


	<p>THE UNIVERSITY OF THE WEST INDIES</p>
	<p>MASc. (Engineering)</p> <p>Department of Electrical and Computer Engineering</p> <p>ECNG 6023: MASc Project</p> <p>Project Title</p> <p>Application for Image Based Classroom Attendance System</p> <p>David Akim</p> <p>807002346</p> <p>July, 22, 2016</p> <p>Project Supervisor: Dr. Akash Pooransingh</p>



THE UNIVERSITY OF THE WEST INDIES
ST. AUGUSTINE, TRINIDAD & TOBAGO, WEST INDIES
FACULTY OF ENGINEERING
Department of Electrical & Computer Engineering
B. Sc. in Electrical & Computer Engineering

CHEATING, PLAGIARISM AND COLLUSION DECLARATION FORM

For the purpose of this declaration form the following Faculty Regulations apply:

Rule 32 , *The Faculty of Engineering: Undergraduate Regulations 2008-2009*, states:

“ Cheating, Plagiarism and Collusion are serious offences under University Regulations.

- a) Cheating is any attempt to benefit one's self or another by deceit or fraud.
- b) Plagiarism is the unauthorised and/or unacknowledged use of another person's intellectual efforts and creations howsoever recorded, including whether formally published or in manuscript or in typescript or other printed or electronically presented form and includes taking passages, ideas or structures from another work or author without proper and unequivocal attribution of such source(s), using the conventions for attributions or citing used in this University. Plagiarism is a form of cheating.
- c) For the purposes of these Regulations, ‘collusion’ shall mean the unauthorized or unlawful collaboration or agreement between two or more students in the preparation, writing or production of a course assignment for examination and assessment, to the extent that they have produced the same or substantially the same paper, project report, as the case may be, as if it were their separate and individual efforts, in circumstances where they knew or had reason to know that the assignment or a part thereof was not intended to be a group project, but was rather to be the product of each student’s individual efforts. Where two or more students have produced the same or substantially the same assignment for examination and assessment in circumstances that the assignment was to be the product of each student’s individual efforts, they shall receive a failing grade in the course. ”

I DAVID AKIM, 807002346 declare that the material submitted for assessment in my MASc final project report is my own work and does not involve plagiarism and collusion.

.....

Student’s Signature with Date

Abstract

Recording the class attendance is very important in an educational system. Traditionally, the class attendance is recorded by the teacher who calls out each student's name. This can be a very time consuming process. Therefore, there is a need for an automatic class attendance system. An automatic class attendance system using face detection and face recognition was developed. An extension of the Viola and Jones algorithm was used for face detection and eye detection. The addition of eye detection significantly improved the performance of face detection. There were zero missed faces and zero falsely detected faces. However, face detection was restricted to a limited distance between the camera and the student. A database of students' faces was created for face recognition. However, the database did not contain all the students for all the courses. The Eigenfaces, Fisherfaces and LBPH methods were examined for face recognition. The Eigenfaces method was found to be the best with only one image per person in the training set and was used for face recognition. Face recognition was restricted to faces looking directly at the camera in order to reduce the number of incorrect face recognitions. However, incorrect face recognitions still exists. The class attendance system required a smartphone, a PC and a webserver. An image is captured and uploaded to the webserver using the smartphone. The attendance record is updated and retrieved using the PC. The results of this research show that the performance of the class attendance system was above average because of the deficiencies in the face recognition method. Further research must be done in finding the best face recognition method. The class attendance system was compared against existing class attendance systems and it was found that the presented system had a lower cost.

Table of Contents

Abstract	iii
Table of Contents	iv
List of Abbreviations	vii
List of Figures	viii
List of Tables	xii
Chapter 1 Introduction	1
1.1 Scope	2
1.2 Objectives.....	2
1.3 Hardware and Software Requirements.....	3
Chapter 2 Literature Review	4
2.1 Related Work.....	4
2.2 Face Detection.....	6
2.3 Face Recognition.....	9
2.4 Face Database.....	13
Chapter 3 Background	19
3.1 Viola and Jones algorithm.....	19
3.1.1 Find Haar-like Features	19
3.1.2 Find integral Image.....	20

3.1.3 Construct classifier using AdaBoost algorithm	22
3.1.4 Construct a cascade of classifiers	28
3.2 Extension of Viola and Jones Algorithm	30
3.3 Eigenfaces Method	34
3.4 Fisherfaces Method	38
3.5 Local Binary Patterns Histograms.....	42
Chapter 4 Methodology	48
4.1 Android application.....	48
4.2 Webserver.....	52
4.3 Windows Application.....	53
4.3.1 Face Detection	53
4.3.2 Face Database	59
4.3.3 Face Recognition	61
4.3.4 Class Attendance	73
Chapter 5 Testing and Results	86
5.1 Android Application.....	86
5.2 Face Detection Application.....	86
5.3 Face Recognition Application.....	94
5.3.1 Eigenfaces Method	96
5.3.2 Fisherfaces Method.....	101

5.3.3 LBPH Method.....	106
5.3.4 Face Recognition Comparisons	111
5.3.5 Class images	115
5.4 Class Attendance Application	121
Chapter 6 Discussion	127
6.1 Face Detection.....	127
6.2 Face Recognition.....	132
6.2.1 Eigenfaces Method	133
6.2.2 Fisherfaces Method.....	134
6.2.3 LBPH Method.....	136
6.2.4 Face Recognition Comparisons	137
6.2.5 Class Images	140
6.3 Class Attendance System	145
Chapter 7 Conclusion.....	149
References.....	151
Appendix.....	158
What is on the CD	158
Gant Chart	159

List of Abbreviations

API - Application Programming Interface

FLD - Fisher's Linear Discriminant

IDE - Integrated Development Environment

LDA - Linear Discriminant Analysis

LBP - Local Binary Patterns

LBPH - Local Binary Patterns Histogram

OpenCV - Open Source Computer Vision

PCA - Principal Component Analysis

SVM - Support Vector Machine

List of Figures

Figure 2.1: Sample of faces from SCface Surveillance Cameras Database (Grgic, Delac, and Grgic 2011)	14
Figure 2.2: Sample of faces from Yale Face Database (UCSD 2011).....	15
Figure 2.3: Sample of faces from AT&T Database (UC 2002)	16
Figure 2.4: Sample of faces from Labeled Faces in the Wild database (Huang et al. 2007)..	17
Figure 2.5: Example of frames from the spectrum of videos available in the Youtube Faces Database (Wolf, Hassner, and Maoz 2011)	18
Figure 3.1: Rectangle features within the detection window	20
Figure 3.2: Calculation of an Integral image at a point (x, y).....	20
Figure 3.3: Reference points for calculating the sum of all pixel values in rectangle D	22
Figure 3.4: Reference points for the rectangle features	22
Figure 3.5: Flowchart for the AdaBoost algorithm.....	24
Figure 3.6: Examples of hypotheses	25
Figure 3.7: All face images captured using strong classifier	25
Figure 3.8: Features selected by AdaBoost algorithm for face detection.....	27
Figure 3.9: Schematic of detection cascade	28
Figure 3.10: Flowchart for constructing a cascaded classifier.....	29
Figure 3.11: Examples of additional features for object detection (Lienhart, Kuranov, and Pisarevsky 2002).....	30
Figure 3.12: Reference points for calculating the sum of all the pixel values in the 45° rotated rectangle D	31
Figure 3.13: Flowchart for Gentle AdaBoost algorithm.....	32

Figure 3.14: Example of eigenfaces (Coro 2015).....	35
Figure 3.15: Flowchart for Eigenfaces method.....	37
Figure 3.16: Flowchart for Fisherfaces method.....	41
Figure 3.17: Basic LBP operator	42
Figure 3.18: Example of a circular (8, 2) neighborhood LBP descriptor (Ahonen, Hadid, and Pietikäinen 2004)	43
Figure 3.19: Face description using LBP (Pietikäinen 2010).....	45
Figure 3.20: Flowchart for Local Binary Patterns Histograms method.....	47
Figure 4.1: Basic procedure for using the ECNG Class Attendance System	48
Figure 4.2: Layout of Android application	50
Figure 4.3: Flowchart for Android application	51
Figure 4.4: Layout of face detection application	54
Figure 4.5: Flowchart for face detection application	57
Figure 4.6: Flowchart for improved face detection application.....	59
Figure 4.7: Sample of faces in ECNG face database	61
Figure 4.8: Layout of face recognition application.....	63
Figure 4.9: Flowchart for face recognition application	67
Figure 4.10: Histogram of minimum distances for Eigenfaces method	69
Figure 4.11: Histogram of minimum distances for Fisherfaces method.....	70
Figure 4.12: Histogram of minimum distances for LBPH method.....	71
Figure 4.13: Layout of class attendance application main window	74
Figure 4.14: Flowchart for class attendance application main window.....	75
Figure 4.15: Flowchart for updating attendance records	78

Figure 4.16: Flowchart for getting the class attendance	79
Figure 4.17: Flowchart for getting the student attendance	81
Figure 4.18: Flowchart for getting student information.....	82
Figure 4.19: Flowchart for getting attendance offenders.....	84
Figure 4.20: Flowchart for getting the attendance record table	85
Figure 5.1: Setup for testing face detection algorithm for a constant distance.....	88
Figure 5.2: Image of students for ECNG 3034 on 2016-04-04	115
Figure 5.3: Image of student for ECNG 6706 on 2016-04-06.....	116
Figure 5.4: Image 1 of students for ECNG 7000 on 2016-04-08	116
Figure 5.5: Image 2 of student for ECNG 7000 on 2016-04-08.....	117
Figure 5.6: Image of students for ECNG 7000 on 2016-04-15	118
Figure 5.7: Image 1 of students for ECNG 7000 on 2016-04-22	118
Figure 5.8: Image 2 of students for ECNG 7000 on 2016-04-22	119
Figure 5.9: Image 3 of student for ECNG 7000 on 2016-04-22.....	119
Figure 5.10: Image 1 of student for ECNG 7000 on 2016-04-29.....	120
Figure 5.11: Image 2 of student for ECNG 7000 on 2016-04-29.....	120
Figure 5.12: Image 3 of student for ECNG 7000 on 2016-04-29.....	120
Figure 5.13: Class attendance for ECNG 7000 between 2016-04-01 and 2016-06-06	124
Figure 5.14: Student attendance for ECNG 7000 between 2016-04-01 and 2016-06-06 with a 50% attendance requirement.....	125
Figure 5.15: Attendances table for ECNG 7000 between 2016-04-01 and 2016-06-06	125
Figure 5.16: Attendance Offenders for ECNG 7000 between 2016-04-01 and 2016-06-06 with a 50% attendance requirement.....	126

Figure 6.1: Instance where false detections occurred	130
Figure 6.2: Examples of incorrect recognitions	143

List of Tables

Table 5.1: Android application tests and results.....	86
Table 5.2: Face Detection Test Conditions.....	87
Table 5.3: Hit, Miss and False Positive Rates for all the tests using 13 MP camera.....	89
Table 5.4: Hit, Miss and False Positive Rates for distance tests using 8 MP camera.....	90
Table 5.5: Hit, Miss and False Positive Rates for distance tests using 5 MP camera.....	90
Table 5.6: Hit, Miss and False Positive Rates for all the tests after re-adjusting the parameters using 13 MP camera.....	91
Table 5.7: Hit, Miss and False Positive Rates for distance tests after re-adjusting the parameters using 8 MP camera	91
Table 5.8: Hit, Miss and False Positive Rates for distance tests after re-adjusting the parameters using 5 MP camera	92
Table 5.9: Hit, Miss and False Positive Rates for all the tests with the aid of eye detection using 13 MP camera.....	92
Table 5.10: Hit, Miss and False Positive Rates for distance tests with the aid of eye detection using 8 MP camera.....	93
Table 5.11: Hit, Miss and False Positive Rates for distance tests with the aid of eye detection using 5 MP camera.....	93
Table 5.12: Face recognition results for the Eigenfaces method using 13 MP camera and full training set.....	96
Table 5.13: Face recognition results for the Eigenfaces method using 8 MP camera and full training set.....	96

Table 5.14: Face recognition results for the Eigenfaces method using 5 MP camera and full training set.....	97
Table 5.15: Face recognition results for the Eigenfaces method using 13 MP camera and training set without side views.....	97
Table 5.16: Face recognition results for the Eigenfaces method using 8 MP camera and training set without side views.....	98
Table 5.17: Face recognition results for the Eigenfaces method using 5 MP camera and training set without side views.....	98
Table 5.18: Face recognition results for the Eigenfaces method using 13 MP camera and training set with frontal views.....	99
Table 5.19: Face recognition results for the Eigenfaces method using 8 MP camera and training set with frontal views.....	99
Table 5.20: Face recognition results for the Eigenfaces method using 5 MP camera and training set with frontal views.....	100
Table 5.21: Face recognition results for the Fisherfaces method using 13 MP camera and full training set.....	101
Table 5.22: Face recognition results for the Fisherfaces method using 8 MP camera and full training set.....	101
Table 5.23: Face recognition results for the Fisherfaces method using 5 MP camera and full training set.....	102
Table 5.24: Face recognition results for the Fisherfaces method using 13 MP camera and training set without side views.....	102

Table 5.25: Face recognition results for the Fisherfaces method using 8 MP camera and training set without side views.....	103
Table 5.26: Face recognition results for the Fisherfaces method using 5 MP camera and training set without side views.....	103
Table 5.27: Face recognition results for the Fisherfaces method using 13 MP camera and training set with frontal views.....	104
Table 5.28: Face recognition results for the Fisherfaces method using 8 MP camera and training set with frontal views.....	104
Table 5.29: Face recognition results for the Fisherfaces method using 5 MP camera and training set with frontal views.....	105
Table 5.30: Face recognition results for the LBPH method using 13 MP camera and full training set.....	106
Table 5.31: Face recognition results for the LBPH method using 8 MP camera and full training set.....	106
Table 5.32: Face recognition results for the LBPH method using 5 MP camera and full training set.....	107
Table 5.33: Face recognition results for the LBPH method using 13 MP camera and training set without side views.	107
Table 5.34: Face recognition results for the LBPH method using 8 MP camera and training set without side views.	108
Table 5.35: Face recognition results for the LBPH method using 5 MP camera and training set without side views.	108

Table 5.36: Face recognition results for the LBPH method using 13 MP camera and training set with frontal views.	109
Table 5.37: Face recognition results for the LBPH method using 8 MP camera and training set with frontal views.	109
Table 5.38: Face recognition results for the LBPH method using 5 MP camera and training set with frontal views.	110
Table 5.39: Eigenfaces method performance results for various camera resolutions and full training set.	111
Table 5.40: Eigenfaces method performance results for various camera resolutions and training set without side views.	111
Table 5.41: Eigenfaces method performance results for various camera resolutions and training set with frontal views.	111
Table 5.42: Fisherfaces method performance results for various camera resolutions and full training set.	112
Table 5.43: Fisherfaces method performance results for various camera resolutions and training set without side views.	112
Table 5.44: Fisherfaces method performance results for various camera resolutions and training set with frontal views.	112
Table 5.45: LBPH method performance results for various camera resolutions and full training set.	113
Table 5.46: LBPH method performance results for various camera resolutions and training set without side views.	113

Table 5.47: LBPH method performance results for various camera resolutions and training set with frontal views.	113
Table 5.48: Face recognition performance results for all face recognition methods and full training set.	114
Table 5.49: Face recognition performance results for all face recognition methods and training set without side views.	114
Table 5.50: Face recognition performance results for all face recognition methods and training set with frontal views.	114
Table 5.51: Face recognition results for all methods using class images for all courses	117
Table 5.52: Face recognition results for all methods using class image for ECNG 7000 on 2016-04-15	118
Table 5.53: Face recognition results for all methods using class images for ECNG 7000 on 2016-04-22	119
Table 5.54: Face recognition results for all methods using class images for ECNG 7000 on 2016-04-29	120
Table 5.55: Class attendance for all courses and dates	122
Table 5.56: Student attendance for all courses and dates	122
Table 5.57: Detailed attendances for all courses and dates	123

Chapter 1 Introduction

Having a high class attendance is very important in improving the quality of education. Therefore it is very crucial that the class attendance be recorded for each class session, in order to determine if there was a high attendance. Additionally this information can be used to account for the students' performances. In some institutions, there is a class attendance requirement. For example, at the University of the West Indies, students are required to have a minimum of 75% class attendance (UWI 2014).

There are many methods of identification such as iris scans, fingerprint scans and face recognition. Although iris and fingerprint scans can be more accurate than face recognition, they are very intrusive (Kar et al. 2012). Face recognition is gradually becoming a universal biometric solution because minimal effort is required from the user (Kar et al. 2012). Face recognition can be used in authorization systems, visitor management systems, class attendance systems and employee management systems.

Traditionally, the attendance of students is recorded by the teacher who calls out the student's name or checks the student's identification card (Shehu and Dika 2010). A considerable amount of time is taken away from teaching using this traditional method. Alternatively, the attendance of students can be recorded manually by using an attendance sheet given out by the lecturer or teaching assistant (Kar et al. 2012). This is a very time consuming process, especially with a large number of students. Additionally this method is very intrusive since it requires every student to sign the attendance sheet, which will disrupt their attention in class. Furthermore, it is very difficult to verify if each student did sign the attendance sheet in a large

class or if the students correctly signed in their respective entries. The attendance sheet is subjected to damage and loss during the transition between different teaching staff. Hence there is a need for a faster and efficient method of recording class attendance.

This research explores the possible techniques for an automatic class attendance system. It attempts to demonstrate the use of face detection and face recognition for the automatic class attendance system.

1.1 Scope

The automatic class attendance system will be restricted to using only static images. Live or pre-recorded video streams will not be used. Furthermore, the environment for collecting data and testing the system will be restricted to the classrooms in Engineering Block 1 and 13 in UWI, St. Augustine.

1.2 Objectives

The objectives of this project were to:

1. Acquire image data from classroom environment
2. Implement face detection and recognition algorithm to detect and identify students in a classroom from acquired image
3. Report detected/recognized/unrecognized faces and class statistics
4. Implement a database of faces of students in a course
5. Test the accuracy of the method used
6. Compare with current class attendance methods

1.3 Hardware and Software Requirements

Hardware requirement:

- 1 PC
- 1 Mobile smart phone

Software requirement:

- Android Studio
- Microsoft Visual Studio 2015
- XAMPP

Chapter 2 Literature Review

2.1 Related Work

Biometrics is widely used in identification and access control systems. Individuals are identified based on their biological and behavioral characteristics. Systems which use biometrics such as iris, retina or fingerprint recognition have better accuracy than face recognition (Kar et al. 2012). However, they can be intrusive since the users have to provide an input, for example, their fingerprint. Furthermore, the devices which capture biometric information can be costly and would require the institution to collect biometric information from all students which would introduce privacy concerns. Additionally, these devices are subjected to physical damage from users.

Shehu and Dika (2010) proposed an automatic attendance management system which uses computer vision and face recognition algorithms. These algorithms were integrated with an existing Learning Management System (LMS). This system requires the installation of a rotating camera at the front and center of the classroom which captures the classroom photo that can then be used to identify the faces of the students. The camera would continuously take photos at regular intervals until all faces are identified or until the system is told to stop. The HAAR classifier was used for face detection and the Eigenfaces algorithm was used for face recognition. This system has some limitations which include every classroom having internet access, an installed camera and a computer. An experiment was performed using 147 students divided into ten groups. The performance of the face detection algorithm was good. However, the performance of the face recognition algorithm was poor. The poor performance was due to

old images of students being used for face recognition. A majority of the older students would have a different facial appearance compared to when they were first registered at the institution.

Kar et al. (2012) proposed a method for a student's attendance system which incorporates face recognition technology using the Principal Component Analysis (PCA) algorithm. This system automatically records the attendance of students in a classroom and provides the ability to access the attendance records. Open source computer vision library (OpenCV) and Light Tool Kit (FLTK) were the main components in the implementation. OpenCV provides many functions which make building sophisticated vision applications simple and fast. After the image is captured, the frontal face is extracted using the Viola Jones algorithm. It is then converted to gray scale and stored. The PCA algorithm is used for face recognition. PCA is commonly used to find patterns in data. The data is expressed as eigenvectors in order to find the similarities and differences between dissimilar data (Zhang and Liu 2002). Whenever a new face is added to the database, the PCA algorithm is performed on the training set and the results are saved. An experiment was performed using 30 different images of 10 persons for the training set. The results showed that as the angle of the face orientation increases the face detection and recognition rates decreased. With no angle of orientation, the face detection and recognition rates were excellent.

Patil and Shukla (2014) proposed a method for classroom attendance system using face detection and face recognition. This system uses the Viola Jones algorithm for face detection and a combination of PCA and LDA (Linear discriminant analysis) for face recognition. Histogram equalization, noise filtering and skin classification were performed on the captured images in order to improve the face detection rate. This system is comprised of a camera connected to a Raspberry pi module, a database for storing face images and another database

for storing information about student and teachers. The camera is installed at the front side of the class and it continuously takes photos until all the students are identified. There were no results from this proposed strategy by Patil and Shukla (2014).

Samlal (2013) proposed an automatic attendance management system using image processing and a mobile application. This system is comprised of a cloud application which handles face detection and face recognition, an Android application which gathers information from the classroom and a database storage for storing students' photos and attendance records. A smartphone with a camera was used to capture photos of the class. This system is less costly compared to the other proposed systems. The Viola Jones algorithm was used for face detection and the Eigenfaces algorithm was used for face recognition. These were implemented on the Azure Cloud and retrieved using Mobile Web Services. An experiment was performed using a picture which contained 33 faces and another picture which contained 14 faces. The performance of the face detection and face recognition methods were evaluated by matching the faces in both pictures to the students in the database. The results showed that the performance of the face detection algorithm was excellent. However, the performance for the face recognition algorithm was exceptional only for a small number of students. As the number of students in the database increased, the performance decreased.

2.2 Face Detection

Before face recognition can take place, face detection must first be performed. This is the process of differentiating the facial regions from the background image. King (2003) performed a survey on six methods of face detection based on the work done by Osuna, Freund,

and Girosi (1997), Romdhani et al. (2001), Rowley, Baluja, and Kanade (1998), Schneiderman and Kanade (2000), Kah-Kay and Poggio (1998), and Viola and Jones (2001).

Osuna, Freund, and Girosi (1997) proposed a method based on support vector machines (SVMs). SVM is a learning technique created by Vapnik (1995) which can be regarded as a new method for training polynomial, neural network or Radial Basis Functions classifiers. By solving a linearly constrained quadratic programming problem the decision surfaces can be found. This is a difficult optimization problem since the quadratic form is completely dense and the memory requirements increase with square of the number of data points. Osuna, Freund, and Girosi (1997) presented a decomposition algorithm which iteratively solves sub-problems and evaluates optimality conditions which can then be used to create better iterative values and the stopping criteria for the algorithm.

Romdhani et al. (2001) improved on that method by producing even smaller training vector sets. This was accomplished by calculating a set of reduced set vectors from the support vectors. The reduced set vectors are considered sequentially and when a face is not detected, the evaluation is stopped. This eliminated the need to evaluate all the reduced set vectors, which reduces the computation time.

Rowley, Baluja, and Kanade (1998) presented a technique using neural network-based filters. This technique works only for upright frontal views of faces. An image is divided into small windows which are examined by a neural network. The neural network determines if each window contains a face. Arbitration between multiple networks is performed in order to improve the performance over a single network. A bootstrap algorithm was used to collect negative face examples for training. Negative face examples are required so that false

detections will be added to the training set. For the positive face examples, a straightforward method was used.

Schneiderman and Kanade (2000) proposed a statistical method for 3D object detection applied to faces and cars. The statistics of “object” and “non-object” appearances were represented using a product of histograms. The joint statistics of a subset of wavelet coefficients and their position on the object is represented by each histogram. A wide range of visual attributes can be represented using these histograms.

Kah-Kay and Poggio (1998) presented an example-based learning approach for face detection. The distribution of human face patterns is modeled by using a few view-based “face” and “nonface” model clusters. At each location in the image, a difference feature vector is calculated between the distribution-based model and the local image pattern. Using the difference feature vector, a trained classifier determines if a human face was detected at the current location in the image.

Viola and Jones (2001) presented a method for robust real time object detection. There are three important concepts introduced in this method. The first concept is an “integral image” which is a new image representation. This enables fast computation of the features used by the proposed detector. The second concept is a learning algorithm based on AdaBoost. This allows for the selection of a small number of important visual features and the generation of efficient classifiers. The third concept is the combination of classifiers in a “cascade”. This allows for the quick elimination of background regions in the image and additional computation for object regions.

King (2003) found that the method presented by Viola and Jones (2001) had the best results in terms of computation time. Additionally, the results in terms of the error rates were excellent.

2.3 Face Recognition

Face recognition has been used mainly for verification and identification (Jafri and Arabnia 2009). In the verification process, the identity of an individual is validated by comparing the image of that individual with an image of the person that the individual claimed to be. In the identification process, the identity of an individual is determined by comparing the image of that individual with a database of images of registered individuals. The method of obtaining face images is determined by the application. For example, applications where static images are required are best served by capturing face images using a standard camera. Jafri and Arabnia (2009) classified face recognition techniques into three categories: methods that operate on video sequences, those that require sensory data such as 3D information and those that operate on intensity images. The scope of this research is limited to methods that operate on intensity images. Therefore only these methods will be examined. Jafri and Arabnia (2009) found that face recognition methods for intensity images can either be feature-based or holistic.

Feature-based methods identify and extract the distinctive facial features in the input image such as the eyes, nose and mouth. The geometric relationships among the facial points are then calculated. This process converts the face in the input image to a vector of geometric features. In order to match faces, standard statistical pattern recognition methods are used along with these measurements. Jafri and Arabnia (2009) investigated feature-based methods for face recognition. Some of methods found were the work done by Kanade (1973), Brunelli and Poggio (1993).

Kanade (1973) used image processing techniques to extract a vector containing 16 facial parameters, which include angles and areas. A simple Euclidean distance measure was used for matching faces. This approach had a maximum performance of 75% using a database of 20 individuals with 2 images for each individual.

Brunelli and Poggio (1993) proposed two methods of face recognition. One of the methods is based on almost-grey-level template matching and the other method is based on computing a set of geometrical features such as chin shape and mouth position. An experiment was performed using a database of 47 individuals (21 females and 26 males) with 4 images for each individual. The geometrical features approach had a correct recognition rate of 90%. However, the template matching approach had a correct recognition rate of 100%.

The main advantage of feature-based methods is that they are robust to position changes in the input image (Jebara 1996). Another advantage is that they are unaffected by orientation, size and lighting (Cox, Ghosn, and Yianilos 1996). The main disadvantage of these methods is the difficulty of automatic feature detection. The decision on which feature to detect is very significant (Cendrillon 1999).

Holistic methods identify faces using global representations. They use descriptions based on the whole image instead of local features of the face. These approaches can either be statistical or AI (Jafri and Arabnia 2009). In statistical methods, the image is represented as a 2D array of intensity values. Face recognition is accomplished by comparing the correlation between the input face and all the faces stored in the database. Some holistic statistical methods include Principal Component Analysis, Eigenfaces and Fisherfaces. In AI methods, tools such as

machine learning techniques and neural networks are used in face recognition. One example of the AI method is the work done by Zhang et al. (2004).

Sirovich and Kirby (1987) presented a method which uses Principal Component Analysis to represent face images. In this method a specific face is represented along the eigenpictures coordinate space. By utilizing only a small collection of eigenpictures and corresponding projections along each eigenpicture, any face can be recreated.

Turk and Pentland (1991) presented a method which uses the projections along eigenpictures as classification features for face recognition. In this face recognition method, eigenfaces are created. These eigenfaces correspond to the eigenvectors which are associated with the dominant eigenvalues of the known face covariance matrix. In order to accomplish face recognition, a comparison is made between their projections along the eigenfaces and those of the known face images. The dimensionality of the original space is reduced by a feature space that is defined by the eigenfaces. Face recognition is performed in this reduced space. An experiment was performed using a database of 2500 images of 16 individuals. Those individuals were subjected to different light conditions, head orientations and head sizes. Under those conditions the face recognition rates were found to be 96%, 85% and 64% respectively.

Belhumeur, Hespanha, and Kriegman (1997) proposed using Fisher's Linear Discriminant (FLD) analysis done by Fisher (1936) for face recognition. In FLD analysis, the ratio of the between-class scatter and the within-class scatter is maximized. Therefore it is supposedly better than PCA for classification. Belhumeur, Hespanha, and Kriegman (1997) developed a method called Fisherfaces. In this method, subspace projection is used just before LDA

projection, which prevents the within-class scatter matrix from degenerating. An experiment was performed using 5 individuals, with 66 images per person. The result showed that this method can handle different facial expressions and lighting conditions better than PCA. It should be noted that the work done by Martinez and Kak (2001) showed that PCA can perform better than LDA when the training data set is small. Additionally PCA is more tolerant to different training sets.

Ahonen, Hadid, and Pietikäinen (2004) presented a face recognition method using local binary patterns histograms (LBPH). In this method the face is partitioned into small regions. The local binary patterns (LBP) are obtained from the regions and then concatenated into a single feature histogram which represents the face. Recognition is achieved using a nearest neighbor classifier in the calculated feature space. Chi square is used as a dissimilarity measure for comparing two faces. This method was tested against the Elastic Bunch Graph Matching, PCA and Bayesian Intra/extrapersonal Classifier methods using the FERET database. The results showed that the method using LBP outperformed the other methods.

The main advantage of holistic methods is that no information in the image is destroyed, since these methods do not focus on limited regions (Jebara 1996). However, these methods are computationally expensive. They also require a high degree of correlation between training and test images. Additionally they do not perform well under significant changes in scale, pose and illumination (Beumier and Acheroy 2000).

2.4 Face Database

In order to evaluate the performance of any face recognition algorithm, a face database must be used. There are many face databases available. An appropriate face database can be selected based on the application, for example, the individuals are subjected to varying facial expressions. Selection can also be done based on the property of the face recognition algorithm to be tested, for example, how the algorithm performs in the presence of varying lighting conditions (Grgic and Delac 2005). Some of the face databases used for face recognition are:

- SCface Surveillance Cameras Database
- Yale Face Database
- AT&T “The Database of Faces” Database
- Labeled Faces in the Wild Database
- YouTube Faces Database

The SCface Surveillance Cameras Database by Grgic, Delac, and Grgic (2011) contains static images of human faces. There are 4160 static images of 130 subjects which provides 32 images per subject. The subjects were placed under surveillance conditions. Hence this database can be used to test a face recognition algorithm's robustness under such conditions. The subjects underwent 9 different poses and had a frontal mugshot taken. The images in this database were captured using surveillance cameras of varying qualities, under uncontrolled illumination conditions and from various distances. Images in the visible and infrared spectrum were captured. Figure 2.1 illustrates a sample of faces from this database.

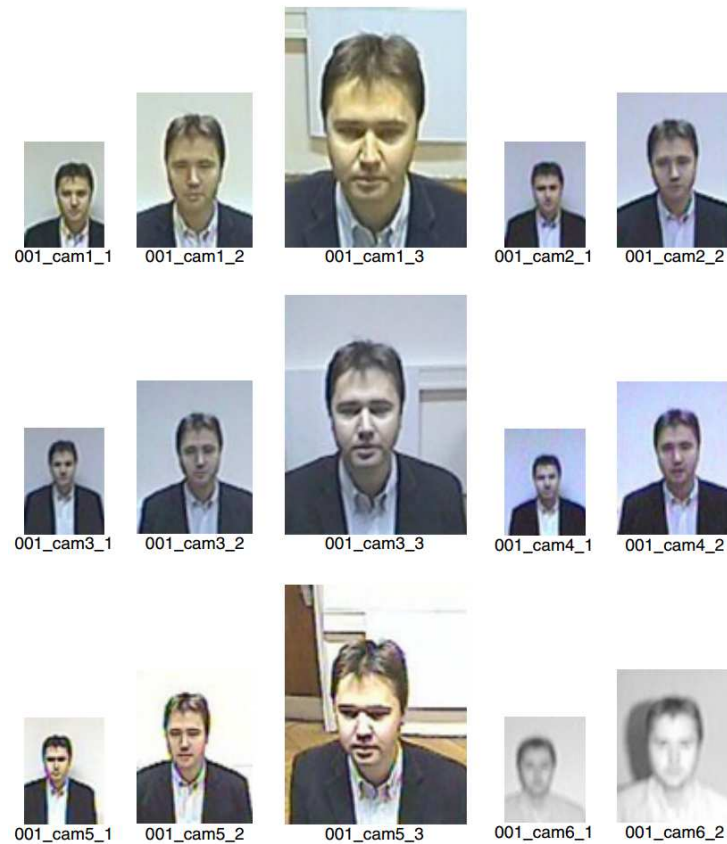


Figure 2.1: Sample of faces from SCface Surveillance Cameras Database (Grgic, Delac, and Grgic 2011)

The Yale Face Database by Yale University, (website created by Georgiades (1997)) contains 165 images of 15 subjects which provides 11 images per subject. The images are in grayscale and GIF format. The subjects underwent different configurations which includes normal, happy, sad, surprised, sleepy, wink, center-light, left-light, right-light, with glasses and without glasses. It should be noted that the database is no longer available at the website created by Georgiades (1997). However, the database can be obtained at UCSD (2011). Figure 2.2 illustrates a sample of faces from this database.

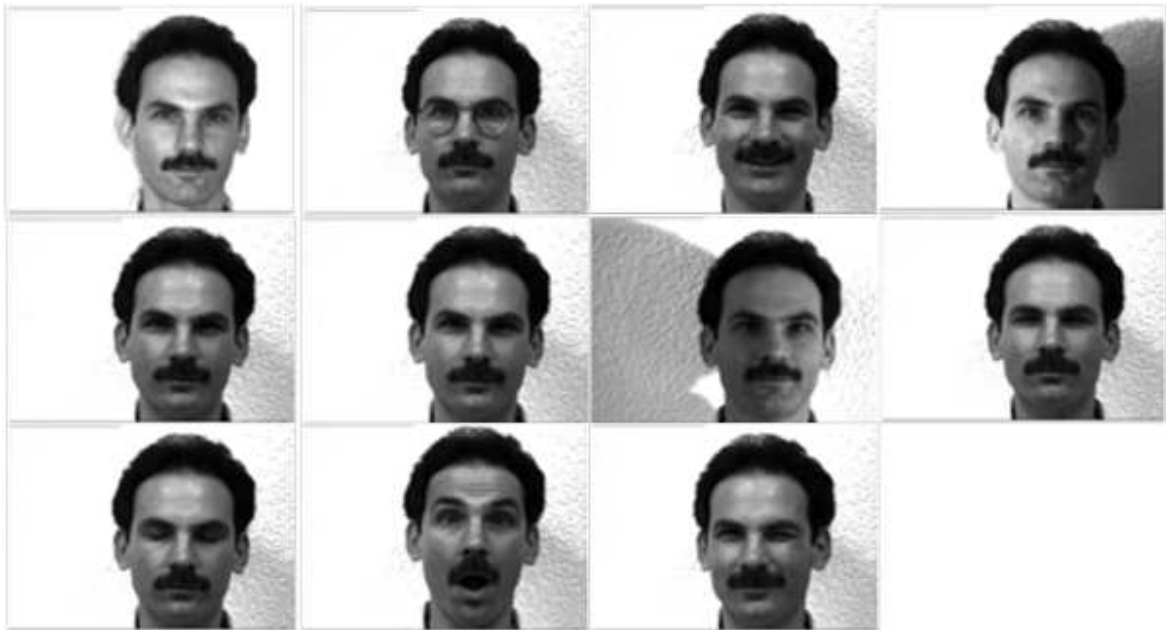


Figure 2.2: Sample of faces from Yale Face Database (UCSD 2011)

The AT&T “The Database of Faces” Database by AT&T Laboratories Cambridge UC (2002) contains the images of 40 different subjects, with each subject having 10 images. The subjects underwent different configurations which includes not smiling, smiling, open eyes, closed eyes, with glasses and without glasses. They were also in an upright, frontal position. Tolerance for some side movement was allowed. The images were taken at different times, against a dark homogenous background under varying lighting conditions. The images are in grayscale, have a size of 92×112 pixels and are stored in PGM format. Figure 2.3 illustrates a sample of faces from this database.



Figure 2.3: Sample of faces from AT&T Database (UC 2002)

The Labeled Faces in the Wild Database by Huang et al. (2007) contains 13233 images of 5749 subjects, from which 1680 subjects have two or more images and the remaining 4069 subjects have one image. All images were collected from the internet. Most databases have been created under controlled conditions for example, illumination, position and expression. However, this database was created under uncontrolled conditions. It exhibits “natural” changes in lighting, resolution, pose, facial expression, race, gender, age, photographic quality, occlusions and background. The extracted faces were acquired using the Viola-Jones face detector. The resulting images have a size of 250×250 pixels and are in JPEG format. All false positive face detections and unidentified subjects were manually removed. Most images in the database are in color, but some are in grayscale. Figure 2.4 illustrates a sample of faces from this database.



Figure 2.4: Sample of faces from Labeled Faces in the Wild database (Huang et al. 2007)

The YouTube Faces Database by Wolf, Hassner, and Maoz (2011) contains 3425 videos of 1595 subjects. All videos were obtained from YouTube. The distribution of videos per subject is not equal. There is an average of 2.15 videos for each subject. The number of frames in a video varied from 48 frames to 6070 frames with an average of 181.3 frames. Videos provide more information than single images and many video face recognition methods were developed using controlled footage or high quality videos. However, many videos found on the internet are produced by amateurs, under various uncontrolled conditions. Additionally, due to storage and bandwidth restrictions, the compression applied to videos may further deteriorate the quality of the videos. This database can be used to evaluate the performance of a video face recognition under uncontrolled conditions. Figure 2.5 illustrates a sample of faces from this database.



Figure 2.5: Example of frames from the spectrum of videos available in the Youtube Faces Database (Wolf, Hassner, and Maoz 2011)

Chapter 3 Background

3.1 Viola and Jones algorithm

The method presented by Viola and Jones (2001) will be explored in this research, since the work done by King (2003) showed that this method had the best performance in terms of computation time and error rate. This algorithm contains the following steps:

1. Find Haar-like features
2. Find integral image
3. Construct classifier using AdaBoost algorithm
4. Construct a cascade of classifiers

3.1.1 Find Haar-like Features

The Viola and Jones (2001) method detects faces using features. Features were utilized because the operation of a feature-based system is much faster than a pixel-based system. Furthermore, features are able to encode ad-hoc domain knowledge which can difficult to learn using a limited amount of training data. Three types of features were used:

- Two-rectangle feature
- Three-rectangle feature
- Four-rectangle feature

Figure 3.1 illustrates the rectangle features within the detection window. A two-rectangle feature is shown in windows A and B. A three-rectangle feature is shown in window C and a four-rectangle feature is shown in window D. The value of any rectangle feature is the

difference between the sum of the pixel values in the black region and the sum of the pixel values in the white region. The base size of the detection window is 24×24 . Considering all possible parameters of the features such as type, position and scale, the exhaustive set of rectangle features is over 45,000.

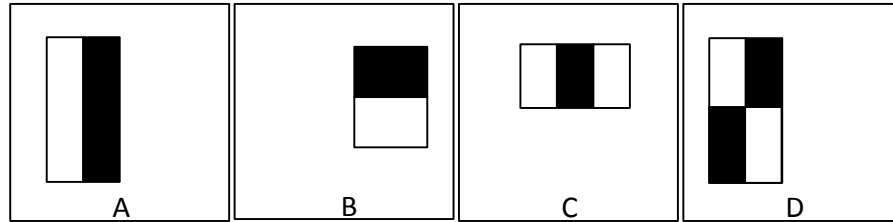


Figure 3.1: Rectangle features within the detection window

3.1.2 Find integral Image

The computational speed of the rectangle features can be greatly increased by using an intermediate representation for the image developed by Viola and Jones (2001) called the integral image.

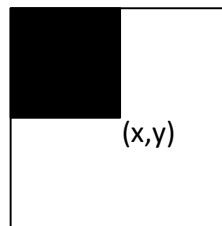


Figure 3.2: Calculation of an Integral image at a point (x, y)

Figure 3.2 illustrates the calculation of an integral image at a point (x, y) . The integral image at a point (x, y) is calculated by finding the sum of all the pixel values above and to the left of the point (x, y) inclusive. Viola and Jones (2001) represented it as:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \quad (3.1)$$

where $i(x, y)$ is the original image and $ii(x, y)$ is the integral image. According to Viola and Jones (2001), the integral image can be calculated in a single pass using the following pair of recurrences:

$$s(x, y) = s(x, y - 1) + i(x, y) \quad (3.2)$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \quad (3.3)$$

where $s(x, y)$ is the cumulative row sum, $ii(-1, y) = 0$ and $s(x, -1) = 0$.

Using the integral image, the sum of all the pixel values within any rectangle can be calculated from the values at four reference points. Figure 3.3 illustrates the reference points for calculating the sum of all pixels in rectangle D. At point 1, the value of the integral image is the sum of all the pixel values in rectangle A. The value at point 2 is the sum of all the pixel values in rectangles A and B. The value at point 3 is the sum of all the pixel values in rectangles A and C. At point 4, the value of the integral image is the sum of all the pixel values in rectangles A, B, C and D. In order to find the sum of all the pixel values in rectangle D, the sum of the values at points 2 and 3 is subtracted from the sum of the values at points 4 and 1. Similarly, to find the sum of all the pixel values in rectangle C, the value at point 1 is subtracted from the value at point 3. The sum of all the pixel values in rectangle B is calculated by subtracting the value at point 1 from the value at point 2.

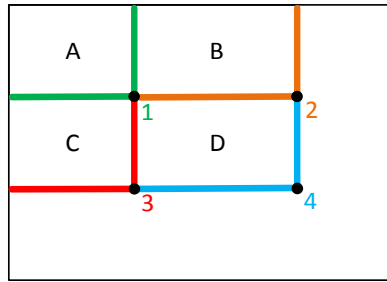


Figure 3.3: Reference points for calculating the sum of all pixel values in rectangle D

The difference between any two rectangular sums can be found using eight reference points. However, the rectangle features discussed before only involve adjacent rectangles. Therefore, the two-rectangle feature can be calculated using 6 reference points. The three-rectangle feature can be calculated using 8 reference points and the four-rectangle feature can be calculated using 9 reference points. As a result, the computational speed of the rectangle features is faster. Figure 3.4 illustrates the reference points for the rectangle features.

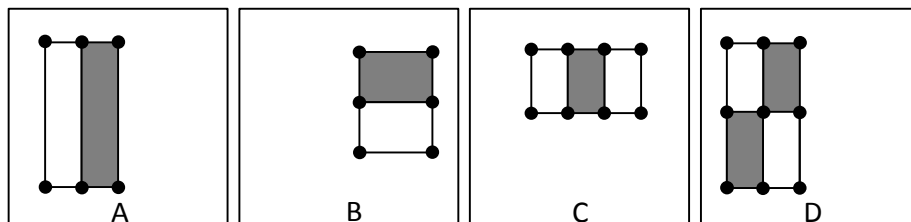


Figure 3.4: Reference points for the rectangle features

3.1.3 Construct classifier using AdaBoost algorithm

While the computation of the features is faster using the integral image, the number of features is still large. In order to choose the important features among all the possible features generated, the AdaBoost algorithm by Freund and Schapire (1997) was used. Viola and Jones (2001) applied this algorithm to face detection. The classification performance of a simple learning algorithm is boosted using the AdaBoost algorithm. This is accomplished by

combining a group of weak classifiers to form a stronger classifier. In the context of boosting, a simple learning algorithm is called a weak classifier. A weak classifier is one that detects faces correctly 51% of the time at its best performance. It uses a single rectangle feature. The weak classifier is boosted through an iterative process. After each iteration, the example images are re-weighted to highlight the example images that were incorrectly classified by the preceding weak classifier. The final strong classifier is a weighted combination of weak classifiers.

Figure 3.5 shows the flowchart for the AdaBoost algorithm. The AdaBoost algorithm is performed on training data that is provided as N example images (x_i, y_i) , where $i = 1, 2, 3, \dots, N$, x_i is the i th example image and $y_i = 0, 1$ which indicates if the example is negative or positive. N is the total number of examples which could vary for positive and negative examples. A positive example is an image that contains a face and a negative example is an image that does not contain a face. The weights are initialized to $\frac{1}{2p}$ for positive examples and $\frac{1}{2q}$ for negative examples where p and q are the total number of positive and negative examples respectively.

The initialization of weights is followed by an iterative process. The process iterates through all the possible hypotheses. A weak classifier is considered to be a hypothesis. It is much easier to find a weak classifier than a stronger classifier. Figure 3.6 illustrates examples of hypotheses or weak classifiers.

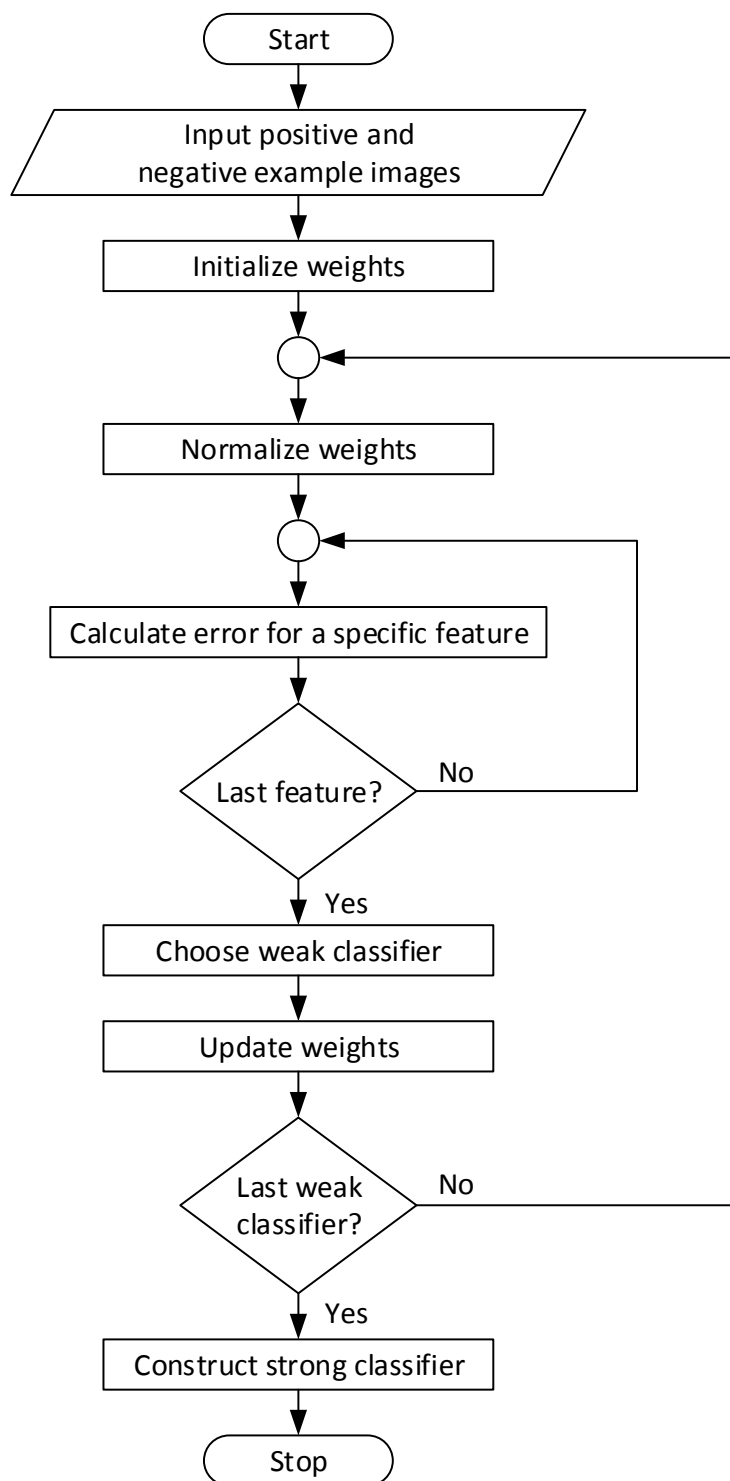


Figure 3.5: Flowchart for the AdaBoost algorithm



Figure 3.6: Examples of hypotheses

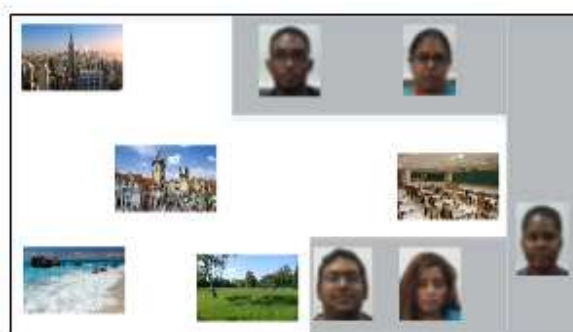


Figure 3.7: All face images captured using strong classifier

Referring to Figure 3.6, one hypothesis could be that all the images to the top (images in gray area) are all face images. Another hypothesis could be that all images to the right (images in gray area) are all face images. Hypothesis 1 and Hypothesis 2 can be easily achieved by applying a threshold (line between the two areas). While the gray areas captured some of the face images, some of the non-face images were captured as well. The end result of the Adaboost algorithm is to create a strong classifier that produces a better result as seen in Figure 3.7. It can be seen that all face images were captured in the gray area.

Referring back to Figure 3.5, the first step in the iterative process is to normalize the weights.

The weights are normalized to:

$$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^n w_{t,j}} \quad (3.4)$$

(Viola and Jones 2001)

where $w_{t,i}$ is the weight of the classifier and $t = 1, 2, 3 \dots T$ weak classifiers.

The second step in the iterative process is to calculate the error of the weight for each rectangle feature. It is given by the following equation:

$$\epsilon_j = \sum_i w_i |h_j(x_i) - y_i| \quad (3.5)$$

(Viola and Jones 2001)

where ϵ_j is the error calculated for a particular feature, $h_j(x_i)$ is the weak classifier for a particular feature and $j = 1, 2, 3 \dots$ Total number of features. $h_j(x_i)$ is given as:

$$h_j(x_i) = \begin{cases} 1, & \text{if } p_j f_j(x_i) < p_j \theta_j \\ 0, & \text{otherwise} \end{cases} \quad (3.6)$$

(Viola and Jones 2001)

where $f_j(x_i)$ is a feature, θ_j is a threshold and p_j is a parity which indicates the direction of the inequality sign.

The next step in the AdaBoost algorithm is to select weak classifier $h_j(x_i)$ with the lowest error ϵ_j . The selected classifier (h_t) will be used in constructing the strong classifier. After

choosing the weak classifier, the weights are updated according to the error e_t associated the selected classifier:

$$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i} \quad (3.7)$$

(Viola and Jones 2001)

where $\beta_t = \frac{e_t}{1-e_t}$, $e_i = 0$, if x_i is classified correctly and $e_i = 1$ otherwise.

At the end of the iterative process, the strong classifier is constructed:

$$h(x) = \begin{cases} 1, & \sum_{t=1}^T \log \frac{1}{\beta_t} h_t(x) \geq \frac{1}{2} \sum_{t=1}^T \log \frac{1}{\beta_t} \\ 0, & \text{otherwise} \end{cases} \quad (3.8)$$

(Viola and Jones 2001)

where T is the total number of weak classifiers.

Viola and Jones (2001) found that the AdaBoost algorithm reduced the number of features needed to yield reasonable results from over 45,000 to 200. There are two important features selected by the AdaBoost algorithm for face detection as seen in Figure 3.8. One of the features focuses on the property that the eyes region is darker than the upper cheeks region. The other feature focuses on the property that the eyes region is darker than the nose bridge region.



Figure 3.8: Features selected by AdaBoost algorithm for face detection

3.1.4 Construct a cascade of classifiers

Viola and Jones (2001) proposed constructing a cascade of classifiers for increasing the face detection rate and reducing the computation time. Figure 3.9 illustrates the schematic of the detection cascade. Each stage in the cascade is created by training the classifiers using the AdaBoost algorithm. The first stage begins with a two-feature classifier. A two-feature classifier can be used to achieve a detection rate of 100% and a false positive rate of 40%. While this classifier is unacceptable for face detection, it can be used to significantly reduce the number of detection windows with little processing. The classifier becomes stronger after each stage. Each stage determines whether or not the given detection window contains a face. If the detection window contains a face then it is passed on to the next stage. If the detection window does not contain a face then it is rejected. This allows for more focus on probable face regions and quick elimination of background regions. Figure 3.10 illustrates the flowchart for constructing a cascaded classifier.

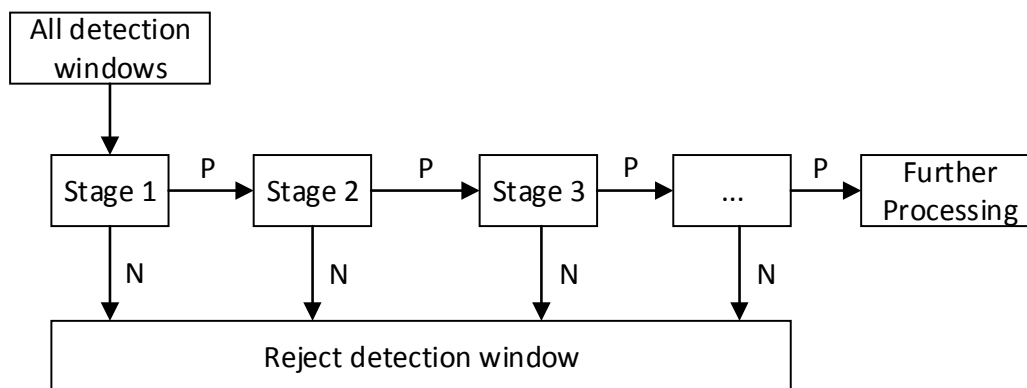


Figure 3.9: Schematic of detection cascade

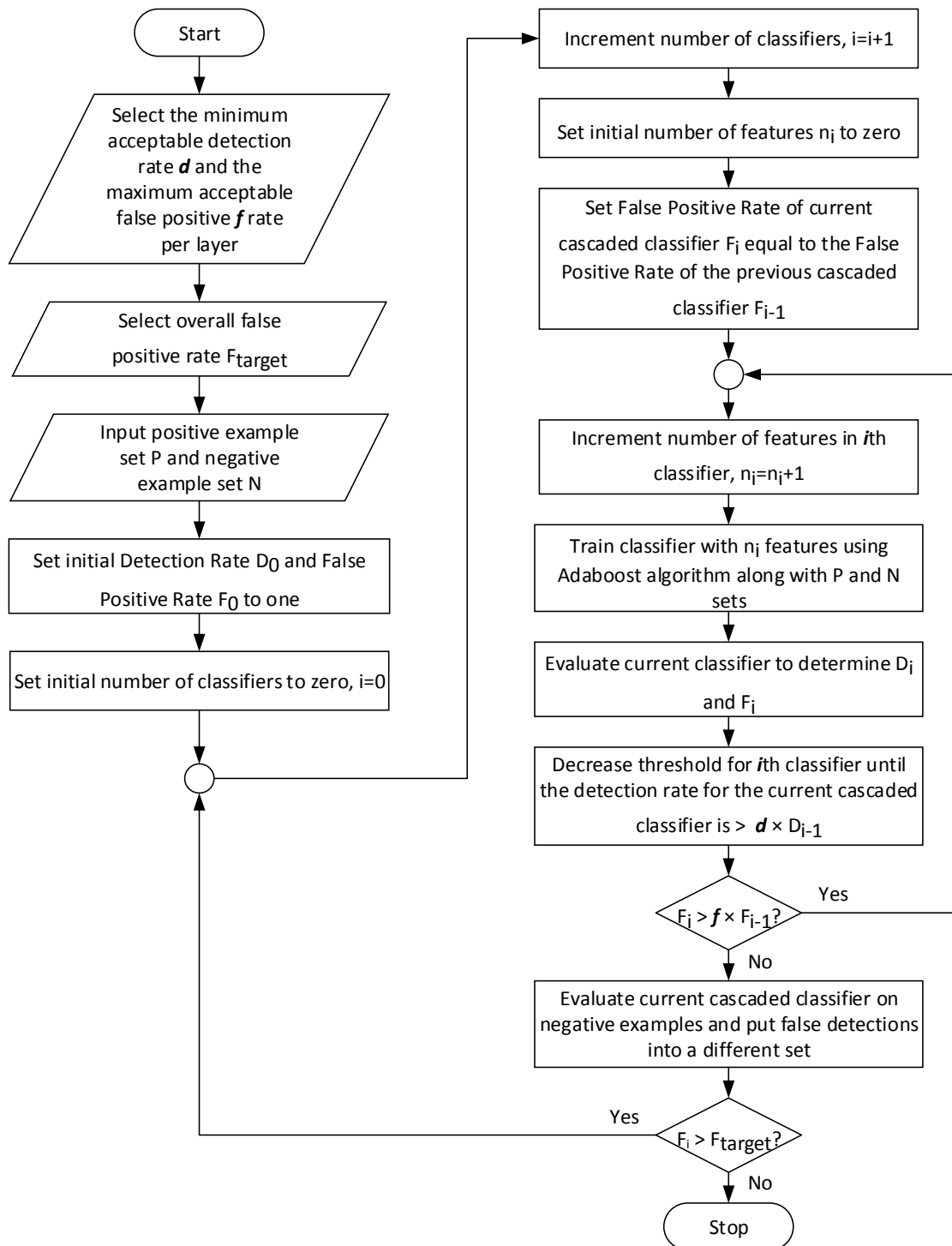


Figure 3.10: Flowchart for constructing a cascaded classifier

3.2 Extension of Viola and Jones Algorithm

An extension of the Viola and Jones algorithm can be found in the Emgu CV library. This library is a .NET compatible version of the OpenCV image processing library, which allows OpenCV functions to be called from languages such as C# (Emgu 2015).

The algorithm by Viola and Jones (2001) was extended by Lienhart, Kuranov, and Pisarevsky (2002) (OpenCV 2014). The Haar-like features were extended by an efficient set of 45° rotated features as seen in Figure 3.11. Additional features other than the 45° rotated features were also used. It should be noted that the four-rectangle feature was not used. Lienhart, Kuranov, and Pisarevsky (2002) found that the extended features improved the performance of object detection.

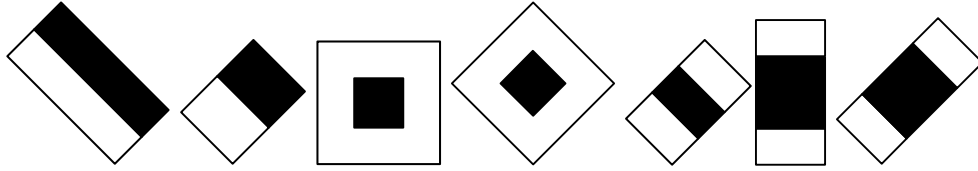


Figure 3.11: Examples of additional features for object detection (Lienhart, Kuranov, and Pisarevsky 2002)

The sum of all the pixel values within any 45° rotated rectangle is represented as:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y - |x - x'|} i(x', y') \quad (3.9)$$

where $i(x, y)$ is the original image and $ii(x, y)$ is the integral image.

The reference points for finding the sum of all the pixel values in the 45° rotated rectangle D can be seen in Figure 3.12.

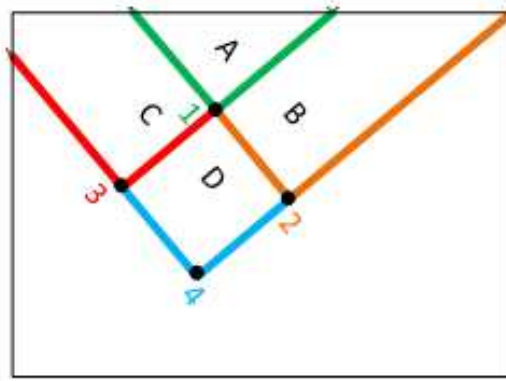


Figure 3.12: Reference points for calculating the sum of all the pixel values in the 45° rotated rectangle D

Similarly to the non-rotated rectangle case, in order to find the sum of all the pixel values in rectangle D, the sum of the values at points 2 and 3 is subtracted from the sum of the values at points 4 and 1.

Lienhart, Kuranov, and Pisarevsky (2002) found that the Gentle AdaBoost algorithm outperformed the Discrete AdaBoost algorithm (previously described in the Viola Jones algorithm). The Gentle AdaBoost algorithm was adapted from Freund and Schapire (1996) and is illustrated in the flowchart in Figure 3.13. It can be seen that this algorithm is similar to the Discrete AdaBoost algorithm.

The Gentle AdaBoost algorithm is performed on training data that is provided as N example images (x_i, y_i) , where $i = 1, 2, 3, \dots, N$, x_i is the i th example image and $y_i = -1, 1$ which indicates if the example is negative or positive. N is the total number of examples which could vary for positive and negative examples. A positive example is an image that contains a face and a negative example is an image that does not contain a face. The weights are initialized to

$\frac{1}{p}$ for positive examples and $\frac{1}{q}$ for negative examples where p and q are the total number of positive and negative examples respectively.

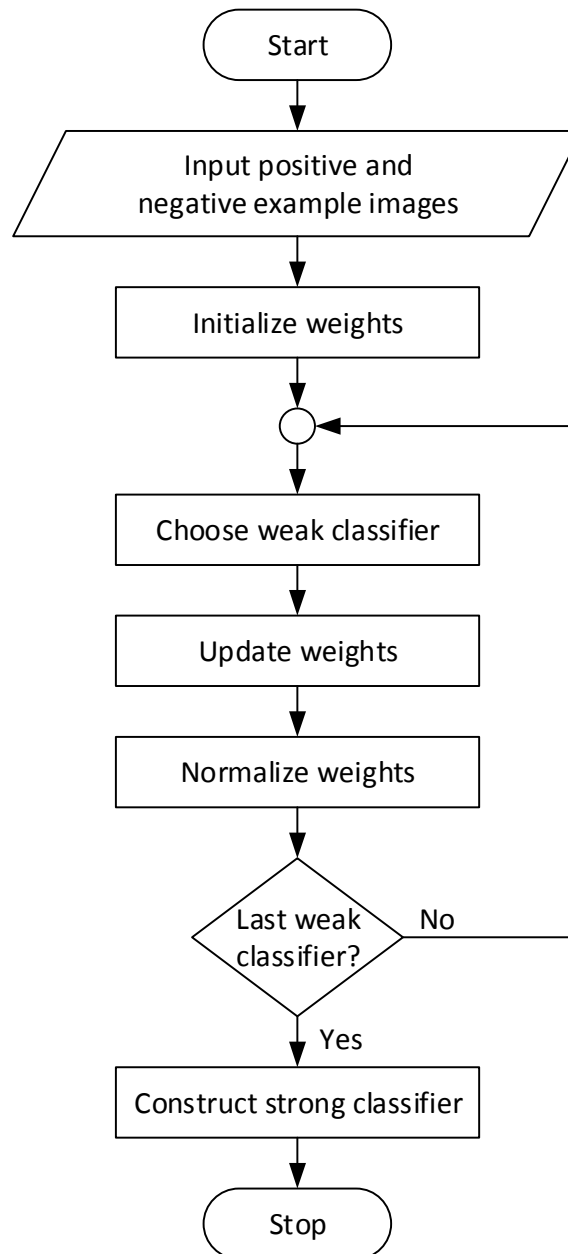


Figure 3.13: Flowchart for Gentle AdaBoost algorithm

The initialization of weights is followed by an iterative process. The process iterates through all the possible hypotheses or weak classifiers. At each iteration, the weak classifier $h_t(x_i)$ is fitted by the weighted least squares of y_i to x_i with weights w_i using regression analysis, where $t = 1, 2, 3, \dots$. Total number of weak classifiers. The weights are updated as follows:

$$w_i \leftarrow w_i e^{-y_i h_t(x_i)} \quad (3.10)$$

(Lienhart, Kuranov, and Pisarevsky 2002)

The weights are then normalized such that:

$$\sum_i w_i = 1 \quad (3.11)$$

(Lienhart, Kuranov, and Pisarevsky 2002)

At the end of the iterative process, the strong classifier is constructed:

$$h(x) = \text{sgn} \left[\sum_{t=1}^T h_t(x) \right] \quad (3.12)$$

(Lienhart, Kuranov, and Pisarevsky 2002)

3.3 Eigenfaces Method

The Eigenfaces method will be explored since it is commonly used in face recognition (Belhumeur, Hespanha, and Kriegman 1997). Additionally the Eigenfaces method is available in the Emgu CV library. The algorithm for this method presented by Turk and Pentland (1991) is illustrated in Figure 3.15 and outlined in the following steps:

1. Obtain a set of M face images with each image having a size of $N \times N$. This is the face database and will be referred to as the training set.
2. Convert the face images in the training set into vectors denoted as $x_1, x_2, x_3, \dots, x_M$.
3. Normalize the vectors in the training set
 - a. Calculate the average vector μ is:

$$\mu = \frac{1}{M} \sum_{i=1}^M x_i \quad (3.13)$$

- b. Find the difference between the vectors in the training set and the average vector. This is the normalized vector ϕ_i .

$$\phi_i = x_i - \mu \quad (3.14)$$

4. Find the covariance matrix C .

$$C = \frac{1}{M} \sum_{i=1}^M \phi_i (\phi_i^T) \quad (3.15)$$

(Turk and Pentland 1991)

The dimension of the covariance matrix C is $N^2 \times N^2$. Therefore, the number of eigenvectors and eigenvalues will be N^2 . An eigenvector has a face-like appearance

and is referred to as an eigenface. It is a characteristic feature image which is a principal component of the set of faces (Turk and Pentland 1991). Figure 3.14 illustrates an example of eigenfaces.

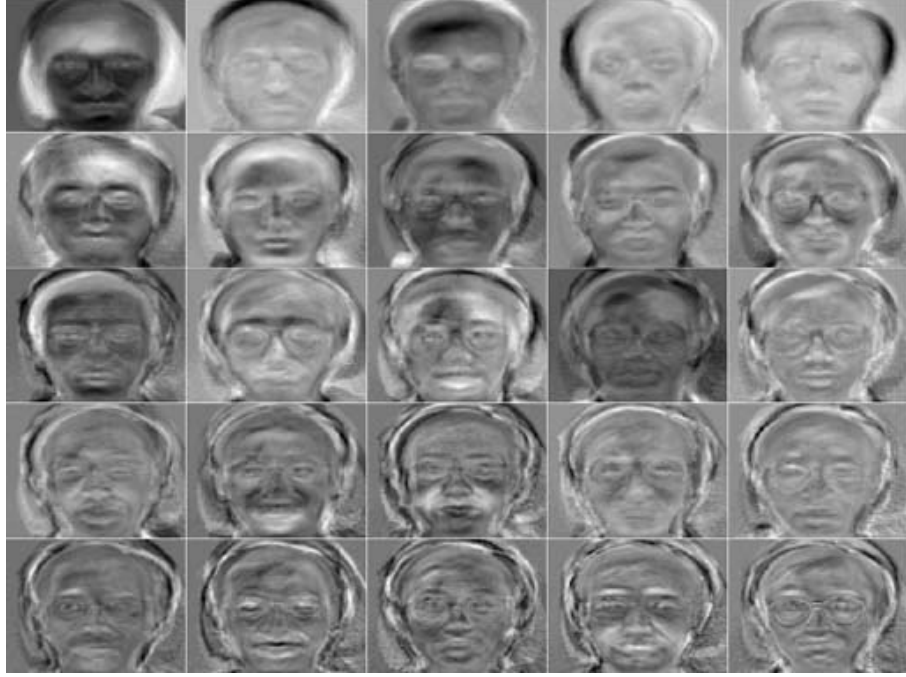


Figure 3.14: Example of eigenfaces (Coro 2015)

The task of finding N^2 eigenfaces is computational expensive. Turk and Pentland (1991) found that if $M < N^2$, then the eigenvectors can be found by solving a $M \times M$ matrix and taking the linear combinations of the resulting vectors. The number of calculations is reduced from the number of pixels in the image N^2 to the number of images in the training set M . Generally, $M \ll N^2$ so the calculations are manageable.

5. Calculate the eigenvectors v_i and eigenvalues λ_i of the covariance matrix.

$$Cv_i = \lambda_i v_i \quad (3.16)$$

6. Sort the eigenvectors in descending order by their eigenvalues.

7. Select M' eigenvectors with the largest eigenvalues where $M' < M$. These eigenvectors span the feature space.
8. Transform each face x_i in the training set into its eigenface components. This is known as projecting the face image into the face space (Turk and Pentland 1991).

$$\omega_k = v_k^T (x_i - \mu) \quad (3.17)$$

(Turk and Pentland 1991)

where $k = 1, 2, 3, \dots, M'$

9. The weights ω_k form a vector $\Omega_{x_i}^T$ which indicates the contribution of each eigenface representing face x_i .

$$\Omega_{x_i}^T = \omega_1, \omega_2, \omega_3, \dots, \omega_{M'} \quad (3.18)$$

(Turk and Pentland 1991)

10. Convert an input face into a vector x' and repeat steps 8 to 9

$$\omega_k = v_k^T (x' - \mu) \quad (3.19)$$

$$\Omega_{x'}^T = \omega_1, \omega_2, \omega_3, \dots, \omega_{M'} \quad (3.20)$$

11. Find the minimum Euclidian distance ϵ between the input face and each face in the training set.

$$\epsilon = \min \|\Omega_{x'}^T - \Omega_{x_i}^T\| \quad (3.21)$$

12. If the minimum Euclidian distance ϵ is below some threshold θ , then the input face x' is classified as face x_i in the training set, else the face is unknown.

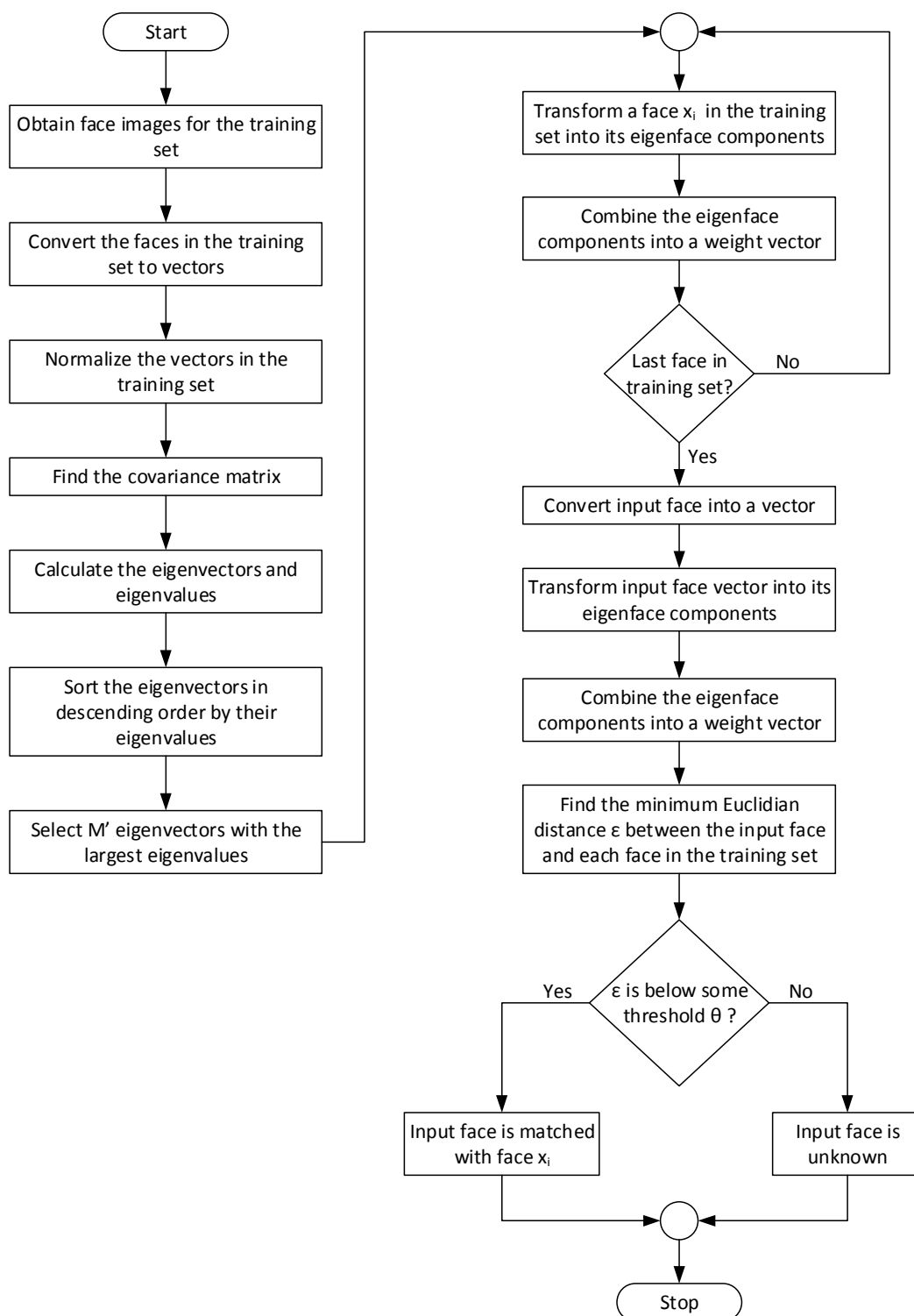


Figure 3.15: Flowchart for Eigenfaces method

3.4 Fisherfaces Method

In addition to the Eigenfaces method, the Fisherfaces method is also available in the Emgu CV library and will be explored. The Eigenfaces method does not consider a labelled training set. A labelled training set consists of different groups of face images for each person. These groups are referred to as classes. This information can be used to create a more reliable method to decrease the dimensionality of the feature space. The Fisherfaces method maximizes the ratio of the determinant of the between-class scatter and the within-class scatter. The algorithm for this method presented by Belhumeur, Hespanha, and Kriegman (1997) is illustrated in Figure 3.16 and outlined in the following steps.

1. Obtain a set of M face images with each image having a size of $N \times N$ for the training set.
2. Convert the face images in the training set into vectors denoted as $x_1, x_2, x_3, \dots, x_M$.
3. Find the average vector μ using Equation 3.13.
4. Find the average vector μ_q for class q .

$$\mu_q = \frac{1}{M_q} \sum_{j=1}^{M_q} x_{qj} \quad (3.22)$$

Where M_q is the number of faces in class q and x_{qj} is the j th vector in class q .

5. Calculate the between-class scatter matrix C_B .

$$C_B = \sum_{q=1}^{N_C} M_q (\mu_q - \mu)(\mu_q - \mu)^T \quad (3.23)$$

(Belhumeur, Hespanha, and Kriegman 1997)

Where N_C is the number of classes in the training set.

6. Calculate the within-class scatter matrix C_W .

$$C_W = \sum_{q=1}^{N_C} \sum_{j=1}^{M_q} (x_{qj} - \mu_q)(x_{qj} - \mu_q)^T \quad (3.24)$$

(Belhumeur, Hespanha, and Kriegman 1997)

7. The goal of the Fisherfaces method is to maximize the ratio $\frac{\det|C_B|}{\det|C_W|}$. This can be achieved by calculating the eigenvectors v_i and eigenvalues λ_i of the scatter matrices C_B and C_W .

$$C_B v_q = \lambda_q C_W v_q \quad (3.25)$$

$$C_W^{-1} C_B v_q = \lambda_q v_q \quad (3.26)$$

8. Sort the eigenvectors in descending order by their eigenvalues.
9. Select N_C' eigenvectors with the largest eigenvalues where $N_C' < N_C$.
10. Project each face in the training set x_i into the face space.

$$\omega_r = v_r^T x_i \quad (3.27)$$

Where $r = 1, 2, 3, \dots, N_C'$ and $i = 1, 2, 3, \dots, M$

11. The weights ω_r form a vector $\Omega_{x_i}^T$ which indicates the contribution of each vector representing face x_i .

$$\Omega_{x_i}^T = \omega_1, \omega_2, \omega_3, \dots, \omega_{N_C'} \quad (3.28)$$

12. Convert an input face into a vector x' and repeat steps 10 to 11.

$$\omega_r = v_r^T x' \quad (3.29)$$

$$\Omega_{x'}^T = \omega_1, \omega_2, \omega_3, \dots, \omega_{N_C'} \quad (3.30)$$

13. Find the minimum Euclidian distance ϵ between the input face and each face in the training set.

$$\epsilon = \min \|\Omega_{x'}^T - \Omega_{x_i}^T\| \quad (3.31)$$

14. If the minimum Euclidian distance ϵ is below some threshold θ , then the input face x' is classified as face x_i in the training set, else the face is unknown.

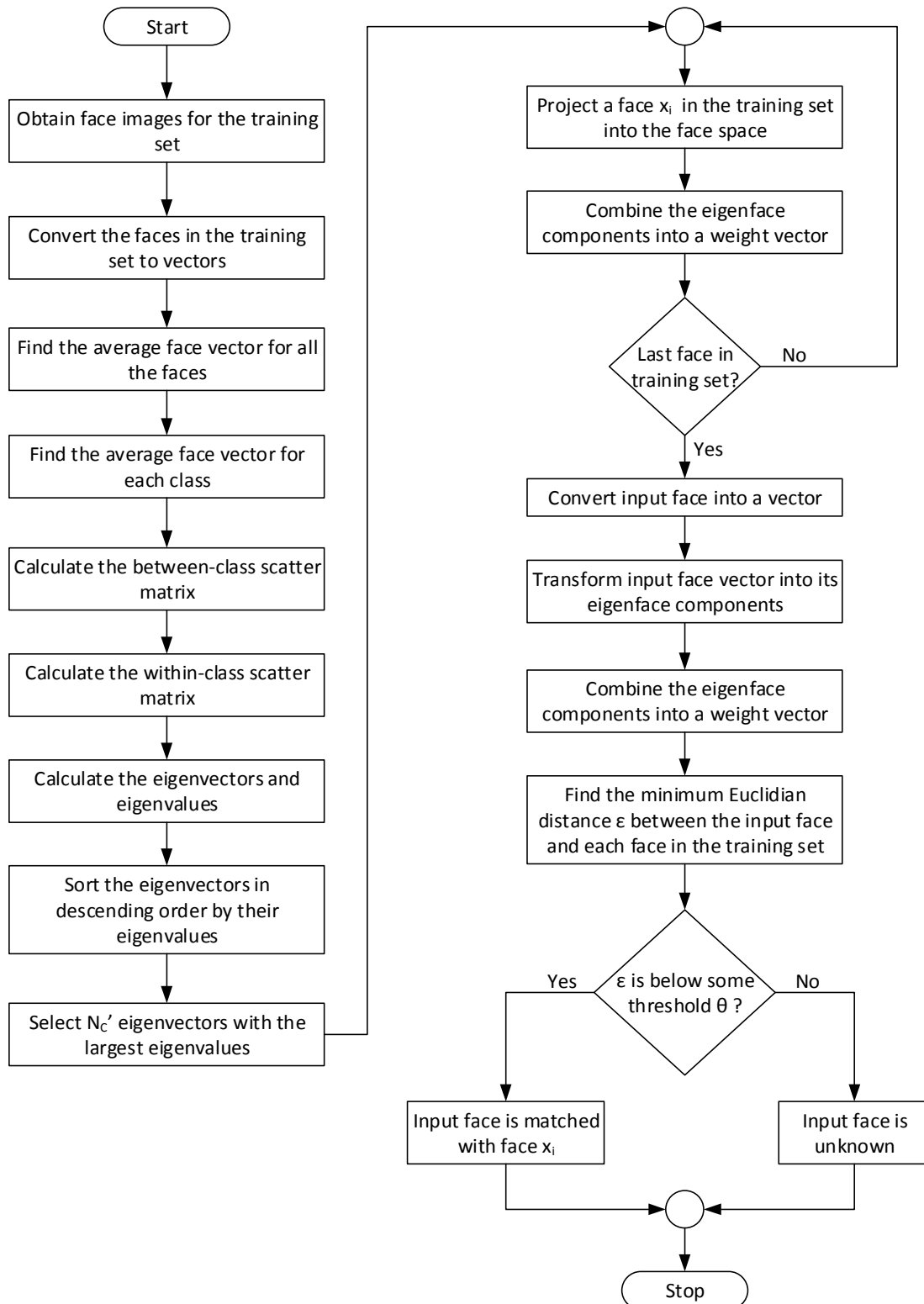


Figure 3.16: Flowchart for Fisherfaces method

3.5 Local Binary Patterns Histograms

In addition to the two previous face recognition methods, the LBPH method is also available in the Emgu CV library and will be explored. The local binary pattern operator by Ojala, Harwood, and Pietikäinen (1996) compares the 3x3 neighborhood of each pixel with the center pixel. If the neighborhood pixel value is greater than or equal to the center pixel value the result is 1, else the result is 0. The resulting 1's and 0's form a binary number which is converted to a decimal number and used as a label for a particular pixel. A histogram of the labels can be utilized as a texture descriptor. Figure 3.17 illustrates the basic LBP operator.

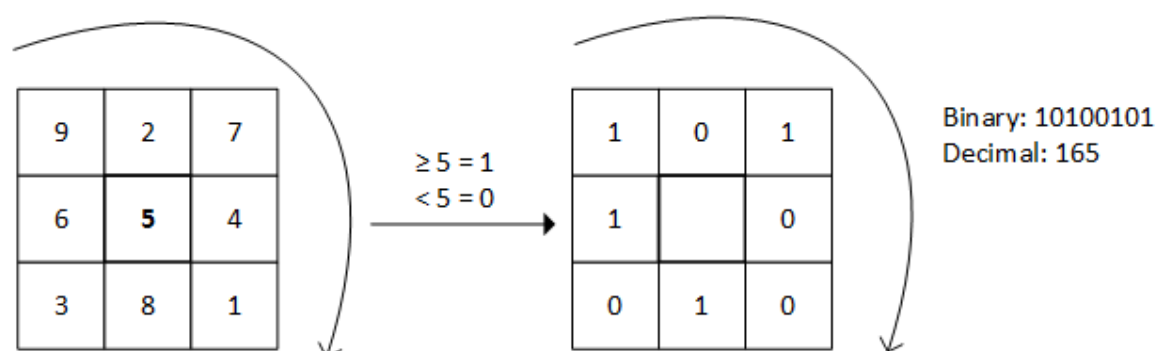


Figure 3.17: Basic LBP operator

The LBP operator was extended to use varying neighborhood sizes by Ojala, Pietikainen, and Maenpaa (2002). This is accomplished by using circular neighborhoods and bilinear interpolation. The neighborhood size can be changed by setting the radius of the circle and the number of pixels in the neighborhood. Another extension to the LBP operator also by Ojala, Pietikainen, and Maenpaa (2002) was the use of uniform patterns. A LBP is uniform if it contains at most two bitwise transitions from 0 to 1 or vice versa given that the binary string is circular. Some examples of uniform LBP are 01110000, 11000011 and 11100111. A uniform pattern is used to reduce the total number of labels. Each uniform pattern has a unique label

while all non-uniform patterns have a single label. For example, if the neighborhood size is 8, the total number of labels is 256. There are 58 uniform patterns (Pietikäinen 2010), therefore the number of labels is reduced to 59. Ahonen, Hadid, and Pietikäinen (2004) used the notation (P, R) , where P is the number of sample points on the circle and R is the radius of the circle. Figure 3.18 illustrates an example of a circular $(8, 2)$ neighborhood LBP descriptor.

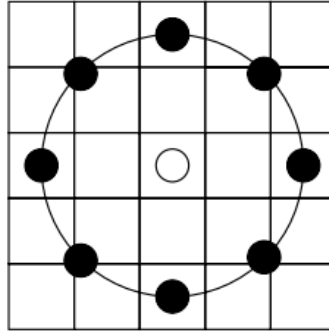


Figure 3.18: Example of a circular $(8, 2)$ neighborhood LBP descriptor (Ahonen, Hadid, and Pietikäinen 2004)

The output of a LBP operator is given by the following equation:

$$LBP = \sum_{i=0}^{P-1} 2^i s(i_k - i_c) \quad (3.32)$$

Where i_k is the i th neighbor pixel value, i_c is the center pixel value and the function s is given as:

$$s = \begin{cases} 1, & x \geq 0 \\ 0, & \text{otherwise} \end{cases} \quad (3.33)$$

A histogram of the labeled image $f_l(x, y)$ is represented as:

$$H_i = \sum_{x,y} I(f_l(x,y) = i) \quad (3.34)$$

Where $i = 0, \dots, n - 1$ and n is the number of labels generated by the LBP operator.

(Ahonen, Hadid, and Pietikäinen 2004)

The function $I(A)$ is given as:

$$I(A) = \begin{cases} 1, & A \text{ is true} \\ 0, & A \text{ is false} \end{cases} \quad (3.35)$$

(Ahonen, Hadid, and Pietikäinen 2004)

This histogram H_i has information about the distribution of local micro-patterns, for example, edges. Ahonen, Hadid, and Pietikäinen (2004) modified the histogram such that it would also contain spatial information by dividing the image into regions. The modified histogram is represented as:

$$H_{i,j} = \sum_{x,y} I(f_l(x,y) = i) I((x,y) \in R_j) \quad (3.36)$$

(Ahonen, Hadid, and Pietikäinen 2004)

Where $i = 0, \dots, n - 1$, $j = 0, \dots, m - 1$, n is the number of labels generated by the LBP operator and m is the number of regions.

Figure 3.19 illustrates the description of a face using the modified histogram.

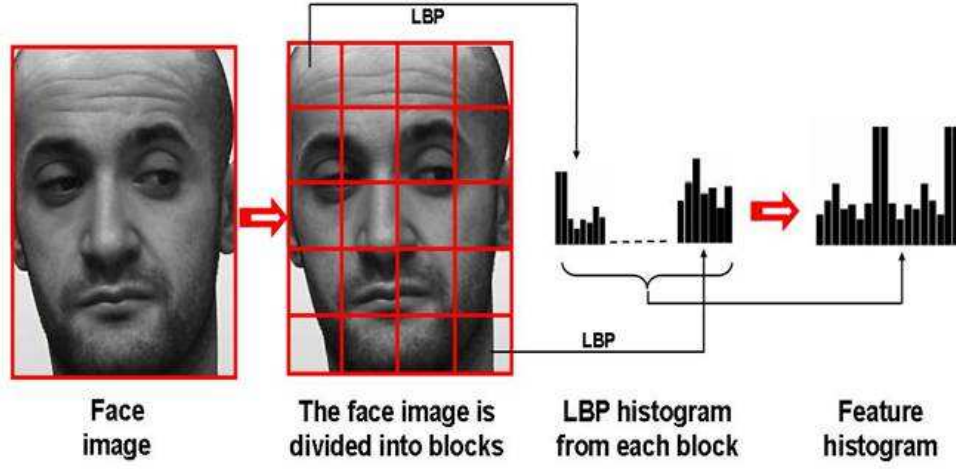


Figure 3.19: Face description using LBP (Pietikäinen 2010)

Chi square distance is used as a dissimilarity measure for comparing two faces. It is given by the following equation:

$$\chi^2(S, M) = \sum_i \frac{(S_i - M_i)^2}{S_i + M_i} \quad (3.37)$$

(Ahonen, Hadid, and Pietikäinen 2004)

Where S is the input face histogram and M is a face histogram in the training set.

Taking into account the regions in the face, the Chi square distance becomes:

$$\chi_w^2(S, M) = \sum_{i,j} w_j \frac{(S_{i,j} - M_{i,j})^2}{S_{i,j} + M_{i,j}} \quad (3.38)$$

(Ahonen, Hadid, and Pietikäinen 2004)

Where w_j is the weight for region j .

Recognition is achieved by finding the minimum distance ϵ between the input face and each face in the training set.

$$\epsilon = \min \left(\chi^2_w(S, M_t) \right) \quad (3.39)$$

Where $t = 1, 2, 3, \dots, N$ images in the training set

If the minimum distance ϵ is below some threshold θ , then the input face x' that corresponds to the face histogram S is classified as face x_t in the training set that corresponds to face histogram M_t , else the face is unknown.

The algorithm for the LBPH method is illustrated in Figure 3.20 and summarized in the following steps:

1. Obtain a set of images for the training set.
2. Divide each face in the training set into small regions.
3. Find the label for each pixel in the face image using the LBP operator.
4. Find the histogram of labels for each region.
5. Concatenate the histograms for all the regions
6. Divide an input face into small regions and repeat steps 3 to 5
7. Find the minimum distance between the input face histogram and each face histogram in the training set.
8. If the minimum distance is below some threshold then the input face corresponds to the face in the training set with the minimum distance, else the face is unknown.

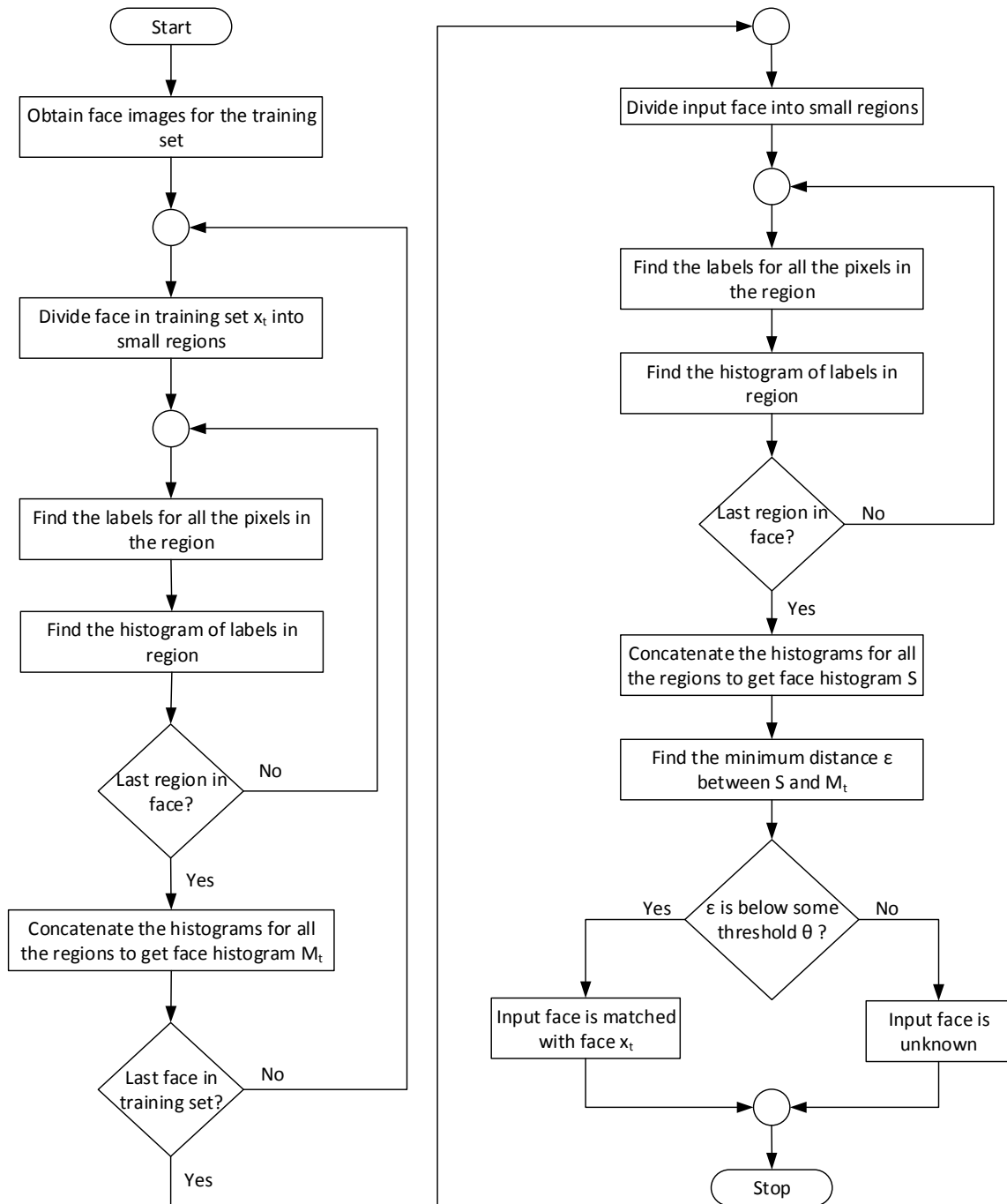


Figure 3.20: Flowchart for Local Binary Patterns Histograms method

Chapter 4 Methodology

A class attendance system called ECNG Class Attendance was developed and tested. This system is comprised of an Android application, a Windows application and a webserver. An image is captured from an Android smartphone with a 13 MP camera or selected from the smartphone's storage using the Android application. The image is then uploaded to the webserver using the Android application. The attendance record is updated and retrieved using a PC with the Windows application. Figure 4.1 illustrates the basic procedure for using the ECNG Class Attendance system.

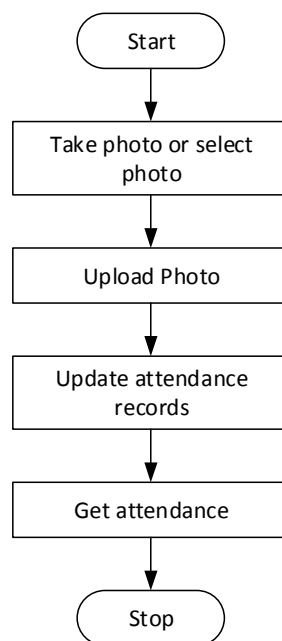


Figure 4.1: Basic procedure for using the ECNG Class Attendance System

4.1 Android application

Android was selected as the platform for smartphone programming because there are many smartphones that support Android. Additionally, there are online support, APIs and an IDE

specifically for programming in Android. The Android Studio IDE is free and compatible with Windows, Mac OS X and Linux.

The Android application consists of an image view, a drop down list and 4 buttons. Figure 4.2 illustrates the layout of the Android application and Figure 4.3 illustrates the flowchart for the application. The course must first be selected by using the drop down list. The next step depends if the user decides to take a photo or select a photo. Either action is accomplished by using the respective button. If the user decides to take a photo, the camera application is deployed. After capturing the photo, the user can either save or discard the photo. If the user saves the photo, the Android application will check the smartphone's storage to see if an album named ECNG_Attendance exists. If the album does not exist, the application will create the album. The captured photo is stored in this album and has the following naming convention "<course code>_<date>_<time>.jpg". The date format is "<year>-<month>-<day>" and the time format is "<hour>-<minute>-<second>". The date and time fields contain the date and time the photo was taken. If the user decides to select a photo, the attendance date must first be selected using the Select Date button. After selecting the date, the Select Photo button is used to browse for the desired photo in the smartphone's storage. The selected photo has the same naming convention as before. After capturing or selecting a photo, the photo is displayed in the image view. The next step is to upload the photo to the webserver using the Upload button. The application checks if there is an Internet connection as well as if a photo was captured or selected. If there is an Internet connection and a photo was taken or selected, the application constructs a POST message. The POST message is constructed from the photo which is converted to a base64 string as well as the image name, course and date. After constructing the POST message, the message is sent to the webserver for further processing.

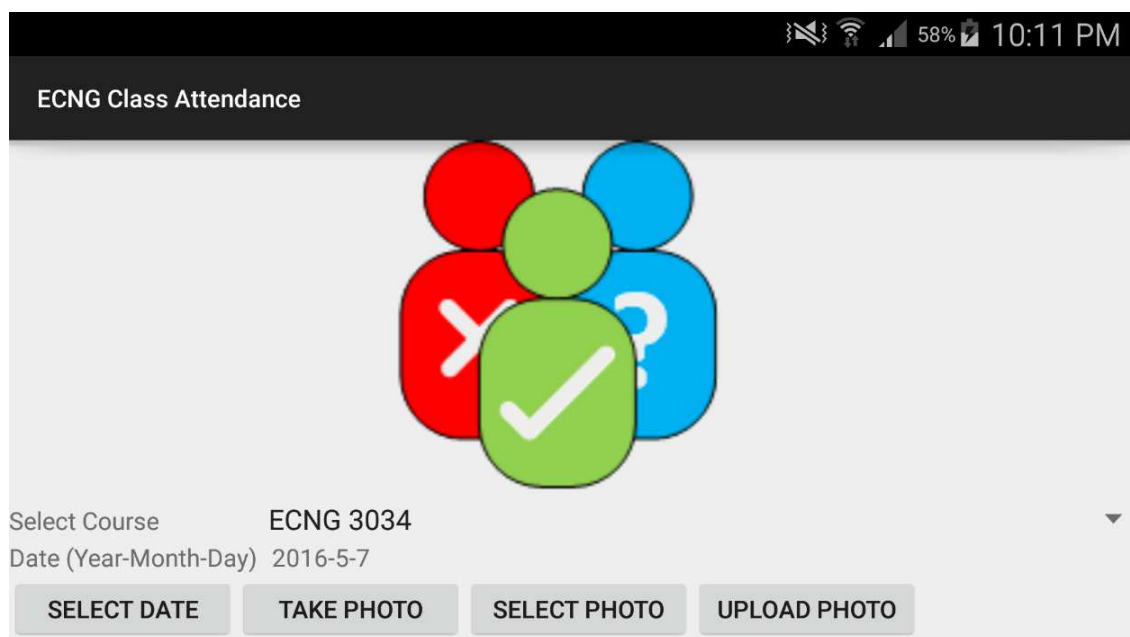


Figure 4.2: Layout of Android application

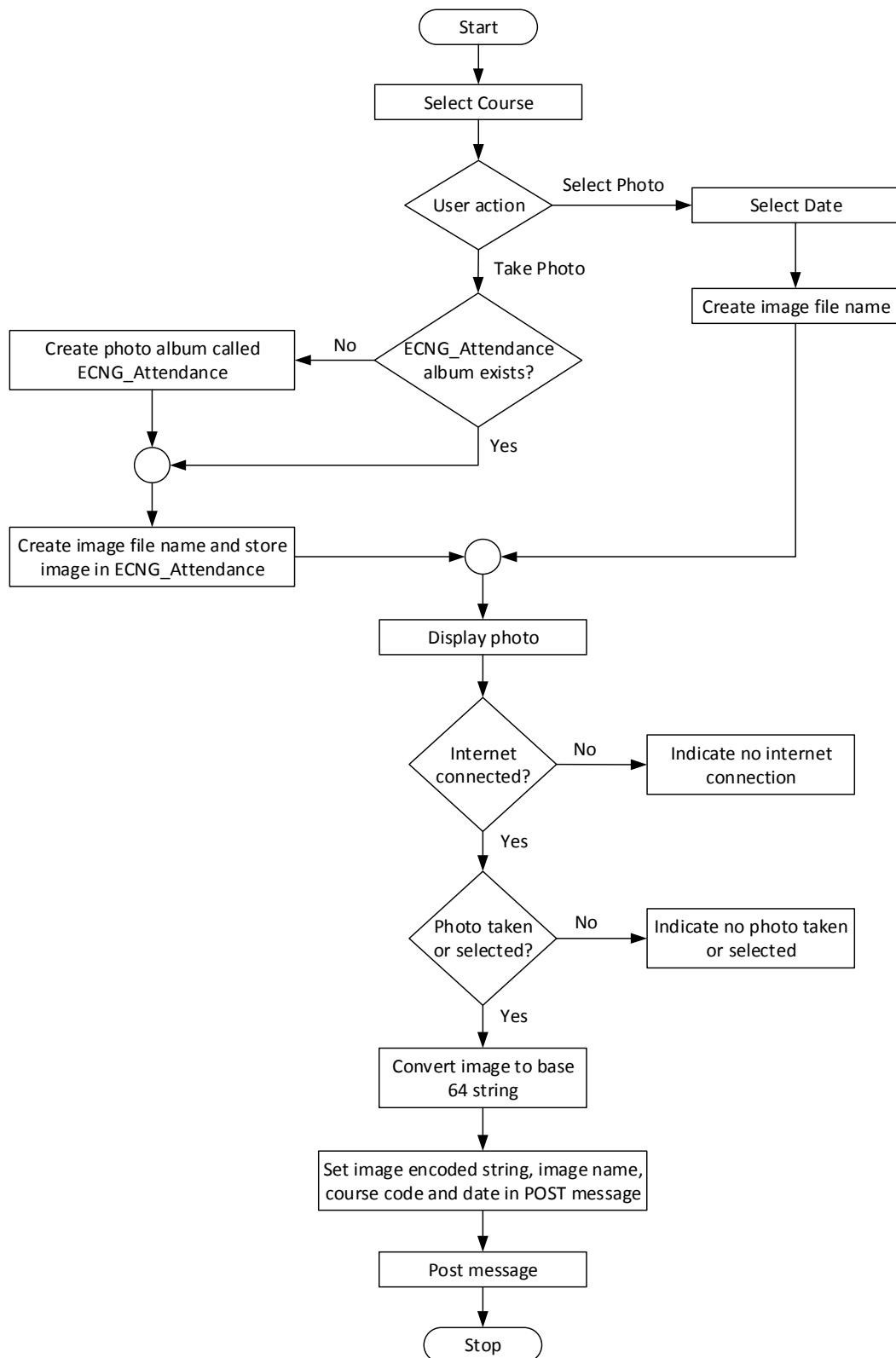


Figure 4.3: Flowchart for Android application

4.2 Webserver

A local webserver was created using XAMPP. The webserver stores the attendance records, course information, student face information and the captured photo information in a database. The webserver also converts the base64 image string sent by the Android application back into an image and stores it in a specific location.

There are 6 tables in database. Three of the tables are used to store the attendance records. Each table from the three tables stores the attendance record for a particular course. For the purpose of demonstration, only 3 courses were used. Each table contains 5 fields: id, student id, first name, last name and date. The id field contains a unique number used to identify the row in the table. The student id, first name and last name fields store the student's identification number, first name and last name respectively. The date field stores the date for the attendance.

One of the tables is used to store the course information. It contains 5 fields: id, course, student id, first name and last name. This table contains all the courses and the students in those courses. The id, student id, first name and last name fields contain the same information as before. The course field contains the course code.

Another table is used to store the student face information. It contains 6 fields: id, student id, first name, last name, image name and image path. The id, student id, first name and last name fields contain the same information as before. The image name field contains the name of the face image and the image path contains the path on the webserver where the image is stored.

Another table is used to store the information on the captured image. It contains 4 fields: id, course, date and path. The id and course fields contain the same information as before. The date field stores the date and time of the attendance. This is done to cater for the case where 2

images were uploaded to the webserver having the same course code and date. The time information is used to differentiate the 2 rows in the table with the same course code and date. The path field contains the path on the webserver where the decoded image is stored.

4.3 Windows Application

The Windows application was created in Microsoft Visual Studio 2015 using C#. This application uses face detection and face recognition to obtain the class attendance. Before this application can be created, the face detection and face recognition algorithms need to be tested. This was accomplished by creating separate Windows applications for face detection and face recognition.

4.3.1 Face Detection

OpenCV is a library of computer vision and machine learning algorithms (Itseez 2015). Some of these algorithms can be used to identify objects, track moving objects, detect faces and recognize faces. As mentioned before the Emgu CV library is a library that allows for OpenCV functions to be called in C#. The Emgu CV library was used because it was easy to learn and fast enough for most applications. Furthermore, it allows for much better user interfaces.

The face detection application consists of a picture box and 2 buttons. Figure 4.4 illustrates the layout of the face detection application. The first step in using this application is to use the Load Image button to browse for the desired image stored in the PC. The selected image is displayed in the picture box. The next step is to use the Detect Face button to find the faces in the image. Rectangular borders are drawn around the detected faces in the image.

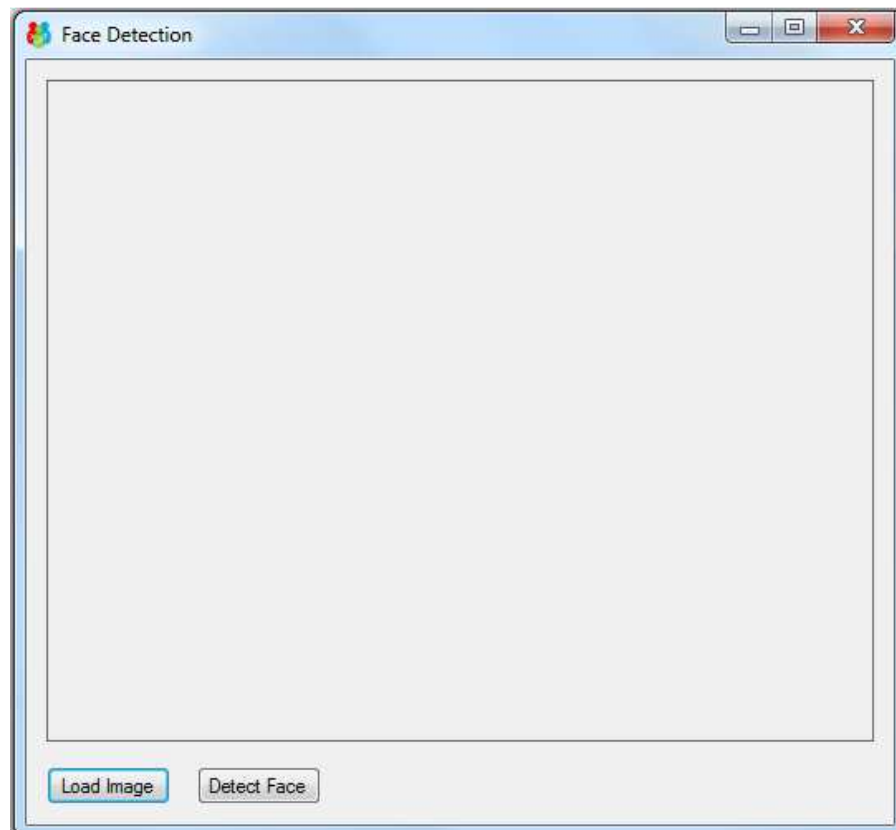


Figure 4.4: Layout of face detection application

In order to detect the faces in an image, the `DetectMultiScale` function from the Emgu CV library was used. This method finds the rectangular regions in the image that are most likely to contain the objects in which the cascade has been trained for and returns those regions as a sequence of rectangles. The image is scanned several times at different scales. Each time the overlapping regions are considered. After all the regions that have passed the classifier cascade are collected, they are grouped and returned as a sequence of average rectangles for each large enough group (Emgu 2013a). The theory for the `DetectMultiScale` function was described in one of the previous sections (Extension of Viola Jones Algorithm).

The syntax for the DetectMultiScale function is given as follows:

```
public Rectangle[] DetectMultiScale(
    Image<Gray, byte> image,
    double scaleFactor,
    int minNeighbors,
    Size minSize,
    Size maxSize
)
```

(Emgu 2013a)

There are 5 input parameters for this method as described in Emgu (2013a): image, scaleFactor, minNeighbors, minSize and maxSize. The image parameter is a gray scale image from which the objects will be detected. The scaleFactor parameter is a factor by which search window is scaled between successive scans. For a factor of 1.2, the window is scaled by 20% on each scan. A larger factor increases the speed of the face detection process. However, the number of missing faces is increased. A lower factor decreases the number of missing faces. However, the computational speed decreases. The minNeighbors parameter is the minimum number (minus 1) of neighbor rectangles which makes up an object. The minSize parameter is the minimum window size. It is set to the size of the samples the classifier was trained on. A larger value reduces the number of false detections, but increases the number of missed faces. A lower value reduces the number of missed detections, but the number of false detections increases. The maxSize parameter is the maximum window size. The maximum window size was set to the default value. When the default value is used the maximum window size parameter is ignored.

Lienhart, Kuranov, and Pisarevsky (2002), the authors who extended the Viola and Jones algorithm found that the ideal scaling factor was 1.1 and the optimal minimum window size was 20×20 . Schmidt and Kasiński (2007) investigated the influence of constraining the

minimum number of neighbors on the efficiency of the Haar Cascade Classifier. They found that the ideal value for the minimum number of neighbor rectangles was 5. Therefore the scaling factor, minimum window size and minimum number of neighbor rectangles were set to 1.1, 20 and 5 respectively in the DetectMultiScale function.

As discussed in the previous section, one of the stages in face detection involves training the classifiers in the cascade using face and non-face images. The Emgu CV library provided a cascaded classifier that was already trained for faces in the form of an XML file. The XML file was examined and the minimum window size was found to be 24×24 . Therefore the minimum window size parameter in the DetectMultiScale function was changed to 24. Figure 4.5 illustrates the flowchart for the face detection application.

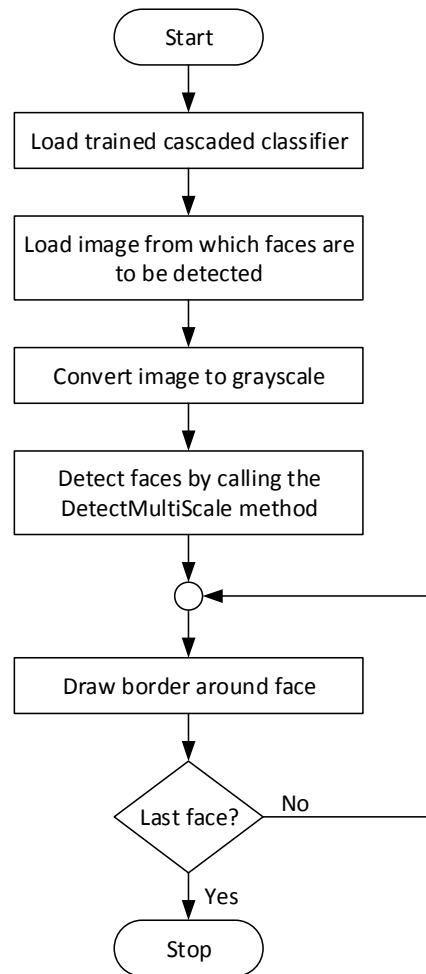


Figure 4.5: Flowchart for face detection application

All face detection tests are described in Chapter 5 Testing and Results. The values for the assigned parameters yielded some falsely detected faces as seen in Table 5.3 in the same chapter. As mentioned before, the ECNG Class Attendance system was designed for a 13 MP camera, so only the results for the 13 MP camera resolution are observed. An attempt was made to reduce the number of false detections by changing some of the parameter values. Viola and Jones (2001) found that a scaling factor of 1.25 produced good results. However, there were still some false detections. The minimum number of neighbor rectangles was selected by experimenting with its value. The value which produced the best result was found to be 6. Therefore the scaling factor, minimum window size and minimum number of neighbor

rectangles were set to 1.25, 24 and 6 respectively in the DetectMultiScale function. Table 5.6 in Chapter 5 Testing and Results shows that the number of falsely detected faces was reduced but the number of faces which were not detected was similar.

Another attempt was made to reduce the number of false detections. The Emgu CV library also supports eye detection by providing a cascaded classifier that was already trained for eyes in the form of an XML file. The goal was to detect eyes within the detected faces. If eyes were found within a detected face, the detected face is confirmed to be an actual face. If eyes were not found within a detected face, the detected face is not an actual face. The DetectMultiScale function used for face detection was also used for eye detection. The scaling factor and minimum number of neighbor rectangles were set to 1.1 and 5 respectively in the DetectMultiScale function for face and eye detection. The XML file for the trained eye classifier was examined and the minimum window size was found to be 20×20 . The minimum window size was set to 20 for eye detection and 24 for face detection. This setting was used to minimize the number of undetected eyes and faces. Figure 4.6 illustrates the flowchart for the improved face detection algorithm. Table 5.9 in Chapter 5 Testing and Results shows that the number of falsely detected faces was zero but the number of faces which were not detected increased. However, the undetected faces were as a result of the face being increasingly further away from the camera. This modified face detector was used in the ECNG Class Attendance system. The results for this modified face detector show that the maximum distance for detecting faces is restricted to 2.54 m.

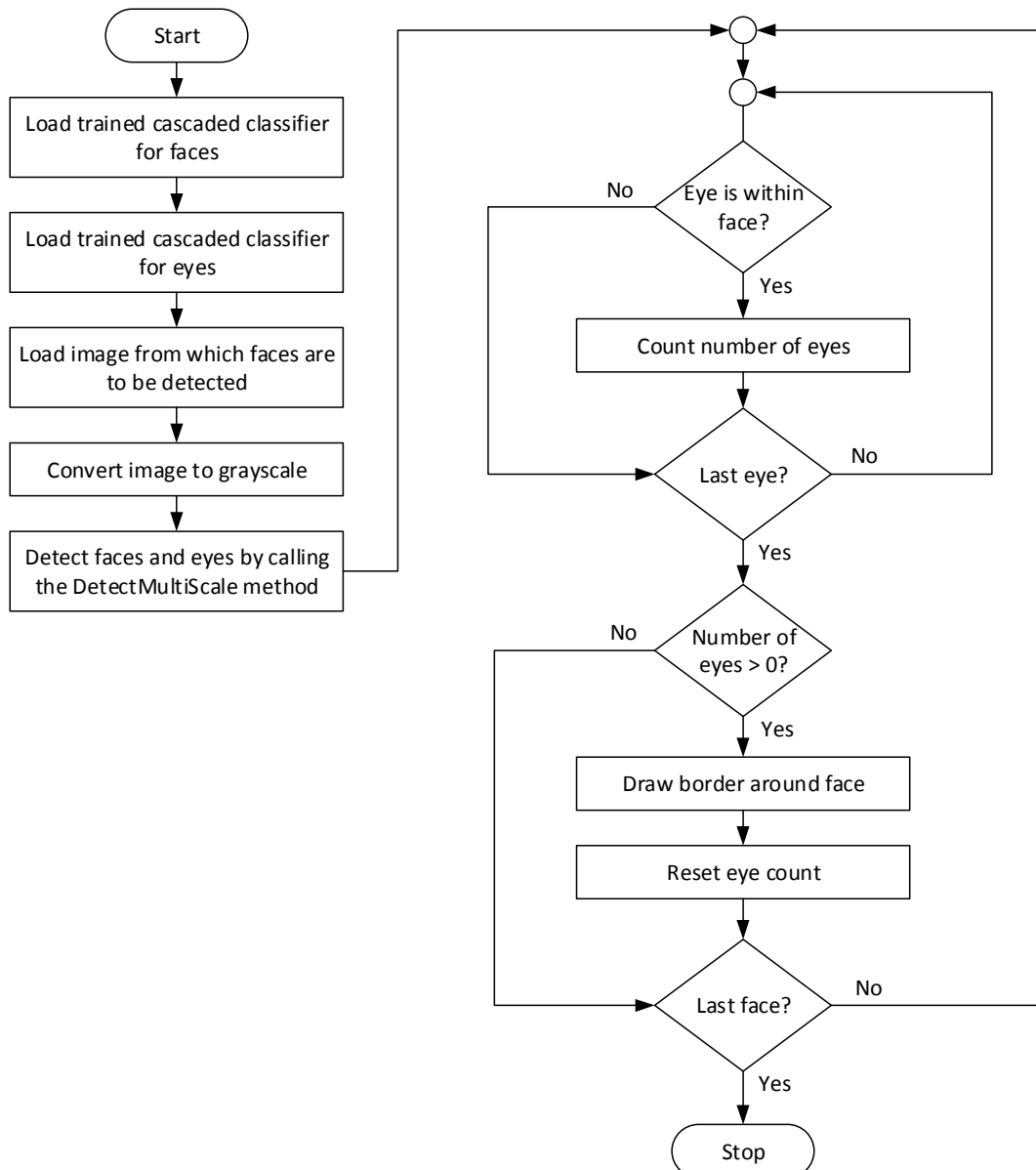


Figure 4.6: Flowchart for improved face detection application

4.3.2 Face Database

As mentioned before a face database must be obtained in order to evaluate the performance of any face recognition algorithm. Additionally, the faces in the database must be suitable for the application. For this application, the environment is a classroom. The students are expected to look towards the camera when the image is being taken. The lighting in the classroom will

vary due to the presence of windows. The amount of light coming from the windows changes according to the time of the day.

A face database was created from the images of 16 student volunteers. There were ten images for each student for a total of 160 images. The volunteers were subjected to the following conditions:

- Upright frontal position with smile/no smile
- Upright frontal position with smile/no smile, head tilted up
- Upright frontal position with smile/no smile, head tilted down
- Upright right side position (from observer's view) with smile/no smile
- Upright left side position (from observer's view) with smile/no smile

Given that the students are expected to face the camera, the upright frontal position with smile/no smile views are sufficient. However, additional views were captured to compensate for intentional or unintentional head movements as well as to test the robustness of the face recognition algorithm. A 13 MP camera from a Samsung Galaxy S4 was used to collect data for the face database. Furthermore, the distance between the volunteer and the camera was maintained at 1.13 m.

From the results of the face detection application in Chapter 5 Testing and Results, the angle for the side views and head tilt were limited. Points in the room were marked out for volunteers to look at, so that the angles for the head tilt and side views would remain constant. Points were also marked out for the location of the camera and the volunteer to ensure a constant distance. Since the volunteers were students who were doing different courses, the time for capturing their images varied for different groups of students. However, their images were

captured in the afternoon period. The images were cropped so that only the face is contained in the image. The detected face from the face detection application is contained within a square. Furthermore, the face is center aligned and not stretched or skewed. Center alignment is aligning the eyes, nose and mouth to the center of the image. The images in the face database were manipulated to mimic the output of the face detection application. Since the size of each face varied, the cropped face images were resized to 280×280 . This was the maximum allowable size by the Emgu CV library using 160 images. Any size greater than 280×280 resulted in a memory issue. The face images were larger than 280×280 so they were down sampled. The images were resized to the maximum allowable size in order to retain as much information as possible about the face. A sample of the faces in the created face database called ECNG face database can be seen below.



Figure 4.7: Sample of faces in ECNG face database

4.3.3 Face Recognition

A face recognition application was created to test the face recognition methods discussed previously. The application consists of 3 picture boxes, 3 radio buttons, 7 buttons and text labels. The first step in using this application is to use the Load Image button to browse for the desired image stored in the PC. The selected image is displayed in the picture box. The next

step is to use the Detect Face button to find the faces in the image. Rectangular borders are drawn around the detected faces in the image. After detecting the faces in the image, a face recognition method has to be selected using one of the radio buttons. The Recognize Face button is then used to recognize the detected faces. The detected and recognized faces are displayed in separate picture boxes. The picture box on the left displays the detected face and the picture box on the right displays the recognized face. Only one detected face and one recognized face are displayed at a time. The 4 navigation buttons are used to navigate between all the faces. The student id, first name and last name corresponding to the recognized face are displayed in the text labels. The selected face recognition method is also displayed in one of the text labels. As discussed previously, all the available face recognition methods find the distance between the detected face and each face in the training set. The face in the training set that generates the minimum distance is used as the recognized face. The minimum distance parameter is available in the Emgu CV library. This value is displayed in one of the text labels. Figure 4.8 illustrates the layout of the face recognition application.

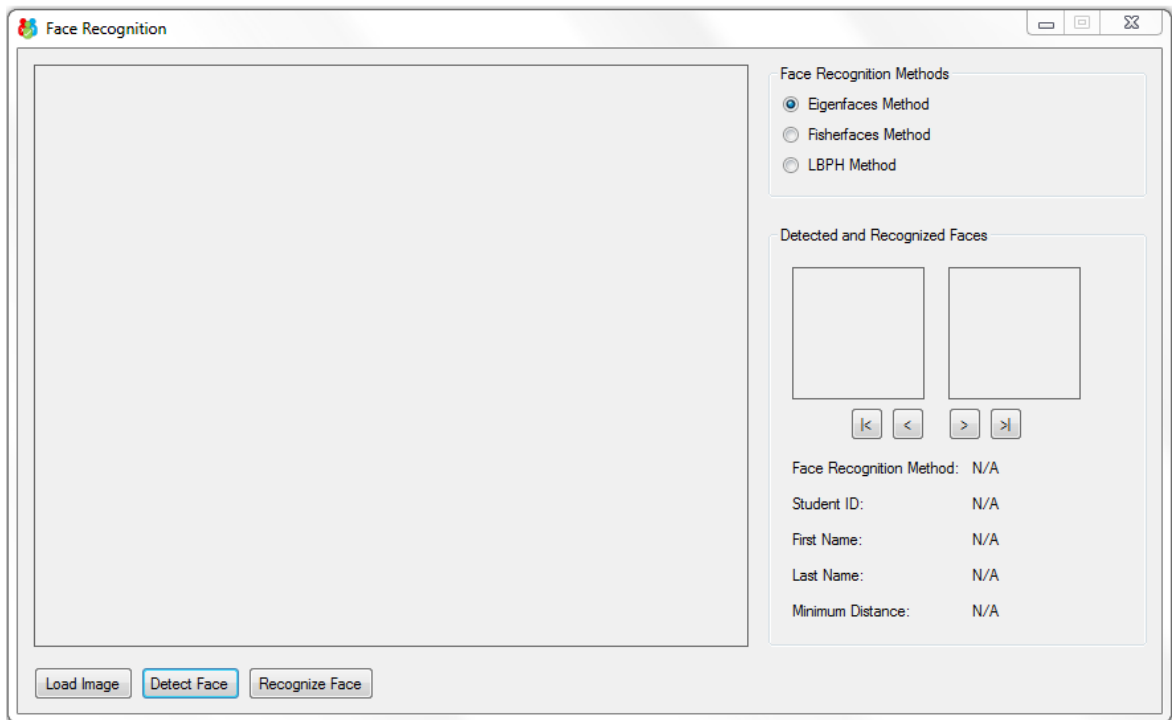


Figure 4.8: Layout of face recognition application

A face database was created as discussed previously and is stored on the webserver. In order to retrieve the images in the face database, the application reads the data stored in the student face information table in the database. The retrieved face images are converted to grayscale images and histogram equalization is performed on the images. Before the recognition process can begin, a face recognizer class must first be created for the selected face recognition method. For example, if the Eigenfaces method was selected, then the EigenFaceRecognizer class must be created. There are 3 face recognition classes which are available in the Emgu CV library:

```
public EigenFaceRecognizer(
    int numComponents,
    double threshold
)
```

(Emgu 2013b)

```
public FisherFaceRecognizer(
    int numComponents,
    double threshold
)
```

(Emgu 2013e)

```
public LBPHFaceRecognizer(
    int radius,
    int neighbors,
    int gridX,
    int gridY,
    double threshold
)
```

(Emgu 2013f)

The EigenFaceRecognizer class constructor requires 2 parameters: the number of components and the threshold. As discussed before, in the Eigenfaces method the eigenvectors with the largest eigenvalues are selected. The number of components parameter is the number of eigenvectors selected. A low value results in a faster execution time. However, more information is lost since less eigenvectors are selected. A high value results in a slower execution time. However, less information is lost since more eigenvectors are selected. The maximum number of eigenvectors that can be selected is the total number of faces in the training set. The threshold parameter determines if the detected face corresponds with a face in the training set. If the minimum distance is below the threshold value then the detected face corresponds to the face in the training set with the minimum distance, else the face is unknown. A high value results in a higher probability of the detected face being incorrectly matched with a face in the training set. A low value results in a higher probability of the detected face being classified as unknown. The FisherFaceRecognizer class constructor requires the same

parameters as the EigenFaceRecognizer class constructor. Similar to the Eigenfaces method, in the Fisherfaces method the largest components are selected and the threshold determines if the detected face corresponds with a face in the training set. The LBPHFaceRecognizer class constructor requires 5 parameters: the radius, the neighborhood size, the width of the region, the height of the region and the threshold. A large radius, neighborhood size and region size result in poor recognition performance since more information is lost. Similar to the previous face recognizer class constructors, the threshold determines if the detected face corresponds with a face in the training set. After creating the face recognizer class, the class needs to be trained by calling the Train function. This function trains the face recognizer with the labels and face images in the training set. The syntax for this function is given as follows:

```
public void Train(
    IImage[] images,
    int[] labels
)
```

(Emgu 2013d)

The faces are extracted using the face detection process from the face detection application. A modification was made to the process such that the detected faces are stored in a variable. Each detected face is converted to a grayscale image and resized to the size of the image in the training set. Histogram equalization is then performed on the detected face images. Histogram equalization distributes the pixel intensities evenly in the image. This is done to improve the contrast of the image. The Predict function from the face recognizer class is used to recognize the detected faces. This function returns the index of the face image in the training set which

corresponds to the detected face. If the minimum distance exceeds the threshold, the function returns -1. The syntax for the Predict function is given as follows:

```
public FaceRecognizer.PredictionResult Predict(  
    IImage image  
)
```

(Emgu 2013c)

The minimum distance parameter is obtained from the face recognizer class for each recognized face. If the output from the Predict function is -1, the student id, first name and last name are set to “Unknown”. The image is set to a customized image indicating that the face is unknown. If the output from the Predict function is not -1, the student id, first name and last name are set to the corresponding fields for the recognized face. The image is set to the face of the recognized person. Figure 4.9 illustrates the flowchart for the face recognition application.

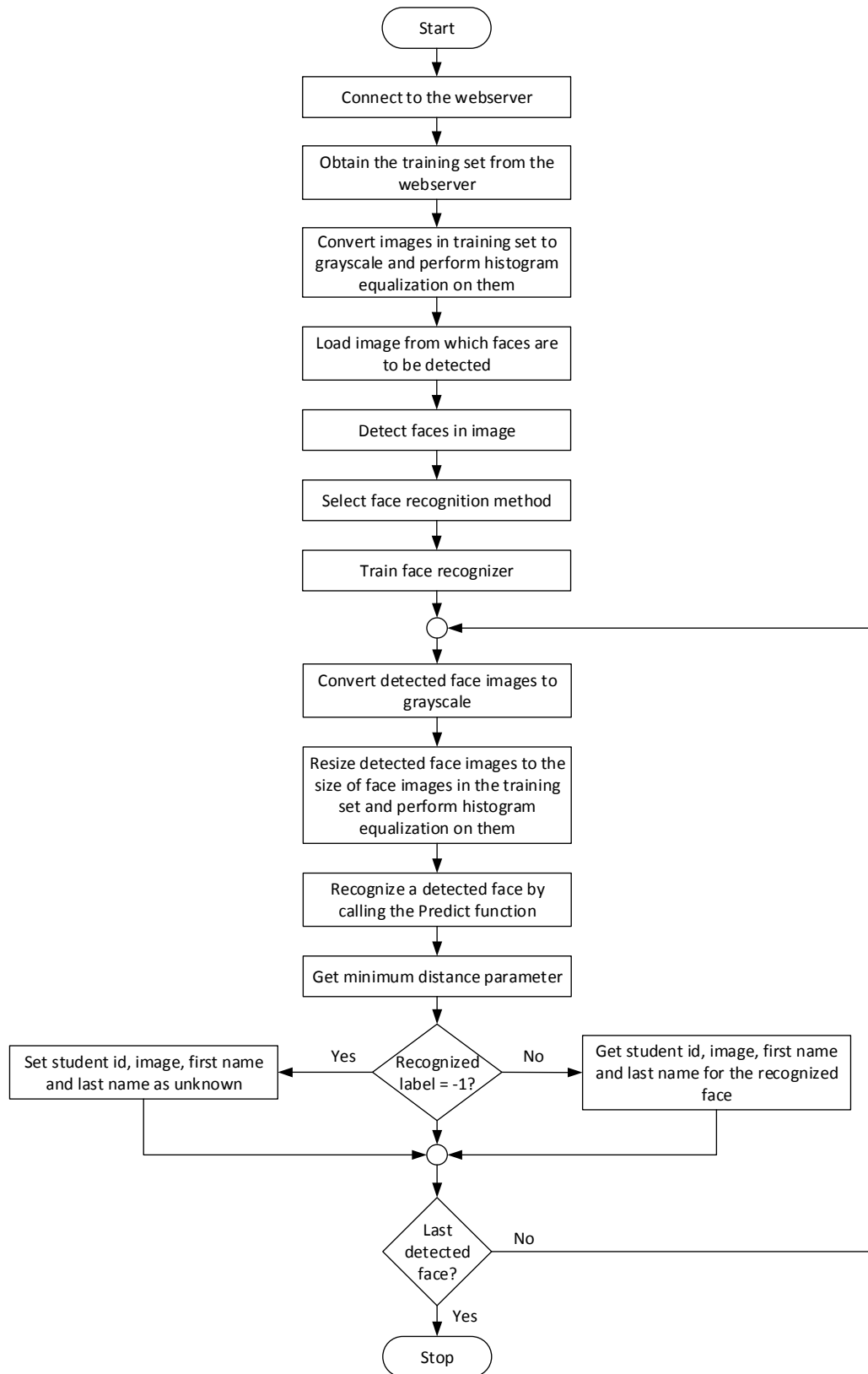


Figure 4.9: Flowchart for face recognition application

In constructing the EigenFaceRecognizer class, the number of components was set to 160 since there were 160 images in the training set. The number of components should be less than the number of faces in the training set so that the computational time is reduced. However, the number of components was set to the number of faces in the training set so that no information would be lost. The threshold was determined by first setting the threshold value to infinity. This results in no faces being labelled as unknown. The face recognition application was used to recognize the faces in the test images described in Table 5.2 in Chapter 5 Testing and Results for different resolutions. The parameters for face detection were set to 1.1, 24 and 5 for the scaling factor, minimum window size and minimum number of neighbor rectangles respectively. Eye detection was not used in this case for finding the threshold, but was used in the final face recognition application and class attendance application. This was done to obtain as much falsely detected faces as possible. The face recognition application will incorrectly recognize the falsely detected faces. A histogram was done on the minimum distances for the incorrect recognitions and the correct recognitions as seen in Figure 4.10.

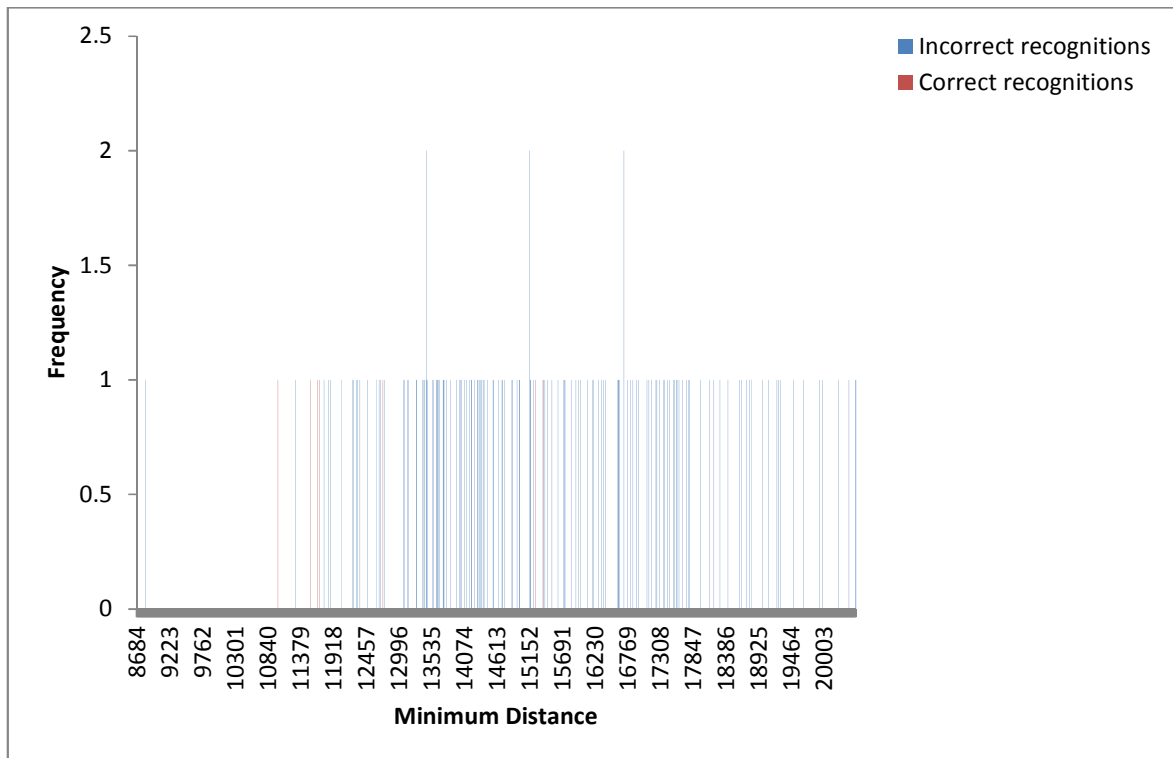


Figure 4.10: Histogram of minimum distances for Eigenfaces method

It is expected that the minimum distances for the correctly recognized faces should be at the lower end of the histogram and the minimum distances for the incorrectly recognized faces should be at the higher end of the histogram. Figure 4.10 shows that the minimum distances for the correct recognition fall within the range of the minimum distances for the incorrect recognition. This occurrence is caused by any flaws of the face recognition method. However, most of the minimum distances for the correct recognition still lie at the lower end of the histogram. It can be seen that the high frequency minimum distances for the incorrect recognitions are clustered around a center value. The average of the minimum distances for the incorrect recognitions was used as the threshold parameter. This value was found to be 15530 and is used for the threshold parameter.

Similar to constructing the EigenFaceRecognizer class, the number of components for the FisherFaceRecognizer class was set to 160 since there were 160 images in the training set. The number of components should be less than the number of faces in the training set so that the computational time is reduced. However, the number of components was set to the number of faces in the training set so that no information would be lost. The threshold for the FisherFaceRecognizer class was found using the same procedure for the EigenFaceRecognizer class. The histogram of the minimum distances seen in Figure 4.11 is similar to the one for the EigenFaceRecognizer class. The average of the minimum distances for the incorrect recognitions was used as the threshold parameter. This value was found to be 14217 and is used for the threshold parameter.

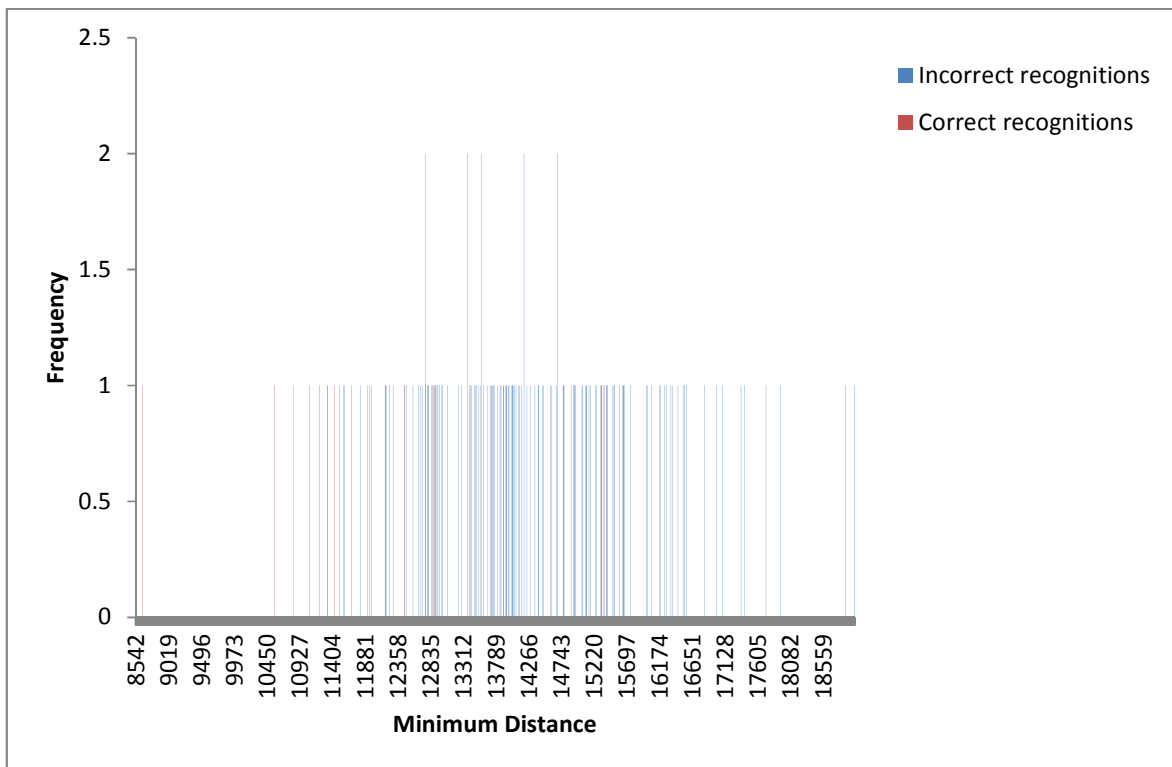


Figure 4.11: Histogram of minimum distances for Fisherfaces method

The radius, neighborhood size, width of the region and height of the region for the LBPHFaceRecognizer class were selected such that little information is lost. Therefore, the minimum values for the parameters were selected. The radius, neighborhood size, width of the region and height of the region were 1, 8, 8 and 8 respectively. The threshold for the LBPHFaceRecognizer class was found using the same procedure for the EigenFaceRecognizer class. Figure 4.12 shows that minimum distances for the correct recognitions lie at the lower end of the histogram. It can be seen that the high frequency minimum distances for the incorrect recognitions are clustered around a center value. The average of the minimum distances for the incorrect recognitions was used as the threshold parameter. This value was found to be 477 and is used for the threshold parameter.

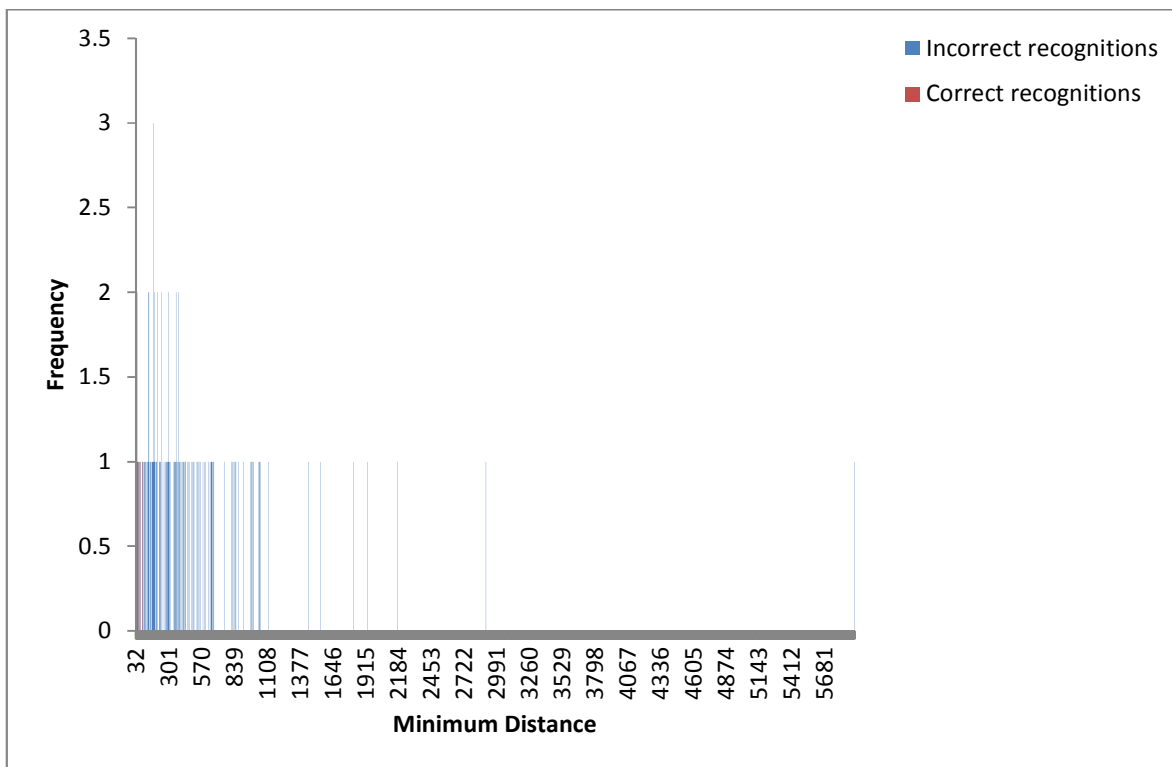


Figure 4.12: Histogram of minimum distances for LBPH method

A series of tests was performed on the face recognition methods. The results in Chapter 5 Testing and Results show that the Eigenfaces method was the best. Therefore, this face recognition method was used in the ECNG Class Attendance system. For best results, the maximum distance from the camera is restricted to 2.54 m. Furthermore, the individual has to look straight ahead, without turning or tilting the head. Additionally, the face database only contained faces that are in the frontal position with no smile.

4.3.4 Class Attendance

The class attendance application consists of 2 date time pickers, 1 textbox, 1 list box and 7 buttons. Figure 4.13 illustrates the layout of the class attendance application main window. When the application first starts, the main window is displayed. Figure 4.14 illustrates the flowchart for the class attendance application main window. As seen in Figure 4.14, the application will first connect to the database on the webserver. It will then populate the list box with the courses from the course information table in the database. The desired course must be selected from the list box. The Update Attendance Records button is then used to update the attendance records table for the selected course in the database. After updating the attendance records, the period for the class attendances is selected by using the date time pickers. The textbox is then used to enter the minimum attendance or the attendance requirement. At this point, the user can perform any of the following actions by using the respective button:

- Get class attendance
- Get student attendance
- Get attendance offenders
- Get attendance record table
- Get student information
- Launch face recognition application

The Get Class Attendance button is used to get the attendance of the entire class for the specified course and period. A chart is displayed where the y axis represents the class attendance (%) and the x axis represents the date. The Get Student Attendance button is used to get the attendance of the students for the specified course and period. A plot is displayed where the y axis represents the student attendance (%) and the x axis represents the student ids.

A line representing the attendance requirement is drawn on the plot. The Get Attendance Offenders button is used to get a list of all the students with attendances that are below the attendance requirement for the specified course and period. The student's id, first name, last name, image and attendance (%) are displayed. The Get Attendance Record Table button is used to get the attendance record table for the specified course and period. A table is displayed showing the present students with their student ids, first names and last names, as well as the attendance dates. The Student Info button is used to get information on all the students. The student's id, first name, last name, image and the courses which the student is registered for are displayed. The Face Recognition button is used to launch the face recognition application. The face recognition application was included to test the class attendance system. Each button except the Update Attendance Records button opens its respective window, for example, the Get Class Attendance button opens the Class Attendance Result window.

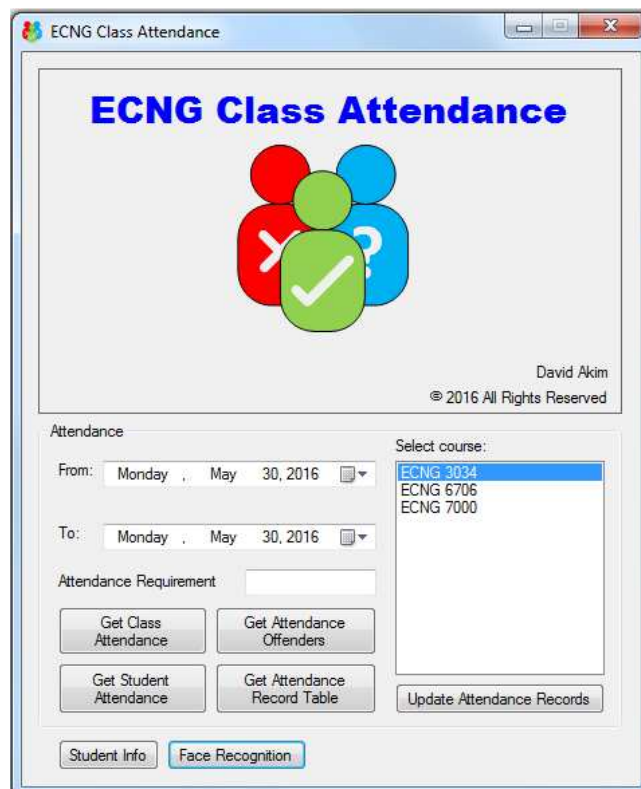


Figure 4.13: Layout of class attendance application main window

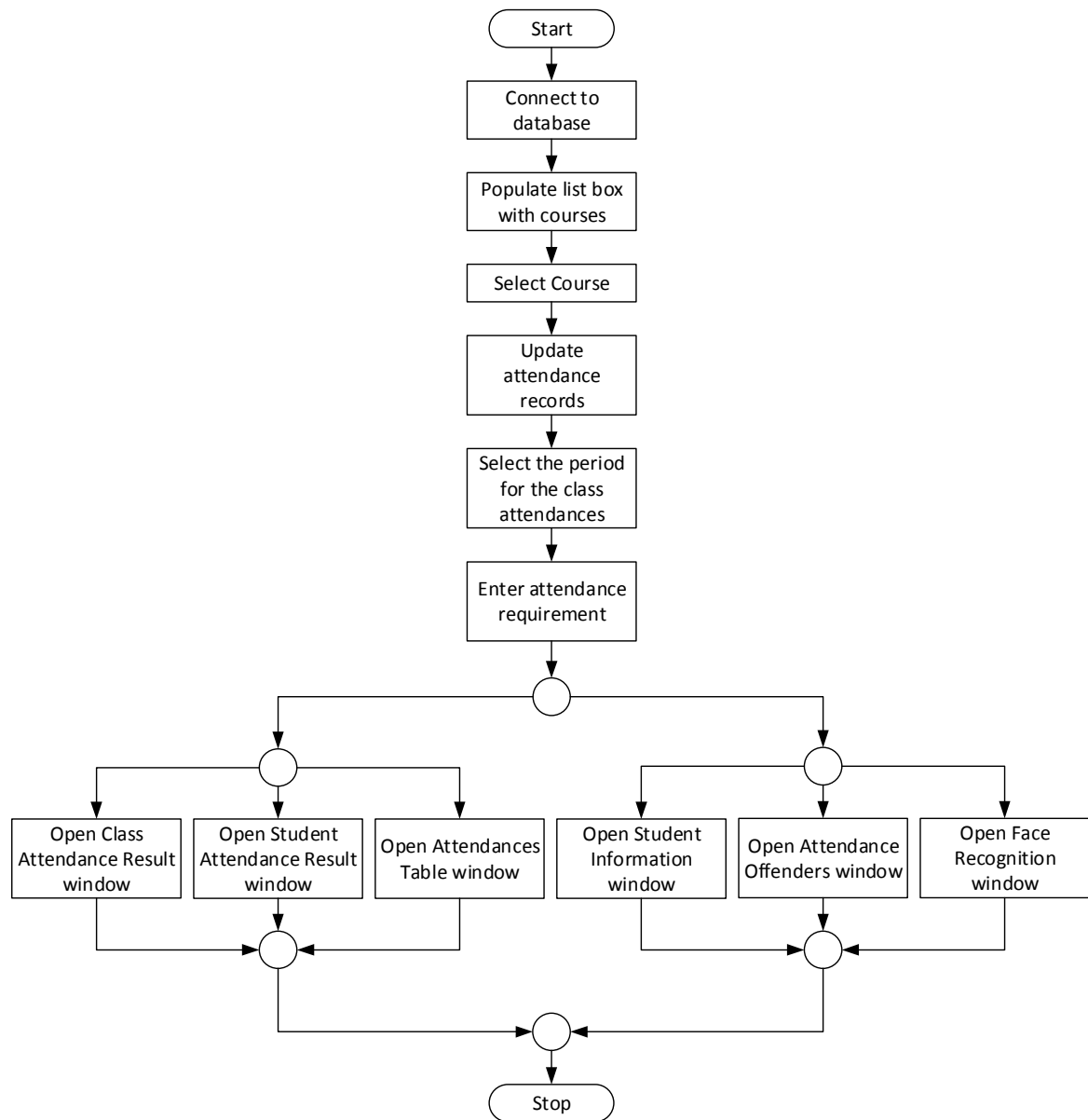


Figure 4.14: Flowchart for class attendance application main window

Updating attendance records

Figure 4.15 illustrates the flowchart for updating the attendance records. The first step is to retrieve the images in the training set from the webserver. The images are then converted to grayscale and histogram equalization is performed on them. The attendance dates for the uploaded images are retrieved from the table containing the captured image information in the database. Since, there can be multiple class images with the same attendance dates, only distinct dates for the specified course were retrieved. The next step is to retrieve the class images for the specified course from same table in the database. The students' ids, first names and last names for the desired course are then retrieved from the course information table in the database. After obtaining the identities of the students for the course, the attendance dates are retrieved from the attendance records table for the specified course in the database. All the class images are examined in an iterative process. At each step in the iterative process, the application checks if the attendance date for the uploaded image is already in the attendance record. This is done by comparing the attendance date for the uploaded image with all the attendance dates in the attendance record. If the attendance date exists in the attendance record, no processing is done on the image. If the attendance date does not exist, further processing is done on the image. Face detection and face recognition are performed on the image. The application then checks if any students were recognized. If no students were recognized, no further processing is done. If students were recognized, the application checks if the recognized student belongs to the specified course. This was done by comparing the recognized student's id with all the students' ids for the specified course. If the recognized student does not belong to the specified course, no further processing is done. If the recognized student belong to the specified course, the student is marked as present. The application then checks if the present

student is already in the attendance record with respect to the course and date. If the student already exists in the attendance record, no further processing is done. If the student does not exist in the attendance record, the student is added to the attendance record. The student's id, first name, last name and attendance date is added to the attendance record table in the database for the specified course. At the end of the iterative process, the application displays a message indicating that the operation is complete.

Get class attendance

Figure 4.16 illustrates the flowchart for getting the class attendance. The application first retrieves all the attendance dates for the specified course and period from the attendance record table in the database. If there were no attendance dates, the application displays a message indicating that there is no data for the specified course and period. If there were attendance dates, the Class Attendance Result window is opened. After opening the window, the total number of students for the specified course is obtained using the course information table in the database. The application then sets the chart title and labels the axes. The attendance dates are examined in an iterative process. At each step of the iteration the total number of students present for a specific date is found using the attendance record table for the specified course. The class attendance percentage is calculated from the total number of present students found and the total number of students for the course. The percentage of present students is plotted on a chart where the y axis represents the class attendance (%) and the x axis represents the attendance dates. The iterative process ends when it reaches the last attendance date for the specified period.

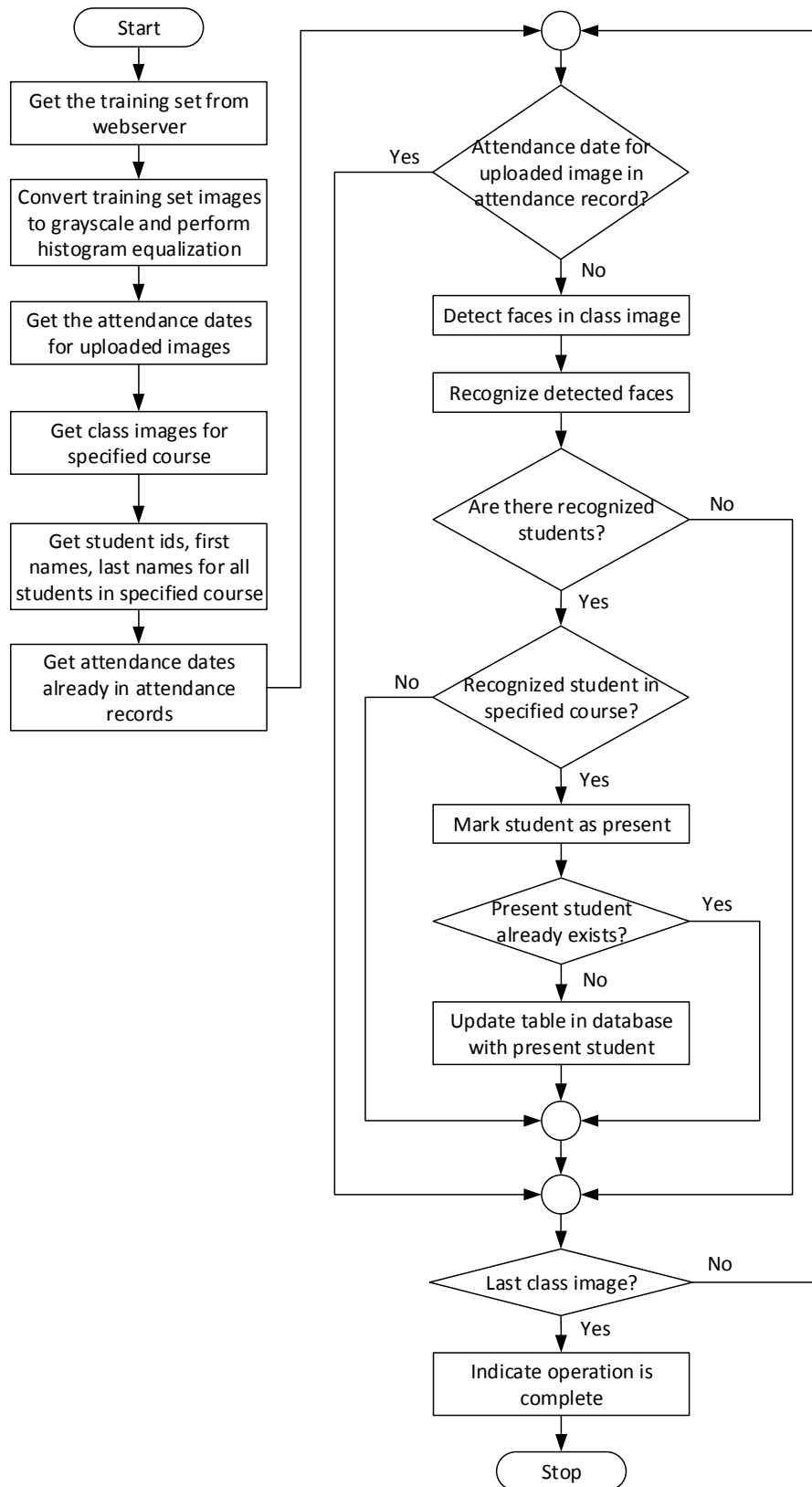


Figure 4.15: Flowchart for updating attendance records

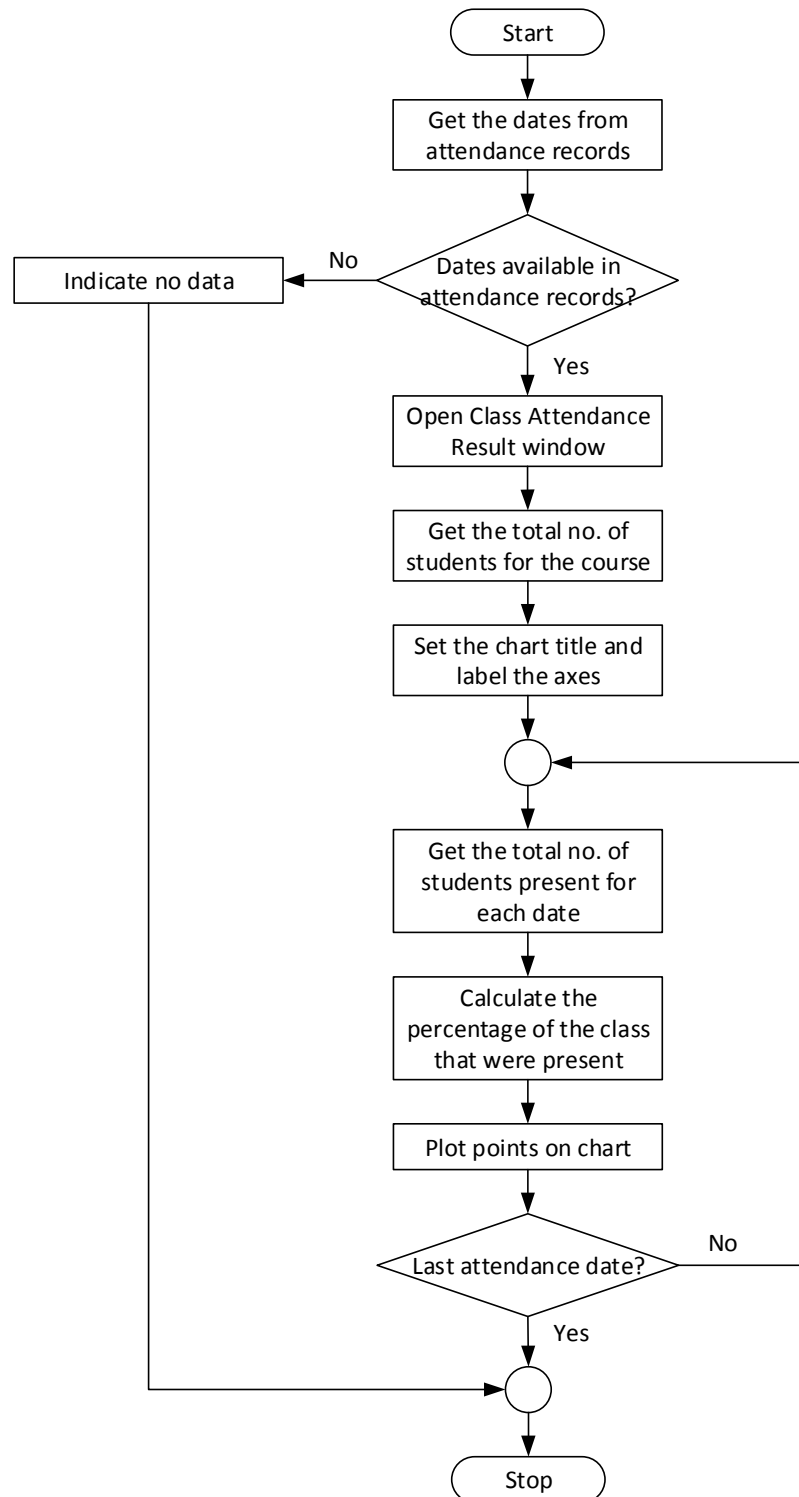


Figure 4.16: Flowchart for getting the class attendance

Get student attendance

Figure 4.17 illustrates the flowchart for getting the student attendance. The application first retrieves all the attendance dates for the specified course and period from the attendance record table in the database. If there were no attendance dates, the application displays a message indicating that there is no data for the specified course and period. If there were attendance dates, the Student Attendance Result window is opened. After opening the window, the student ids for the selected course are obtained from the course information table in the database. The application then retrieves the total number of attendance days or classes within the specified period using the attendance record table for the specified course in the database. The chart title is set and the axes are labelled. The student ids are examined in an iterative process. At each step of the iterative process, the total number of days present for a particular student is obtained. The student attendance percentage is calculated from the total number of days present and the total number of classes. The percentage of attendance is plotted on a chart where the y axis represents the student attendance (%) and the x axis represents the student id. The iterative process ends when it reaches the last student id.

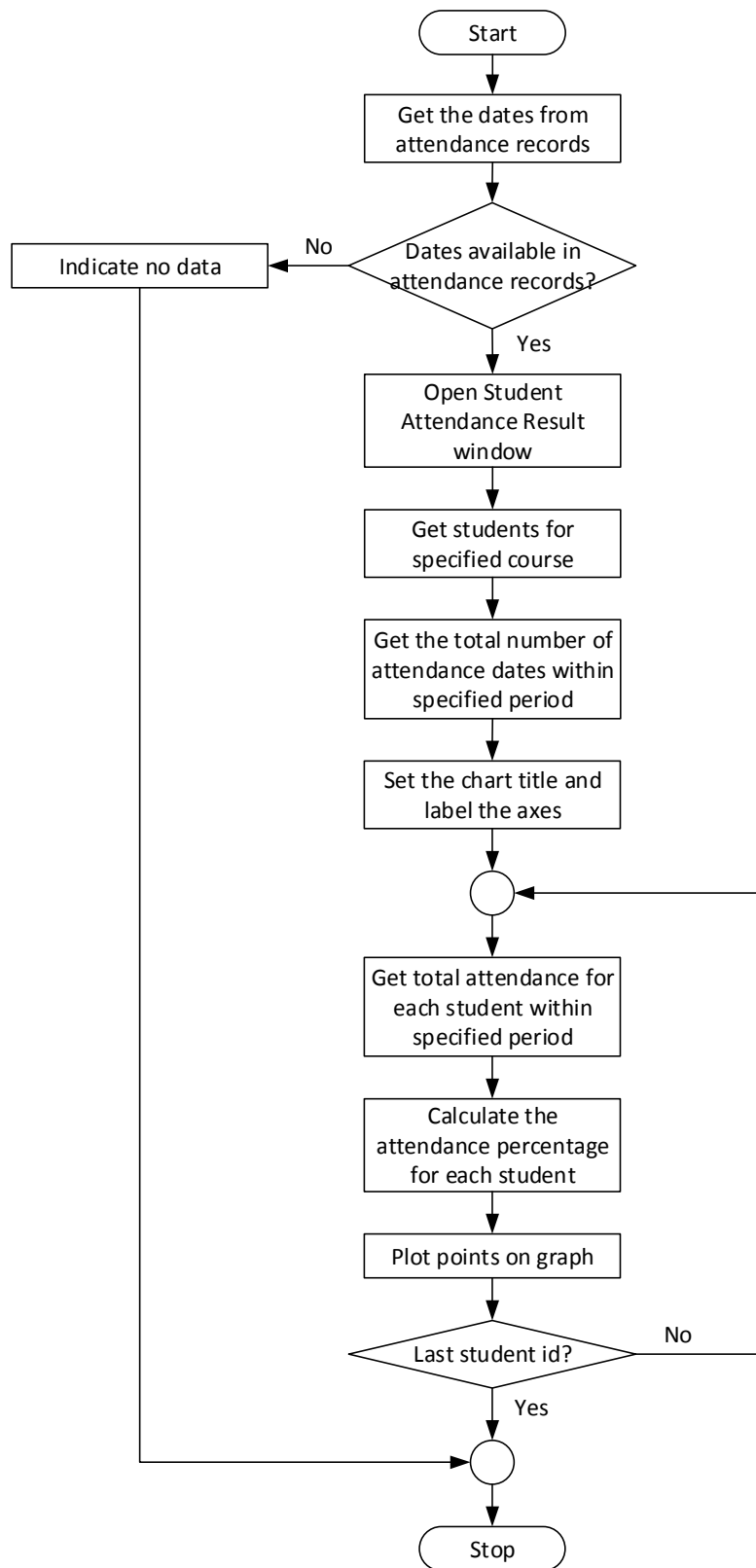


Figure 4.17: Flowchart for getting the student attendance

Get student information

Figure 4.18 illustrates the flowchart for getting the student information. The flowchart shows that the Student Information window is first opened. After opening the window, the students' ids were obtained from the student face information table in the database. The list box is then populated with the students' ids. After obtaining the students' ids, the students' first names, last names and images are retrieved from the student face information table in the database. The students' courses are retrieved from the course information table in the database. The student's information is displayed after being retrieved. The information is displayed for one student at a time. If the user wants to see the information for another student, the user must make another selection in the list box.

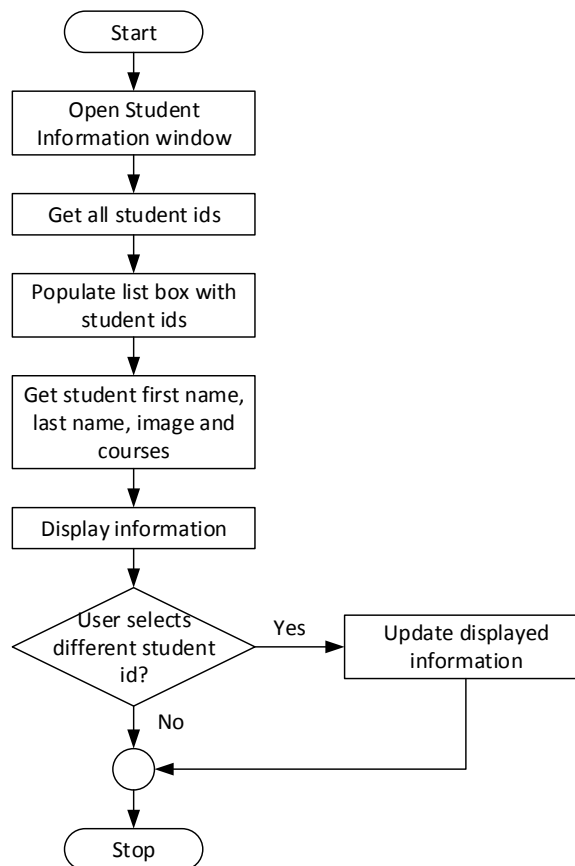


Figure 4.18: Flowchart for getting student information

Get attendance offenders

Figure 4.19 illustrates the flowchart for getting the attendance offenders. The application first retrieves the attendance dates for the specified course and period from the attendance record table in the database. The text in the Attendance Requirement field is then converted to an integer. If there were no attendance dates, the application displays a message indicating that there is no data for the specified course and period. If there were attendance dates, further processing is done. The student ids for the selected course are retrieved from the course information table in the database. After retrieving the student ids, the application obtains the total number of attendance days or classes within the specified period using the attendance record table for the specified course in the database. The student ids are examined in an iterative process. At each step of the iterative process, the total number of days present for a particular student is obtained. The attendance percentage is calculated from the total number of days present and the total number of classes. The attendance percentage for the student is then compared with the attendance requirement. If the attendance percentage is greater than the attendance requirement, no further processing is done. If the attendance percentage is less than the attendance requirement, the student id and attendance percentage is stored. The iterative process ends when it reaches the last student id. The stored offenders are then displayed on the Attendance Offenders window. The algorithm that is executed when this window opens is similar to one described in the Get Student Information section.

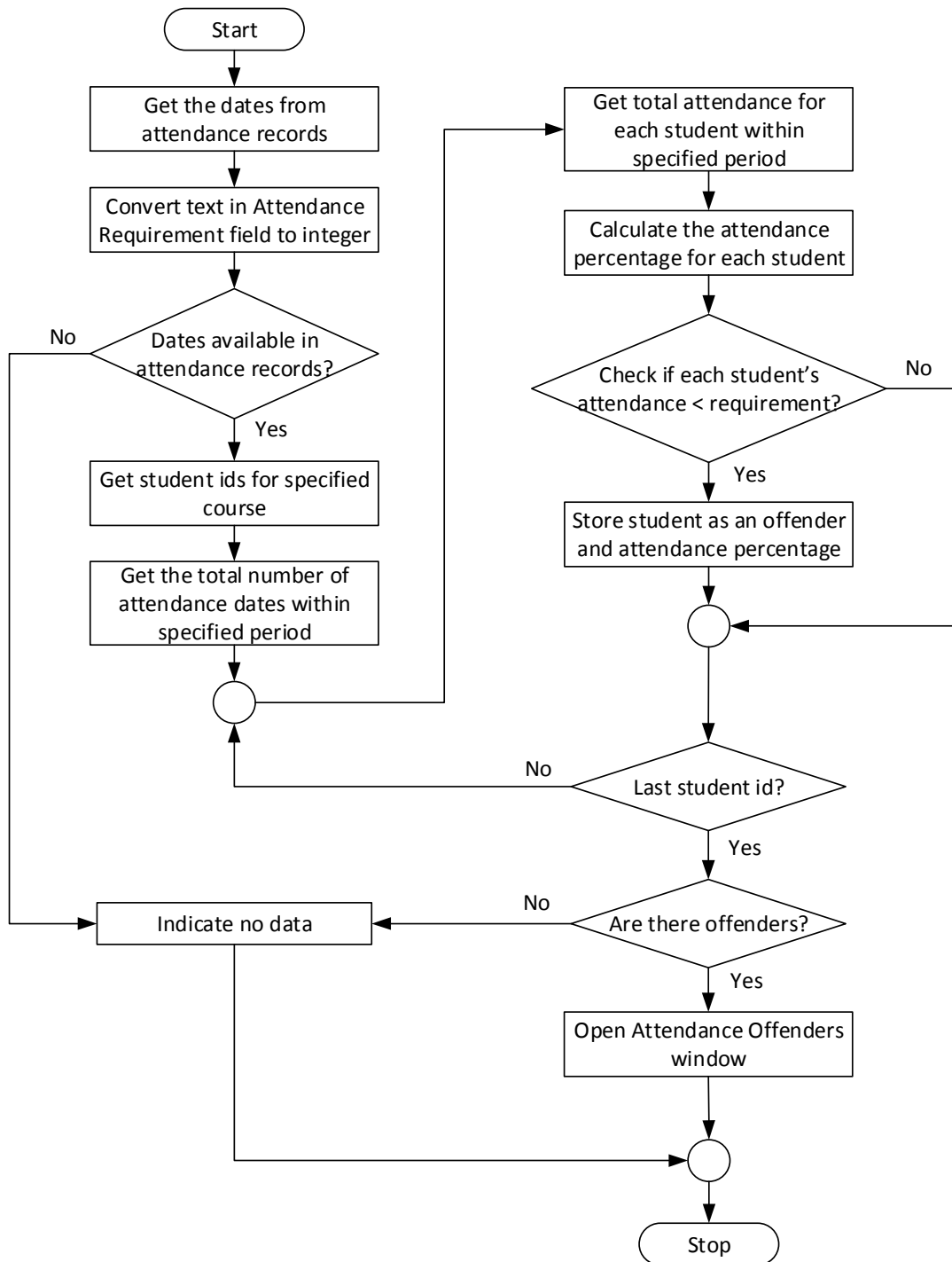


Figure 4.19: Flowchart for getting attendance offenders

Get attendance record table

Figure 4.20 illustrates the flowchart for getting the attendance record table. The application first retrieves the data table for specified course and period. If the data table has no data, the application displays a message indicating that there is no data for the specified course and period. If the data table has data, the Attendances Table window is opened. This table shows all the present students for the specified course and period.

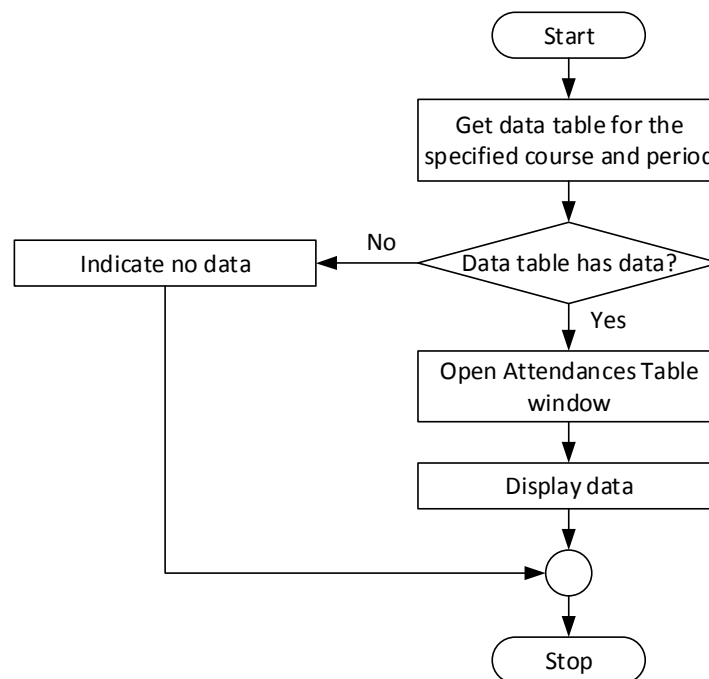


Figure 4.20: Flowchart for getting the attendance record table

Chapter 5 Testing and Results

5.1 Android Application

The following table shows the tests and results for the Android application.

Table 5.1: Android application tests and results

Test	Result
Installation on Android smartphone	Success
Capturing image	Success
Saving image on smartphone	Success
Loading image from smartphone	Success
Displaying captured or loaded image	Success
Selecting Course	Success
Selecting Date	Success
Uploading image to webserver	Success
Saving uploaded image on the webserver	Success
Updating database on the webserver	Success

5.2 Face Detection Application

As mentioned before, the students are expected to face the camera when taking their photos.

However, there could be the possibility of head movements. Therefore, images were captured for different head positions in order to evaluate the face detection algorithm under such conditions. Furthermore, the face detection algorithm can be affected by the size of the class, specifically the distance from the front of the class to the back of the class. A volunteer was subjected to the following conditions:

- Different angles for a single side view
- Different up/down head tilts
- Different distances

Table 5.2 describes the test conditions for the face detection algorithm. The angle between the volunteer facing the camera and the volunteer facing to the side is called the side angle (Refer to Figure 5.1). The head tilt was not measured since it was difficult to obtain a reference point. A 13 MP camera from a Samsung Galaxy S4 was used to capture the images, which produced an image size of 4128×2322 . For the constant distance test conditions, the distance between the volunteer and the camera was 1.13 m. Figure 5.1 illustrates the setup for the constant distance tests. For the varying distance conditions, a classroom was used since it is the environment for capturing the images of students. The volunteer sat at different locations starting from the front of the class to the back of the class. The volunteer also looked at the camera with no side angle and no head tilt.

Table 5.2: Face Detection Test Conditions

Tests	Description
Test 1	No head tilt, 0 degree side angle, constant distance
Test 2	No head tilt, 10 degree side angle, constant distance
Test 3	No head tilt, 20 degree side angle, constant distance
Test 4	No head tilt, 30 degree side angle, constant distance
Test 5	No head tilt, 90 degree side angle, constant distance
Test 6	Little upward head tilt, 0 degree side angle, constant distance
Test 7	Moderate upward head tilt, 0 degree side angle, constant distance
Test 8	Large upward head tilt, 0 degree side angle, constant distance
Test 9	Little downward head tilt, 0 degree side angle, constant distance
Test 10	Moderate downward head tilt, 0 degree side angle, constant distance
Test 11	Large downward head tilt, 0 degree side angle, constant distance
Test 12	No head tilt, 0 degree side angle, distance of 1.65 m
Test 13	No head tilt, 0 degree side angle, distance of 2.54 m
Test 14	No head tilt, 0 degree side angle, distance of 3.43 m
Test 15	No head tilt, 0 degree side angle, distance of 4.32 m
Test 16	No head tilt, 0 degree side angle, distance of 5.21 m
Test 17	No head tilt, 0 degree side angle, distance of 6.10 m
Test 18	No head tilt, 0 degree side angle, distance of 6.99 m
Test 19	No head tilt, 0 degree side angle, distance of 7.87 m

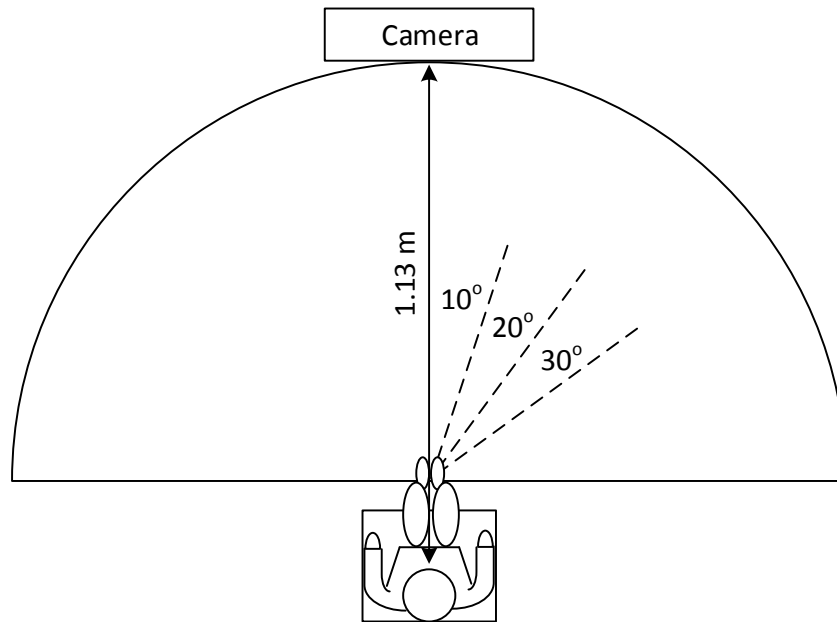


Figure 5.1: Setup for testing face detection algorithm for a constant distance

The ECNG Class Attendance system was designed for a 13 MP camera. However, different camera resolutions were also investigated in testing the face detection algorithm. In addition to the 13 MP camera from a Samsung Galaxy S4, a 5 MP camera from a Sony Ericsson Xperia active and an 8 MP camera from a Titans2 DG700 Doogee were also used. Test 12 to Test 19 were repeated under different camera resolutions. Test 1 to Test 11 were not repeated because the distances in those tests were constant and close to the camera.

The face detection algorithm was evaluated in terms of hit rate, miss rate and false positive rate. The hit rate is defined as the total number of correctly detected faces divided by the total number of actual faces in the image. The miss rate is defined as the total number of undetected faces divided by the total number of actual faces in the image. The false positive rate is defined as the total number of incorrectly detected faces divided by the total number of detected faces in the image. The hit rate, miss rate and false positive rate are expressed as a percentage.

Notes:

- Observed face is the face that is subjected to the test. For example a face that is 1.65 m away from the camera. There is only one observed face.
- AF – No. of actual faces
- DF – No. of detected faces
- MF – No. of missed faces
- FD – No. of false detections

Table 5.3: Hit, Miss and False Positive Rates for all the tests using 13 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 1	1	7	0	6	100.00	0.00	85.71	Yes
Test 2	1	8	0	7	100.00	0.00	87.50	Yes
Test 3	1	8	0	7	100.00	0.00	87.50	Yes
Test 4	1	8	0	7	100.00	0.00	87.50	Yes
Test 5	1	12	1	12	0.00	100.00	100.00	No
Test 6	1	10	0	9	100.00	0.00	90.00	Yes
Test 7	1	13	0	12	100.00	0.00	92.31	Yes
Test 8	1	9	0	8	100.00	0.00	88.89	Yes
Test 9	1	10	0	9	100.00	0.00	90.00	Yes
Test 10	1	5	0	4	100.00	0.00	80.00	Yes
Test 11	1	8	0	7	100.00	0.00	87.50	Yes
Test 12	2	3	1	2	50.00	50.00	66.67	Yes
Test 13	2	10	1	9	50.00	50.00	90.00	Yes
Test 14	2	11	0	9	100.00	0.00	81.82	Yes
Test 15	1	8	0	7	100.00	0.00	87.50	Yes
Test 16	2	6	0	4	100.00	0.00	66.67	Yes
Test 17	2	11	0	9	100.00	0.00	81.82	Yes
Test 18	2	9	0	7	100.00	0.00	77.78	Yes
Test 19	1	7	0	6	100.00	0.00	85.71	Yes

Table 5.4: Hit, Miss and False Positive Rates for distance tests using 8 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 12	1	5	0	4	100.00	0.00	80.00	Yes
Test 13	1	5	0	4	100.00	0.00	80.00	Yes
Test 14	1	5	0	4	100.00	0.00	80.00	Yes
Test 15	1	4	0	3	100.00	0.00	75.00	Yes
Test 16	2	4	1	3	50.00	50.00	75.00	Yes
Test 17	2	3	1	2	50.00	50.00	66.67	Yes
Test 18	2	4	1	3	50.00	50.00	75.00	Yes
Test 19	2	2	1	1	50.00	50.00	50.00	Yes

Table 5.5: Hit, Miss and False Positive Rates for distance tests using 5 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 12	2	5	0	3	100.00	0.00	60.00	Yes
Test 13	1	5	0	4	100.00	0.00	80.00	Yes
Test 14	1	6	0	5	100.00	0.00	83.33	Yes
Test 15	2	4	0	2	100.00	0.00	50.00	Yes
Test 16	2	3	0	1	100.00	0.00	33.33	Yes
Test 17	2	4	0	2	100.00	0.00	50.00	Yes
Test 18	2	4	0	2	100.00	0.00	50.00	Yes
Test 19	1	3	0	2	100.00	0.00	66.67	Yes

Table 5.6: Hit, Miss and False Positive Rates for all the tests after re-adjusting the parameters using 13 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 1	1	1	0	0	100.00	0.00	0.00	Yes
Test 2	1	3	0	2	100.00	0.00	66.67	Yes
Test 3	1	2	0	1	100.00	0.00	50.00	Yes
Test 4	1	2	0	1	100.00	0.00	50.00	Yes
Test 5	1	0	1	0	0.00	100.00	0.00	No
Test 6	1	3	0	2	100.00	0.00	66.67	Yes
Test 7	1	2	0	1	100.00	0.00	50.00	Yes
Test 8	1	2	0	1	100.00	0.00	50.00	Yes
Test 9	1	3	0	2	100.00	0.00	66.67	Yes
Test 10	1	2	0	1	100.00	0.00	50.00	Yes
Test 11	1	1	0	0	100.00	0.00	0.00	Yes
Test 12	2	1	1	0	50.00	50.00	0.00	Yes
Test 13	2	1	1	0	50.00	50.00	0.00	Yes
Test 14	2	2	1	1	50.00	50.00	50.00	Yes
Test 15	1	2	0	1	100.00	0.00	50.00	Yes
Test 16	2	3	0	1	100.00	0.00	33.33	Yes
Test 17	2	2	0	0	100.00	0.00	0.00	Yes
Test 18	2	3	1	2	50.00	50.00	66.67	Yes
Test 19	1	1	0	0	100.00	0.00	0.00	Yes

Table 5.7: Hit, Miss and False Positive Rates for distance tests after re-adjusting the parameters using 8 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 12	1	1	0	0	100.00	0.00	0.00	Yes
Test 13	1	3	0	2	100.00	0.00	66.67	Yes
Test 14	1	2	0	1	100.00	0.00	50.00	Yes
Test 15	1	1	0	0	100.00	0.00	0.00	Yes
Test 16	2	1	1	0	50.00	50.00	0.00	Yes
Test 17	2	1	1	0	50.00	50.00	0.00	Yes
Test 18	2	1	2	1	0.00	100.00	100.00	No
Test 19	2	1	1	0	50.00	50.00	0.00	Yes

Table 5.8: Hit, Miss and False Positive Rates for distance tests after re-adjusting the parameters using 5 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 12	2	1	1	0	50.00	50.00	0.00	Yes
Test 13	1	1	0	0	100.00	0.00	0.00	Yes
Test 14	1	3	0	2	100.00	0.00	66.67	Yes
Test 15	2	1	1	0	50.00	50.00	0.00	Yes
Test 16	2	1	1	0	50.00	50.00	0.00	Yes
Test 17	2	1	1	0	50.00	50.00	0.00	Yes
Test 18	2	2	1	1	50.00	50.00	50.00	Yes
Test 19	1	1	0	0	100.00	0.00	0.00	Yes

Table 5.9: Hit, Miss and False Positive Rates for all the tests with the aid of eye detection using 13 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 1	1	1	0	0	100.00	0.00	0.00	Yes
Test 2	1	1	0	0	100.00	0.00	0.00	Yes
Test 3	1	1	0	0	100.00	0.00	0.00	Yes
Test 4	1	1	0	0	100.00	0.00	0.00	Yes
Test 5	1	0	1	0	0.00	100.00	0.00	No
Test 6	1	1	0	0	100.00	0.00	0.00	Yes
Test 7	1	1	0	0	100.00	0.00	0.00	Yes
Test 8	1	1	0	0	100.00	0.00	0.00	Yes
Test 9	1	1	0	0	100.00	0.00	0.00	Yes
Test 10	1	1	0	0	100.00	0.00	0.00	Yes
Test 11	1	1	0	0	100.00	0.00	0.00	Yes
Test 12	2	1	1	0	50.00	50.00	0.00	Yes
Test 13	2	1	1	0	50.00	50.00	0.00	Yes
Test 14	2	0	2	0	0.00	100.00	0.00	No
Test 15	1	0	1	0	0.00	100.00	0.00	No
Test 16	2	0	2	0	0.00	100.00	0.00	No
Test 17	2	0	2	0	0.00	100.00	0.00	No
Test 18	2	0	2	0	0.00	100.00	0.00	No
Test 19	1	0	1	0	0.00	100.00	0.00	No

Table 5.10: Hit, Miss and False Positive Rates for distance tests with the aid of eye detection using 8 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 12	1	1	0	0	100.00	0.00	0.00	Yes
Test 13	1	1	0	0	100.00	0.00	0.00	Yes
Test 14	1	1	0	0	100.00	0.00	0.00	Yes
Test 15	1	0	1	0	0.00	100.00	0.00	No
Test 16	2	1	1	0	50.00	50.00	0.00	Yes
Test 17	2	1	1	0	50.00	50.00	0.00	Yes
Test 18	2	0	2	0	0.00	100.00	0.00	No
Test 19	2	0	2	0	0.00	100.00	0.00	No

Table 5.11: Hit, Miss and False Positive Rates for distance tests with the aid of eye detection using 5 MP camera

Tests	AF	DF	MF	FD	Hit Rate (%)	Miss Rate (%)	False Positive Rate (%)	Observed face detected?
Test 12	2	1	1	0	50.00	50.00	0.00	Yes
Test 13	1	1	0	0	100.00	0.00	0.00	Yes
Test 14	1	1	0	0	100.00	0.00	0.00	Yes
Test 15	2	0	2	0	0.00	100.00	0.00	No
Test 16	2	1	1	0	50.00	50.00	0.00	Yes
Test 17	2	0	2	0	0.00	100.00	0.00	No
Test 18	2	0	2	0	0.00	100.00	0.00	No
Test 19	1	0	1	0	0.00	100.00	0.00	No

Further Observation: The third modification to the face detection application which uses eye detection resulted in a longer execution time.

5.3 Face Recognition Application

The face recognition algorithms were tested under the same conditions as the face detection algorithm. Results were collected for two cases. In one case, the face recognition algorithms were tested to determine how well they can reject non-faces and faces that are not in the training set. In order to achieve this, the parameters for the face detection function were set to their minimum values which resulted in a high hit rate, low miss rate and high false positive rate. Eye detection was not used for testing the face recognition algorithms but was used in the final class attendance application. The performances of the face recognition algorithms were not evaluated under the specific tests. For example, the face recognition algorithm was not tested to determine if it correctly recognized an individual sitting 1.65 m away from the camera. In the other case, the performances of the face recognition algorithms were evaluated under the specific tests. Only the face of interest is observed.

In addition to the face detection tests, the effect of the number of faces in the training set on the face recognition methods was also investigated. The face recognition methods were tested using 160 faces (full training set), 96 faces (no side angle faces) and 16 faces (frontal view, no smile faces) in the training set. As discussed before, the EigenFaceRecognizer and the FisherFaceRecognizer classes require 2 parameters: the number of components and the threshold. Both of these parameters were discussed previously. The maximum value for the number of components parameter is the total number of faces in the training set. If this value is exceeded, the face recognizer class uses the total number of faces in the training set as the number of components. If this value is less than the total number of faces in the training set, the face recognizer class uses the value that was selected by the user as the number of

components. Therefore, this parameter was not altered, since it was set to 160 for both face recognizer classes.

The face recognition algorithms were evaluated using the performance metrics for classifiers by Fawcett (2006). These performance metrics were applied to face recognition. The performance metrics were true positive (TP), true negative (TN), false positive (FP), false negative (FN), precision, accuracy and recall. True positive is the number of predicted faces that were correctly matched with the faces in the training set. True negative is the number of predicted faces that were correctly classified as unknown faces (faces not in training set) or rejected. False positive is the number of predicted faces that were incorrectly matched with the faces in the training set. False negative is the number of predicted faces that were incorrectly classified as unknown faces. Precision indicates how many predicted faces that were matched with the faces in the training set are correct. Accuracy indicates how many predicted faces are correctly matched with the faces in the training set and correctly rejected. Recall indicates how many predicted faces that exists in the training set are correctly matched. Precision, accuracy and recall are expressed as percentages and are given as follows:

$$Precision = \frac{TP}{TP + FP} \times 100\% \quad (5.1)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \quad (5.2)$$

$$Recall = \frac{TP}{TP + FN} \times 100\% \quad (5.3)$$

5.3.1 Eigenfaces Method

Full training set (160 faces)

Table 5.12: Face recognition results for the Eigenfaces method using 13 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	0	5	2	0	0.00	71.43	0.00	No
Test 2	0	2	6	0	0.00	25.00	0.00	No
Test 3	1	5	2	0	33.33	75.00	100.00	Yes
Test 4	1	4	3	0	25.00	62.50	100.00	Yes
Test 5	0	8	4	0	0.00	66.67	0.00	No
Test 6	0	7	3	0	0.00	70.00	0.00	No
Test 7	0	7	6	0	0.00	53.85	0.00	No
Test 8	0	5	4	0	0.00	55.56	0.00	No
Test 9	0	3	7	0	0.00	30.00	0.00	No
Test 10	0	1	4	0	0.00	20.00	0.00	No
Test 11	0	6	2	0	0.00	75.00	0.00	No
Test 12	1	1	1	0	50.00	66.67	100.00	Yes
Test 13	1	3	6	0	14.29	40.00	100.00	Yes
Test 14	0	3	8	0	0.00	27.27	0.00	No
Test 15	1	1	6	0	14.29	25.00	100.00	Yes
Test 16	1	2	3	0	25.00	50.00	100.00	Yes
Test 17	1	3	7	0	12.50	36.36	100.00	Yes
Test 18	0	4	5	0	0.00	44.44	0.00	No
Test 19	1	2	4	0	20.00	42.86	100.00	Yes

Table 5.13: Face recognition results for the Eigenfaces method using 8 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	3	1	0	50.00	80.00	100.00	Yes
Test 13	0	0	4	1	0.00	0.00	0.00	No
Test 14	1	2	2	0	33.33	60.00	100.00	Yes
Test 15	0	2	2	0	0.00	50.00	0.00	No
Test 16	1	2	1	0	50.00	75.00	100.00	Yes
Test 17	1	2	0	0	100.00	100.00	100.00	Yes
Test 18	1	2	1	0	50.00	75.00	100.00	Yes
Test 19	0	1	1	0	0.00	50.00	0.00	No

Table 5.14: Face recognition results for the Eigenfaces method using 5 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	1	3	0	25.00	40.00	100.00	Yes
Test 13	1	1	3	0	25.00	40.00	100.00	Yes
Test 14	1	1	4	0	20.00	33.33	100.00	Yes
Test 15	1	2	1	0	50.00	75.00	100.00	Yes
Test 16	1	0	2	0	33.33	33.33	100.00	Yes
Test 17	1	1	1	0	50.00	66.67	100.00	Yes
Test 18	1	2	1	0	50.00	75.00	100.00	Yes
Test 19	1	2	0	0	100.00	100.00	100.00	Yes

Training set without side angle faces (96 faces)

Table 5.15: Face recognition results for the Eigenfaces method using 13 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	0	3	4	0	0.00	42.86	0.00	No
Test 2	0	2	6	0	0.00	25.00	0.00	No
Test 3	0	5	3	0	0.00	62.50	0.00	No
Test 4	0	3	5	0	0.00	37.50	0.00	No
Test 5	0	6	6	0	0.00	50.00	0.00	No
Test 6	0	3	7	0	0.00	30.00	0.00	No
Test 7	0	6	7	0	0.00	46.15	0.00	No
Test 8	0	4	5	0	0.00	44.44	0.00	No
Test 9	0	3	7	0	0.00	30.00	0.00	No
Test 10	0	0	5	0	0.00	0.00	0.00	No
Test 11	0	5	3	0	0.00	62.50	0.00	No
Test 12	1	1	1	0	50.00	66.67	100.00	Yes
Test 13	1	3	6	0	14.29	40.00	100.00	Yes
Test 14	0	2	9	0	0.00	18.18	0.00	No
Test 15	1	0	7	0	12.50	12.50	100.00	Yes
Test 16	1	2	3	0	25.00	50.00	100.00	Yes
Test 17	1	3	7	0	12.50	36.36	100.00	Yes
Test 18	0	4	5	0	0.00	44.44	0.00	No
Test 19	1	0	6	0	14.29	14.29	100.00	Yes

Table 5.16: Face recognition results for the Eigenfaces method using 8 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	3	1	0	50.00	80.00	100.00	Yes
Test 13	0	0	4	1	0.00	0.00	0.00	No
Test 14	1	0	4	0	20.00	20.00	100.00	Yes
Test 15	0	0	4	0	0.00	0.00	0.00	No
Test 16	1	3	0	0	100.00	100.00	100.00	Yes
Test 17	1	1	1	0	50.00	66.67	100.00	Yes
Test 18	1	2	1	0	50.00	75.00	100.00	Yes
Test 19	1	1	0	0	100.00	100.00	100.00	Yes

Table 5.17: Face recognition results for the Eigenfaces method using 5 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	3	1	0	50.00	80.00	100.00	Yes
Test 13	1	1	3	0	25.00	40.00	100.00	Yes
Test 14	1	0	5	0	16.67	16.67	100.00	Yes
Test 15	1	3	0	0	100.00	100.00	100.00	Yes
Test 16	1	1	1	0	50.00	66.67	100.00	Yes
Test 17	1	1	1	0	50.00	66.67	100.00	Yes
Test 18	1	2	1	0	50.00	75.00	100.00	Yes
Test 19	1	1	1	0	50.00	66.67	100.00	Yes

Training set without head tilt, side angle and smiling faces (16 faces)

Table 5.18: Face recognition results for the Eigenfaces method using 13 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	1	1	5	0	16.67	28.57	100.00	Yes
Test 2	1	0	7	0	12.50	12.50	100.00	Yes
Test 3	1	1	6	0	14.29	25.00	100.00	Yes
Test 4	1	0	7	0	12.50	12.50	100.00	Yes
Test 5	0	0	12	0	0.00	0.00	0.00	No
Test 6	1	2	7	0	12.50	30.00	100.00	Yes
Test 7	0	2	11	0	0.00	15.38	0.00	No
Test 8	1	0	8	0	11.11	11.11	100.00	Yes
Test 9	0	1	9	0	0.00	10.00	0.00	No
Test 10	1	0	4	0	20.00	20.00	100.00	Yes
Test 11	0	1	7	0	0.00	12.50	0.00	No
Test 12	1	0	2	0	33.33	33.33	100.00	Yes
Test 13	1	0	9	0	10.00	10.00	100.00	Yes
Test 14	0	0	11	0	0.00	0.00	0.00	No
Test 15	1	0	7	0	12.50	12.50	100.00	Yes
Test 16	1	0	5	0	16.67	16.67	100.00	Yes
Test 17	1	1	9	0	10.00	18.18	100.00	Yes
Test 18	0	0	9	0	0.00	0.00	0.00	No
Test 19	1	0	6	0	14.29	14.29	100.00	Yes

Table 5.19: Face recognition results for the Eigenfaces method using 8 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	1	3	0	25.00	40.00	100.00	Yes
Test 13	0	0	5	0	0.00	0.00	0.00	No
Test 14	1	0	4	0	20.00	20.00	100.00	Yes
Test 15	0	0	4	0	0.00	0.00	0.00	No
Test 16	1	1	2	0	33.33	50.00	100.00	Yes
Test 17	1	1	1	0	50.00	66.67	100.00	Yes
Test 18	1	0	3	0	25.00	25.00	100.00	Yes
Test 19	1	0	1	0	50.00	50.00	100.00	Yes

Table 5.20: Face recognition results for the Eigenfaces method using 5 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	1	3	0	25.00	40.00	100.00	Yes
Test 13	1	1	3	0	25.00	40.00	100.00	Yes
Test 14	1	0	5	0	16.67	16.67	100.00	Yes
Test 15	1	1	2	0	33.33	50.00	100.00	Yes
Test 16	1	0	2	0	33.33	33.33	100.00	Yes
Test 17	1	0	2	0	33.33	33.33	100.00	Yes
Test 18	1	0	3	0	25.00	25.00	100.00	Yes
Test 19	1	0	2	0	33.33	33.33	100.00	Yes

5.3.2 Fisherfaces Method

Full training set (160 faces)

Table 5.21: Face recognition results for the Fisherfaces method using 13 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	0	4	3	0	0.00	57.14	0.00	No
Test 2	1	2	5	0	16.67	37.50	100.00	Yes
Test 3	1	3	4	0	20.00	50.00	100.00	Yes
Test 4	1	5	2	0	33.33	75.00	100.00	Yes
Test 5	0	7	5	0	0.00	58.33	0.00	No
Test 6	0	7	3	0	0.00	70.00	0.00	No
Test 7	0	10	3	0	0.00	76.92	0.00	No
Test 8	0	3	6	0	0.00	33.33	0.00	No
Test 9	0	4	6	0	0.00	40.00	0.00	No
Test 10	0	1	4	0	0.00	20.00	0.00	No
Test 11	0	5	3	0	0.00	62.50	0.00	No
Test 12	0	1	1	1	0.00	33.33	0.00	No
Test 13	0	2	7	1	0.00	20.00	0.00	No
Test 14	0	3	8	0	0.00	27.27	0.00	No
Test 15	1	2	5	0	16.67	37.50	100.00	Yes
Test 16	1	2	3	0	25.00	50.00	100.00	Yes
Test 17	1	3	7	0	12.50	36.36	100.00	Yes
Test 18	0	2	7	0	0.00	22.22	0.00	No
Test 19	1	2	4	0	20.00	42.86	100.00	Yes

Table 5.22: Face recognition results for the Fisherfaces method using 8 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	0	2	2	1	0.00	40.00	0.00	No
Test 13	0	0	4	1	0.00	0.00	0.00	No
Test 14	1	2	2	0	33.33	60.00	100.00	Yes
Test 15	0	1	3	0	0.00	25.00	0.00	No
Test 16	1	2	1	0	50.00	75.00	100.00	Yes
Test 17	1	1	1	0	50.00	66.67	100.00	Yes
Test 18	1	2	1	0	50.00	75.00	100.00	Yes
Test 19	1	1	0	0	100.00	100.00	100.00	Yes

Table 5.23: Face recognition results for the Fisherfaces method using 5 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	2	2	0	33.33	60.00	100.00	Yes
Test 13	1	1	3	0	25.00	40.00	100.00	Yes
Test 14	1	1	4	0	20.00	33.33	100.00	Yes
Test 15	1	2	1	0	50.00	75.00	100.00	Yes
Test 16	1	0	2	0	33.33	33.33	100.00	Yes
Test 17	1	1	1	0	50.00	66.67	100.00	Yes
Test 18	1	2	1	0	50.00	75.00	100.00	Yes
Test 19	0	1	2	0	0.00	33.33	0.00	No

Training set without side angle faces (96 faces)

Table 5.24: Face recognition results for the Fisherfaces method using 13 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	0	1	6	0	0.00	14.29	0.00	No
Test 2	0	1	7	0	0.00	12.50	0.00	No
Test 3	0	4	3	1	0.00	50.00	0.00	No
Test 4	0	3	4	1	0.00	37.50	0.00	No
Test 5	0	4	8	0	0.00	33.33	0.00	No
Test 6	1	2	7	0	12.50	30.00	100.00	Yes
Test 7	0	5	8	0	0.00	38.46	0.00	No
Test 8	0	2	7	0	0.00	22.22	0.00	No
Test 9	0	3	7	0	0.00	30.00	0.00	No
Test 10	0	0	5	0	0.00	0.00	0.00	No
Test 11	0	3	5	0	0.00	37.50	0.00	No
Test 12	1	0	2	0	33.33	33.33	100.00	Yes
Test 13	1	2	7	0	12.50	30.00	100.00	Yes
Test 14	0	2	9	0	0.00	18.18	0.00	No
Test 15	1	1	6	0	14.29	25.00	100.00	Yes
Test 16	1	1	4	0	20.00	33.33	100.00	Yes
Test 17	1	1	9	0	10.00	18.18	100.00	Yes
Test 18	0	4	5	0	0.00	44.44	0.00	No
Test 19	1	1	5	0	16.67	28.57	100.00	Yes

Table 5.25: Face recognition results for the Fisherfaces method using 8 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	1	3	0	25.00	40.00	100.00	Yes
Test 13	0	0	5	0	0.00	0.00	0.00	No
Test 14	1	0	4	0	20.00	20.00	100.00	Yes
Test 15	1	1	2	0	33.33	50.00	100.00	Yes
Test 16	1	2	1	0	50.00	75.00	100.00	Yes
Test 17	1	0	2	0	33.33	33.33	100.00	Yes
Test 18	1	2	1	0	50.00	75.00	100.00	Yes
Test 19	1	1	0	0	100.00	100.00	100.00	Yes

Table 5.26: Face recognition results for the Fisherfaces method using 5 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	3	1	0	50.00	80.00	100.00	Yes
Test 13	1	1	3	0	25.00	40.00	100.00	Yes
Test 14	1	0	5	0	16.67	16.67	100.00	Yes
Test 15	1	2	1	0	50.00	75.00	100.00	Yes
Test 16	1	1	1	0	50.00	66.67	100.00	Yes
Test 17	1	1	1	0	50.00	66.67	100.00	Yes
Test 18	1	1	2	0	33.33	50.00	100.00	Yes
Test 19	1	1	1	0	50.00	66.67	100.00	Yes

Training set without head tilt, side angle and smiling faces (16 faces)

Table 5.27: Face recognition results for the Fisherfaces method using 13 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	1	0	6	0	14.29	14.29	100.00	Yes
Test 2	1	0	7	0	12.50	12.50	100.00	Yes
Test 3	0	0	8	0	0.00	0.00	0.00	No
Test 4	0	0	8	0	0.00	0.00	0.00	No
Test 5	0	0	12	0	0.00	0.00	0.00	No
Test 6	1	0	9	0	10.00	10.00	100.00	Yes
Test 7	1	0	12	0	7.69	7.69	100.00	Yes
Test 8	0	0	9	0	0.00	0.00	0.00	No
Test 9	0	0	10	0	0.00	0.00	0.00	No
Test 10	0	0	5	0	0.00	0.00	0.00	No
Test 11	0	0	8	0	0.00	0.00	0.00	No
Test 12	1	0	2	0	33.33	33.33	100.00	Yes
Test 13	1	0	9	0	10.00	10.00	100.00	Yes
Test 14	0	0	11	0	0.00	0.00	0.00	No
Test 15	1	0	7	0	12.50	12.50	100.00	Yes
Test 16	1	0	5	0	16.67	16.67	100.00	Yes
Test 17	1	0	10	0	9.09	9.09	100.00	Yes
Test 18	0	0	9	0	0.00	0.00	0.00	No
Test 19	1	0	6	0	14.29	14.29	100.00	Yes

Table 5.28: Face recognition results for the Fisherfaces method using 8 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	0	4	0	20.00	20.00	100.00	Yes
Test 13	0	0	5	0	0.00	0.00	0.00	No
Test 14	1	0	4	0	20.00	20.00	100.00	Yes
Test 15	1	0	3	0	25.00	25.00	100.00	Yes
Test 16	1	0	3	0	25.00	25.00	100.00	Yes
Test 17	1	0	2	0	33.33	33.33	100.00	Yes
Test 18	1	0	3	0	25.00	25.00	100.00	Yes
Test 19	1	0	1	0	50.00	50.00	100.00	Yes

Table 5.29: Face recognition results for the Fisherfaces method using 5 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	0	4	0	20.00	20.00	100.00	Yes
Test 13	1	0	4	0	20.00	20.00	100.00	Yes
Test 14	1	0	5	0	16.67	16.67	100.00	Yes
Test 15	1	0	3	0	25.00	25.00	100.00	Yes
Test 16	1	0	2	0	33.33	33.33	100.00	Yes
Test 17	1	0	2	0	33.33	33.33	100.00	Yes
Test 18	1	0	3	0	25.00	25.00	100.00	Yes
Test 19	1	0	2	0	33.33	33.33	100.00	Yes

5.3.3 LBPH Method

Full training set (160 faces)

Table 5.30: Face recognition results for the LBPH method using 13 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	1	2	4	0	20.00	42.86	100.00	Yes
Test 2	1	6	1	0	50.00	87.50	100.00	Yes
Test 3	1	5	2	0	33.33	75.00	100.00	Yes
Test 4	0	2	6	0	0.00	25.00	0.00	No
Test 5	0	8	4	0	0.00	66.67	0.00	No
Test 6	0	5	5	0	0.00	50.00	0.00	No
Test 7	0	3	10	0	0.00	23.08	0.00	No
Test 8	0	3	6	0	0.00	33.33	0.00	No
Test 9	0	3	7	0	0.00	30.00	0.00	No
Test 10	0	1	4	0	0.00	20.00	0.00	No
Test 11	0	3	5	0	0.00	37.50	0.00	No
Test 12	1	1	1	0	50.00	66.67	100.00	Yes
Test 13	0	4	6	0	0.00	40.00	0.00	No
Test 14	0	1	10	0	0.00	9.09	0.00	No
Test 15	0	1	7	0	0.00	12.50	0.00	No
Test 16	1	0	5	0	16.67	16.67	100.00	Yes
Test 17	1	0	10	0	9.09	9.09	100.00	Yes
Test 18	0	0	9	0	0.00	0.00	0.00	No
Test 19	0	1	6	0	0.00	14.29	0.00	No

Table 5.31: Face recognition results for the LBPH method using 8 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	2	2	0	33.33	60.00	100.00	Yes
Test 13	0	2	3	0	0.00	40.00	0.00	No
Test 14	0	0	5	0	0.00	0.00	0.00	No
Test 15	1	1	2	0	33.33	50.00	100.00	Yes
Test 16	0	3	1	0	0.00	75.00	0.00	No
Test 17	0	1	2	0	0.00	33.33	0.00	No
Test 18	1	1	2	0	33.33	50.00	100.00	Yes
Test 19	1	0	1	0	50.00	50.00	100.00	Yes

Table 5.32: Face recognition results for the LBPH method using 5 MP camera and full training set.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	1	3	0	25.00	40.00	100.00	Yes
Test 13	1	1	3	0	25.00	40.00	100.00	Yes
Test 14	0	2	4	0	0.00	33.33	0.00	No
Test 15	0	1	3	0	0.00	25.00	0.00	No
Test 16	1	1	1	0	50.00	66.67	100.00	Yes
Test 17	0	2	1	0	0.00	66.67	0.00	No
Test 18	0	1	3	0	0.00	25.00	0.00	No
Test 19	1	1	1	0	50.00	66.67	100.00	Yes

Training set without side angle faces (96 faces)

Table 5.33: Face recognition results for the LBPH method using 13 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	1	2	4	0	20.00	42.86	100.00	Yes
Test 2	1	6	1	0	50.00	87.50	100.00	Yes
Test 3	1	5	2	0	33.33	75.00	100.00	Yes
Test 4	0	3	5	0	0.00	37.50	0.00	No
Test 5	0	8	4	0	0.00	66.67	0.00	No
Test 6	0	6	4	0	0.00	60.00	0.00	No
Test 7	0	3	10	0	0.00	23.08	0.00	No
Test 8	0	3	6	0	0.00	33.33	0.00	No
Test 9	0	4	6	0	0.00	40.00	0.00	No
Test 10	0	1	4	0	0.00	20.00	0.00	No
Test 11	0	3	5	0	0.00	37.50	0.00	No
Test 12	1	1	1	0	50.00	66.67	100.00	Yes
Test 13	0	5	5	0	0.00	50.00	0.00	No
Test 14	0	1	10	0	0.00	9.09	0.00	No
Test 15	0	1	7	0	0.00	12.50	0.00	No
Test 16	1	0	5	0	16.67	16.67	100.00	Yes
Test 17	1	0	10	0	9.09	9.09	100.00	Yes
Test 18	0	0	9	0	0.00	0.00	0.00	No
Test 19	0	1	6	0	0.00	14.29	0.00	No

Table 5.34: Face recognition results for the LBPH method using 8 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	2	2	0	33.33	60.00	100.00	Yes
Test 13	0	2	3	0	0.00	40.00	0.00	No
Test 14	0	0	5	0	0.00	0.00	0.00	No
Test 15	1	1	2	0	33.33	50.00	100.00	Yes
Test 16	0	3	1	0	0.00	75.00	0.00	No
Test 17	0	1	2	0	0.00	33.33	0.00	No
Test 18	1	1	2	0	33.33	50.00	100.00	Yes
Test 19	1	0	1	0	50.00	50.00	100.00	Yes

Table 5.35: Face recognition results for the LBPH method using 5 MP camera and training set without side views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	1	3	0	25.00	40.00	100.00	Yes
Test 13	1	1	3	0	25.00	40.00	100.00	Yes
Test 14	0	2	4	0	0.00	33.33	0.00	No
Test 15	0	2	2	0	0.00	50.00	0.00	No
Test 16	1	1	1	0	50.00	66.67	100.00	Yes
Test 17	0	2	1	0	0.00	66.67	0.00	No
Test 18	0	1	3	0	0.00	25.00	0.00	No
Test 19	1	1	1	0	50.00	66.67	100.00	Yes

Training set without head tilt, side angle and smiling faces (16 faces)

Table 5.36: Face recognition results for the LBPH method using 13 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 1	1	2	4	0	20.00	42.86	100.00	Yes
Test 2	1	6	1	0	50.00	87.50	100.00	Yes
Test 3	1	5	2	0	33.33	75.00	100.00	Yes
Test 4	0	3	5	0	0.00	37.50	0.00	No
Test 5	0	8	4	0	0.00	66.67	0.00	No
Test 6	1	6	3	0	25.00	70.00	100.00	Yes
Test 7	1	4	8	0	11.11	38.46	100.00	Yes
Test 8	0	3	6	0	0.00	33.33	0.00	No
Test 9	0	5	5	0	0.00	50.00	0.00	No
Test 10	0	3	2	0	0.00	60.00	0.00	No
Test 11	0	3	5	0	0.00	37.50	0.00	No
Test 12	1	1	1	0	50.00	66.67	100.00	Yes
Test 13	0	5	5	0	0.00	50.00	0.00	No
Test 14	0	1	10	0	0.00	9.09	0.00	No
Test 15	0	1	7	0	0.00	12.50	0.00	No
Test 16	0	1	5	0	0.00	16.67	0.00	No
Test 17	0	1	10	0	0.00	9.09	0.00	No
Test 18	0	0	9	0	0.00	0.00	0.00	No
Test 19	0	2	5	0	0.00	28.57	0.00	No

Table 5.37: Face recognition results for the LBPH method using 8 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	2	2	0	33.33	60.00	100.00	Yes
Test 13	0	2	3	0	0.00	40.00	0.00	No
Test 14	0	0	5	0	0.00	0.00	0.00	No
Test 15	1	1	2	0	33.33	50.00	100.00	Yes
Test 16	0	2	1	0	0.00	66.67	0.00	No
Test 17	0	1	2	0	0.00	33.33	0.00	No
Test 18	0	1	3	0	0.00	25.00	0.00	No
Test 19	0	0	2	0	0.00	0.00	0.00	No

Table 5.38: Face recognition results for the LBPH method using 5 MP camera and training set with frontal views.

Tests	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)	Observed face correctly classified?
Test 12	1	1	3	0	25.00	40.00	100.00	Yes
Test 13	1	1	3	0	25.00	40.00	100.00	Yes
Test 14	0	2	4	0	0.00	33.33	0.00	No
Test 15	0	2	2	0	0.00	50.00	0.00	No
Test 16	0	1	2	0	0.00	33.33	0.00	No
Test 17	0	2	1	0	0.00	66.67	0.00	No
Test 18	0	1	3	0	0.00	25.00	0.00	No
Test 19	0	1	2	0	0.00	33.33	0.00	No

5.3.4 Face Recognition Comparisons

Note: The averages were found using Test 12 to Test 19 (distance tests).

Eigenfaces Method

Table 5.39: Eigenfaces method performance results for various camera resolutions and full training set.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	17.01	41.58	75.00
8 MP	35.42	61.25	62.50
5 MP	44.17	57.92	100.00

Table 5.40: Eigenfaces method performance results for various camera resolutions and training set without side views.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	16.07	35.31	75.00
8 MP	46.25	55.21	75.00
5 MP	48.96	63.96	100.00

Table 5.41: Eigenfaces method performance results for various camera resolutions and training set with frontal views.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	12.10	13.12	75.00
8 MP	25.42	31.46	75.00
5 MP	28.12	33.96	100.00

Fisherfaces Method

Table 5.42: Fisherfaces method performance results for various camera resolutions and full training set.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	9.27	33.69	50.00
8 MP	35.42	55.21	62.50
5 MP	32.71	52.08	87.50

Table 5.43: Fisherfaces method performance results for various camera resolutions and training set without side views.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	13.35	28.88	75.00
8 MP	38.96	49.17	87.50
5 MP	40.63	57.71	100.00

Table 5.44: Fisherfaces method performance results for various camera resolutions and training set with frontal views.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	11.99	11.99	75.00
8 MP	24.79	24.79	87.50
5 MP	25.83	25.83	100.00

LBPH Method

Table 5.45: LBPH method performance results for various camera resolutions and full training set.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	9.47	21.04	37.50
8 MP	18.75	44.79	50.00
5 MP	18.75	45.42	50.00

Table 5.46: LBPH method performance results for various camera resolutions and training set without side views.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	9.47	22.29	37.50
8 MP	18.75	44.79	50.00
5 MP	18.75	48.54	50.00

Table 5.47: LBPH method performance results for various camera resolutions and training set with frontal views.

Camera resolutions	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
13 MP	6.25	24.07	12.50
8 MP	8.33	34.38	25.00
5 MP	6.25	40.21	25.00

All face recognition methods

Note: The averages were found using Test 1 to Test 11 (all tests except distance tests).

Table 5.48: Face recognition performance results for all face recognition methods and full training set.

Face recognition method	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
Eigenfaces	5.30	55.00	18.18
Fisherfaces	6.36	52.79	27.27
LBPH	9.39	44.63	27.27

Table 5.49: Face recognition performance results for all face recognition methods and training set without side views.

Face recognition method	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
Eigenfaces	0.00	39.18	0.00
Fisherfaces	1.14	27.80	9.09
LBPH	9.39	47.59	27.27

Table 5.50: Face recognition performance results for all face recognition methods and training set with frontal views.

Face recognition method	Average Precision (%)	Average Accuracy (%)	Average Recall (%)
Eigenfaces	9.05	16.14	63.64
Fisherfaces	4.04	4.04	36.36
LBPH	12.68	54.44	45.45

Further Observation: The execution time for the LBPH method is shorter than the execution times for the Eigenfaces and Fisherfaces methods using the full training set. Additionally, the execution times for the Eigenfaces and Fisherfaces methods using the reduced training sets were shorter than the execution times for the same methods using the full training set.

5.3.5 Class images

The previous face recognition tests were performed on only one person. Therefore, the images of multiple persons were captured for further testing. The images for all of the students whose faces are in the face database could not be obtained because of their unavailability. Hence, the images for some of the students were captured. The ideal conditions for the face detection and face recognition methods were used. The distance between the students and the camera was less than 2.54 m. The students looked straight ahead, without turning or tilting their heads. Additionally, the face database only contained faces that are in the frontal position with no smile. The class images were captured using the 13 MP camera and were also used to test the class attendance application. The detected faces in the captured class images are illustrated in Figures 5.2 to 5.5.



Figure 5.2: Image of students for ECNG 3034 on 2016-04-04

Note: One of students did not adhere to the ideal conditions, hence the face of that student was not detected and recognized.



Figure 5.3: Image of student for ECNG 6706 on 2016-04-06

Note: The undetected faces were the faces of students who are not in the face database and not doing the course.

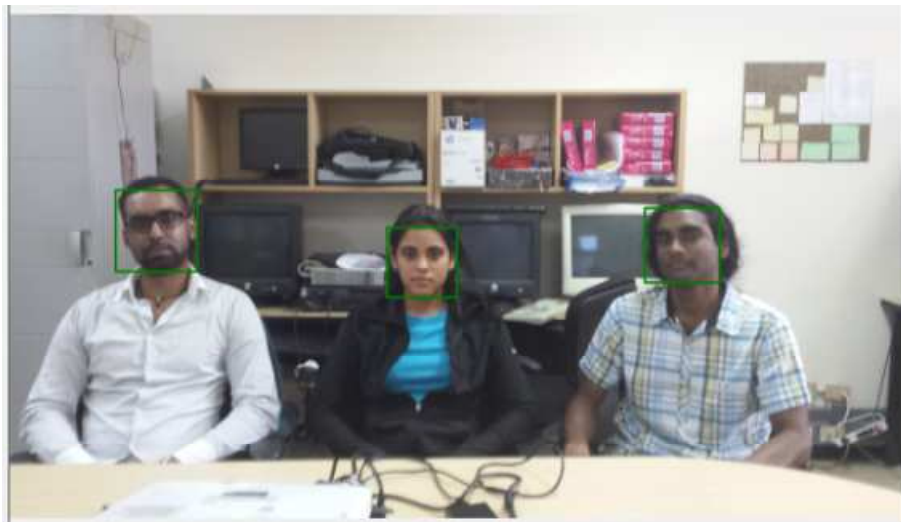


Figure 5.4: Image 1 of students for ECNG 7000 on 2016-04-08



Figure 5.5: Image 2 of student for ECNG 7000 on 2016-04-08

Table 5.51: Face recognition results for all methods using class images for all courses

Face recognition method	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)
Eigenfaces	4	0	3	0	57.14%	57.14%	100.00%
Fisherfaces	3	0	4	0	42.86%	42.86%	100.00%
LBPH	3	0	4	0	42.86%	42.86%	100.00%

Further Tests

More images of students were captured for one of the courses. The images for the other students could not be captured due to their unavailability. The images were captured under ideal conditions. However, in this case the images were captured with the students located progressively closer to the camera. The additional images were also used to test the class attendance application. The detected faces in the captured class images are illustrated in Figures 5.6 to 5.12.

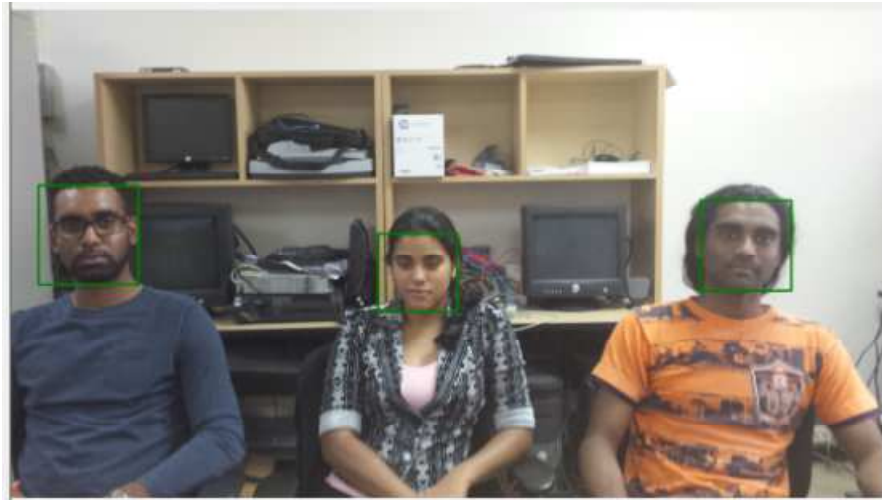


Figure 5.6: Image of students for ECNG 7000 on 2016-04-15

Table 5.52: Face recognition results for all methods using class image for ECNG 7000 on 2016-04-15

Face recognition method	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)
Eigenfaces	1	0	2	0	33.33%	33.33%	100.00%
Fisherfaces	1	0	2	0	33.33%	33.33%	100.00%
LBPH	0	0	3	0	0.00%	0.00%	0.00%

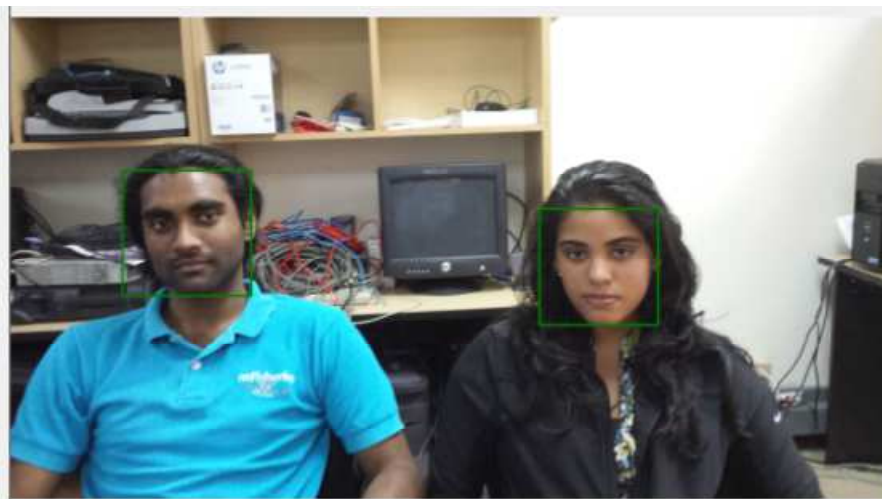


Figure 5.7: Image 1 of students for ECNG 7000 on 2016-04-22



Figure 5.8: Image 2 of students for ECNG 7000 on 2016-04-22



Figure 5.9: Image 3 of student for ECNG 7000 on 2016-04-22

Table 5.53: Face recognition results for all methods using class images for ECNG 7000 on 2016-04-22

Face recognition method	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)
Eigenfaces	2	0	3	0	40.00%	40.00%	100.00%
Fisherfaces	1	0	4	0	20.00%	20.00%	100.00%
LBPH	3	0	2	0	60.00%	60.00%	100.00%

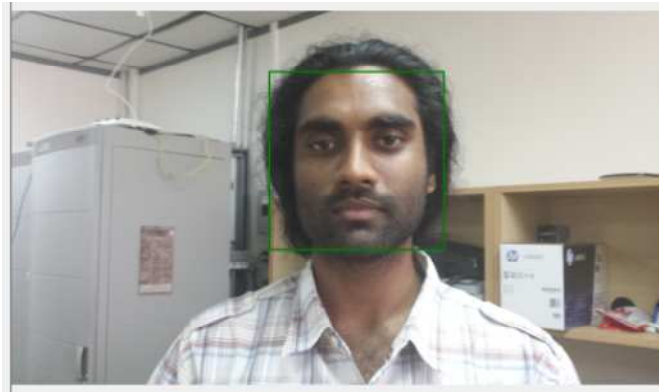


Figure 5.10: Image 1 of student for ECNG 7000 on 2016-04-29

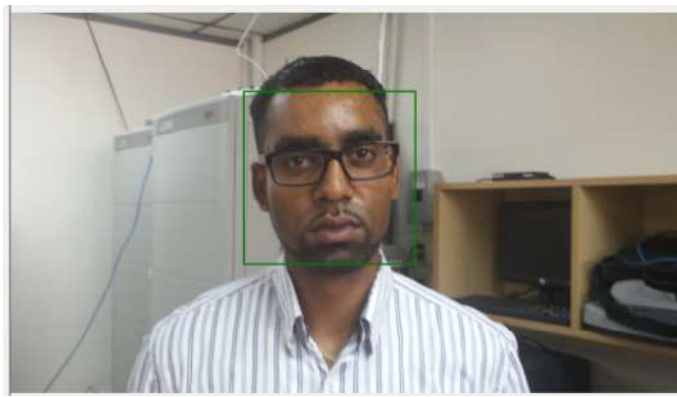


Figure 5.11: Image 2 of student for ECNG 7000 on 2016-04-29

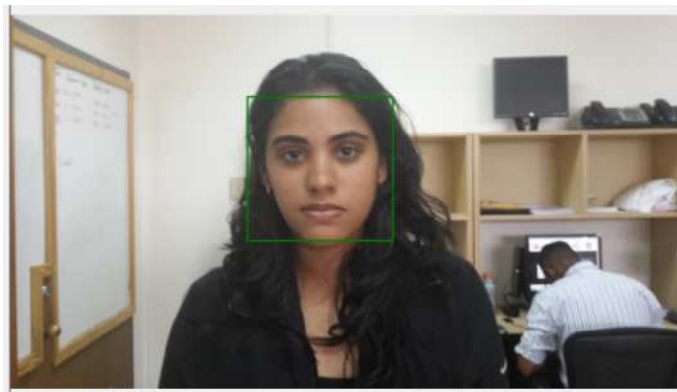


Figure 5.12: Image 3 of student for ECNG 7000 on 2016-04-29

Table 5.54: Face recognition results for all methods using class images for ECNG 7000 on 2016-04-29

Face recognition method	TP	TN	FP	FN	Precision (%)	Accuracy (%)	Recall (%)
Eigenfaces	2	0	1	0	66.67%	66.67%	100.00%
Fisherfaces	1	0	2	0	33.33%	33.33%	100.00%
LBPH	1	0	2	0	33.33%	33.33%	100.00%

5.4 Class Attendance Application

As mentioned before, the images of the students illustrated in the previous section were used to test the class attendance system. The class attendance system is tested for 3 cases: class attendance, student attendance and detailed attendance. The class attendance is the total number of present students divided by the total number of students for a specific course. The class attendance is calculated on a daily basis. The student attendance is the total number of class sessions for which the student is present divided by the total number of class sessions for a particular student. The student attendance is calculated over a period. The detailed attendance indicates the present students in all courses and the corresponding dates.

Similar to evaluating the face recognition methods, the class attendance system was evaluated using true positive (TP), true negative (TN), false positive (FP), false negative (FN), precision, accuracy and recall. In this case, true positive is the number of students who were reported as being present and were actually present. True negative is the number of students who were reported as being absent and were actually absent. False positive is the number of students who were reported as present but were actually absent. False negative is the number of students who were reported as absent but were actually present. Precision indicates how many students that were reported as being present were actually present. Accuracy indicates how many students were correctly reported as present and absent. Recall indicates how many students that were actually present were reported as being present. The equations for precision, accuracy and recall are the same as before.

Table 5.55: Class attendance for all courses and dates

Course	Actual Class Attendance	Reported Class Attendance	Date
ECNG 3034	100.00%	33.33%	2016-04-04
ECNG 6706	12.50%	12.50%	2016-04-06
ECNG 7000	80.00%	40.00%	2016-04-08
ECNG 7000	60.00%	20.00%	2016-04-15
ECNG 7000	80.00%	20.00%	2016-04-22
ECNG 7000	60.00%	40.00%	2016-04-29

Table 5.56: Student attendance for all courses and dates

Course	Student Id	Actual Attendance	Reported Attendance	Period
ECNG 3034	000000010	100.00%	0.00%	2016-04-04 to 2016-04-04
ECNG 3034	000000011	100.00%	100.00%	2016-04-04 to 2016-04-04
ECNG 3034	000000012	100.00%	0.00%	2016-04-04 to 2016-04-04
ECNG 6706	000000002	0.00%	0.00%	2016-04-06 to 2016-04-06
ECNG 6706	000000005	0.00%	0.00%	2016-04-06 to 2016-04-06
ECNG 6706	000000006	0.00%	0.00%	2016-04-06 to 2016-04-06
ECNG 6706	000000007	100.00%	100.00%	2016-04-06 to 2016-04-06
ECNG 6706	000000008	0.00%	0.00%	2016-04-06 to 2016-04-06
ECNG 6706	000000009	0.00%	0.00%	2016-04-06 to 2016-04-06
ECNG 6706	000000013	0.00%	0.00%	2016-04-06 to 2016-04-06
ECNG 6706	000000014	0.00%	0.00%	2016-04-06 to 2016-04-06
ECNG 7000	000000001	100.00%	0.00%	2016-04-08 to 2016-04-29
ECNG 7000	000000003	50.00%	0.00%	2016-04-08 to 2016-04-29
ECNG 7000	000000004	0.00%	0.00%	2016-04-08 to 2016-04-29
ECNG 7000	000000015	100.00%	100.00%	2016-04-08 to 2016-04-29
ECNG 7000	000000016	100.00%	50.00%	2016-04-08 to 2016-04-29

Note: The student ids are fictional.

Table 5.57: Detailed attendances for all courses and dates

Course	Student	Present?	Student correctly recognized?	Student reported as present?	Date
ECNG 3034	Gerard Clayton	Yes	No	No	2016-04-04
ECNG 3034	Alison Brock	Yes	Yes	Yes	2016-04-04
ECNG 3034	Benjamin Carson	Yes	N/A	No	2016-04-04
ECNG 6706	Samuel Green	No	N/A	No	2016-04-06
ECNG 6706	Tamara Wilkerson	No	N/A	No	2016-04-06
ECNG 6706	Kerry Davis	No	N/A	No	2016-04-06
ECNG 6706	Pete Sanders	Yes	Yes	Yes	2016-04-06
ECNG 6706	Brooke Clarke	No	N/A	No	2016-04-06
ECNG 6706	Iris Henderson	No	N/A	No	2016-04-06
ECNG 6706	Walter Hayes	No	N/A	No	2016-04-06
ECNG 6706	Vincent Stewart	No	N/A	No	2016-04-06
ECNG 7000	Jane Harper	Yes	No	No	2016-04-08
ECNG 7000	Gordon Richardson	Yes	No	No	2016-04-08
ECNG 7000	Zane Knight	No	N/A	No	2016-04-08
ECNG 7000	Guillermo Garcia	Yes	Yes	Yes	2016-04-08
ECNG 7000	Randall Walter	Yes	Yes	Yes	2016-04-08
ECNG 7000	Jane Harper	Yes	No	No	2016-04-15
ECNG 7000	Gordon Richardson	No	N/A	No	2016-04-15
ECNG 7000	Zane Knight	No	N/A	No	2016-04-15
ECNG 7000	Guillermo Garcia	Yes	Yes	Yes	2016-04-15
ECNG 7000	Randall Walter	Yes	No	No	2016-04-15
ECNG 7000	Jane Harper	Yes	No	No	2016-04-22
ECNG 7000	Gordon Richardson	Yes	No	No	2016-04-22
ECNG 7000	Zane Knight	No	N/A	No	2016-04-22
ECNG 7000	Guillermo Garcia	Yes	Yes	Yes	2016-04-22
ECNG 7000	Randall Walter	Yes	No	No	2016-04-22
ECNG 7000	Jane Harper	Yes	No	No	2016-04-29
ECNG 7000	Gordon Richardson	No	N/A	No	2016-04-29
ECNG 7000	Zane Knight	No	N/A	No	2016-04-29
ECNG 7000	Guillermo Garcia	Yes	Yes	Yes	2016-04-29
ECNG 7000	Randall Walter	Yes	Yes	Yes	2016-04-29

Note: The student names are fictional.

Using the information in Table 5.57 the following performance metrics were calculated:

- $TP = 8$
- $TN = 13$
- $FP = 0$
- $FN = 10$
- $Precision = 100.00\%$
- $Accuracy = 67.74\%$
- $Recall = 44.44\%$

The following figures illustrate an example of the results produced by the class attendance application.

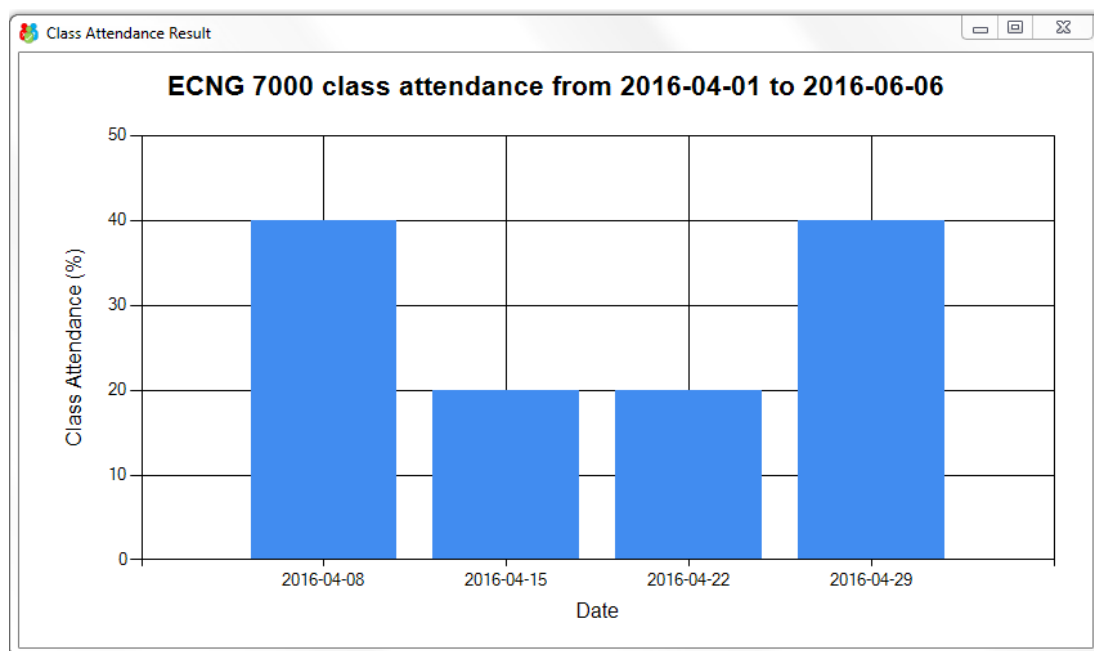


Figure 5.13: Class attendance for ECNG 7000 between 2016-04-01 and 2016-06-06

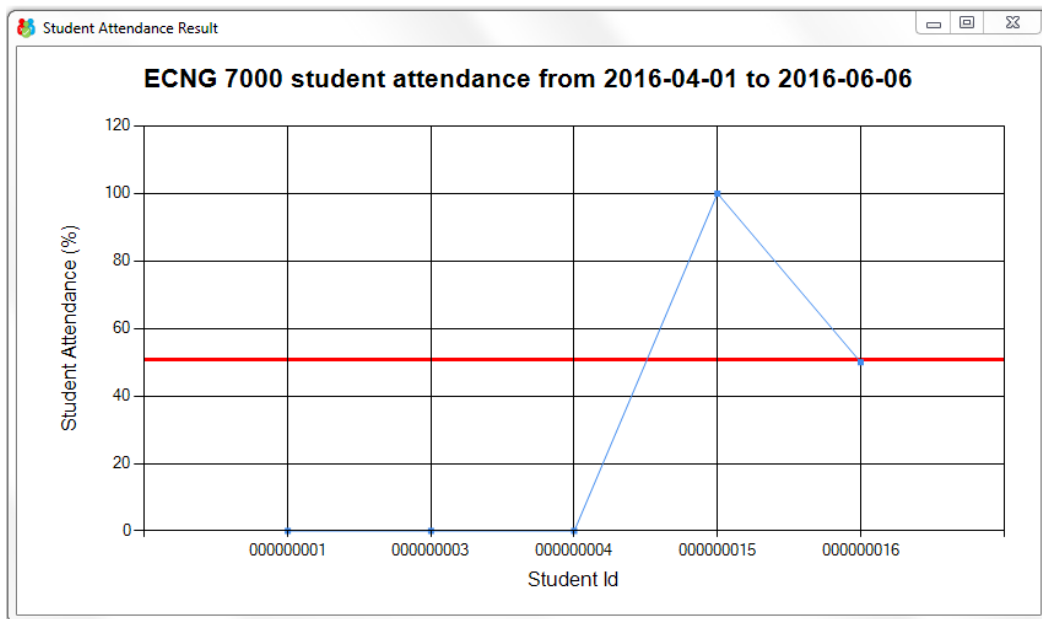


Figure 5.14: Student attendance for ECNG 7000 between 2016-04-01 and 2016-06-06 with a 50% attendance requirement

Attendances Table				
Course: ECNG 7000		From: 2016-04-01		To: 2016-06-06
	Student Id	First Name	Last Name	Date
▶	000000015	Guillermo	Garcia	2016-04-08
	000000016	Randall	Walter	2016-04-08
	000000015	Guillermo	Garcia	2016-04-15
	000000015	Guillermo	Garcia	2016-04-22
	000000015	Guillermo	Garcia	2016-04-29
	000000016	Randall	Walter	2016-04-29

Figure 5.15: Attendances table for ECNG 7000 between 2016-04-01 and 2016-06-06

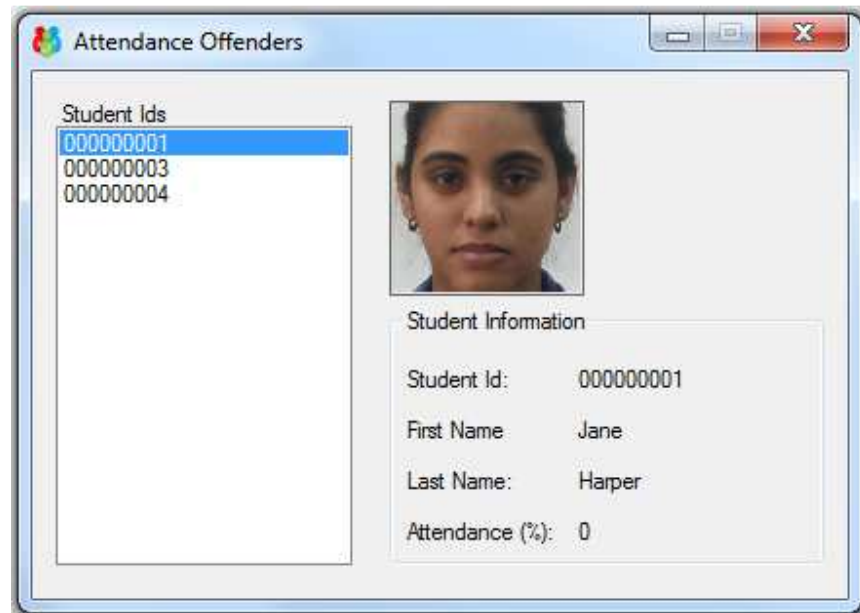


Figure 5.16: Attendance Offenders for ECNG 7000 between 2016-04-01 and 2016-06-06 with a 50% attendance requirement

Chapter 6 Discussion

6.1 Face Detection

Ideally, the hit rates should be 100% while the false positive rates and miss rates should be 0%.

Any errors in the face detection process can result in an absent student being recorded as present or a present student being recorded as absent. For example, if a non-face region was detected as a face and the face recognition process incorrectly matches that face to a student, then that student will be recorded as present even if that student was absent. Additionally, if the student was present, then that student will be recorded as present twice. If a face was not detected, then the student whose face was not detected will be recorded as absent.

Tables 5.3, 5.4 and 5.5 in Chapter 5 Testing and Results show the results for the face detection algorithm using the 13 MP, 8 MP and 5 MP camera respectively. Test 1 to Test 11 evaluated the face detection algorithm for different head tilts and head turns. Test 12 to Test 19 evaluated the face detection algorithm for different distances. Only one individual was subjected to each test. However, there was additional individual for the distance testing. The distance testing was done in a classroom where students can occupy the room even when there is no class session. In this case, there was a student present in the room. However, this student was not subjected to the tests and did not comply with the rules of the tests. Therefore, there would be instances of missed detections as a result of the presence of this student. Table 5.3 shows that a face was not detected in Test 5 (90 degree side angle test). Test 5 had a 100% miss rate or 0% hit rate, since the face of interest was not detected. For all tests, there was only one face of interest and if it was detected, the hit rate would be 100%. Table 5.3 to Table 5.5 show that that the hit rate was 100% for most tests. For the tests that had a 50% hit rate or 50% miss rate, there were 2

actual faces and 1 detected face. This was due to the presence of the non-participating student. Table 5.4 and Table 5.5 show that the number of detected faces were generally lower using the 8 MP and 5 MP cameras than using the 13 MP camera. However, the ratio of false detections to total detections were similar, which resulted in similar false positive rates using the 13 MP and 8 MP cameras. The false positive rates were generally the lowest using the 5 MP camera.

Tables 5.6, 5.7 and 5.8 in Chapter 5 Testing and Results show the results for the face detection algorithm using the 13 MP, 8 MP and 5 MP camera respectively. These tables in particular show the results after modifying the parameters of the DetectMultiScale function (face detection function). These tables show that number of detected faces and falsely detected faces were reduced. As a result the false positive rates were also reduced. In some tests, a 0% false positive rate was achieved. Similar to before, the face in Test 5 in Table 5.6 was not detected. There were also miss rates of 50% for some tests. Test 18 in Table 5.7 had a miss rate of 100%. Table 5.6 to Table 5.8 show that that the hit rate was 100% for most tests. In this case, the number of detected faces were similar using the 13 MP, 8 MP and 5 MP cameras.

Tables 5.9, 5.10 and 5.11 in Chapter 5 Testing and Results show the results for the face detection algorithm using the 13 MP, 8 MP and 5 MP camera respectively. These tables in particular show the results after combining face detection and eye detection. The goal in this case was to use eye detection to eliminate the falsely detected faces. If the detected face contained eyes, then the detected face is confirmed to be an actual face. The parameters for the eye and face detection methods were set such that there would be no missed detections. However, this would result in a high number of falsely detected eyes and faces. Since the detection of eyes is not the main goal, the high number of falsely detected eyes is not an issue. The correctly detected eyes would be found in the correctly detected faces. Therefore, the

falsely detected faces are eliminated. Tables 5.9, 5.10 and 5.11 show that the false positive rates were zero. Similar to before, the face in Test 5 in Table 5.9 was not detected. There were also miss rates of 50% for some tests. Despite having 0% false positive rates, there were also 0% hit rates. Table 5.9 shows that Test 14 to Test 19 had a 0% hit rate or 100% miss rate. These tests correspond to distances of 3.43 m to 7.87 m. Table 5.10 shows that the hit rate was 0% for Test 15, 18 and 19. Table 5.11 shows that the hit rate was 0% for Test 15, 17, 18 and 19. Therefore the modified face detector using eye detection is safely effective for distances less than or equal to 2.54 m. However, the execution time was longer compared to the previous cases. This is expected since the process of checking for eyes in each detected face requires more processing. The face detection performance was similar using the 13 MP, 8 MP and 5 MP cameras. Given that the face detection with the aid of eye detection performed the best for distances less than or equal to 2.54 m, this face detection modification was used in the class attendance system.

The face detection application was implemented using the DetectMultiScale function in the Emgu CV library. Therefore, the factors that will affect the hit rate, miss rate and false positive rate are the input parameters to this function. For example, increasing the minimum number of neighbor rectangles can reduce the number of false detections, but the number of missed detections is increased. Additionally, the images used for training the cascaded classifier can affect the performance of the face detection algorithm. For example, a large difference between the resolution of the training images and the test image may have a negative impact on the performance.

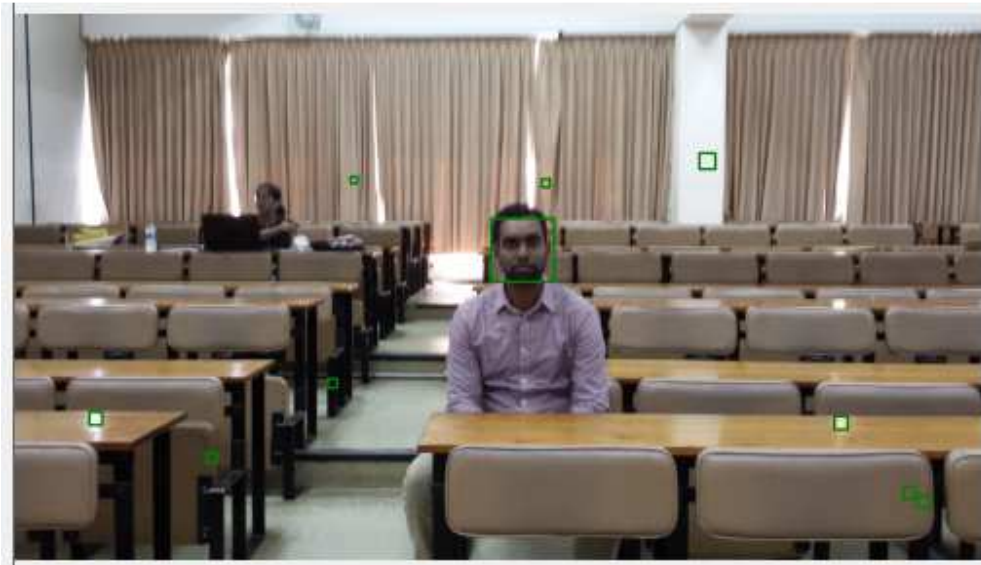


Figure 6.1: Instance where false detections occurred

An instance where false detections occurred is illustrated in Figure 6.1. As seen in Figure 6.1, a part of the wall, desk, chair and curtain were falsely detected as faces. For some false detections, the detected region was homogenous. One factor that may cause a homogenous region to be incorrectly detected as a face is the different pixel values within a region that appears to be homogeneous. For example, it is expected that for a white background, all the pixel values are supposed to be 255. However, this is not the case as the pixel values can vary. Matlab was used to read in the image and convert it to grayscale. The pixel values in the detected homogenous region were found to be different. This may be due to the small movements that occurred when capturing the images, since the camera was held by hand. Considering the size of the detection window (24×24) and the resolution of the image (4128×2322), even if an area was homogenous, any dissimilarity in pixels within the window can result in an incorrect detection.

For detected regions that were not homogenous, the false detections may be caused by the input parameters in the DetectMultiScale function and the images used for the cascaded classifier. Additionally, the Emgu CV library may have bugs which may result in erroneous output from the DetectMultiScale function.

The camera's autofocus feature can affect the quality of the image which may have an impact on the performance of the face detection algorithm. For example, when the camera focuses on an object, the image may be blurred temporarily to varying degrees which degrades the quality of the image. Hence, the quality of the image will not correspond to the resolution of the image. A high resolution image can have the quality of a low resolution image. The autofocus feature in the camera can be disabled. However, the distance between the individual and the camera will have to be constant in order to ensure that the optimum quality is achieved.

Another factor which may have an impact on the performance of the face detection algorithm is camera movement. For example, the test images were captured from a camera held by hand (unstable base), which causes small movements when capturing the image. This may result in incorrect detections, since the movement alters the quality of the image. The camera can be placed on a tripod for a stable base.

For all test scenarios, the face was not detected in Test 5 (90 degree side angle test). This is expected since the Viola and Jones algorithm is a frontal face detection algorithm. The face detection algorithm will only detect faces of individuals who are facing the camera. However, the results show that the algorithm is resilient to some level of head movement. The performance seem to improve for lower camera resolutions using the modified face detection algorithm with eye detection. Referring to Figure 6.1, the non-participating student seen in the

back of the class accounts for all tests that had a 50% miss rate. The non-participating student did not comply with the rules of the tests, for example, being present for all tests and looking forward without obstructing the face. However, there were some instances of compliance which would produce a 100% hit rate, given that the other individual's face was detected.

6.2 Face Recognition

As mentioned before, the same tests used for face detection were also used for face recognition. Recall indicates how many predicted faces that exists in the training set are correctly matched. Since there was only one face to be recognized for all the tests except the class images tests, the recall would be 100% if the recognition was correct and 0% if the recognition was incorrect. Additionally, the precision would be a non-zero value if there was correct recognition (face of interest correctly recognized) and 0% if there was incorrect recognition. Precision indicates how many predicted faces that were matched with the faces in the training set are correct. For example, if there were 3 faces that were matched with the faces in the training set and 1 was correctly matched, then the precision would be 33.33%. Accuracy indicates how many predicted faces are correctly matched with the faces in the training set and correctly rejected (classified as unknown). For example, if there were 8 predicted faces, with 1 correctly matched and 5 correctly rejected, then the accuracy would be 75%. There can be instances where the precision is zero and the accuracy is non zero. In these instances, the recall would be zero because the face of interest was not correctly recognized. Additionally, the accuracy would only indicate how well the faces were rejected in these instances.

6.2.1 Eigenfaces Method

Tables 5.12, 5.13 and 5.14 in Chapter 5 Testing and Results show the results for the Eigenfaces method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the full training set (160 faces). Table 5.12 shows that there were correct recognitions for 2 side angle tests and incorrect recognitions for all of head tilt tests. For the distance tests there were instances of incorrect recognition, as seen in Tables 5.12 and 5.13. It is expected that the face would not be correctly recognized as the distance from the camera increased. However, this was not the case. For example, the face was incorrectly recognized in the 3.43 m distance test but correctly recognized in the 4.32 m distance test as seen in Table 5.12. This occurrence was observed using the 13 MP and 8 MP cameras. There were 6 and 5 correct recognitions using the 13 MP and 5 MP cameras respectively for the distance tests. There were correct recognitions for all distance tests using the 5 MP camera as seen in Table 5.14.

Tables 5.15, 5.16 and 5.17 in Chapter 5 Testing and Results show the results for the Eigenfaces method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the training set without side angle faces (96 faces). In this case there were incorrect recognitions for all side angle and head tilt tests as seen in Table 5.15. For the distance tests, there were instances of incorrect recognition as seen in Table 5.15 and Table 5.16. Similar to the previous case, there were correct recognitions for longer distances even if there were incorrect recognitions for shorter distances using the 13 MP and 8 MP cameras. There were 6 correct recognitions using the 13 and 8 MP cameras. There were correct recognitions for all distance tests using the 5 MP camera as seen in Table 5.17.

Tables 5.18, 5.19 and 5.20 in Chapter 5 Testing and Results show the results for the Eigenfaces method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the training set with only frontal views (16 faces). In this case there were 4 correct recognitions for the side angle tests as seen in Table 5.18. Additionally, there were 3 correct recognitions for head tilt tests. For the distance tests, there were instances of incorrect recognition as seen in Table 5.18 and Table 5.19. Similar to the previous case, there were correct recognitions for longer distances even if there were incorrect recognitions for shorter distances using the 13 MP and 8 MP cameras. There were 6 correct recognitions using the 13 MP and 8 MP cameras. There were correct recognitions for all distance tests using the 5 MP camera as seen in Table 5.20.

6.2.2 Fisherfaces Method

Tables 5.21, 5.22 and 5.23 in Chapter 5 Testing and Results show the results for the Fisherfaces method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the full training set (160 faces). There were correct recognitions for 3 side angle tests as seen in Table 5.21. Similar to the results for Eigenfaces method, there were incorrect recognitions for all head tilt tests. Additionally, there were correct recognitions for longer distances even if there were incorrect recognitions for shorter distances using the 13 MP and 8 MP cameras as seen in Table 5.21 and Table 5.22 respectively. There were 4 and 5 correct recognitions using the 13 MP and 8 MP cameras respectively. Furthermore, there was an incorrect recognition for the shortest distance using the 13 MP and 8 MP cameras. However, using the Eigenfaces method, there was a correct recognition for the shortest distance. There

were correct recognitions for all distance tests except the longest distance test using the 5 MP camera as seen in Table 5.23.

Tables 5.24, 5.25 and 5.26 in Chapter 5 Testing and Results show the results for the Fisherfaces method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the training set without side angle faces (96 faces). Similar to the results for the Eigenfaces method, there were incorrect recognitions for all side angle tests as seen in Table 5.24. However, in this case there was one correct recognition for the head tilt tests. For the distance tests, there were instances of incorrect recognition as seen in Table 5.24 and Table 5.25. Similar to the previous case, there were correct recognitions for longer distances even if there were incorrect recognitions for shorter distances using the 13 MP and 8 MP cameras. There were 6 and 7 correct recognitions using the 13 MP and 8 MP cameras respectively. Similar to the results for the Eigenfaces method, there were correct recognitions for all distance tests using the 5 MP camera as seen in Table 5.26.

Tables 5.27, 5.28 and 5.29 in Chapter 5 Testing and Results show the results for the Fisherfaces method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the training set with only frontal views (16 faces). There were 2 correct recognitions for the side angle and head tilt tests as seen in Table 5.27. For the distance tests, there were instances of incorrect recognition as seen in Table 5.27 and Table 5.28. Similar to the previous case, there were correct recognitions for longer distances even if there were incorrect recognitions for shorter distances using the 13 MP and 8 MP cameras. There were 6 and 7 correct recognitions using the 13 MP and 8 MP cameras respectively. Similar to the results for the Eigenfaces method, there were correct recognitions for all distance tests using the 5 MP camera as seen in Table 5.29.

6.2.3 LBPH Method

Tables 5.30, 5.31 and 5.32 in Chapter 5 Testing and Results show the results for the LBPH method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the full training set (160 faces). There were correct recognitions for 3 side angle tests as seen in Table 5.30. Similar to the results for the Eigenfaces and Fisherfaces methods, there were incorrect recognitions for all head tilt tests. For the distance tests, there were correct recognitions for longer distances even if there were incorrect recognitions for shorter distances using the 13 MP, 8 MP and 5 MP cameras as seen in Tables 5.30, 5.31 and 5.32 respectively. There were 3, 4 and 4 correct recognitions using the 13 MP, 8 MP and 5 MP cameras respectively.

Tables 5.33, 5.34 and 5.35 in Chapter 5 Testing and Results show the results for the LBPH method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the training set without side angle faces (96 faces). Like the case using 160 faces, there were correct recognitions for 3 side view tests as seen in Table 5.33. Similar to the results for the Eigenfaces method, there were incorrect recognitions for all head tilt tests. For the distance tests, there were correct recognitions for longer distances even if there were incorrect recognitions for shorter distances using the 13 MP, 8 MP and 5 MP cameras as seen in Tables 5.33, 5.34 and 5.35 respectively. There were 3, 4 and 4 correct recognitions using the 13 MP, 8 MP and 5 MP cameras respectively.

Tables 5.36, 5.37 and 5.38 in Chapter 5 Testing and Results show the results for the LBPH method using the 13 MP, 8 MP and 5 MP cameras respectively. These tables in particular show the results using the training set with only frontal views (16 faces). Like the case using 160 faces, there were correct recognitions for 3 side angle tests as seen in Table 5.36. Similar to

the results for the Fisherfaces method, there were correct recognitions for 2 head tilt tests. For the distance tests, there was a correct recognition only for the 1.65 m distance test using the 13 MP camera as seen in Table 5.36. Using the 8 MP camera, there were correct recognitions for longer distances even if there were incorrect recognitions for shorter distances as seen in Table 5.37. Using the 5 MP camera there were correct recognitions for the 1.65 m and 2.54 m distance tests.

6.2.4 Face Recognition Comparisons

Tables 5.39, 5.40 and 5.41 in Chapter 5 Testing and Results show the average precision, accuracy and recall for the Eigenfaces method. These tables in particular pertain to the distance tests. Table 5.39 shows that as the camera resolution decreased, the average precision increased for the 160 faces training set. However, the average accuracy and recall varied. The average recall was equivalent to the pass rate for the tests. For example, an average recall of 75% indicated that the face was correctly recognized in 6 out of 8 tests. Therefore, this was used as the deciding factor for selecting the face recognition method. The average accuracy accounts for correctly matched and rejected faces while the average precision accounts for correctly matched faces among all the predicted faces. Both performance metrics are affected by the threshold parameter for the Eigenfaces and Fisherfaces methods. If the threshold was set to a lower value then more faces would be rejected. This could increase the average accuracy and precision since the number of false positives would be reduced. However, setting the threshold too low can incorrectly reject faces. Table 5.40 shows that as the camera resolution decreased, the average precision and accuracy increased for the 96 faces training set. The average recall was similar using the 13 MP and 8 MP cameras. However, it was 100% using the 5 MP camera.

Table 5.41 shows a similar behavior for the 16 faces training set. However, the average precision and accuracy seem to be lower compared to the 160 faces and 96 faces training sets.

Tables 5.42, 5.43 and 5.44 in Chapter 5 Testing and Results show the average precision, accuracy and recall for the Fisherfaces method. These tables in particular pertain to the distance tests. Table 5.42 shows that as the camera resolution decreased, the average recall increased for the 160 faces training set. However, the average precision and accuracy varied. Table 5.43 shows that as the camera resolution decreased, the average precision, accuracy and recall increased for the 96 faces training set. Table 5.41 shows a similar behavior for the 16 faces training set. The average recall improved using the 96 faces training set. However, there was no further improvements using the 16 faces training set.

Tables 5.45, 5.46 and 5.47 in Chapter 5 Testing and Results show the average precision, accuracy and recall for the LBPH method. These tables in particular pertain to the distance tests. Table 5.45 shows that as the camera resolution decreased, the average accuracy increased for the 160 faces training set. Table 5.46 and Table 5.47 show a similar behavior for the 96 and 16 faces training sets respectively. The average precision and recall improved using the 8 MP and 5 MP camera resolutions. However, the average precision and recall degraded for the 16 faces training set, but were similar for the 160 and 96 faces training sets. Generally, for the distance tests, the performance of the Eigenfaces and Fisherfaces methods were almost similar and the performance of the LBPH method was the worst. The performance for all the face recognition methods seem to improve for decreasing camera resolutions. The average precision, accuracy and recall were found using one face for recognition. More tests were done on class images where there were more than one face to be recognized. These tests will be examined later in the chapter.

Tables 5.48, 5.49 and 5.50 in Chapter 5 Testing and Results show the average precision, accuracy and recall for all the face recognition methods. These tables in particular pertain to the side angle and head tilt tests. The average precision was the worst using the Eigenfaces method and best using the LBPH method for the 160 and 96 faces training sets. However, for the 16 faces training set the average precision was the worst using the Fisherfaces method and the best using the LBPH method. The average accuracy was the worst using the LBPH method and the best using the Eigenfaces method for the 160 faces training set. However, for the 96 and 16 faces training sets the average accuracy was the worst using the Fisherfaces method and the best using the LBPH method. The average recall was the worst using the Eigenfaces method and the best using the LBPH method for the 160 and 96 faces training sets. The Fisherfaces method had the same average recall as the LBPH method for the 160 faces training set. However, for the 16 faces training set the average recall was the best using the Eigenfaces method and the worst using the Fisherfaces method. For all face recognition methods, the average recall was the best using the 16 faces training set. Generally, for the side angle and head tilt tests, the performance of the Eigenfaces method was the best using the 16 faces training set.

The 160 faces training set contained faces that had side angles and head tilts for each person. It was expected that this training set is supposed to have the best performance for face recognition since there is more information for each person. However, it seems that having more images per person had a negative impact on the performance. Face images with side angles contained some non-face data. For example, if the person was looking to the right, there was some non-face data (background) on the right side of the image. Therefore, the faces with side angles were removed for the 96 faces training set. However, the performance was worse.

Therefore all faces with side angles and head tilts were removed for the 16 faces training set. Additionally, faces with smiles were removed for the 16 faces training set. The performance was the best using this training set. Therefore, the ideal training set for the Eigenfaces, Fisherfaces and LBPH methods should contain one image per person.

The LBPH method was faster than the Eigenfaces and Fisherfaces methods using the full training set. In the Eigenfaces and Fisherfaces methods, the input face has to be compared against all the components (weights) for each face in the training set. In the LBPH method, the input face has to be compared to only one component (face histogram) for each face in the training set. Therefore, the LBPH method will be faster if the number of components is large for the Eigenfaces and Fisherfaces methods. Furthermore, if the number of components is small, then the Eigenfaces and Fisherfaces methods will perform faster. The number of components was set to the total number of faces in the training set. Hence, the number of faces in the training set affected the speed of the face recognition algorithms as noted in the results. Given that the Eigenfaces method was found to be the best using the 16 faces training set, this face recognition method with the reduced training set was used in the class attendance system.

6.2.5 Class Images

Class images were captured using the 13 MP camera for further testing. The 8 MP and 5 MP cameras were not used because at the time of doing this test, the cameras were no longer available. The class images contained one or more students. Not all the students whose faces are in the database were captured in the images. The first round of testing captured images of students for each course. As seen in Table 5.51, the recall was 100% for all the face recognition methods. The precision and accuracy were similar for the Fisherfaces and LBPH methods.

However, for the Eigenfaces method the precision and accuracy were the best at 57.14%. Using the Eigenfaces method, 4 faces were correctly recognized out of 7 faces.

The second round of testing captured images of students for one course. As seen in Table 5.52, the recall was 100% for all the face recognition methods except the LBPH method. The precision and accuracy were similar for the Eigenfaces and Fisherfaces methods. However, for the LBPH method the precision and accuracy were zero. The Eigenfaces and Fisherfaces methods had the best precision and accuracy at 33.33%. Using the Eigenfaces and Fisherfaces methods, 1 face was correctly recognized out of 3 faces.

The third round of testing captured images of students at a closer distance for one course. As seen in Table 5.53, the recall was 100% for all the face recognition methods. The LBPH method had the best precision and accuracy at 60%. However, the Fisherface method had the worst precision and accuracy at 20%. Using LBPH method, 3 faces were correctly recognized out of 5 faces.

The final round of testing captured images of students at an even closer distance for one course. As seen in Table 5.54, the recall was 100% for all the face recognition methods. The precision and accuracy were similar for the Fisherfaces and LBPH methods. However, for the Eigenfaces method the precision and accuracy were the best at 66.67%. Using the Eigenfaces method, 2 faces were correctly recognized out of 3 faces. Generally, the Eigenfaces method performed the best.

The face recognition application was implemented using the face recognition classes in the Emgu CV library. The face recognition classes are in the form of constructors. Therefore, one factor which may affect the performance of the face recognition methods is the input

parameters of the constructors. For example, setting a high threshold parameter for the Eigenfaces method may increase the number of faces being incorrectly matched with the faces in the training set. Additionally, the Emgu CV library may have bugs which may result in erroneous output from the Predict function in the face recognizer class.

Similar to the face detection case, the performance of the face recognition methods may be affected by the camera's autofocus feature which can affect the quality of the image. For example, when the camera focuses on an object, the image may be blurred temporarily to varying degrees which degrades the quality of the image. Hence, the quality of the image will not correspond to the resolution of the image. A high resolution image can have the quality of a low resolution image. The autofocus feature in the camera can be disabled. However, the distance between the individual and the camera will have to be constant in order to ensure that the optimum quality is achieved.

Another factor which may have an impact on the performance of the face recognition methods is camera movement. For example, the test images were captured from a camera held by hand (unstable base), which causes small movements when capturing the image. This may result in incorrect recognitions, since the movement alters the quality of the image. The camera can be placed on a tripod for a stable base.

There can be cases where there are similar faces. In these cases, the face recognition methods may have issues differentiating them. Figure 6.2 illustrates some examples of incorrect recognitions. Examples 1 and 3 show a case where the face is matched with the wrong face that has some resemblance. Example 2 shows a case where the face is matched with the wrong

face that also has glasses. In these examples, the similarity was not strong, but there can be cases where there is a strong similarity, for example, twins.

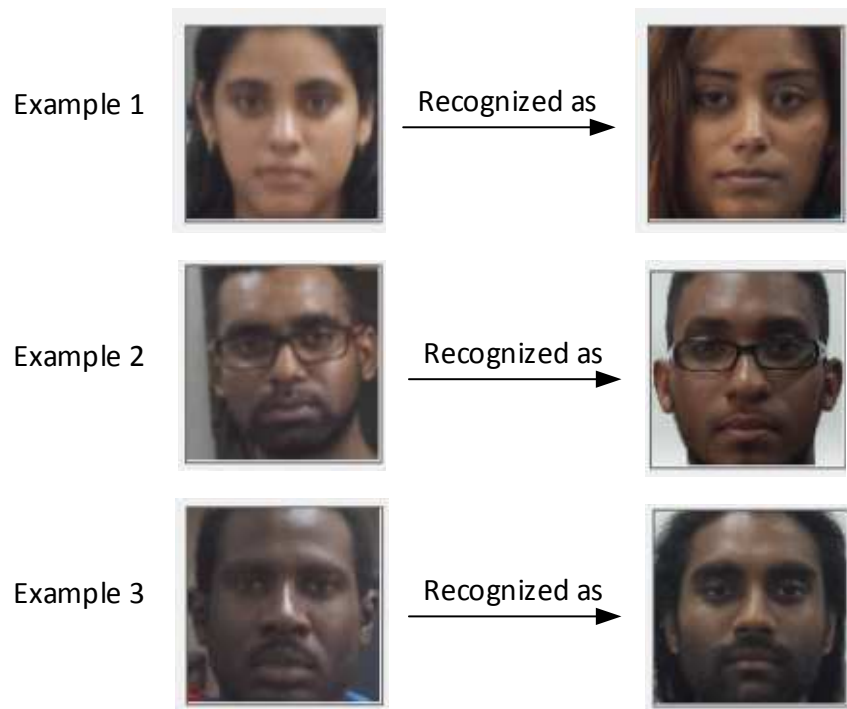


Figure 6.2: Examples of incorrect recognitions

The performance of the face recognition methods can be improved by aligning the input face with each face in the training set. However, there is the issue of finding the appropriate landmarks. The eyes can be used as a landmark. However, any errors in detecting the eyes will have a negative impact on the face alignment. For example, if more than two eyes were detected in a face, then it would be impossible to align the detected eyes with the landmark eyes. The performance of the face recognition methods can also be improved by applying the same conditions as in the training set for capturing the class image such as similar lighting, distance, camera resolution and background. However, this would impose too many constraints making the process impractical.

The face recognition methods presented by Turk and Pentland (1991), Belhumeur, Hespanha, and Kriegman (1997) and Ahonen, Hadid, and Pietikäinen (2004) seem to be tested using the face images in the training set. There was no indication of using separate test images. Furthermore, there was also no indication of using face detection. This would imply that the whole image was compared against the images in the training set. The whole image may not necessarily contain only the face. It may also contain some level of non-face data, for example, background. Additionally, if the images in the training set were used for testing, then the conditions are controlled. For example, the distance between the individual and the camera is constant and known. In some training sets, there are variations in the conditions, for example, lighting. However, the variations in the conditions are known and controlled. The tests performed using the same images in the training set evaluate how well the face recognition methods are able to match faces under controlled conditions. However, this may not be practical for real world applications, since having controlled conditions may not be possible all the time.

In this research, face detection was used to extract only the face data. Additionally, the test images had variations in lighting, background, camera resolution, head orientation and distance. The variations in lighting were not directional as in the case presented by Belhumeur, Hespanha, and Kriegman (1997). The changes in lighting were due to the time of day and the source of the light, for example sunlight and/or fluorescent light. The lighting and background conditions were uncontrolled. The distance and head orientation conditions were controlled and uncontrolled. This research provided more in depth testing compared to the previous work done by the mentioned authors. The wide range of tests evaluate the face recognition methods under real world conditions.

6.3 Class Attendance System

Table 5.55 shows that there was one correctly reported class attendance out of 6 reported class attendances. Table 5.56 shows that there were 11 correctly reported student attendances out of 16 reported student attendances. Table 5.57 shows that incorrectly recognized students have a negative impact on the student attendance. There were 21 correctly reported attendances out of 31 attendances. The precision, accuracy and recall were 100.00%, 67.74% and 44.44% respectively. The results show that the students who were incorrectly recognized were reported as absent. Additionally, one student whose face was not detected was also reported as absent. The student's face was not detected because the student was not looking at the camera. The performance of the class attendance system was above average because of the flaws in face recognition.

The use of face detection and face recognition in a class attendance system is beneficial, since it is a non-intrusive method of capturing biometric information. For example, a device does not have to be passed around in class for students to enter their information. However, the results show that the performance of the face detection and recognition methods must be perfect. Any deficiencies in the methods will have adverse effects on the class attendance system. The face detection method combined with eye detection used in this research yielded perfect results when the individual is facing directly at the camera within a limited distance. Therefore, further research must be done in finding the best face recognition method. Ideally, the face recognition method should be resilient to camera resolutions, changes in lighting, distance from the camera and head orientations. Additionally, it should require a minimal amount of images per person in the training set. Similar to combining eye detection and face detection for improved performance, a configurable combination of face recognition and

another type of biometric recognition such as fingerprint recognition can be used for improved performance.

In this research, the results for the face recognition methods show that having only one face for each person in the training set will produce the best results. However, this may not be the case for new and improved face recognition methods. The new face recognition methods may require multiple images of one person for best results. Although the time spent on capturing one image of one person under controlled settings may be insignificant, the time spent on capturing multiple images of one person will be considerable. Additionally, an image may be captured more than once under the same setting because the student failed to comply with the rules, for example, the student blinked. This was one of the challenges faced in this research. Furthermore, the images of the students need to be updated every year since the appearance of the students may change, for example, weight gain. This process may be impractical for an institution with a large number of students, since all the images may not be captured before classes begin. However, this issue can be resolved through the institution's administration. For example, setting a sufficient period for capturing the images of all the students before the start of classes. With improved face recognition, the presented class attendance system would be extremely useful.

The class attendance system developed in this research reduces the time and effort required by students and teachers in recording the class attendance. An image of the class is first captured and uploaded to the webserver using a smartphone. The attendance records are then updated and retrieved using an application installed on a PC. This method is much faster than the traditional method of calling out each student's name and recording the attendance. One challenge that may arise is the uploading of images to the webserver. The size of the captured

images from a smartphone ranges from 1 MB to 3 MB. Images of these sizes may take some time to be uploaded to the server. However, this issue can be resolved by having a high speed internet connection and multiple wireless access points.

The class attendance system provided the class attendance, student attendance, students who failed to meet the attendance requirement, the details of the attendance and the student information. The class attendance showed the number of students who were present for a particular day (as a percentage). The student attendance showed the number of days a particular student was present (as a percentage). Only the student id is displayed. However, the student information can be used to get the student's name, picture and the courses that the student is registered for. The information on the students who failed to meet the attendance requirement is useful for institutions that have this requirement. Furthermore, this information can be used to determine if there is any connection between the pass rate and the attendance.

The details of the class attendance system done by Kar et al. (2012), Shehu and Dika (2010), Patil and Shukla (2014) and Samlal (2013) were already discussed in Chapter 2 Literature Review. Kar et al. (2012) presented a class attendance system which required a webcam and a minimum distance of 0.5 m between the camera and the student. Shehu and Dika (2010) presented a system which required a digital camera to be installed in the classroom. Furthermore, a computer with internet access is required in each classroom. Patil and Shukla (2014) presented a system which used a camera connected to a Raspberry pi module, that is installed at the front of the classroom. Samlal (2013) presented a system which used a cloud application and an Android application. Furthermore, smartphones were used to capture the images of the classroom. The class attendance systems done by Shehu and Dika (2010) and Patil and Shukla (2014) continuously capture images of the classroom. This method of

capturing class images requires no action by the students and teacher. However, the systems done by Kar et al. (2012) and Samlal (2013) require the teacher or lecturer to capture the class images and the students have to look at the camera.

In this research, the class image is captured from a smartphone by the teacher or lecturer. Although, some action is required by the students and teacher, this method is still faster than the traditional method of calling out students' names. Additionally, smartphones are becoming more affordable and it is highly likely that teachers or lecturers have smartphones. This reduces the cost of the class attendance system, since no additional expense is required to purchase and install cameras in the classroom. The smartphones have the ability to upload the class images to the webserver. No additional cables are required since smartphones are equipped with Wi-Fi. However, wireless internet access and wireless access points are required. Many large institutions such as universities already have this requirement. Since the database is stored on a webserver, the attendance records can be accessed from any PC with internet access and the class attendance application installed. Therefore, the attendance records can be retrieved from outside of the institution's compound. In this research, the webserver was local (not online). However, the database and face images can be easily exported to an online webserver.

The work done by Kar et al. (2012), Shehu and Dika (2010), Patil and Shukla (2014) and Samlal (2013) did not provided any results for the class attendance system. Hence, there were no performance metrics. Kar et al. (2012), Shehu and Dika (2010) and Samlal (2013) presented the results for the face detection and face recognition. Patil and Shukla (2014) presented a screen shot of the class attendance system. In this research, the results and the performance metrics for the class attendance system were presented. This is important since it provides the overall result of the combined effects from face detection and face recognition.

Chapter 7 Conclusion

A class attendance system using face detection and face recognition was developed. An extension of the Viola and Jones algorithm was used for face detection and eye detection. The combination of eye detection and face detection resulted in perfect performance. There were zero missed faces and zero falsely detected faces. Furthermore, the modified face detection algorithm was resilient to some level of head movement. Tests were done to detect the face of a single student and the performance of the face detection algorithm improved for lower camera resolutions. However, face detection was restricted to a limited distance between the camera and the student. A database containing the student's faces was created for face recognition. However, the database did not contain all the students for all the courses.

The Eigenfaces, Fisherfaces and LBPH methods were examined for face recognition. The Eigenfaces method was found to be the best with only one image per person in the training set and was used for face recognition. Additionally, this method was resilient to some level of side movements of the head, but not resilient to up/down movements. Therefore, face recognition was restricted to faces looking directly at the camera in order to reduce the number of incorrect face recognitions. However, incorrect face recognitions still exists. Tests were done to recognize the face of a single student and the performance of all the face recognition methods improved for lower camera resolutions. Furthermore, the performance for all the face recognition methods except the LBPH method improved with one image per person in the training set for the distance tests. However, the performance for all the face recognition methods improved with one image per person in the training set for the head orientation tests.

The performance of the class attendance system was above average due to the deficiencies in the face recognition method. Students were incorrectly reported as absent because their faces were not correctly recognized. Therefore, further research must be done in finding the best face recognition method. Similar to the system done by Samlal (2013), the presented system has a lower cost compared to the existing class attendance systems.

References

- Ahonen, Timo, Abdenour Hadid, and Matti Pietikäinen. 2004. "Face Recognition with Local Binary Patterns." In *Computer Vision - ECCV 2004: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004. Proceedings, Part I*, edited by Tomás Pajdla and Jiří Matas, 469-481. Berlin, Heidelberg: Springer Berlin Heidelberg.
- Belhumeur, P. N., J. P. Hespanha, and D. Kriegman. 1997. "Eigenfaces vs. Fisherfaces: recognition using class specific linear projection." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* no. 19 (7):711-720. doi: 10.1109/34.598228.
- Beumier, C., and M. Acheroy. 2000. Automatic Face Recognition. Paper read at Symposium IMAGING., at The Netherlands.
- Brunelli, Roberto, and T. Poggio. 1993. "Face recognition: features versus templates." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* no. 15 (10):1042-1052. doi: 10.1109/34.254061.
- Cendrillon, Raphael. 1999. *Real Time Face Recognition using Eigenfaces*, Department of Computer Science and Electrical Engineering, University of Queensland.
- Coro, Christopher De. *Face Recognition using Eigenfaces*. Princeton University 2015 [cited 14 April 2016. Available from <https://www.cs.princeton.edu/~cdecoro/eigenfaces/>.
- Cox, Ingemar J., J. Ghosn, and P. N. Yianilos. 1996. Feature-based face recognition using mixture-distance. Paper read at Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on, 18-20 Jun 1996.

- Emgu. *CascadeClassifier.DetectMultiScale Method*. Emgu Corporation 2013a [cited 10 December 2015. Available from <http://www.emgu.com/wiki/files/2.4.10/document/html/944b73ce-a95c-5547-3651-12b691fdeb46.htm>.
- Emgu. *EigenFaceRecognizer Constructor*. Emgu Corporation 2013b [cited 10 December 2015. Available from <http://www.emgu.com/wiki/files/2.4.10/document/html/d221454e-64fc-3526-270e-201570563cc6.htm>.
- Emgu. *FaceRecognizer.Predict Method*. Emgu Corporation 2013c [cited 10 December 2015. Available from <http://www.emgu.com/wiki/files/2.4.10/document/html/86a38b0c-314e-949d-f7b3-3e0d8846ba2d.htm>.
- Emgu. *FaceRecognizer.Train Method*. Emgu Corporation 2013d [cited 10 December 2015. Available from <http://www.emgu.com/wiki/files/2.4.10/document/html/2c4e34d5-0dab-a959-e0e6-aa86065c2031.htm>.
- Emgu. *FisherFaceRecognizer Constructor*. Emgu Corporation 2013e [cited 10 December 2015. Available from <http://www.emgu.com/wiki/files/2.4.10/document/html/c9ed754a-f3fc-b48e-8d08-9767d5055141.htm>.
- Emgu. *LBPFaceRecognizer Constructor*. Emgu Corporation 2013f [cited 10 December 2015. Available from <http://www.emgu.com/wiki/files/2.4.10/document/html/ff50adf5-db52-6e39-c694-7a06846b6f2c.htm>.

- Emgu. *Main Page*. Emgu Corporation 2015 [cited 10 December 2015. Available from http://www.emgu.com/wiki/index.php/Main_Page.
- Fawcett, Tom. 2006. "An introduction to ROC analysis." *Pattern Recognition Letters* no. 27 (8):861-874.
- Fisher, R. A. 1936. "The use of multiple measures in taxonomic problems." *Annals of Eugenics* no. 7:179-188.
- Freund, Yoav, and Robert E. Schapire. 1996. Experiments with a New Boosting Algorithm. Paper read at Proceedings of the Thirteenth International Conference on Machine Learning.
- Freund, Yoav, and Robert E. Schapire. 1997. "A decision-theoretic generalization of on-line learning and an application to boosting." *Journal of Computer and System Sciences* no. 55:119-139.
- Georghiades, Athinodoros S. *The Yale Face Database* 1997 [cited 10 November 2015. Available from <http://cvc.cs.yale.edu/cvc/projects/yalefaces/yalefaces.html>.
- Grgic, Mislav, and Kresimir Delac. *Face Recognition Homepage* 2005 [cited 10 November 2015. Available from <http://face-rec.org/databases/>.
- Grgic, Mislav, Kresimir Delac, and Sonja Grgic. 2011. "SCface - surveillance cameras face database." *Multimedia Tools and Applications* no. 51 (3):863-879.
- Huang, Gary B., Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. University of Massachusetts, Amherst.
- Itseez. *OpenCV About*. Itseez 2015 [cited 10 December 2015. Available from <http://opencv.org/about.html>.

- Jafri, Rabia, and Hamid R. Arabnia. 2009. "A Survey of Face Recognition Techniques." *Journal of Information Processing Systems* no. 5 (2):41-68.
- Jebara, Tony S. 1996. *3D Pose Estimation and Normalization for Face Recognition*, Department of Electrical Engineering, McGill University.
- Kah-Kay, Sung, and T. Poggio. 1998. "Example-based learning for view-based human face detection." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* no. 20 (1):39-51. doi: 10.1109/34.655648.
- Kanade, Takeo. 1973. *Picture Processing System by Computer Complex and Recognition of Human Faces*, Department of Information Science, Kyoto University, Japan.
- Kar, Nirmalya, Mrinal Kanti Debbarma, Ashim Saha, and Dwijen Rudra Pal. 2012. "Study of Implementing Automated Attendance System Using Face Recognition Technique." *International Journal of Computer and Communication Engineering* no. 1 (2):100-103.
- King, Andrew. 2003. A Survey of Methods for Face Detection.
- Lienhart, Rainer, Alexander Kuranov, and Vadim Pisarevsky. 2002. Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection. Intel Corporation, Santa Clara, CA 95052, USA: Microprocessor Research Lab, Intel Labs.
- Martinez, Aleix M., and A. C. Kak. 2001. "PCA versus LDA." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* no. 23 (2):228-233. doi: 10.1109/34.908974.
- Ojala, T., M. Pietikainen, and T. Maenpaa. 2002. "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns." *IEEE Transactions on Pattern Analysis and Machine Intelligence* no. 24 (7):971-987. doi: 10.1109/TPAMI.2002.1017623.

- Ojala, Timo, David Harwood, and Matti Pietikäinen. 1996. "A comparative study of texture measures with classification based on feature distributions." *Pattern Recognition* (29):51–59.
- OpenCV. *Cascade Classification*. OpenCV Dev Team 2014 [cited 10 December 2015]. Available from http://docs.opencv.org/2.4.10/modules/objdetect/doc/cascade_classification.html#cascadeclassifier-cascadeclassifier.
- Osuna, E., R. Freund, and F. Girosi. 1997. Training support vector machines: an application to face detection. Paper read at Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on, 17-19 Jun 1997.
- Patil, Ajinkya, and Mrudang Shukla. 2014. "Implementation of classroom attendance system based on face recognition in class." *International Journal of Advances in Engineering & Technology* no. 7 (3):974-979.
- Pietikäinen, Matti. *Local Binary Patterns*. Scholarpedia 2010 [cited 1 May 2016]. Available from http://www.scholarpedia.org/article/Local_Binary_Patterns.
- Romdhani, S., P. Torr, B. Scholkopf, and A. Blake. 2001. Computationally efficient face detection. Paper read at Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on, 2001.
- Rowley, H. A., S. Baluja, and T. Kanade. 1998. "Neural network-based face detection." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* no. 20 (1):23-38. doi: 10.1109/34.655647.
- Samlal, Daryl. 2013. Mobile Application For Image Based Classroom Attendance. The University of the West Indies.

- Schmidt, Adam, and Andrzej Kasiński. 2007. "The Performance of the Haar Cascade Classifiers Applied to the Face and Eyes Detection." In *Computer Recognition Systems 2*, edited by Marek Kurzynski, Edward Puchala, Michal Wozniak and Andrzej Zolnierok, 816-823. Springer Berlin Heidelberg.
- Schneiderman, H., and T. Kanade. 2000. A statistical method for 3D object detection applied to faces and cars. Paper read at Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on, 2000.
- Shehu, V., and A. Dika. 2010. Using real time computer vision algorithms in automatic attendance management systems. Paper read at Information Technology Interfaces (ITI), 2010 32nd International Conference on Information Technology Interfaces, 21-24 June 2010.
- Sirovich, L., and M. Kirby. 1987. "Low-dimensional Procedure for the Characterization of Human Faces." *Journal of the Optical Society of America A* no. 4:519-524.
- Turk, M. A., and A. P. Pentland. 1991. Face recognition using eigenfaces. Paper read at Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on, 3-6 Jun 1991.
- UC. *AT&T The Database of Faces*. AT&T Laboratories Cambridge 2002 [cited 10 November 2015. Available from <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>.
- UCSD. *Yale Face Database*. UCSD Computer Vision 2011 [cited 10 November 2015. Available from <http://vision.ucsd.edu/content/yale-face-database>.
- UWI, St Augustine Campus. 2014. *Undergraduate Regulations and Syllabuses 2014 - 2015, The Faculty of Engineering*.

- Vapnik, Vladimir N. 1995. *The Nature of Statistical Learning Theory*. New York: Springer.
- Viola, Paul, and Michael Jones. 2001. Robust Real-time Object Detection. In *Second International Workshop on Statistical and Computational Theories of Vision-Modeling, Learning, Computing, And Sampling*.
- Wolf, Lior, Tal Hassner, and Itay Maoz. 2011. Face recognition in unconstrained videos with matched background similarity. Paper read at Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, 20-25 June 2011.
- Zhang, Guangcheng, Xiangsheng Huang, Stan Z. Li, Yangsheng Wang, and Xihong Wu. 2004. "Boosting Local Binary Pattern (LBP)-Based Face Recognition." *Advances In Biometric Person Authentication* no. 3338:179-186.
- Zhang, Yankun, and Chongqing Liu. 2002. Face recognition using kernel principal component analysis and genetic algorithms. Paper read at Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on, 2002.

Appendix

What is on the CD

The CD attached to this report contains the following items:

1. **ECNG 6023 - David Akim 807002346.docx** – ECNG 6023 Microsoft Word report
2. **ECNG 6023 - David Akim 807002346.pdf** – ECNG 6023 PDF report
3. **ECNGClassAttendanceAndroidApp** – Class attendance Android application
4. **ECNG_Class_Attendance_Windows_App** – Class attendance Windows application
5. **ECNG_FaceDetection_with_EyeDetection_Windows_App** – Face detection Windows application with eye detection
6. **ECNG_FaceDetection_without_EyeDetection_Windows_App** – Face detection Windows application without eye detection
7. **ECNG_FaceRecognition_Windows_App** – Face recognition Windows application
8. **Webserver** – This folder contains images for the face database, attendance images, a PHP script to convert the base64 image string back to an image and a SQL file of the exported database

