phi_i = T_i - \Psi$ [3.3]

Step 4: Calculate the covariance matrix

These vectors are then subjected to PCA which seeks a set of $M$ orthonormal vectors $u_n$ and their associated Eigen values $\lambda_k$ that best represent the distribution of the data. The vectors $u_n$ and $\lambda_k$ are the eigenvectors and Eigen values, respectively of the covariance matrix

$$C = \frac{1}{M} \sum_{n=1}^{M} \Phi_n \Phi_n^T$$

$$C = AA^T$$ where the matrix $A = [\Phi_1, \Phi_2, \Phi_3, \Phi_4,...,\Phi_M]$ [3.5]

Step 5: Calculate the eigenvectors and Eigen values of the covariance matrix.

Since the covariance matrix $C$ is of size $N^2$ by $N^2$, determining $N^2$ will be a huge task for typical image sizes. So the alternative is to determine the eigenvectors by solving a smaller $M$ by $M$ matrix and taking linear combinations of the resulting vectors. This would reduce the calculation from the order of the number of pixels in the images $N^2$ to
the order of the images in the training set $M$.

Step 6: Select the principal components.

Usually, we will use only a subset of $M$ Eigen faces, the $M'$ Eigen faces with the largest Eigen values. Eigen faces with low Eigen values are omitted, as they explain only a small part of characteristic features of the faces. This completes the training phase of the face recognition.

Step 7: Face recognition – classifying the face

The next task is the face recognition. The test image $T_{test}$ is projected into face space by the following operation:

$$\omega_k = u_k^T(T_{test} - \psi)$$

where $k = 1, 2, ..., M$ (3.6)

Weights form a vector $\Omega^T = (\omega_1, \omega_2, \omega_3, ..., \omega_M)$ describing the contribution of each Eigen face in representing the input face image.

Whichever face class provides a minimum of Euclidean distance of $\varepsilon_k = ||\Omega - \Omega_k||$, where $\Omega_k$ is vector defining $k$th face class, is the matching face image.

A threshold of $\theta$ defines the maximum allowable distance from the face space beyond which the face is considered as “unknown”.

3.3 Program Flow

This section gives the flow of the JAVA program. Java is used for the development of face recognition system using Eigen faces. The program uses
EigenFaceComputation Java class [9] that in turn uses the linear algebra methods to compute Eigen values and Eigen vectors by JAMA, a Java Matrix package [10].

We need to prepare training set first in order to run the program. Training set preparation involves following steps:

Firstly, sixteen different photographs showing the face are captured in a digital camera and uploaded to the computer. Using photo editor, the color images are converted to gray scale images. Then all the images are resized to the resolution of 320 X 240 pixels. The training images are named as 1.jpeg, 2.jpeg…..16.jpeg and copied to a folder named “EigenfacesDB” in the same folder where the program is going to be run. Also, capture some test input images that will be used for comparing against the training database. Test images are also subjected to the same steps as described in steps 2 and 3. These are then saved under the folder “TestImagesForEigenFaces”.

3.4 Running the Program

- Make sure that the TestFaceRecognition Java class is in a folder which is same as the folder where the “EigenFacesDB” is present.
- From the command prompt, go to the folder where the TestFaceRecognition java class is installed / copied and run the command “java TestFaceRecognition”. This will bring-up the screen.
- The same program is used for both methods of face recognition – using Eigen faces and using wavelet sub-bands. Wavelet sub-bands are explained later in this
document. Select the menu “Face Recognition Method” and choose the second menu item “Eigen faces” as shown.

- Select “Using Eigen Faces” menu and choose the menu item “Choose image to compare…”. This brings up the screen, where the input test images are placed.
- Choose any of the test input images and click on “Open” button. The input image is displayed on the screen.
- Go back to “Using Eigen Faces” menu and now choose the menu item “Search for matching image” as shown in Fig.2.
- In the above case, the test image is already in our database and the exact match is found and displayed on the screen as shown in Fig.3.
- We can test other images by repeating the same steps defined above.
- Fig.4 is another example of the testing. Note that the input image is slightly different than the training set image.

Figure 2 – Input test image for Eigen Faces
This concludes the discussion of Face recognition using Wavelet sub-bands. Following chapter discusses Video Face detection and Tracking in detail.
Chapter 4

FACE RECOGNITION USING WAVELET SUB BANDS

4.1 Introduction

In this Chapter we learn about the implementation of the Face Recognition System using Wavelet sub-bands.

Face recognition using PCA, though a good method still has some drawbacks. If the image resolution is too high, that will still result in a lot of computation thereby taking away a lot of CPU time and memory. Another drawback is of relatively poor discriminatory power, that is, even though the two images being compared are different, there are a lot of similarities in the features which could result in incorrect face recognition.

In Eigen Faces method PCA is done on the entire face. Because of the limitations mentioned above, PCA on wavelet sub-bands is performed and found out that this method is better for face recognition than PCA on Eigen faces. Following sections gives the better idea of the algorithm.

Eigen faces technique is extended here to apply PCA on wavelet sub-band. Using Wavelet Transform (WT), a given image is decomposed into a number of sub-bands with different frequency components. Instead of working on the original high resolution \((r \times r)\), we apply PCA on a lower resolution of \((r/n \times r/n)\) where \(r\) is the original resolution
and n is the number of times the original image is decomposed. So complexity in the computation is considerably reduced since we work on a low resolution image.

4.2 Design/Algorithm

This is an extension of the previously explained face recognition using Eigen faces. The images are transformed into lower resolution wavelet sub-bands in order to reduce computational complexity [6].

Steps involved in Wavelet sub-band method are explained here. The main idea is to decompose a given image using Wavelet Transforms (WT) into Wavelet sub-bands [6]. Using a wavelet transform, the transient elements of an image can be represented by a smaller amount of information, that is, a lower resolution image. Wavelet decomposition provides local information in both space domain and frequency domain. Resolution of the image is thereby reduced which would mean reduction in the complexity of the computation.

The following are the examples of three levels of decomposition of an image. The original image is of the resolution n X n. The 1st level of decomposition will result in images of resolution n/2 X n/2 as shown in Fig.11. The top-left image is further decomposed to get n/4 X n/4 resolution sub-bands in the 2nd level of decomposition as shown in Fig.12. In the 3rd level of decomposition, we will have images of resolution n/8 X n/8 as shown in the Fig.13.
The sub-band images have a wide range of variance whose sum is equal to that of the original image. The same steps as mentioned for Eigen Faces are followed for face recognition, but the wavelet sub-band is used instead of the original image.

Figure 5- Original Image of resolution n X n

Figure 6- First Level of Decomposition

Figure 7 - Second Level of Decomposition
Face recognition using PCA on Wavelet sub-bands has two stages namely, Training stage and Recognition stage.

Training stage has 5 sub-stages. In the first sub-stage, 3-level WT using the Daubechies wavelet D4 is applied to decompose the reference images [6]. Every single reference is then decomposed into 10 sub-bands. PCA is then applied on the 16X16 sub-band 4 images. The collection of the sub-band 4 images will form a new set. A correlation matrix will be constructed for this sub-set. We will get set of Eigen vectors and Eigen values as output from this. We have to arrange the result in the descending order. Now we need to select ‘d’ eigenvectors with the largest eigenvalues, they are used as the bases.

These $d'$ eigenvectors span a subspace $E = \text{span} \{ e_1, \ldots, e_{d'} \}[6]$.  \hspace{0.5in} (4.1)
Then, the sub-band 4 images of all reference images are represented by a linear combination of M representational bases by projecting them into the M Eigen subspace. Then training image representation is done by projecting the resultant of the sub-band images into E, Eigen sub-space. Last step is, to store the image representations and the representational bases in the library.

Recognition stage has 3 sub-stages. In the first step, an unknown image is presented to the recognition stage. This unknown image is subtracted by the mean value of the reference images and a resultant image is calculated. Then, a 3-level WT is applied to transform the resultant image, similar to training stage.

In the second step, the 16 x 16 sub-band 4 sub image is represented as a probe image representation by projecting its sub-band 4 image into the subspace \( E = \text{span}\{e_1, \ldots, e_M\} \), which is obtained in the training stage [6]. This type of representation is known as probe image representation.

Finally, the comparison between the probe image and the reference images in the library is done to find out if the input image matches with any of the images in the library. Flow chart of the algorithm is as shown in the Fig.9.
Start training set image preparation with the same resolution images

Decompose each image using Wavelet Transform into wavelet sub-bands

Store image as a matrix represented as a vector with \((r \times c)\) elements

Calculate average image from training images

Calculate and store deviation of each image from the average image

Calculate covariance matrix

Calculate Eigenvectors and Eigenvalues of the covariance matrix

Store the principal components

End training set preparation

Test Input Image

Decompose test image using Wavelet Transform into wavelet sub-bands

Subtract from the average image

Project into Eigenface space

Compare with training vectors

Determine the minimum Euclidean distance which is the matching image

End face recognition

Figure 9 – Flowchart of Face Recognition using Wavelet sub-bands
4.3 Program Flow and Running the Program

The same Java program used for face recognition using Eigen faces is used. The only additional step is the wavelet transforms process that is done before applying the PCA using Eigen faces. Running the program requires training set preparation like in Eigen faces. Preparation of training set is followed below.

1. Sixteen different photographs showing the face are captured in a digital camera and uploaded to the computer.

2. Using photo editor, the color images are converted to gray scale images.

3. All the images are resized to the resolution of 128 X 128 pixels.

4. For each training image, use the “Convert to wavelet image” sub-menu option explained later in the “Running the program” section to load an image and convert it into wavelet sub-band. Choosing that option will generate a temporary file “temp.jpg” in the work directory of the Java program and that file can be renamed.

5. The training images are named as 1.jpeg, 2.jpeg…..16.jpeg and copied to a folder named “WaveletDB” in the same folder where the program is going to be run.

6. Also, capture some test input images that will be used for comparing against the training database.

7. Test images resolution is changed to 128 X 128 and the images are then saved under the folder “TestImagesForWavelet”.
Running the program is explained in this section. For launching the program, refer to the steps explained in “Running the Program” section 3.4 of Chapter 3. Then choose, “Wavelet Sub-band” from the menu “Face Recognition method”. Select “Choose image to compare…” sub-option from the menu option “Using Wavelet Sub-bands”. This will bring up the folder containing the test images for Wavelets. Then choose a test image for comparison and we will see the screen as shown in Fig.10. Choose “Convert to wavelet image…” sub-menu option. The image is converted to a wavelet sub-band as shown in Fig.11. The middle image will be in color and the last one will convert the color image to a gray scale image. In this example, only gray images are used. Then select “Search for matching image” menu option. This will bring up the matching wavelet sub-band after comparing with the wavelet sub-band images in the training set database using Eigen faces face recognition method.

Figure 10 – Test image for Face Recognition using Wavelet sub-bands
Figure 11 – Wavelet Decomposition

Figure 12 – Successful test result of Face Recognition method using Wavelet sub-bands
Different faces with different facial expressions can be tested. Fig.12 shows the successful test of the Face recognition system using Wavelet sub-bands. Discussion on Wavelet sub-bands ends here. Next chapter discussed about Video Face Tracking, design and implementation.
Chapter 5

VIDEO FACE DETECTION USING HAAR-LIKE FEATURES

5.1 Introduction

This chapter gives the description of video face tracking implementation. Face tracking in a moving video first involves Face Detection to be performed on the video frames. For this we are using Haar-like feature [12]. Haar-like features are digital image features used in object recognition that encode information about the object to be detected. They are named as “Haar-like” features as these are similar to the computation in case of Haar wavelet transforms.

The initial step is to prepare the trained Haar cascade profiles. A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window (a rectangular window around the face, for example), sums up the pixel intensities in these regions and calculates the difference between them. This difference is then used to categorize subsections of an image, like the eye region above the cheek region in a face image that is part of a training image database containing human faces.

To detect a face (or any object) with a very good accuracy, a large number of Haar-like features are organized in a cascade to form a strong classifier. Next is the detection phase. The search window is moved across the entire input image at multiple locations and at multiple scales. For each sub-section of the image, the Haar cascade is
applied. The key advantage of a Haar-like feature over most other features is its calculation speed.

5.2 Design/Algorithm

Face Detection using Haar wavelet is explained in this section. The initial step is to prepare the trained Haar cascade profiles. Steps below are followed.

1. A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window (a rectangular window around the face, for example), sums up the pixel intensities in these regions and calculates the difference between them.

2. This difference is then used to categorize subsections of an image, like the eye region above the cheek region in a face image that is part of a training image database containing human faces.

3. To detect a face (or any object) with a very good accuracy, a large number of Haar-like features are organized in a cascade to form a strong classifier.

4. Next is the detection phase. It is explained below.

5. The search window is moved across the entire input image at multiple locations and at multiple scales.

6. For each sub-section of the image, the Haar cascade is applied.

Fig.13 shows the Flow chart of Video face tracking.
Figure 13 – Flowchart of Video Face Tracking using Haar Cascade Profile
5.3 Implementation

There are several libraries that implement Haar-like feature [12] method for object (in our case, face) detection. Open Computer Vision library or OpenCV is an open-source library of computer vision algorithms, which includes object detection code and trained Haar cascade profiles for detecting faces based on Haar classifiers. The Haar cascades generated by the OpenCV are XML files containing a tree structure and floating-point weights denoting the constants resulting from each stage of the cascade. Several files are provided with the OpenCV in the directory data/Haar cascades. File sizes in that directory run from several hundred kilobytes to a few megabytes. So performance would be affected when dealing with such huge size file.

JJIL is a Java library incorporating an image processing architecture and a significant number of well-known image processing algorithms targeted towards mobile platforms [11]. Though this library is mainly meant for mobile platforms, the same can be used for face detection from a computer with a web camera. This library converts the Haar cascade XML files to a smaller size text files.

Face Detection steps are explained in this section. Using Java Media Framework, start the video streaming from the web camera. The Java Media Framework is a Java library that enables audio, video and other time-based media to be added to Java applications and applets. This provides APIs that can capture, play, stream, and trans-code multiple media formats. Grab the image frame (a still image) from the video image
using JMF’s API. Convert the color image to 8-bit gray scale image. Use JJIL library APIs for doing the following [11]. Use a search window to move across the entire input image at multiple locations and at multiple scales. For each sub-section of the image and for each scale, the Haar cascade profile is applied. If the face is detected in the sub-section of the image, determine the rectangular coordinates for the rectangular region enclosing the face. Return the face detected rectangular area.

5.3 Program Flow

Java Media Framework (JMF) provides a platform-neutral framework for displaying time-based media. The Java Media Player APIs are designed to support most standard media content types, including MPEG-1, MPEG-2, QuickTime, AVI, WAV, AU, and MIDI.

Pre-requisites are, JMF 2.1.1e software (Oracle software) needs to be installed in the test computer; an external or in-built web camera needs to be there to capture video. Make sure that in JMF Registry (In Windows, Program Files -> Java Media Framework 2.1.1e -> JMF Registry).

Running the program involves different steps. Firstly, from the command prompt, go to the folder where the class file VideoFaceDetector is installed. Run “java VideoFaceDetector”. Then a window should come up showing the video captured by the web camera. A light colored square should enclose the face indicating that the face was detected as shown in Fig.14. Move the face towards left or right, or go back further or
move front. The square should move towards the position of the face and resize accordingly.

![Successful test result of Video Face Tracking method](image)

**Figure 14** – Successful test result of Video Face Tracking method

This Video face tracking system can be tested for different backgrounds. It can also be tested by varying the distance between the face and the camera.

Next chapter gives the conclusion of the Face Recognition and Face tracking systems.
Chapter 6

CONCLUSION

This project demonstrated the successful implementation of the following tasks:

(a) Face recognition using Eigen faces and wavelet sub-bands,

(b) Video face detection and tracking using Haar-like features.

Test results of face recognition using Eigen faces and wavelet sub-bands are shown below:

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Eigen faces</td>
</tr>
<tr>
<td>Number of images in Eigen faces database</td>
<td>16</td>
</tr>
<tr>
<td>Number of test images</td>
<td>30</td>
</tr>
<tr>
<td>Number of images successfully matched</td>
<td>28</td>
</tr>
<tr>
<td>Percentage of images successfully matched</td>
<td>93.33%</td>
</tr>
</tbody>
</table>

Table 1 – Face recognition results comparison

From the above testing results, we can see that the face recognition using Eigen faces has slightly better success rate than the wavelet sub-bands. However, this difference is not significant when compared to the computational complexity for a large number of images to be matched.
When we have a large number of images in the training database, the computational speed is expected to be faster in case of wavelet sub-bands and the success percentage difference is expected to be slightly less than the face recognition using Eigen faces.

Future enhancements for face recognition can be expanded to handle more than 16 images in the training database. Right now, adjusting the given input images or test images to the same resolution is a manual process. The program can be expanded to adjust the image resolution automatically without manual involvement.

The face detection in the live video streaming is found to accurately detect human face up to 7 feet. The video frame capturing interval is kept as 500 milliseconds. So every 500 milliseconds, the frame is captured and the face detection algorithm is run.

The video frame capturing interval can be made configurable. For example, in a computer with a very fast processing speed, we can keep the interval smaller. If it is a computer with lower processing speed, we can keep the interval higher. This parameter could be read from a file instead of hard-wired value in the program.

Another future enhancement could be to combine both video face detection and face recognition methods to not only detect a face in a video, but also recognize matching face in the test database using wavelet sub-bands.
REFERENCES


[6] G C Feng, P C Yuen and D Q Dai, “Human Face Recognition Using PCA on Wavelet Subband”, Department of Mathematics, Zhongshan University Guangzhou, China, Department of Computer Science, Hong Kong Baptist University, Kowloon, Hong Kong.


[9] Konrad Rzeszutek, Java packages for Eigen Face Recognition

[10] JAMA - product of The MathWorks and the National Institute of Standards and Technology (NIST) released to the public domain.
