Face Detection and Recognition Student Attendance System

Research · August 2018

CITATIONS
3

READS
91,556

1 author:

Jireh Jam
Manchester Metropolitan University
10 PUBLICATIONS **62** CITATIONS

SEE PROFILE

FACULTY OF SCIENCE, ENGINEERING AND COMPUTING

School of *Computer Science & Mathematics*

BSc DEGREE
IN
Computer Science with Network Communications

PROJECT REPORT

Name: Jireh Robert Jam

ID Number: K1557901

Project Title: Face Detection and Recognition Student Attendance System.

Project Type : Build

Date: 23/04/2018

Supervisor: Jean-Christophe Nebel

Kingston University London

Plagiarism Declaration

The following declaration should be signed and dated and inserted directly after the title page of your report:

Declaration

I have read and understood the University regulations on plagiarism and I understand the meaning of the word *plagiarism*. I declare that this report is entirely my own work. Any other sources are duly acknowledged and referenced according to the requirements of the School of Computer Science and Mathematics. All verbatim citations are indicated by double quotation marks ("…"). Neither in part nor in its entirety have I made use of another student's work and pretended that it is my own. I have not asked anybody to contribute to this project in the form of code, text or drawings. I did not allow and will not allow anyone to copy my work with the intention of presenting it as their own work.

Date 23/04/2017

Signature *JirrehJam*

Acknowledgements:

A big thank you to **Dr. Jean-Christophe Nebel** for his support and guidance throughout this project.

I also express gratitude to all my friends for their help and support especially those of the Computer Science Department.

I also thank **Jirdith Esther Qwasinwi Roberts** my daughter for her encouraging words; my parents **Rev. Dr. Ilaja and Susan Jam** for their love and prayers.

Most of all a big thanks to God Almighty for his guidance throughout course.
Thank you all for the encouragement and support!

Abstract:

This paper will show how we can implement algorithms for face detection and recognition in image processing to build a system that will detect and recognise frontal faces of students in a classroom. "A face is the front part of a person's head from the forehead to the chin, or the corresponding part of an animal" (Oxford Dictionary). In human interactions, the face is the most important factor as it contains important information about a person or individual. All humans have the ability to recognise individuals from their faces. The proposed solution is to develop a working prototype of a system that will facilitate class control for Kingston University lecturers in a classroom by detecting the frontal faces of students from a picture taken in a classroom. The second part of the system will also be able to perform a facial recognition against a small database.

In recent years, research has been carried out and face recognition and detection systems have been developed. Some of which are used on social media platforms, banking apps, government offices e.g. the Metropolitan Police, Facebook etc.

Table of Contents

CHAPTER ONE

Introduction:

In Face Detection and Recognition systems, the flow process starts by being able to detect and recognise frontal faces from an input device i.e. mobile phone. In today's world, it has been proven that students engage better during lectures only when there is effective classroom control. The need for high level student engagement is very important. An analogy can be made with that of pilots as described by Mundschenk et al (2011 p101)" Pilots need to keep in touch with an air traffic controller, but it would be annoying and unhelpful if they called in every 5 minutes". In the same way students need to be continuously engaged during lectures and one of the ways is to recognise and address them by their names. Therefore, a system like this will improve classroom control. In my own view based on experience, during my time as a teacher, I realised calling a student by his/her name gives me more control of the classroom and this draws the attention of the other students in the classroom to engage during lectures.

Face detection and recognition is not new in our society we live in. The capacity of the human mind to recognize particular individuals is remarkable. It is amazing how the human mind can still persist in identification of certain individuals even through the passage of time, despite slight changes in appearance.

Anthony (2014 p1) reports that, due to the remarkable ability of the human mind to generate near positive identification of images and facial recognition of individuals, this has drawn considerable attention for researchers to invest time in finding algorithms that will replicate effective face recognition on electronic systems for use by humans.

Wang et al (2015 p318) states that" the process of searching a face is called face detection. Face detection is to search for faces with different expressions, sizes and angles in images in possession of complicated light and background and feeds back parameters of face".

Face recognition processes images and identifies one or more faces in an image by analysing patterns and comparing them. This process uses algorithms which extracts features and compare them to a database to find a match. Furthermore, in one of most recent research, Nebel (2017, p. 1), suggest that DNA techniques could transform facial recognition technology, by the use of video analysis software which can be improved thanks to a completely advance in research in DNA analysis. By so doing, camera-based surveillance systems software to analyze DNA sequences, by treating a video as a scene that evolves the same way DNA does, to detect and recognize human face.

Motivation and Problem Definition:

This project is being carried out due to the concerns that have been highlighted on the methods which lectures use to take attendance during lectures. The use of clickers, ID cards swiping and manually writing down names on a sheet of paper as a method to track student attendants has prompted this project to be carried out. This is not in any way to criticize the various methods used for student attendance, but to build a system that will detect the number of faces present in a classroom as well as recognizing them. Also, a teacher will be able to tell if a student was honest as these methods mentioned can be used by anyone for attendance records, but with the face detection and recognition system in place, it will be easy to tell if a student is actually present in the classroom or not.

This system will not only improve classroom control during lectures, it will also possibly detect faces for student attendance purposes. I will use MATLAB to build and implement this system.

Project Aims and Objectives:

The aim and objectives of this project has been acquired after meeting with the client.

AIM	OBJECTIVES
To develop a prototype that will facilitate classroom control and attendance by face detection and recognition of students faces in a digital image taken by a mobile phone camera.	➢ The system should be able to detect students' frontal faces in a classroom within 30% accuracy. ➢ The system should be able to

	automatically reveal the number of students present on a GUI. ➢ Recognise student stored on a database of faces by matching them to images on a database with an accuracy within 30%. ➢ The system should be able to match detected students faces cropped from an image to those on a database on the system. ➢ The system should be able to process an image within 10 minutes to be able to achieve the objective of recognition by the end of a lecture. i.e. 5 names per hour per lecture. ➢ The algorithm implemented for the system's functionality will achieve system accuracy within 20%. ➢ The positive prediction should be within 20%. ➢ The system designed will be user friendly with a Graphical User Interphase that will serve as an access to the functionalities of the system.

Discussion regarding Ethics:

Regarding ethics aspects as well as data protection and safety in relation to this project, the system will be fully covered and protected by Kingston University data protection law and the UK data protection act (2017). The right to privacy is a priority and students have this will be used strictly for face detection and recognition purposes only. They will be informed of the use of the picture taken in class for the purpose of this project, and how their personal information will be used. The use of this application will be exclusively for facial detection and recognition during lectures in classrooms only.

Report Structure:

The following chapters are in this report

- Literature Review.
- Methodology and Analysis.
- Design.
- Implementation.
- Evaluation and Analysis of the System.
- Conclusion.

CHAPTER TWO

Literature Review:

In this chapter, a brief overview of studies made on face detection and recognition will be introduced alongside some popular face detection and recognition algorithms. This will give a general idea of the history of systems and approaches that have been used so far.

Overview of Face Detection and Recognition:

Most face recognition systems rely on face recognition algorithms to complete the following functional task as suggested by Shang-Hung Lin. (2000, p.2).

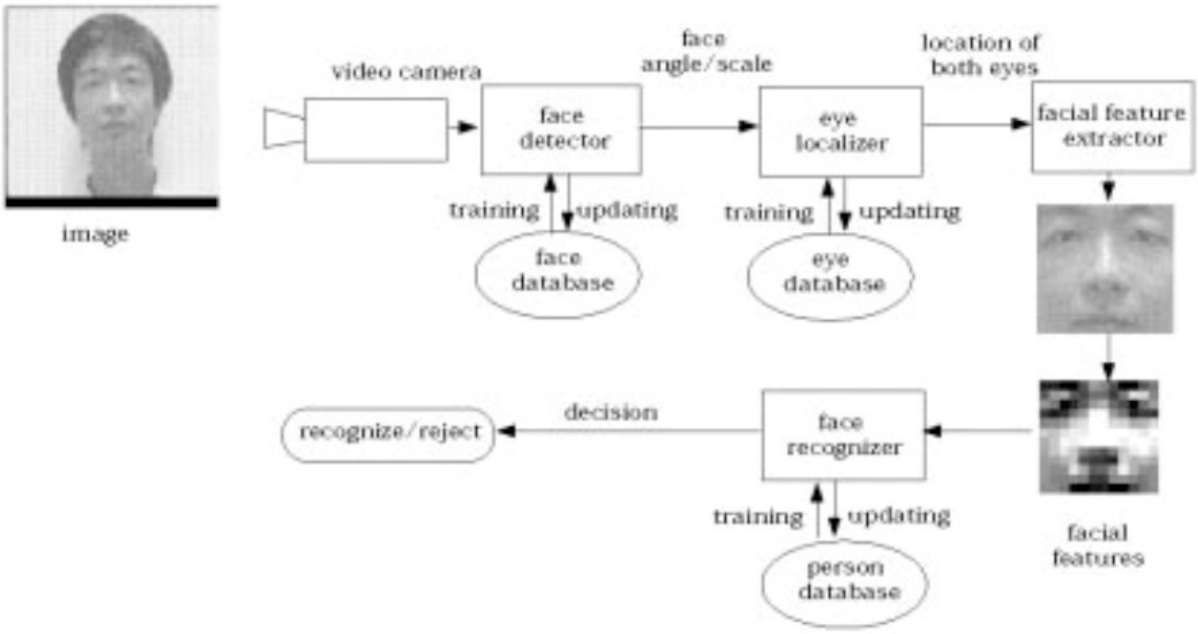


Fig2.2 Face recognition system framework as suggested by Shang-Hung Lin. (2000, p.2).

The figure below shows a simplified diagram from the framework for face recognition from the study suggested by Shang-Hung Lin. (2000).

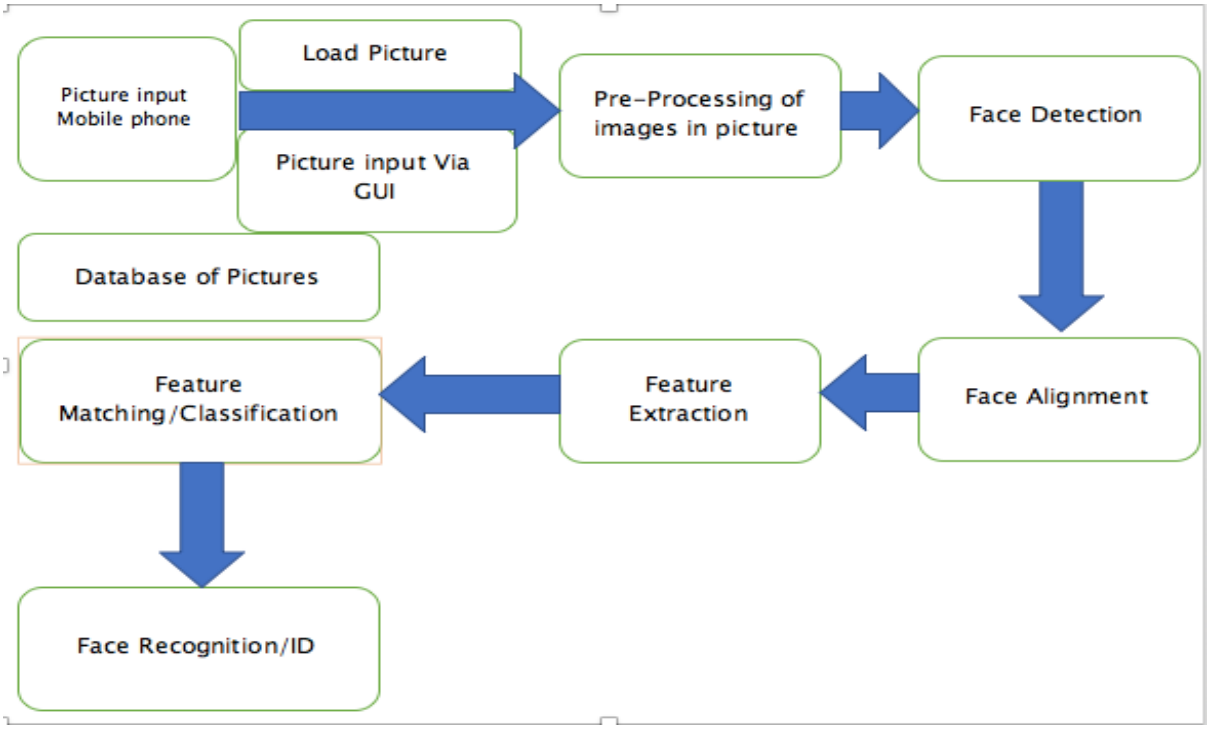


Figure 2.3 Face Detection and Recognition Flow Diagram.

From the figure, above, **Face Detection** or face detector will detect any given face in the given image or input video. **Face localization**, will detect where the faces are located in the given image/video, by use of bounding boxes. **Face Alignment** is when the system will find a face and align landmarks such as nose, eyes, chin, mouth for feature extraction. Feature extraction, extracts key features such as the eyes, nose, mouth to undergo tracking. Feature matching and classification. matches a face based on a trained data set of pictures from a database of about 200 pictures. Face recognition, gives a positive or negative output of a recognized face based on feature matching and classification from a referenced facial image.

Face detection is the process of locating a face in a digital image by any special computer software build for this purpose. Feraud et al (2000 p.77) discuss face detection as "To detect a face in an image means to find its position in the image plane and its size or scale ".

As figure 2.3 shows, the detection of a face in a digital image is a prerequisite to any further process in face recognition or any face processing software.

In early years, face detection algorithms focused mainly on the frontal part of the human face (Srinivasan, Golomb and Martinez, 2016, p.4434).

However, in recent years, Cynganek, (2013, p.346) suggest that newer algorithms take into consideration different perspectives for face detection. Researchers have used such systems but the most challenge that has been faced is to make a system detect faces irrespective of different illumination conditions. This is based on a study by Castrillón et al. (2011, p.483) on the Yale database which contains higher resolution images of 165 frontal faces. Face detection is often classified into different methods. In order to face the first major problem of the project (Detecting students faces), a wide range of techniques have been researched. These several face detection techniques/ methodologies have been proposed by many different researchers and often classified in major categories of different approaches. In this paper, we will look at some reviews and major categories of classification by different groups of researchers and relate it to the system.

Yang et al (2002) classifies face detection methodologies into four major categories: Knowledge-based, Feature invariant, Template matching and appearance-based approaches. **Knowledge Based** Method that uses human

knowledge or human coding to model facial feature based on nature of the human face such as two eyes, mouth and the nose. This is very easy to apply the rules but very difficult to detect in various

background depending on the pose and illumination. Low detection accuracy with small burden of calculation and short detection time.
Yang et al (2002 pp.36-37), in order to investigate this method created multiple resolution hierarchy of images by averaging and subsampling as on the figure 2.4 below.

(a) (b) (c) (d)

. (a) n = 1, original image. (b) n = 4. (c) n = 8. (d) n = 16. Original and corresponding low resolution images. Each square cell consists of pixels in which the intensity of each pixel is replaced by the average intensity of the pixels in that cell.

Figure 2.4 Taken from Detecting Faces in Images: A Survey (Yand et al (2002 p.36).
They subdivided these resolution hierarchies into three levels with level 1 being the lowest resolution which only searches for face candidate and further processed at finer resolutions. At level 2 they used the face candidate in level 1 to alongside local histogram equalization followed by edge detection. At level three, the surviving face candidate region uses a set rule responding to facial features such as mouth and eyes. They conducted their experiment on 60 images. Their system located faces on 50 of these images and 28 images gave false alarm, thus giving a success rate of 83.33% and a false alarm rate at 46.66%. **Feature-Based-Methods** that uses algorithms to look for structural features regardless of pose, viewpoint or lighting conditions to find faces. **Template Matching Methods;** uses standard facial patterns stored for use to correlate an input image with the stored pattern to compute for detection. Appearance Base Methods; uses a set of training sets of images to learn the templates and capture the representative of facial appearance. Furthermore, Yang et al. also carried out their experiments on a standard database set which is shown on the Table2.1 and Table 2.2 Yang et al. (2002, pp53-54) below with the detection rate results and false detection rates.

Test Sets for Face Detection

Data Set	Location	Description
MIT Test Set [154]	http://www.cs.cmu.edu/~har	Two sets of high and low resolution gray scale images with multiple faces in complex background.
CMU Test Set [128]	http://www.cs.cmu.edu/~har	130 gray scale images with a total of 507 frontal faces.
CMU Profile Face Test Set [141]	ftp://eyes.ius.cs.cmu.edu/usr20/ftp/testing_face_images.tar.gz	208 gray scale images with faces in profile views.
Kodak Data Set [94]	Eastman Kodak Corporation	Faces of multiple size, pose and under varying illumination in color images. Designed for face detection and recognition.

Table 2.1 showing standard database test set for Face Detection. Yang et al. (2002, p.53).

Experimental Results on Images from Test Set 1 (125 Images with 483 Faces) and Test Set 2 (23 Images with 136 Faces) (See Text for Details)

Method	Test Set 1		Test Set 2	
	Detection Rate	False Detections	Detection Rate	False Detections
Distribution based [154]	N/A	N/A	81.9%	13
Neural network [128]	92.5%	862	90.3%	42
Naive Bayes classifier [140]	93.0%	88	91.2%	12
Kullback relative information [24]	98.0%	12758	N/A	N/A
Support vector machine [107]	N/A	N/A	74.2%	20
Mixture of factor analyzers [175]	92.3%	82	89.4%	3
Fisher linear discriminant [175]	93.6%	74	91.5%	1
SNoW with primitive features [176]	94.2%	84	93.6%	3
SNoW with multi-scale features [176]	94.8%	78	94.1%	3
Inductive learning [38]	90%	N/A	N/A	N/A

Table 2.2 Results of Two Image Test Sets experimented. Yang et al. (2002, p.54).
As Table 2.2 summarizes, the experimental results show images of different training set with different

parameters of tuning which has a direct impact on the training performance. For example, the dimensionality reduction is carried out to improve computation efficiency and detection efficacy, with image patterns projected to a lower dimensional space to form a discriminant function for classification. Also, the training and execution time and the number of scanning windows in these experimented influenced the performance in some way. Hjelmås and Low, (2001) classifies face detection methodologies into two major categories. Image-based approaches, which is further sub-categorized into Linear subspace methods, Neural networks and statistical approaches.

Image Based Approaches; Most of the recent feature-based attempts in the same study by Hjelmås and Low, (2001 p.252) have improved the ability to cope with variations, but are still limited to head, shoulder and part of frontal faces. There is therefore need for techniques to cope in hostile scenarios such as detecting multiple faces in a cluttered scene, e.g. clutter-intensive background. Furthermore, this method ignores the basic knowledge of the face in general and uses face patterns from a given set of images. This is mostly known as the training stage in the detection method.

From this training stage, the system may be able to detect similar face patterns from an input image. A decision of face existence by the system is now established based on a comparison of the distance between the pattern from the input image and training image with a 2D intensity array extracted from the input image. Most image-based approaches use window-scanning techniques for face detection.

Window scanning algorithm searches for possible face locations at all scales. This method depends on window scanning algorithms. In other research carried out on this method which depends on window scanning algorithms, Ryu et al. (2006), in their study experimented the scanning window techniques discussed by Hjelmas and Low, (2001, p.252) in their system. They go further to experiment their system, based on a combination of various classifiers for a more reliable result compared to a single classifier. They designed multiple face classifiers which can take different representations of face patterns. They used three classifiers, Gradient feature classifier which contains the integral information of pixel distribution that returns certain invariability among facial features. The second classifier is Texture Feature which extracts texture features by correlation (uses joint probability occurrence of specified pixel), variance (measures the amount of local variations in an image) and entropy (measures image disorder). The third classifier used here is Pixel Intensity Feature, which extracts pixel intensity features of the eye, nose and mouth region for determining the face pattern. They further used Coarse-To-Fine Classification approach with their classifications for computational efficiency. Based on 1056 images which were obtained from the AT&T, BioID, Stirling, and Yale dataset, they achieved the results presented in Table 2.4 and Table 2.5 Ryu et al. (2006, p.489) The first face classification of their experiment with respect to shift in both x and y direction achieved a detection rate of 80% when images are shifted within 10 pixels in the x direction and 4 pixels in the y direction. The second and third face of their classification showed a detection rate of over 80% when 2 pixels were shift in both x and y directions respectively.

Test DB	Detection results						Reduction rates of # of scans
	Exhaustive full scanning method			Proposed scanning method			
	Detection rate	# of false	# of scans per image	Detection rate	# of false	# of scans per image	
IMM	96.2 %	28	755,418	95.7 %	8	72,273	90.4 %
Caltech	94.5 %	12	1,369,067	93.0 %	10	176,674	87.1 %
AR	95.7 %	22	1,128,541	95.0 %	6	142,136	87.3 %
WWW	80.7 %	46	4,312,203	83.7 %	12	513,152	88.1 %

Table 2.4: Results showing Exhaustive full scanning method and Proposed scanning method. Ryu et al (2006, p.489)

	Detection rate	# of false
Rowley method [3]	86.2%	23
Froba method [9]	87.8%	120
Feraud method [10]	86.0%	8
Proposed method (coarse-to-fine search)	86.6%	19
Proposed method (full search)	89.1%	32

Table 2.5: Performance Comparison by different Researchers and Proposed System by Ryu et al (2006, p.490)

As seen in Table 2.5, their system achieved a detection rate between 93.0% and 95.7%. Rowley et al. 1998 in their study on Neural Network-Based face detection, experimented on their system which applies a set of neural network-based filters to an image and then uses an arbitrator to combine the outputs. They tested their system against two databases of images. The CMU databased which was made of 130 images and the FERET database achieve a detection rate of 86.2% with 23 false detections. Feraud et al. (2001) also experimented on neural network-based face detection technique. They used a combination of different components in their system (motion filter, colour filter, prenetwork filter and large neural network). The prenetwork filter is a single multilayer perceptron, with 300 inputs corresponding to the extracted sizes of the subwindows, hidden with 20 neurons and outputs a face/nonface for a total of number of weights [reference]. These components, with a combination of neural network achieved an 86.0% detection rate with 8 false detections, based on a face database of 8000 images from Sussex Face Database and CMU Database which is further subdivided into different subsets of equal sizes corresponding to different views. (page 48). Table 2.6 and Table 2.7 Feraud et al. (2001, p.49) below shows the experimental results carried out by these researchers.

Results on Sussex Face Database

orientation (degree)	CGM1	CGM3	CGM5	Ensemble	Conditional ensemble	Conditional mixture
0	100.0 %	100.0 %	100.0 %	100.0 %	100.0 %	100.0 %
10	62.5 %	100.0 %	87.5 %	100.0 %	100.0 %	100.0 %
20	50.0 %	100.0 %	87.5 %	87.5 %	100.0 %	100.0 %
30	12.5 %	100.0 %	62.5 %	62.5 %	100.0 %	100.0 %
40	0.0 %	100.0 %	50.0 %	12.5 %	62.5.0 %	87.5 %
50	0.0 %	75.0 %	0.0 %	0.0 %	37.5 %	62.5 %
60	0.0 %	37.5 %	0.0 %	0.0 %	0.0 %	37.5 %
70	0.0 %	37.5 %	0.0 %	0.0 %	0.0 %	25.0 %

Table 2.6: Showing Results of Sussex Face Database. Feraud et al. (2001, p.49)

Results on the CMU Test Set A

Model	Detection rate	False alarms rate	False alarms
GM	84 %	$1000\ 10^{-6}$	≈ 20000
CGM1	77 %	$5.43\ 10^{-6}$	47
CGM3	85 %	$6.3\ 10^{-6}$	212
CGM5	85 %	$1.36\ 10^{-6}$	46
one SWN (Rowley95)	84 %	$8.13\ 10^{-6}$	179
Ensemble	74 %	$0.71\ 10^{-6}$	24
Conditional ensemble	82 %	$0.77\ 10^{-6}$	26
Conditional mixture	87 %	$1.15\ 10^{-6}$	39

Table 2.7: Showing Results of CMU Test Set A. Feraud et al. (2001, p.49)

Wang et al. (2016) in their study to support neural network face detector used a multi-task convolutional neural network-based face detector, which relies directly on learning features from images instead of hand crafted features. Hence their ability to differentiate faces from uncontrolled backgrounds or environments. The system the experimented on used the Region Proposed Network which generates the candidate proposal and the CNN-Based detector for the final detection output. They experimented this based on 183200 images from their database and used the AFLW dataset for validation. Their face detector system was evaluated on AFW, FDDB and Pascal faces datasets respectively and achieved a 98.1% face detection rate. The authors did not reveal all the facts leading to the development of the

system and I have limited time to implement this on OpenCV. 2.8 (Wang et al. 2016 p.479), shows the different comparisons of their system against other state of the arts. (Wang et al. 2016 p.480), discuss their system (FaceHunter) perform better than all other structured models. However, this cannot be independently verified as this system was commercialised. One cannot conclude if this was for marketing purpose or a complete solution to the problem as I have limited time to implement it.

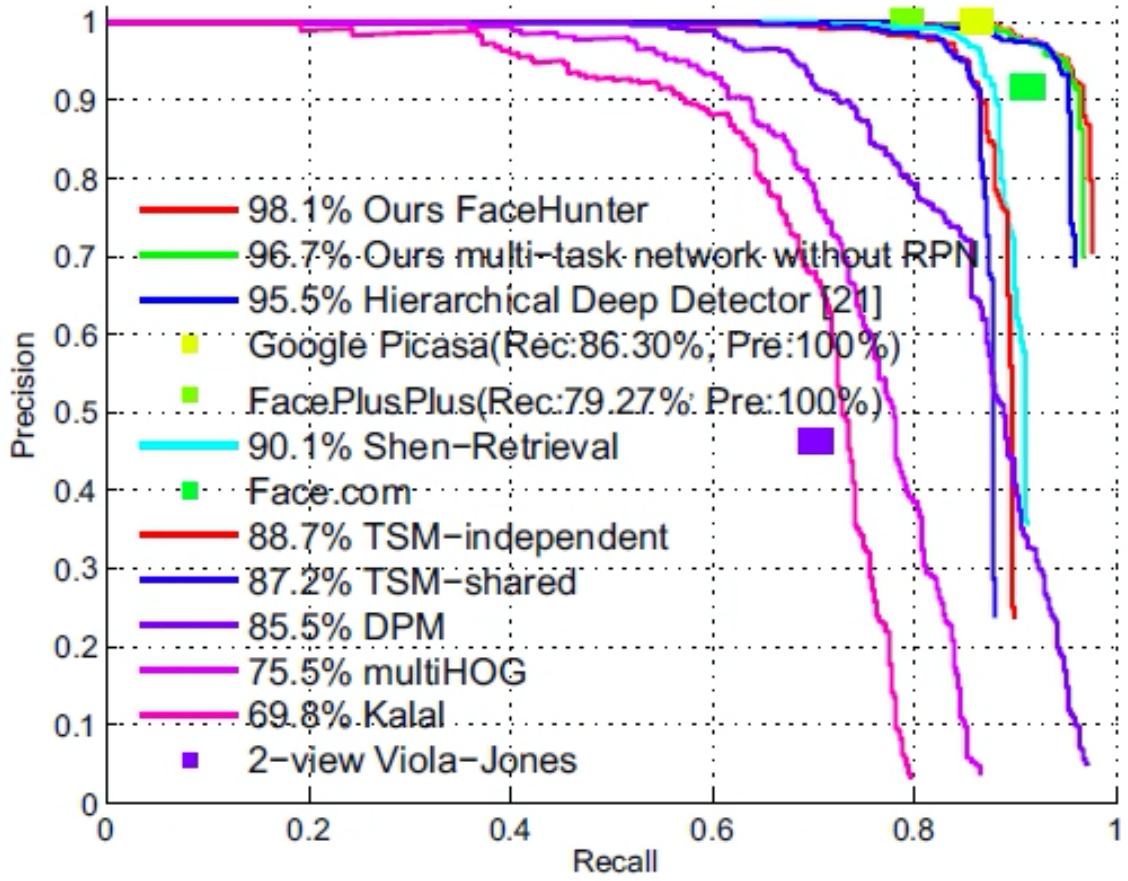


Table 2.8: Showing PR curve on AFW. (Wang et al. 2016 p.479).

The other major category is **Feature-based approaches;** depends on extracted features which are not affected by variations in lighting conditions and pose. This according to these researchers, Hjelmås and Low, (2001 p.241),further clarifies that "visual features are organised into a more global concept of face and facial features using information of face geometry ". This technique in my own opinion will be slightly difficult to use for images containing facial features from uncontrolled background. This technique relies on feature analysis and feature derivation to gain the required knowledge about the face to be detected. The features extracted are the skin colour, face shape, eyes, nose and mouth. On the other hand, in another study by Mohamed et al (2007 p.2) suggest that the "human skin colour is an effective feature used to detect faces, although different people have different skin colour, several studies have shown that the basic difference based on the intensity rather than their chrominance". The texture of the human skin can therefore be separated from different objects. Feature methods for face detection uses features for face detection. Some users depend on the edges and then grouping the edges for face detection. Furthermore, Sufyanu et al (2016) suggest a good extraction process will involve feature points chosen in terms of their reliability of automatic extraction and importance for face representation. Most geometric feature-based approaches use the active appearance model(AAM) as suggested by (S., N.S. and M., M. 2010). This allows localization of facial landmarks in different ways to extract the shape of facial features and movement of these features as expression evolves
Hjelmås and Low, (2001 p.241),further placed the feature-based approach into sub categories of; Low level analysis (Edges, Gray-levels, Colour, motion and generalized measure).
Feature analysis (Feature searching and constellation analysis).

Active shape models (Snakes, Deformable templates and Point distribution models(PDMS).
Figure 2.5 shows the different approaches for Face detection as reported in a study by Hjelmås and Low, (2001), which can be compared with Figure 2.6 showing the exact same classification by Modi and Macwan (2014, p.11108).

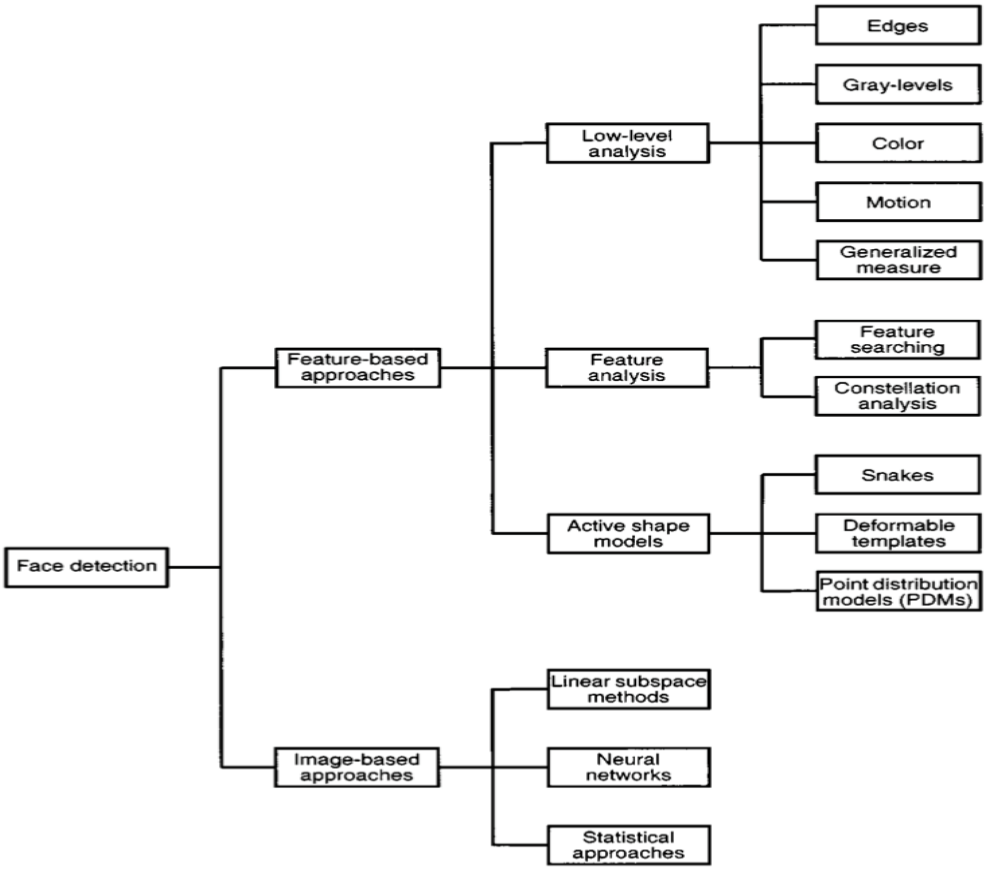


Figure2.5 Face Detection, classified into different methodologies. Hjelmås and Low, (2001, p.238)

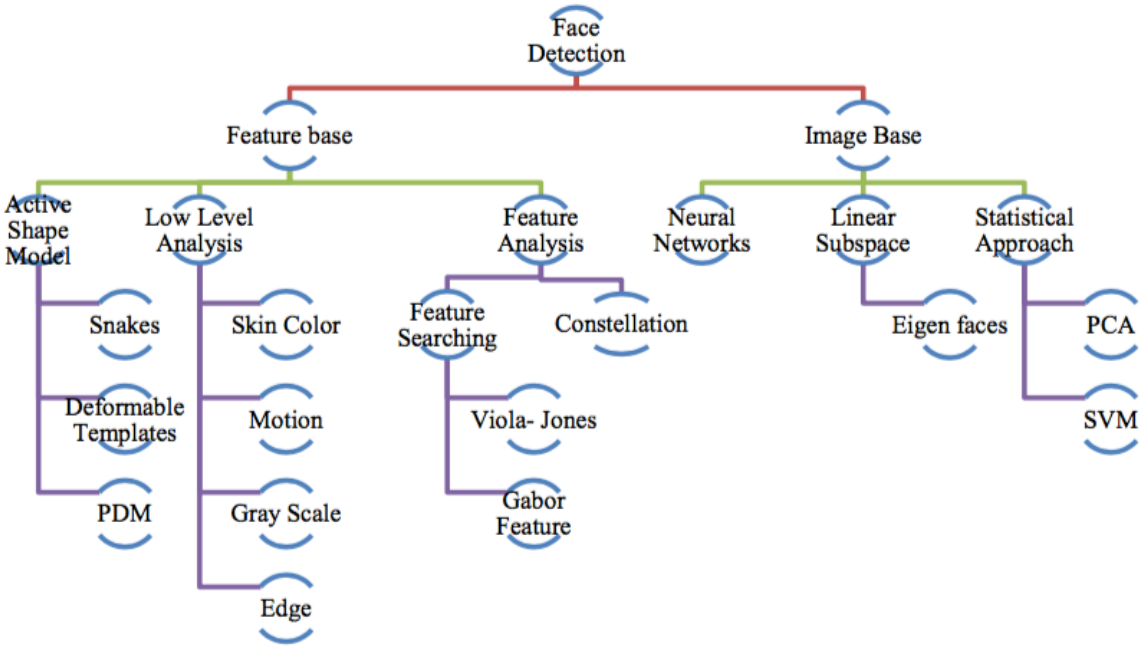


Figure 2.6 Various Face Detection Methodologies. Modi and Macwan (2014, p.11108).

Hjelmås and Low, (2001, p.240) in their study, show experiment based on edge-detection based approach for face detection, on a set of 60 images of 9 faces, with complex backgrounds and correctly detected 76% of faces with an average of two false alarms per image. Nehru and Padmavathi, (2017, p.1), in their study, experimented face detection based on the Viola-Jones algorithm in a dataset of dark and colored men to support their statement which states "It is possible to detect various parts of the human body based on the facial features present", like the eyes, nose and mouth. In this case, systems as such will have to be trained properly to be able to distinguish features like the eyes, nose, mouth etc., when a live dataset is used. The Viola-Jones algorithm to detect faces as seen in the images in Figure 2.7 which shows dark and colored skin faces detected accurately.

Figure2.7 Face Detection in Dark and Colored Men by. Nehru and Padmavathi, (2017, p.1).

Also, in support of the claim made by Nehru and Padmavathi, (2017), the research carried out by Viola-Jones to come up with the Viola-Jones algorithm in face detection, has had the most impact in the past decade. As suggested by Mayank Chauhan et al (2014 p.1615), the Viola-Jones in face detection is widely used in genuine applications such as digital cameras and digital photo managing software. This claim is made based on a study by Viola and Jones (2001). Table 2.9 gives a summary of the results obtained by these experts, showing various numbers of false and positive detections based on the MIT and CMU database set with 130 images and 507 faces.

False detections / Detector	10	31	50	65	78	95	110	167	422
Viola-Jones	78.3%	85.2%	88.8%	89.8%	90.1%	90.8%	91.1%	91.8%	93.7%
Rowley-Baluja-Kanade	83.2%	86.0%	-	-	-	89.2%	-	90.1%	89.9%
Schneiderman-Kanade	-	-	-	94.4%	-	-	-	-	-
Roth-Yang-Ahuja	-	-	-	-	(94.8%)	-	-	-	-

Table 2.9: Various Detection rates by different algorithms showing positive and false detection rates. Viola and Jones (2001, pI-517).

Wang et al, (2015, p.318) states that" the process of searching a face is called face detection. Face detection is to search for faces with different expressions, sizes and angles in images in possession of complicated light and background and feeds back parameters of face". In their study, they tested face detection based on two modules which shows one module uses a combination of two algorithms (PCA with SVM) and the other module based on a real-time field-programmable gate array (FPGA). With these they concluded with their results of face detection accuracy of 89%. Table 2.10 is a screen short taken from this paper to show experimental results of two units combined in order to investigate the accuracy of the system.

The number of testing	Correct detection	False detection	Accuracy
1000	890	110	89 %

Table 2.10. Detection accuracy system by Wang et al, (2015, p.331).

Another method is the **Learning based methods,** that includes machine learning techniques that extract discriminative features from a trained dataset before detection. Some well-known classifiers used for

face detection, based on a study by Thai et al. (2011) are Canny, Principal Component Analysis(PCA), Support Vector Machine(SVM), and Artificial Neural Network (ANN). Although used for facial expression classification, the algorithms are however, also used in the initial stage of their experiment, which is the detection phase. Their experiment as achieved some results which are shown in Table 2.11. A screenshot from Thai et al. (2011, p.392).

Method	Classification Accuracy %
Rapid Facial Expression Classification Using Artificial Neural Networks [10]	73.3%
Facial Expression Classification Using Multi Artificial Neural Network [11]	83.0%
Proposal System (Canny_PCA_ANN)	85.7%

Table 2.11. Comparing Different Algorithms on classification rates. Thai et al. (2011, p.392).

The overall objective of the Face detection part of this project will be to find out if any faces exist in the input image and if present will return the location in bounding boxes and extent of each face, counting the number of faces detected. It is a challenge to this project that due to the variations in location, scale, pose orientation, facial expression, illumination or lighting condition and various appearance features such as facial hair, makeup etc. It will be difficult to achieve an excellent result. However, the performance of the system will be evaluated, taking into consideration the learning time, execution time and number of samples required for training and the ratio between the detection rate and false detections. Table 2.12 below shows experiments from different researchers. They have used different sizes of image dataset. Some have used a combination of different algorithms and applied other methods like colour filtering etc. and different training sets to obtain their results. However, we can conclude the Viola-Jones algorithm which is on its own classifies images based on local features only and can still detect at very high accuracy and rapidly than pixel-based systems. Viola-Jones (2001, p.139).

	Method	Detection Accuracy	#False Detection
Yang et al (2002 pp.36-37)	Knowledge-Based-Method	83.33%	28
Ryu et al. (2006)	Image-Based Method	89.1%	32
Feraud et al. (2001)	neural network-based	86.0%	8
Rowley et al. (1998)	Neural Network-Based	86.2%	23
Wang et al. (2016)	CNN-Based	98.1%	
Hjelmås and Low, (2001, p.240)	Edge Detection-Based	76%	30
Viola and Jones (2001).	Viola-Jones	88.84%	103
Wang et al, (2015, p.318)	PCA with SVM)	89%	110
Thai et al. (2011)	Canny_PCA_ANN	85.7%	N/A

Table 2.12. Comparison of Results by Different Researchers Showing Face Detection Accuracy and False Detection.

Face Recognition:

Face recognition can be defined as the method of identifying an individual based on biometrics by way of comparing a digital captured image or video with the stored record of the person in question.
In the early 90s numerous algorithms were developed for face recognition and increase in the need for face detection. Systems were designed to deal with video streaming. The past few years has proven to have developed more research and systems to deal with such challenges. Dodd, (2017 p.1) reported that in the recent Notting Hill carnival, some arrest resulted due to trial of facial recognition systems. Hence the reason why there is still on-going research on this system. In contrast, the 2011 London riots had just one arrest contributed by facial recognition software out of the 4962 that took place. With the most recent technology of facial recognition and detection techniques, commercial products have emerged on the markets. Despite the commercial success a few issues are still to be explored.
Jafri and Arabnia (2009) in their study discuss Face Recognition in two primary tasks. **Verification**; a one-to-one matching of an unknown face alongside a claim of identity, to ascertain the face of the individual claiming to be the one on the image. **Identification** which is also a one-to-one matching, given an input image of a face for an individual (unknown), to determine their identity by comparing the image against a database of images with known individuals. However, Face Recognition can also be used in numerous applications such as Security, Surveillance, General Identity Verification (electoral registration, national ID cards, passports, driving licenses, student IDs), Criminal Justice systems, Image Database Investigations, Smart Card, Multi-media Environments, Video Indexing and Witness face reconstruction. Face Recognition in most common form is its frontal view which is not unique or rigid as numerous factors cause its appearance to vary. Variations in facial appearance has been categorized in two groups of intrinsic factors (physical nature of the face which is independently of the observer) and extrinsic factors (illumination, pose, scale and imaging parameters such as resolution, noise, focus, imaging) as discussed by Gong et al. (200) and supported by Jafri and Arabnia (2009 p.42).
Lenc and Král (2014 pp.759-769) classify face recognition into various approaches; **Correlation Method**, compares two images by computing the correlation between them, with the images handled as one-dimensional vectors of intensity values. The images are normalized to have zero mean and unit variance with the nearest neighbour classifier used in the image directly. With these considerations stated, the light source intensity and characteristics of the camera are suppressed. The limitations of this method are; Large amount of memory storage needed, the corresponding points in the image space may not be tightly clustered and it is computationally expensive.
Eigenfaces; This method considers the whole image as a vector. With this method, performance depends on alignment of the images with approximately the same pose. The change in lighting conditions, scale, pose and other dissimilarities decreases the recognition rate rapidly.
View-Based Eigenfaces; on like the previous method, evaluates images on a large database and addresses the problem of viewing orientation.
Independent Component Analysis; this separates signal into sub-components with the main aim looking for a linear combination of non-Gaussian data signals that reconstructs the original signal. With this method, images are treated as random variables with pixels as observations and vice-versa.
Fisherfaces; which are derived from Fisher's Linear Discriminant (FLD) and projected into another less dimensional space. The dimensionality, given by the image resolution is reduced to a number of distinct classes. Yi et al. (2000) investigated this method and compared it with SKKUfaces (Sungkyunkwan University faces) an algorithm developed by them which adopts Principal Component Analysis and FLD in series similar to Fisherfaces. They discuss Fisherfaces method as being able to perform dimensionality reduction using a projection matrix and still preserve class separability. With this, FLD is applied to reduce PCA subspace thus achieving more reliability for classification purposes. Their performance comparison of the Fisherfaces method and SKKUfaces for SKKU facial images showed a recognition rate of approximately 88% for Fisherfaces method and 92% for SKKUfaces with changes in illumination and other factors considered. They also compared the Fisherfaces methods and SKKUfaces with Yale facial images and achieved an approximate recognition rate of 92% for

Fisherfaces and 96% recognition rate for SKKUfaces. On the other hand, Jeong and Choi (2013) show the performance of Fisherfaces on recognition for various number of features with a recognition rate of approximately 91% for the Yale database as shown in Figure 2.8. Furthermore, they expanded their research and compared the Fisherfaces method with Feature Feedback and obtained a recognition rate of approximately 95% with Feature Feedback performing at a recognition rate of approximately 96% on the Yale database.

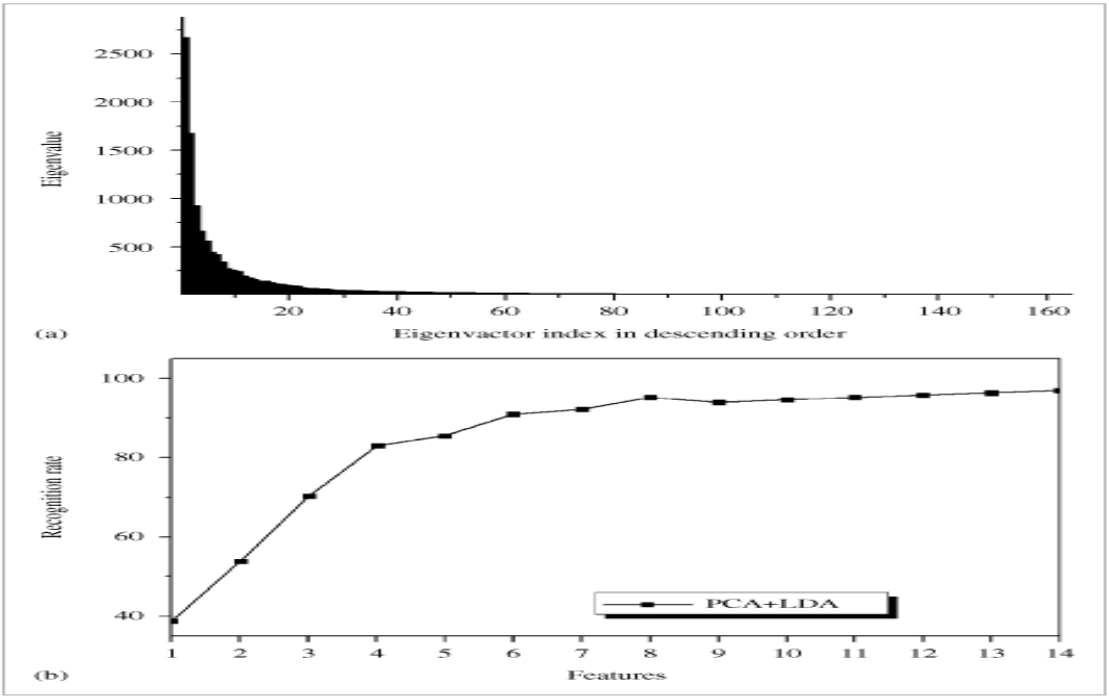


Figure 2.8 Showing Number of Fishers for Recognition. (a) Descending order of EigenValues. (b) Recognition rate as a function of the number of features. Jeong and Choi (2013, p.542).

Kernel Methods; are KPCA and KFLD addressing the issue that original methods are based on second order statistics as discussed by Lenc and Král (2014, p.761). This method makes considerations of multiple pixels' dependencies, allowing more information to be captured for the face representation. Another method based on the kernel method uses Kernel Fisher discriminant analysis, with features extracted by Discrete Cosine Transform (DCT) and Radon Transform. The coefficients for this are used for feature vector. Furthermore, the Kernel Fisher Discriminant (KFD) is applied to increase discrimination capabilities. This method was experimented by Jadhav and Holambe (2010 p.1007) alongside other algorithms and showed an average recognition rate of approximately 90.09% for KPCA on two image sets and 92.01% based on three image sets for KFD on FERET database. Upon evaluation using the ORL database KPCA showed an average recognition rate of 90.65% and KFD 91.54% for three images per training set and on the Yale database showed an average recognition of 90.22% for KPCA and 94.76% for KFD showing an overall average recognition rate of 90.32% for KPCA and 92.77% for KFD.

Adaptive Local Hyperplane; one of the methods suggested by Lenc and Kral (2014) says it's an extension of the K-local Hyperplane Distance Nearest Neghbour HKNN). This method approximates the possibility of missing instances in the manifolds of particular classes by a local hyperplane. With this method, classification of an unknown vector starts with identifying the K-nearest neighbor and based this, the local hyperplane is constructed.

Genetic Algorithms; this approach shows how a facial image is processed in lower dimensional PCA sub-space. It looks for optimal rotation of a basis vector based on a fitness function, as the rotations are random.

Trace Transformation; is invariant to image transformation and it is first transformed into a trace space, thus creating a face representation. The face representation is created.

Linear Regression; assumes that faces from one class are placed on a linear subspace and multiple

training in images for each class (individual).
Active Appearance Models; this method uses a statistical model for grey level appearance and object shape. As stated by Lenc and Kral (2014 p.762), "a set of training examples is used to learn the valid shapes. The shapes must be labelled". This is a clear indication that the landmarks are manually marked, so that the algorithm can try to match the model to an image. by so doing, the distance between the synthesized model and the image is minimized and performed iteratively.
Neural Networks; performs based on neural networks with the images sampled into a set of vectors. The vectors created from the labelled images are used as a training set for Self-Organized Map. In other study carried out by Dhanaseely et al. (2012), discuss the neural Network Classifiers as an Artificial Neural Network (ANN) that comprises of artificial neurons that uses a computational model to process information. They further conducted an experiment based on their proposed system, to measure the performance recognition rate of two of the neural networks, the Feed Forward Neural Network and Cascade Neural Network. A diagram of their proposed system is seen below in Figure 2.9.

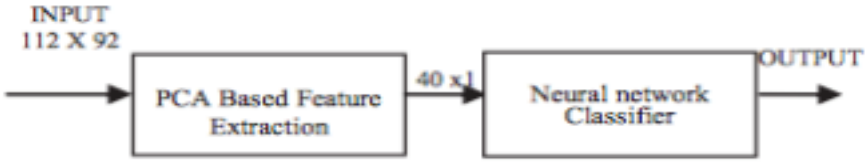


Figure 2.9 Proposed Face Recognition System by Dahanaseely et al. (2012).
Their experiment was based on the following parameters shown on the Table below

Training Parameters

Network Parameter	CASNN	FFNN
Number of layers	Input layer-1 Hidden layer-30 Output layer-1	Inputlayer-1 Hidden layer-2 Output layer-1
Number of neurons at input layer	40	40
Number of neurons at hidden layer	single	15 neurons in each layer
Number of neurons at output layer	40	40
Transfer function	Tansig	Tansig
Training function	Trainrp	Trainrp
Performance function	MSE	MSE
Number of Parameters	4505	1495
Number of Addition	4435	1425
Number of sigmoid	30	30
Number of multiplication	4435	1425

Table 2.13 Parameters for Cascade Neural Network and Feed Forward Neural Network Dahanaseely et al. (2012).
The overall recognition rate based on the parameters shown on the table 2.13 show a 97.5% recognition rate for CASNN and 92.5% for FFNN. This experiment is based on the ORL database.
Hidden Markov Models; associated with the states of the HMM are the subdivided regions of the face (eyes, nose, mouth etc.). the images in this method are sampled with a rectangular window of the same width as the image and shifted downward with a specific block overlap. This is done thanks to the representation of boundaries between regions which are represented by probabilistic transition between the states of the HMM.

Miar-Naimi and Davari (2008) in their study carried out their investigation of the algorithm based on 7-state HMM. Their experiment on this algorithm, was carried out alongside Singular Value Decomposition (SVD) coefficients as extracted features. With the quantized SVD coefficient of each block extracted from the image, each face is a numerical sequence that is modelled by HMM. With an order static filter used for processing of the images, their experiment was carried out on the ORL database of 400 images and 99% recognition rate was achieved. On the YALE database, which has 165 images of 15 subjects, they obtained a 97.78% recognition rate. However, the high recognition rate is achieved based on only three features with the best classification rate per block, alongside image resizing. With different quantization levels and symbols of 24,182 and 630, the recognition rate dropped to 80%, 94.5% and 97% respectively. However, the high recognition obtained by the experiment on this study, is only achieved based on the number of trained images greater than five and forward facing facial images. The recognition rate based on number of trained images less than five shows a recognition rate of 78% on the YALE database.

Lenc and Kral (2014 p.763), in their study investigated this algorithm on a dataset containing 5 images of 24 individuals. The recognition rate using this approach was 84%. In order to confirm this algorithm, they compared the Eigenfaces with the same dataset and obtained a recognition rate of 74%. In another study by Cheng et al. (2014), describe HMM as "a doubly embedded stochastic process with a Markov chain describing the transition of the states while another stochastic process used to describe the statistical corresponding relation between states and observations". With these states hidden, it can only be observable through observations produced by each state. HMM works better by specifying parameters. Cheng et al. (2014) further tested this on the ORL database and obtained an 86.50% recognition rate. This rate is dependent on the parameters (sampling height, overlap, DCT coefficients, training set, training times, and recognition time). With these parameters, the recognition rate can be partially increased by optimizing the parameters of the sampling height, overlap and DCT coefficients. Wang and Abdulghafour. (2015, p.294), in their study also carried out an investigation on two databases by combining DCT-HMM for face recognition. With a parameter of the Overlap value of 9, they obtained a recognition rate of 83.5% for the ORL database and 82.23% for the Yale dataset.

Support Vector Machine; this method is relying on two approaches. Component-based, and global Methods to create vectors that represent the face. In the first approach, the pixel values are considered as the input vector in a Support Vector Machine classifier. In this approach, the separate representations are used for important parts of the face and feed into a classifier for individual classification, and it's less sensitive to image variation. Also, it is used for feature extraction, derived from Linear Discriminant Analysis (LDA). Da Costa et al. (2015) investigated this method by using feature extraction methods and further extract the coefficients generated by different transforms (Shearlet transform, wavelet, contourlet, and curvelet) and variations of PCA and LDA obtained an approximate recognition accuracy of 85.05%.

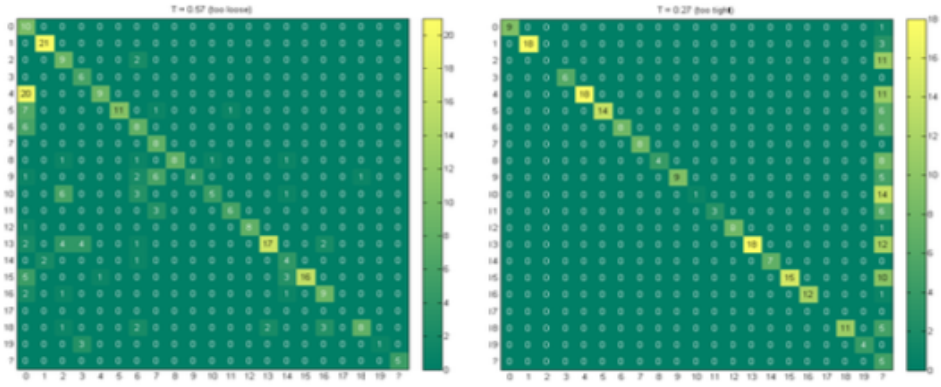
Cost-Sensitive Face Recognition; most researchers always consider the recognition rate but never take into consideration different types of misclassifications which may have an impact on the performance of the system. The loss value depends on the classification error. Exmples of cost- sensitive classifications are mckNN and mcKLR.

Elastic Bunch Graph Matching and Related Approaches; uses features obtained from Gabor wavelet. The first phase of the process is to manually label the landmarks presented to the algorithm. The landmarks are then used to compare the landmark position in an imaginary image. The landmark positions are computed by Gabor wavelets convolutions(Jets) and used for face representation. With a "bunch of graph" created to relate this, each node in the graph contains a set of Jets for each landmark on all the images. Face similarity is obtained by getting the positions of the landmark and jet values.

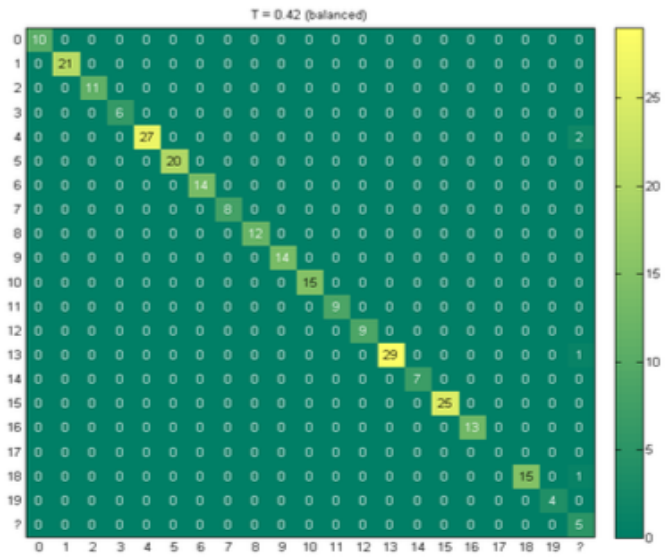
Kepenekci Method; in this algorithm, the landmarks are labelled by Gabor filter and obtained dynamically as compared to the previous algorithm which requires manual labelling of the facial landmarks. It also uses the sliding window to scan the images and identify the maxima of Gabor filter responses within the window. These points are known as fiducial points. The fiducial points are not constant and used to calculate the feature vectors. The cosine similarity is used to calculate the similarity of these vectors. The higher the window size the less fiducial points detected. However, a search for larger window leads to more computational time. The number of fiducial points determines the time needed in the comparison stage.

Local Binary Patterns; first used in texture as texture descriptor, the operator uses the value of the

central pixel to threshold a local image region. The pixels are labelled either as 0 or 1 depending on whether the value is lower or greater than the threshold. Linna et al. (2015) in their study, proposed a system (Online Face Recognition System) that is based on LBP and Facial Landmarks, which uses nearest neighbor classifier in LBP histogram matching. They experimented the system on the videos of Honda/UCSD video database. They used both Offline and Online testing for different distance thresholds and achieved recognition rates of 62.2%,64.0% and 98.6% respectively for the Offline test. The recognition rate was calculated based on a confusion matrix that is shown in Figure 2.10 below, obtained as a screenshot from this paper. The online test performed at a recognition rate of 95.9%. The high achieved recognition rates as per their experiment is based on longer search strategy. The detected face tracked, is used to find the nearest neighbor match and the number of frames from the start of the face tracking are used in the database. This shows that the number of frames decreases as the database gets larger and hence increase in search time. This is because more time is needed to find the nearest match for a single frame. However, as more time is needed to find the nearest match, although recognition rate may be high, it is still not robust enough to compete with other methods.



[redacted] Confusion matrices showing offline test results. In y-axis are actual persons and in x-axis recognized persons. Left: threshold 0.57 (too loose). The first trained person is emphasized because it is searched first and the search stops when the distance is below the threshold. Right: threshold 0.27 (too tight). The unknown class is emphasized, because there are less histograms whose distance is below the threshold.



[redacted] Confusion matrix showing offline results. In y-axis are actual persons and in x-axis recognized persons. Threshold 0.42. Based on recognition rate, this was the best choice.

Figure 2.10: Confusion Matrix showing Offline test results by Linna et al. (2015, p.10).

Local Derivative Patterns; this constructs patterns from local derivative variations. It has an advantage over the LBP in that it has a higher order and can represent more information than the LBP. It can be applied on an original gray scale level image and processed Gabor filter images. Dagher et al. (2013) in their research investigated LDP and other algorithms with their propsed algorithm (Voting Technique) on four databases (ORL, UMIST, Yale, and BIOID). On the different databases, they randomly partitioned the databases into 60% training set and 40% test set the results obtained was approximately 73.59% recognition rate for the LDP.

Scale Invariant Feature Transform (SIFT); originally developed for object recognition by Lowe (1999), creates local features that can lead to high recognition rates. These features are invariant to image scaling, translation, rotation and illumination. With this algorithm, features of the reference image are compared using Euclidean distance of their feature vectors. The algorithm works in four stages, namely extrema detection, removal key-points with low contrast, assignment orientation and descriptor calculation. Lenc and Kral (2014, p.765) report a recognition rate of 96.3% and 91.7% on the ORL and Yale databases respectively. In other research by Sunghoon et al (2016), the recognition rate based on ORL database is 90.0% on an image size of 50X57. Their conclusion was that the usage of "SIFT for face recognition has many problems because face has the landmark of doubleness, non-rigid and smooth character compared to general object", as stated by Sunghoon et al (2016, p.10).

Speeded-Up Robust Features(SURF); another useful method of descriptor creation and Key-point detection. In this method, the process of key-point detection is based on Hessian matrix where box filters approximate the second order Gaussian derivatives. It is invariant to face rotation as one orientation is applied to each key-point. Computation is based on circular neighborhood of the key points. Carro et al. (2015) in investigated this method compared with the SIFT method implemented on OpenCV. Their proposed approached followed a step by step flow process as shown in Figure 2.11 below obtained as a screenshot on their study. They measured the performance of SURF against SIFT, based on Correct Recognition Rate (CRR) and Equal Error Recognition Rate (ERR). The SIFT showed 87.34% CRR and 31.7% ERR with the SURF showing a 98.97% CRR and 29.62% ERR. Figure 2.12 shows the comparison between SIFT and SURF with Key-point matching obtained from the results of their investigation. They further concluded that both algorithms can be used for face recognition with SURF as the most suitable.

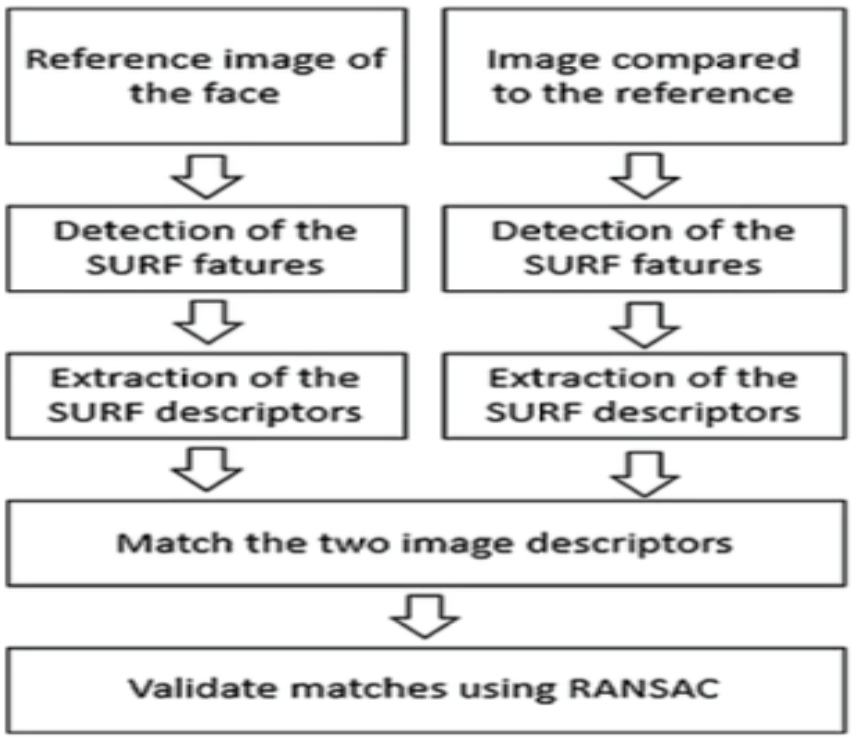


Figure 2.11 Showing Overview of Proposed System by Carro et al. (2015, p.324)

Figure 2.11. Comparison of SIFT (left) and SURF (right) showing Key-point matching Carro et al. (2015, p.325)

3D Face Recognition Methods; performs recognition of faces with any pose by means of linear equations capable of making out face description. It works independent of facial pose and light variations. Unlike other methods it has its limitation which is computational complexity of the face fitting process. Shiwen et al. (2015) in their research on 3D face recognition on the FRGC v2.0 3D Face database evaluated the recognition performance of Single Region by extracting eLBP operator a new local feature proposed in their study known as the Extended Local Binary Pattern (eLBP). They obtained an approximate recognition rate of 97.80%. In another study by Li et al. (2017), on 3D face recognition in cloud environment (cloud computing platform). They used the CASIA 3D face V1 database in order to test their proposed method. The results they obtained showed an approximate result of 89% for their proposed method on a neural expression of the 3D image and 83% recognition rate for LBP based on neural expression. They further computed the False Acceptance Rate and False Reject Rate to obtain Receiver Operating Characteristics curve to measure the performance of these two methods, i.e. the LBP and their proposed method. The Figure 2.12 below is taken as a screenshot from this study to show the FRR against FAR on both methods.

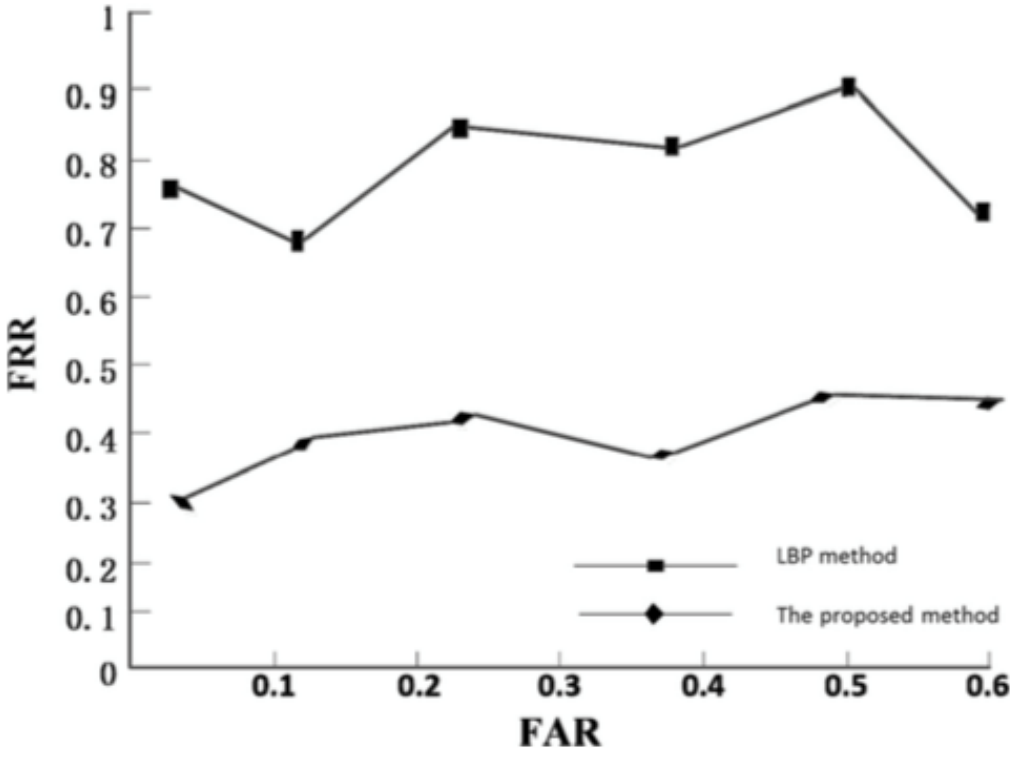


Figure 2.12. Results comparison of ROC curve for two Methods LBP and the Proposed Method by Li et al. (2017, p.17068).

However, the review of the different approaches seen in this section has been evaluated by the experts stated on different databases. There is bound to be disparities in the experimental setup which leaves one to conclude that, there is no worst or best performing approach. But this can lead us into making a choice to target our application which will be discussed in the implementation phase.

Therefore, the main purpose for face recognition related to this paper is to match a given face image of a student captured in a lecture, to a database of known faces, in order to identify the student in the query image. Unlike any other system for face detection and face recognition, the investigation will not be limited to challenges but will test some of the methods in the review to achieve the aim of the project.

CHAPTER THREE

Methodology:

The development of software projects in the past have been carried out based on different methodologies applied by software developers/engineers. As discussed by Veerapaneni and Rao (2011, p.41), each methodology used is informed by the type of project, the organisation and the developers involved in seeing the execution of the project to completion.

The agile project delivery framework is the approach that will be used for the development of the system in this paper. DSDM is an agile methodology approach primarily used as a software development method. The development of this project is requiring the supervisor (user) involvement in order to have timely visible results. Information gathered from the literature review shows researchers using different algorithms in face detection and recognition. However, as it is an ongoing research area, this project requires incremental implementation in smaller functionalities which will be put together at the end for a complete system. With the help of my supervisor, it is important to consider DSDM as the approach to achieve the project objectives. The project objectives specified in the proposal can only be achieved with the expertise of the supervisor, as functionalities will be prioritised in order of importance alongside continuous user involvement. Unlike other approaches (Waterfall) where the stages of implantations are clearly defined, it was preferable to use the approach which will adapt easily to changes made during the implementation. Baseer, K. (2015)

DSDM takes an iterative development approach which is facilitated by the following characteristics.

- The supervisor and the developer (myself) will be actively involved in the development of the system.
- The client will be satisfied with the rapid development of the system as working prototypes will be released.
- The results of the development will be timely and directed by the supervisor.
- More functionality will be delivered at regular intervals with basic functionalities delivered frequently.
- Bureaucracy will be eliminated to break down communication between parties to facilitate incremental development.
- There are early indications of achieving the project objectives rather than surprise at the end of the project as different algorithms will be tried.
- The system is likely to be delivered on time by trying different algorithms stated in the literature review.
- The direction of the Project will be influenced by the client and supervisor.

Below is figure 3.1 showing the DSDM development process (Pierre, H. 2016). The feasibility studies for this project, have been informed by the literature review. Each interactive and iterative stage will be implemented according to changes to functionalities at regular intervals, by putting together different algorithms to achieve project objectives. The risk of building upon the wrong solution will be eliminated as this project will be closely monitored. In this way, the deployment of the project in this case will be a smooth process. DSDM wipes out late delivery as the project will be focused on time frame. Moreover, unused features will be eliminated to be able to meet the project dateline.

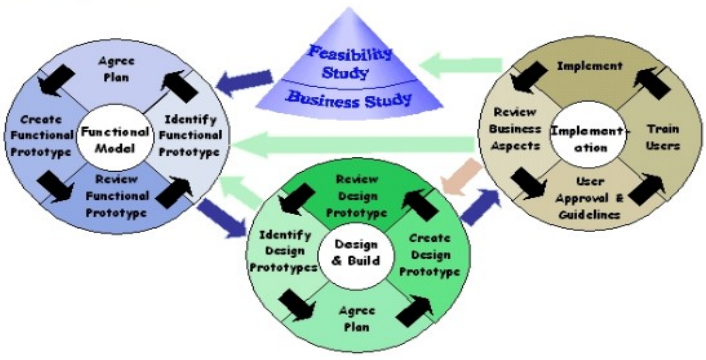


Figure 3.1: DSDM Development Process. (Pierre, H. 2016).

By using DSDM, the techniques to weight the importance of requirements are MoSCoW rules which

will be seen later in this chapter. Also, the principles for evolutionary prototyping for a project using DSDM (frequent delivery and incremental development) supports the development this system. Elbeheri, A. (2016).

Other Methodologies for Project Development:

SCRUM:

Another approach to develop the system is SCRUM which compliments DSDM. This will allow the release of a working prototype every two weeks and get it evaluated by the client before moving to the next phase. In each sprint of this project, algorithms will be evaluated and implemented. During the sprint, if a task to be completed defined, other task with different implementation strategies will not be included (Baseer, K. 2015).

Also, SCRUM requires frequent evaluations by members of the team to see the development of a sprint. However, it is very difficult to use the SCRUM approach for the development of the system in this project as it requires a team of developers and stand-up meetings weekly. With this approach, the allocation of resources and task distribution is done by the development team for successful delivery of functionalities. To support this, Veerapaneni and Rao (2011, p.41), state that "Scrum concentrates on how the team members should function in order to produce the system flexibly in a constantly changing environment". With the nature of this project, which is one developer only and a supervisor, it will be very difficult to follow the scrum approach as it is more team focused and industrial process control for software development.

Extreme Programming(XP):

Other methods that could have been considered is the agile methodology framework Extreme Programming (XP). With this methodology, there is constant release of small prototypes with testing alongside Wood et al (2013). The incremental development of the system in this project will require a little bit of work upfront to understand in wider perspective the system design, before going into the details some aspects to deliver specific features/functionalities. By doing this, design decisions will change based on the most current information and refactoring will be incorporated to remove duplication. As this project is an ongoing research in its field of study and will require small iterations complemented by evaluations and client feedback. Also, the development of this project using this method gives the client a clear idea on user interaction with the system. It is a suitable approach for small teams of developers who are working with projects that have unclear changes in requirements. Furthermore, a negative communication amidst foundations (testing, client acceptance test, test-first design, pair programming and refactoring) is easy to identify and moderated professionally. This will be done through simple design and a pace sustainable to the project incorporated with performance. Wood et al (2013). Also, pair programming which is an aspect of this methodology is going to slow down the development of this project as it is not possible to work with the supervisor daily.

Comparing all the above methodologies, the most suitable framework to the development of this project is DSDM. This project, will be divided into smaller functionalities, as the information gathered from the literature review shows no guarantee an algorithm will work as expected. In this case, there will be amendments during the implementation of each iteration and new solutions can be applied. On the other hand, the SCRUM framework is similar but with the short period for each sprint and pair programming. It is not a good idea to follow as there is one developer.

To sum up, choosing this Agile approach was due to the information gathered from the literature review of the ongoing research on face recognition. Moreover, I had little understanding of Face Detection and Face Recognition and the resources and libraries that could be used to best implement it. However, from more research gathered and having a supervisor who is an expert in the field, I decided to choose this approach for an iterative development throughout the life cycle of the project.

To conclude, with DSDM, there is that flexibility to go back stages and find applicable solution to upgrade functionalities.

Requirements and Analysis:

This stage will discuss the various tools that are used to achieve the project goals and objectives.

Functional Requirements:

Functional requirements are features that the system will need in order to deliver or operate. In the case

of this project, it was important to gather some requirements that will be needed to achieve the objectives set out previously. With client (user) story a use case analysis was implemented which resulted in the following functional and non-functional requirements were captured. The functional requirements have been gathered from the user story developed from the minutes collected during meetings with the client and are outlined here.

- Capture face images via webcam or external USB camera.
- A professional HD Camera
- Faces on an image must be detected.
- The faces must be detected in bounding boxes.
- Compute the total attendance based on detected faces.
- Crop the total number of faces detected.
- Resize the cropped faces to match faces the size required for recognition.
- Store the cropped faces to a folder.
- Load faces on database.
- Train faces for recognition.
- Perform recognition for faces stored on database.
- Compute recognition rate of the system.
- Perform recognition one after the other for each face cropped by Face Detector.
- Display the input image alongside output image side by side on the same plot.
- Display the name of the output image above the image in the plot area.

Non-Functional Requirements:

Non-functional requirements are set of requirements with specific criteria to judge the systems operation. These requirements have been collected based on the following after meetings with the client. They cover ease of use to the client, security, support availability, operational speed, and implementation considerations. More specifically:

- The user will find it very convenient to take photos.
- The user will inform the students when taking a photo with clear instructions on how to position their faces.
- The system is very secure.
- The system will have a response time of 10 seconds.
- The system can be easily installed.
- The system is 100% efficient.
- The system must be fast and reliable.

MoSCoW Analysis:

In order to support the analysis stage, we use MoSCoW a DSDM-Agile Methodology tool to weigh the importance of requirements captured from analyzing the use cases. This will help prioritize the delivery of each requirement under Must Have, Should Have, Could Have, Will Not Have. The “Must Have” must be implemented for the final solution of this system to be acceptable. The “Should Have” are priority features that will be implemented if possible with this project time frame. The “Could Have” is a desirable feature to have if time permits but this system will still function without it. The “Not Have” is a feature of this system that will be implemented in future. Due to the unique login of the university, it is easy to trace who is using this system. However, if this system is to go commercial, this will be a requirement to be implemented.

Must Have: With regards to this project, the “Must Have” are the requirements that have been identified by the client that must be implemented for the final solution. Without these requirements, the final solution will not achieve its aim and objectives.

- The application must detect images by use of bounding boxes.
- Crop the total number of faces detected.
- The application must resize faces to match size of faces stored on the database.
- Compute the total attendance based on the number of faces detected.
- Train Images for recognition.
- Display the input image alongside output image side by side on the same plot.
- Display the name of the output image above the image in the plot area.

Should Have: These are priority features that the system “Should Have” as identified by the client during the meeting. These features will be implemented if possible with this project time frame.

Although these features are priority, the system will still meet its aim and objective.

- Display the name of the input search image and the output image in the command window.
- Determine the percentage Recognition of an image to that found on the database.
- Compute recognition rate of the system.

Could Have: The "Could Have" is a desirable feature for the system of this project but will only be implemented if time permits. This system will still function without it.

- Graphical User Interface (GUI).
- Professional HD Camera.

Will Not Have: The "Not Have" is a feature that was identified during the meeting that will be implemented in future as it is not much of an issue at the moment. Due to the unique login of the university, it is easy to trace who is using this system. However, if this system is to go commercial, this will be a requirement to be implemented.

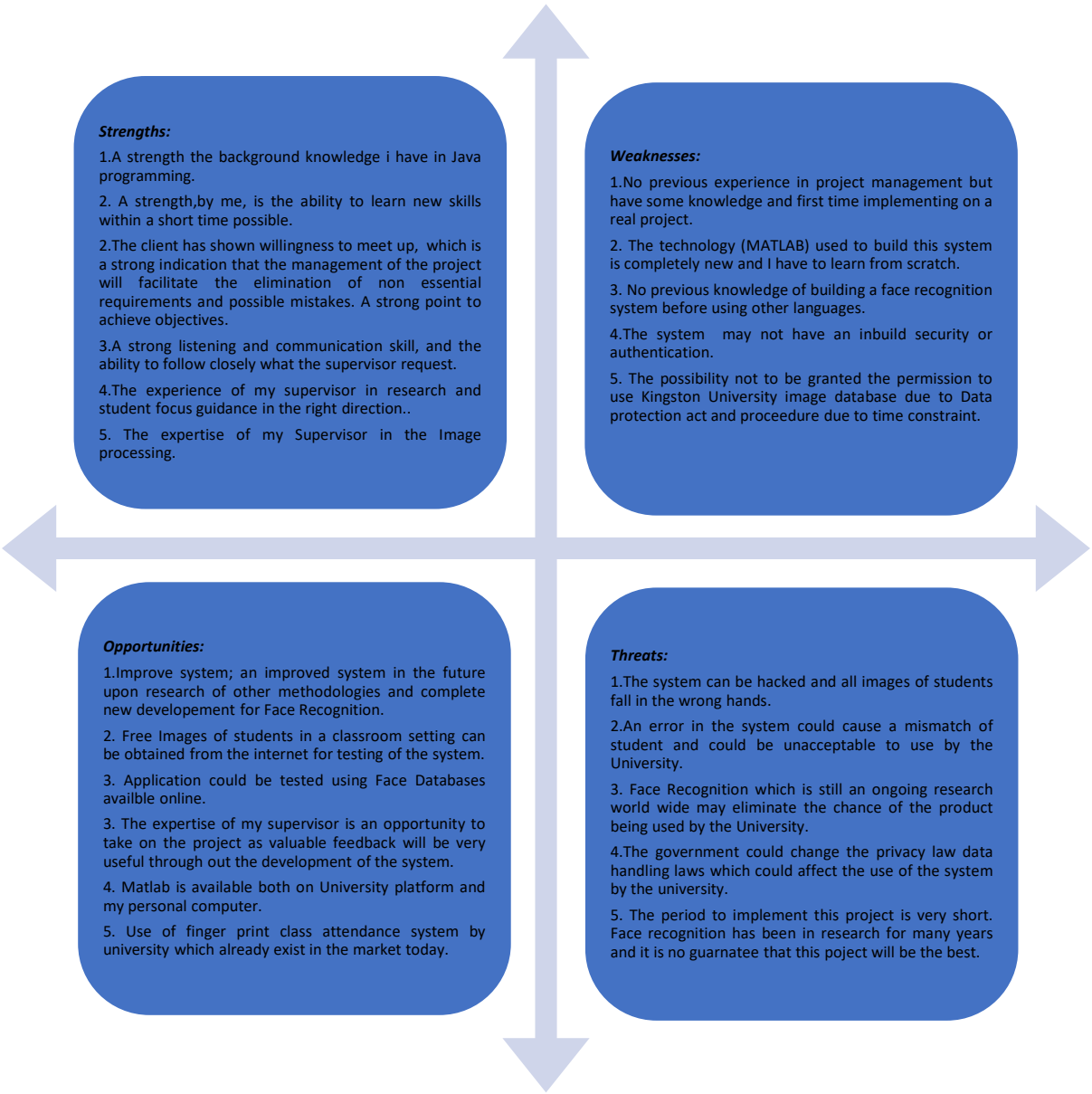
- A login authentication.

SWOT Analysis:

An analysis of the strategic implementation of this project can be done using SWOT. SWOT evaluates the project in terms of Strengths, weaknesses, opportunities and threats. It is simply to identify these points using this type of analysis methodology. By using this method, this section will consider both internal (strengths and weaknesses) and external (opportunities and threats) factors which are either harmful or useful to accomplish the project objectives. This will help estimate the likelihood of the success in accomplishing project objectives. The SWOT acronym represents for Strengths, weakness, opportunities and threats.

Strengths – are factors that influence the success of the project objectives and possessed within.
Weaknesses – are harmful to accomplish project objectives and are internal factors.
Opportunities – are factors that accomplish project objectives from an external point of view.
Threats – are factors harmful to the project aims and objectives and are external.

S.W.O.T.
Matrix



Use Case Diagram:

A use case diagram is a representation of a set of description of actions (use cases) that the system can perform in collaboration with an external factor (user/actor).

The User Story:

As a user, the client wants a system where he/she can load an image that will automatically detect the number of faces on the image. The client also requested that, this system should have the option to capture an image using a mobile phone or an inbuilt webcam on a laptop. As a user, the system should be able to crop the faces on an image after detection and store them on a folder/dataset that will be used for recognition purposes in the second phase of the system. The system should be able to automatically count the number of faces detected on the image.

As a user, the client requests the second phase of the system to be able to match faces stored on a dataset

against input images which are either detected from the first phase or captured by an input device (camera).
The user will start the software used to build this system. On this system, there will be buttons where the user can click to facilitate interaction between certain task as requested. Because the system has two phases, the second phase of the system will involve the training of images on a dataset that are to be used for recognition.
The proposed system behaviour has been captured by the use case diagram in Figure 3.2 below.

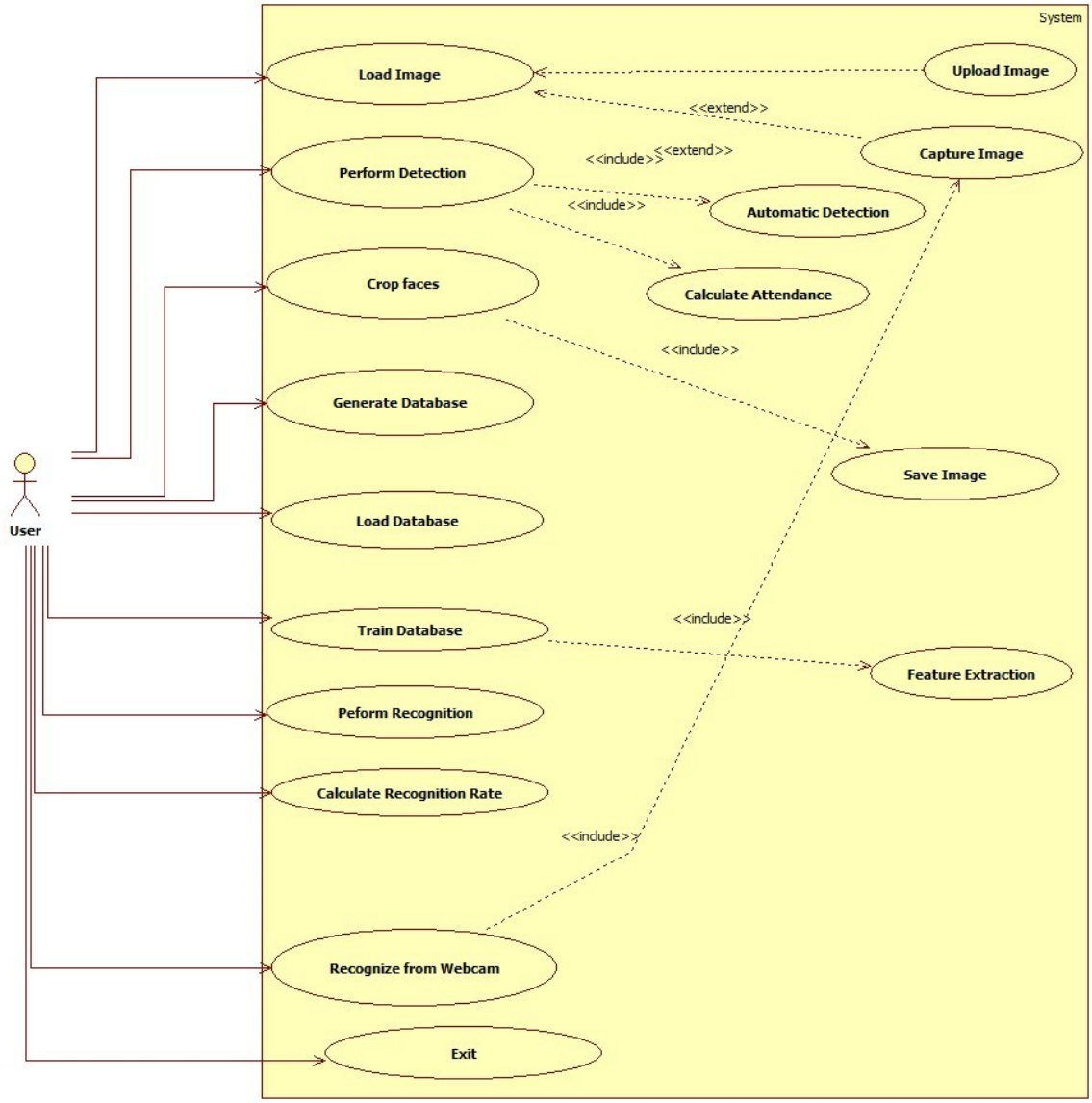


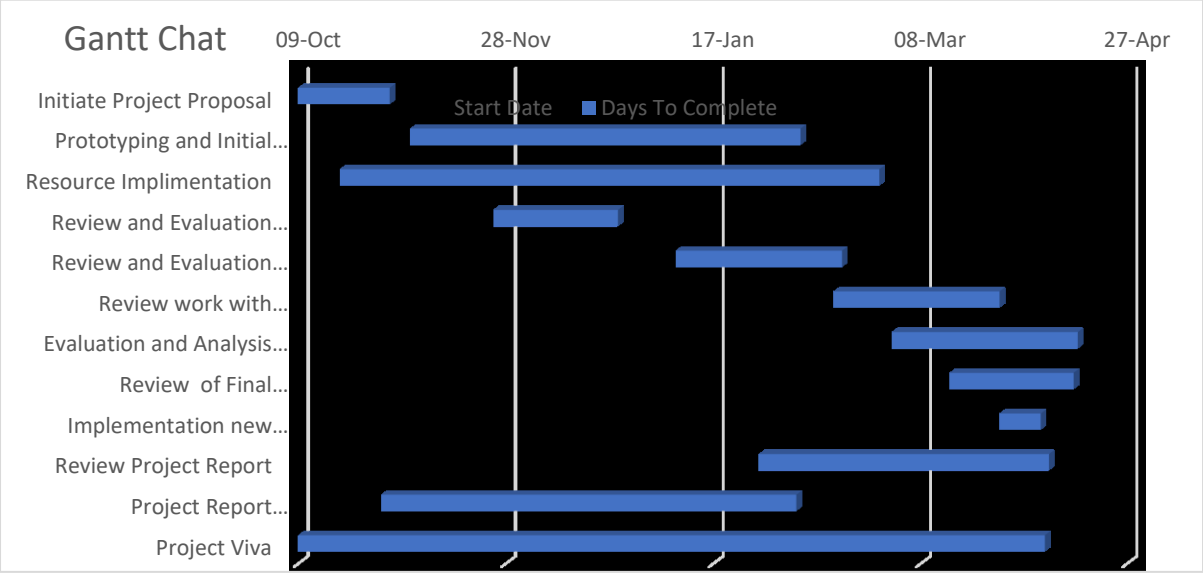
Figure 3.2: Illustration of System Behaviour.

Conclusion:

The requirements analysis of any project has laid the foundation to take the project forward through to the design and implementation phases. Meeting with the client has been very useful in gathering functional and non-functional requirements. Also, information gathered from the literature review have been very useful. MoSCoW, SWOT analysis and Use Cases have been a strong tool to identify how the client wants the system to work. However, because aspects in this project are part of ongoing research, there will be changes during the implementation to achieve more which could lead to more contribution in the future.

Project Plan

This project has been planned and followed using the Gantt chat below. The detail layout of the plan is attached to the Appendix A2 of this report. The detailed layout shows the phases of implementation using the DSDM methodology chosen for this project. The reader can follow these phases on the detailed layout to understand how iteratively this project has been developed.



Project Contingency

The project will be constantly being worked upon as I will be using the DSDM-Agile methodology approach throughout the build. The Gantt chart shows a planned duration based on exaggerated time frame estimation which gives me and my supervisor ample time to meet the dateline. Some of the factors outlined below may hinder the project phases.

RISK FACTORS	EVALUATION	CONTINGENCY
Delay in project implementation	Likelihood-Medium Impact – medium	Online tutorials, attend lectures on Image processing and Vision, Online research on similar projects.
Breakdown in Communication with Supervisor	Likelihood-medium Impact-high	Planned meeting with Supervisor. Communicate via emails and work with calendar.
Work in progress with short deliverables and functionalities may increase time frame as unforeseen functionalities may come into play.	Likelihood- Low Impact-High	More research and online study on areas that may arise with these functionalities. Use more image processing systems already out there to compare functionalities.
Lack of technical expertise	Likelihood-Medium Impact - High	Commit extra hours of work on use of MATLAB and online research.
Technology used may not fulfil project objectives and requirements as stated.	Likelihood-m High Impact-Very High	Carryout research to use best platform or technology to be used.
Ill Health	Likelihood-Medium Impact-Low	Rearrange workload to cope with ill health.
Bugs on coding leading to	Likelihood-Very	Continuous integration, testing and

unsatisfactory results.	Low Impact- Very high	evaluation to ensure the system is satisfactory.

Summary:

Summarily, to understand the goals and objectives of this project, boundaries have been established by using various tools to capture and analyse specific requirements. The user story has enabled us to capture specific requirements, and use non-functional requirement to judge the system, MoSCoW to prioritise functionalities that the system must have. Moreover, SWOT has been used to analyse the internal and external factors which can either be helpful or harmful to the project objectives. Contingency plans to identified potential risk were classified under impact, likelihood and impact for the smooth management of the project.

CHAPTER FOUR

Design:

This chapter represent design concepts that has led to the current implementation of the prototype of this project. The design of this system is going to be carried out using the requirements analysed in the previous chapter in order to produce a description of the systems internal structure that will serve as the basis to implement the system (Bass et al 2013). This will result in the systems architecture, showing how the system will be decomposed and organised into components alongside the interface of those components. The model design of this system, will form a blue print for the implementation that will be put together to achieve the project objectives and best performance for the final product. This system design consists of activities that fit between software requirements analysis and software construction. The algorithms that compute the functionalities of this system have been discussed further in this chapter.

The various outputs for this design are functional specification, detailed design, user interface specification, data model, prototype implementation plan.

Data Flow Model: This is a representation of the system of this project, by way of diagrams to show the exchange of information within the system. It is a diagram that gives an overview of the system described here without going into much detail, which can be later elaborated. (Wikipedia 2018).

Structural Model: A structural model will display of the system of this project in terms of components and their relationships. For example, the system of this project is modelled by architectural design to illustrate the systems responds to events based on the system environment and the interaction between other components both externally and internally. (Sommerville 2011, p.119). The system of this project will also be represented using a behavioural diagram based on the Unified Modelling language (UML).

After a clear understanding of the requirements and assigned components for the system in this project, the method chosen to inform the implementation phase is described below.

During this phase, the client provided feedback to match specific objectives. The prototype has been built iteratively by using the requirements gathered in the requirements and analysis section in the previous chapter.

Architecture Design:

This shows the interaction between software (Internal) components and hardware (External) components with an interface to establish a framework to achieve system objectives. Both external and internal components have been considered. The internal component incorporates all the functionalities with a Graphical User Interface to allow the user to interact with the system.

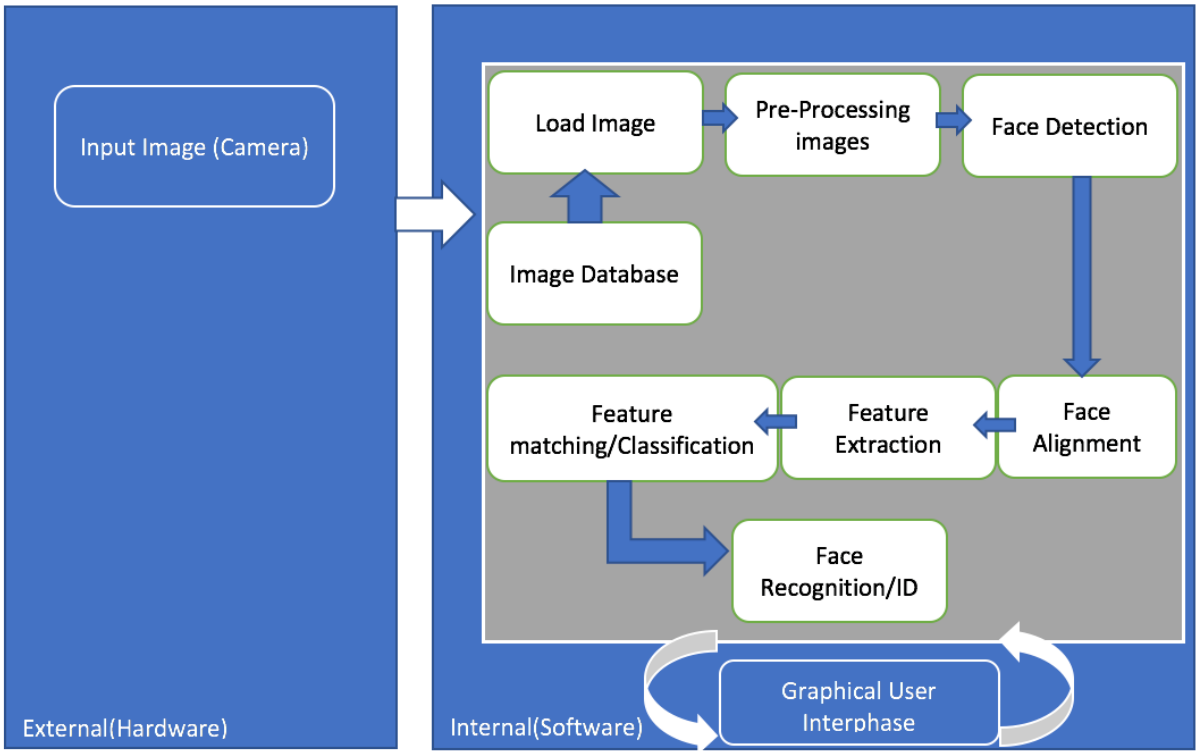


Figure 4.1: External and Internal components of the System.

The Image input and image dataset are all dependencies for the excellent performance of the system. The system will perform best if the images are of good resolution. A good resolution image will enhance face detection to a wider range. Also, a good resolution of the image will have nearly all the pixels required for training if the images which will boost matching of images on the dataset. The accuracy of the system will very much rely on the resolution and quality of the image and how it is trained for recognition.

Interface Design:

The Graphical user interface has been designed to allow the user to interact with the system. This has been implemented using MATLAB's GUI design accessed via GUIDE. A simple menu could still do the job, but using a GUI brings the system together from Face Detection to Recognition.

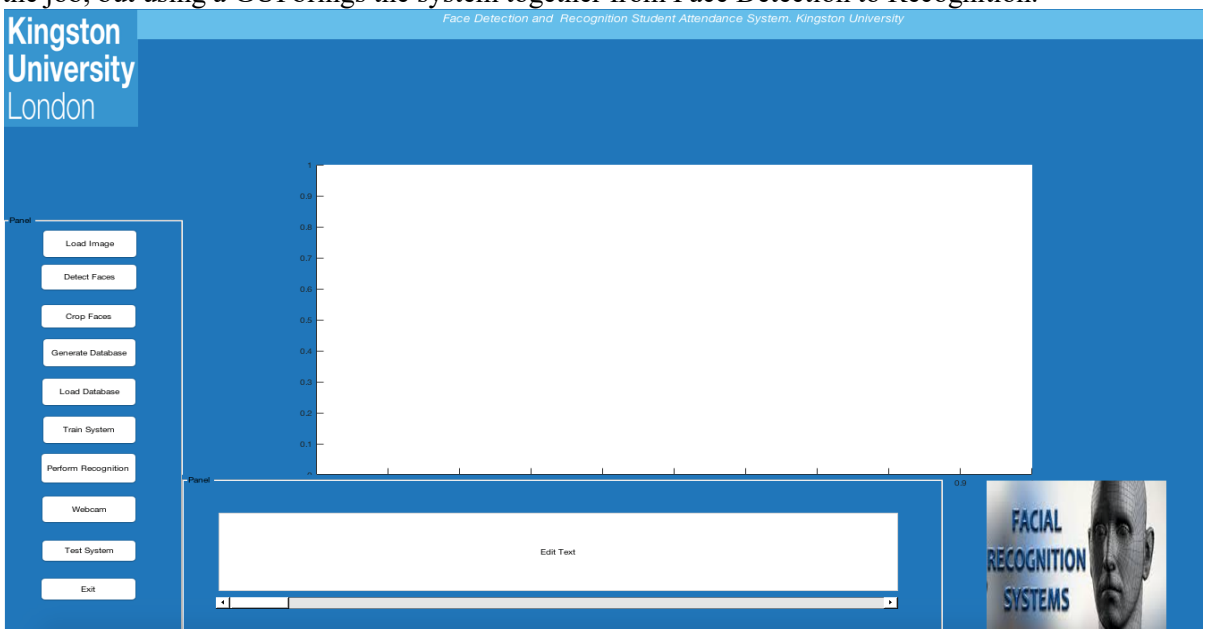


Figure 4.2: GUI designed for User interaction.

The Activity Diagram:

As mentioned above, this will represent the flow of actions that will breakdown most of the interactions into activities in the system.

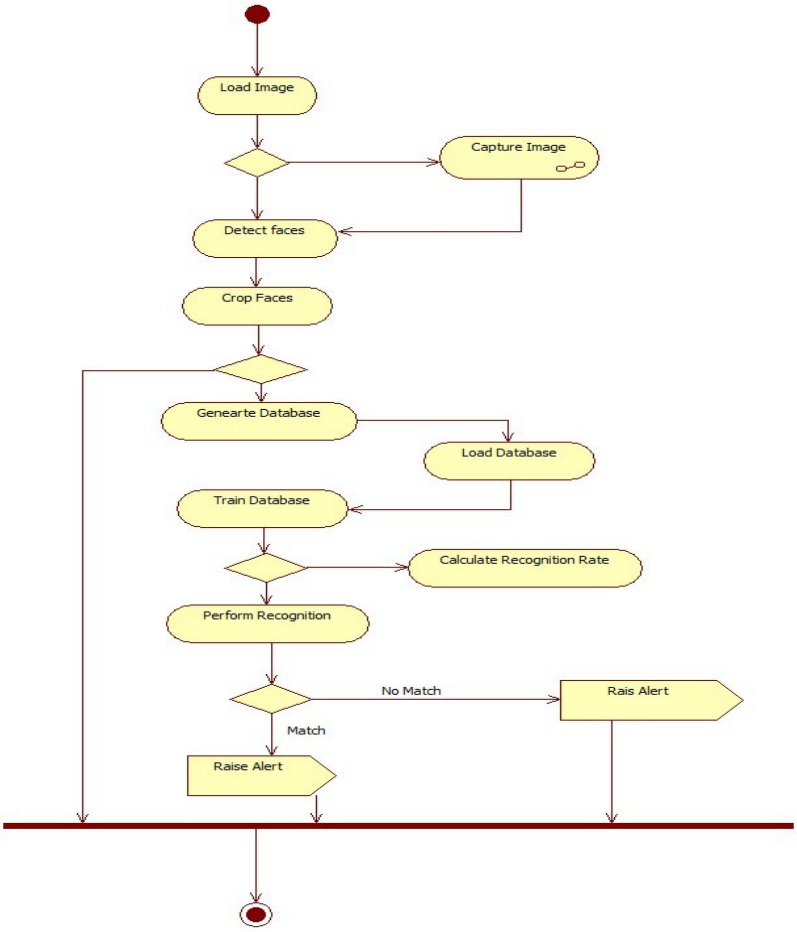


Figure 4.3. Activity Diagram for the system.

The above diagram represents the activity of the functionalities that will be performed by the system. The activities will be actioned by the buttons shown in Figure 4.2 above. The dot without a circle is the start of the system and the dot with the circle is the finish. The diamond triangle represents decisions either carried out by the user or the system. After cropping the detected faces, the user can either decide to exit or carry on to the recognition phase. At each phase, the results will be displayed by way of an alert dialogue box (represented by an arrow rectangle on the diagram) or axis shown on the GUI. At exit the system is shut down and the window is destroyed.

Choice of Methods/Algorithms:

The above figures represent the system's structure, interface and activities that will occur in the system. The figure 4.1 represents the structure show the hardware component (camera) as an input device for images. The software component that performs this same task is a local database (folder) of images. The image is loaded using an internal functionality (Load image) for pre-processing. When the image is processed, the faces in the image are detected and aligned into suitable sizes that is required for feature extraction. The features are then classified and matched to the faces corresponding to that requested by the system or the user. This is then output to the GUI which is part of the internal software components. The figure 4.2 shows the structure of the GUI and the buttons to the various functionalities implemented in the system. The axis will output the image and the second axis to the bottom of the first display axis will output the command window output for the user to view when a task is completed. The figure 4.3 shows how the activities going on in the system. These activities have been described

above. These functionalities will be use the algorithms described below for the design and implemented implementation of the system.

The Viola-Jones Algorithm:

The Viola-Jones algorithm for face detection was proposed by Viola and Jones (2001) as mentioned in the literature review. This algorithm since then has been widely used by researchers for face detection. It has shown the most impact compared to other methods with fast detections due to its broad use in genuine applications Mayank Chauhan et al (2014).

The Viola-Jones algorithm works with a full view of frontal faces (Viola-Jones 2001). The difficulty comes when faces are tilted or on either side but can be adjusted as has been implemented with reference to MATLAB.

The Viola-Jones Algorithm scans a sub-window in order to detect faces across an input image. The standard approach for image processing with this algorithm is to rescales the detector and run it many times through the image rather than rescaling the input image which results to more computational time. Each time the rescaled detector is run against an input image, the size of the image changes each time. The initial step of the Viola-Jones algorithm converts the input image into an integral image. By this, each pixel is equivalent to the entire sum pixels on top and to the left of the pixel in question.

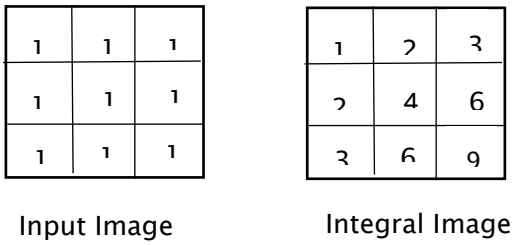


Figure 4.1: Integral Image (Jensen 2008).

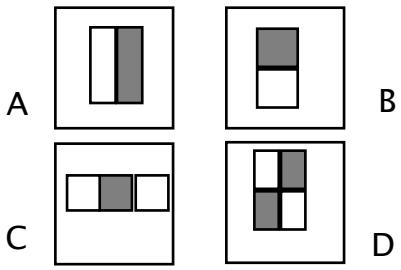


Figure 4.2: Showing different types of Features (Viola-Jones 2001).

The pixels are within rectangular features of random sizes which can be computed in constant time. The given sub-window is analyzed by the Viola-Jones using features of two or more rectangles a number of reference arrays.

The features are as shown in Figure 4.2. where the resulting value are calculated by subtracting the sum of the rectangles from the sum of the gray rectangles. (Viola-Jones 2001).

The recognition part of the system has been implemented using Hidden Markov Model with Singular Value Decomposition (SVD). The use of this has been based on the information gathered during the literature review. The reason why HMM with SVD coefficients was considered is because of the great results obtained by implementing these algorithms together as informed by research. Their results were the motivation to consider as these algorithms for this part of the system. As this area of study is still an ongoing research, this algorithm has been implemented differently and evaluated accordingly.

Hidden Markov Models (HMM):

This is a simple tool that determines a sequence of events (observations), without the state of the model in which the sequence went through to generate the event. (MATLAB R2018a Documentation). In HMM, based on a sequence of observations, the sequence of states can be predicted. Other research defines the HMM as a definite set of state with each state associated with a multidimensional probability distribution. The probability that governs these states are transition probabilities and with the associated probability, an observation can be generated. The training and testing of the images for recognition are performed in the observation vector space (Miar-Naimi and Davari 2008). The image is usually represented in 2D matrix. In the design, the use of one-dimensional model of HMM is used to partition the face.

The face is modelled in one-dimension with seven states. The state in a simpler markov chain is visible to the observer but in HMM, the state is hidden, with the output of the state made visible to the observer. The 7-state of HMM occur from top to bottom of the face and correspond to the Head, forehead, eyebrows, eyes, nose, mouth, and chin. These states will be enhanced with the adjustment of following elements that are needed to complete a HMM (Miar-Naimi and Davari 2008).

- The number of states(N) of the model.
- The number of observations.
- A set of transition probabilities.

Singular Value Decomposition (SVD):

The SVD of a given matrix is the factorization of that matrix into three matrices, where the columns of right and left matrix are orthonormal and the matrix in the middle of the three is a diagonal with real positive entries. This is a tool used for signal processing and analysis for statistical data. (Miar-Naimi and Davari 2008). A data matrix contains singular values with information on noise level, energy and the rank of the matrix. SVD is used as they contain features of patterns embedded in a signal. This is because the singular vectors of the matrix are the span bases of the matrix and orthonormal (Miar-Naimi and Davari 2008). SVD is relevant to face recognition due to its stability on the face image. It is a robust feature extraction technique. The singular value decomposition of a m-by-n matrix is given by $X = U\sum V^T$ where U and V are orthogonal matrix and $\sum$ is a diagonal matrix of singular values of X and V^T is the conjugate transpose of V. The $\sum$ matrix in this case is of grayscale values, with one for each pixel.

Feature Extraction:

The features are extracted using SVD. This is because the coefficients have continuous values and build observation vectors. As these values are continuous, it is clear to encounter an infinite number of possible observations vectors that cannot be modelled by discrete HMM. Quantization is used to model the probability density function by distribution of prototype vectors. It processes by dividing the large set of vectors into groups of approximately the same number of points that may lead to information loss (Miar-Naimi and Davari 2008).

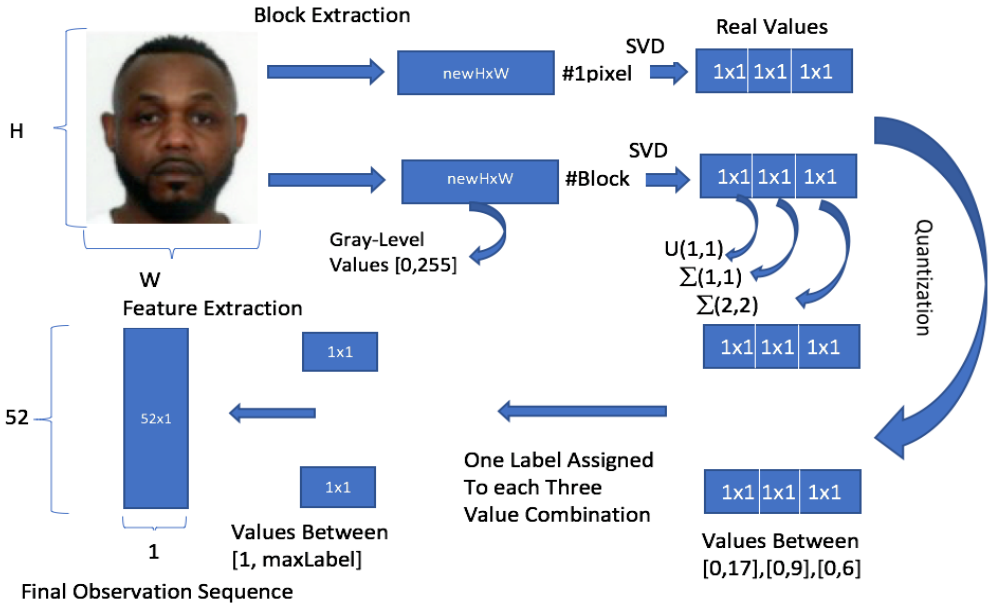


Figure 4.3: Converting Single Face image into observation sequence. (Omid Sahki 2008).

The diagram shows an image of width(W) and height(H). The observation sequence will be divided into overlapping blocks of new height (newH) and width(W). The features will be extracted in seven states (from forehead to the chin). The patch size of (new height by w) slides from top to bottom and the sequence of overlapping blocks is generated. The overlapping size (OL)=new height-1. Showing that each patch is moved only by 1 pixel at a time. Using a for loop, the number of blocks extracted from each face is #Block = (H-newH/newH-OL) +1 =sequence of elements. Each block extracted are converted single values using SVD. Only a 1-by-3 matrix stores the three coefficient values. The values undergo quantization to round to an approximate value. Assume the discrete coefficient values are between 0 and 17, 0 and 9, and 0 and 6. Each block of the face will have three discrete values and only one will be labelled for each block. The possible combinations to consider amounts up to a maximum label(maxLabel) and a minimum label of 1 if the values are zero. From these values, the final observation sequence is generated.

The system is put together using these algorithms and will be informed by the graphical user interface that specifies how the system output each task.

CHAPTER FIVE

Implementation

This chapter will focus on the implementation of the proposed system and how it has been developed iteratively using the DSDM methodology. It has been followed through by structural modelling with an architectural design of the requirements captured during the requirements analysis. Also, a detailed description of the functionalities implemented with stages of how the prototype has evolved to completion will be discussed.

Technology Used:

The key algorithms are Viola-Jones for face detection and Hidden Markov Model with SVD.
The existing implementation of The Viola-Jones algorithm are available for environments MATLAB, OpenCV and Web Browsers (using adobe flash).
The existing implementation of the Hidden Markov Model with SVD for face recognition are available on MATLAB, C++ and OpenCV libraries.
Hence, with advice from my supervisor, I have had to choose MATLAB for the implementation of the Viola-Jones algorithm for face detection due to its full implementation in the Computer Vision System toolbox in MATLAB (R2018a Documentation). Also, from research (literature review) different implementation strategies were considered and the Viola-Jones algorithm had the most impact compared to other algorithms used for face detection. The viola-Jones algorithm according to literature performs with robustness and high accuracy.
Eventually, with MATLAB already chosen as the implementation platform for face detection, it was necessary to consider HMM with SVD for implementation of face recognition. Also, with HMM, it has reduced complexity in training and recognition, a better initial estimates of model parameters are obtained, works well with images with variations in lighting, facial expression and variations (S. Sharavanan et al 2009, p.82). Also, the use of SVD coefficients as features instead of gray values of pixels in the sampling blocks. It also has the ability to integrate with OpenCV (Open Source Computer Vision).

Implementation of Face Detection:

As informed by the design section, the implementation is done using the vision.CascadeObjectDetector in MATLAB 2018a which detects objects using the Viola-Jones algorithm. “The cascade object detector uses the Viola-Jones algorithm to detect people’s faces, noses, eyes, mouth, or upper body” (MATLAB R2018 Documentation). The Viola-Jones algorithm examines an image within a sliding box to match dark or light region to identify a face that contains mouth, eyes and nose. The window size varies with different faces on different scales with the ratio unchanged. The cascade classifier in MATLAB will determine the regions where a face can be detected. According to MATLAB R2018 Documentation, the stages in the cascade classifier are designed to rule out regions that do not have a face in the initial stage (reject negative samples) to save time to analyze regions with possible potential faces in the next stages.

Important Code Details of Face Detector:

Firstly, the input source of the image is implemented. This will be either from the database or directly with a webcam. If loading from the database, the functionality is implemented as show in the figure 5.1 below. Notice that the variable image has been set to global, so it can be called anywhere in the GUI. The “uigetfile” on line 103 is an algorithm in MATLAB that will load images of any file type and the “strcat” on line 108 adds the filename that will read the image using the Matlab 2018a “imread” functionality to extract the file from the path declared on line 103.

```
97    function LoadBtn_Callback(hObject, eventdata, handles)
98    % hObject    handle to LoadBtn (see GCBO)
99    % eventdata  reserved - to be defined in a future version of MATLAB
100   % handles    structure with handles and user data (see GUIDATA)
101 - global image im2 %variable image
102 - if true
103 - [filename,pathname]=uigetfile({'*.jpeg;*.jpg;*.gif;*.tif;*.tiff;*.bmp;*.png',...
104       'all  image file';'*.jpg;*.jpeg','JPEG Files(*.jpg,*.jpeg)';...   % Select the correct color for test
105       '*.gif','GIF Files(*.gif)';'*.tif;*.tiff','TIFF Files(*.tif,*.tiff)';...
106      '*.bmp','Bitmap Files(*.bmp)';'*.png','PNG Files(*.png)';'MultiSelect', 'on'});
107 - end
108 - ab =strcat(pathname, filename); %Adds path to filename and reads the image using Imread function below.
109 - image =imread(ab); %Imread function reads the image (ab) extracted from the path declared above.
110 - axes(handles.axes2) % Handles the pictures in the axis2.
111 - imshow(image)  % this function shows the image in the axis declared above.
```

Figure 5.1 Input image from static source(Folder/Database).

For real time image capture, the user can load the image directly from a webcam or external camera connected to the system. This is implemented in a separated function and can be called in the MenuFigure. The figure 5.2 below shows the implementation of this functionality. This is a MATLAB 2018a functionality which can be implemented with a simple menu choice as shown in line 14 of figure 5.2. The function "webcam ()" starts the camera and it is previewed by the "preview(cam)" function on line 9 in figure 5.2.

```
5     function I = getcam()
6     %vid = videoinput('macvideo',1);
7     %%,'YCbCr422_1280x720'
8 -   cam = webcam();
9 -   preview(cam);
10    %I = snapshot(cam);
11    %imshow(I);
12
13    %preview(vid);
14 -  choice = menu(' Capture Frame  ','  Capture  ','  Exit  ');
15 -  I = [];
16 -  if (choice == 1)
17        %I = getsnapshot(vid);
18        %I = snapshot(vid);
19 -      I = snapshot(cam);
20        % figure,imshow(I);
21 -      try
22 -          I = rgb2gray(I);
23 -      catch
24 -          warning('Problem using getcam(). Please try again');
25 -      end
```

Figure 5.2: Implementation of real-time image capture.

For the cascade object detector to work, we reference a variable "FaceDtect" to the vision.CascadeObjectDetector as stated above, is a system object detector that detects faces using the Viola-Jones algorithm and it is in the vision toolbox of Matlab 2018a. The algorithm was implemented with the relevant parameters (properties) as shown on the figure 5.3 below. This will detect frontal faces with the default parameters as shown on line 375 on figure 5.3. The training cascade classifier can be made to custom with the parameters of MinSize, ScaleFactor is shown on lies 377 and 378 respectively on the same figure. These parameters have been described below.

The line FaceDtect = vision.CascadeObjectDetector('FrontalFaceCART','MergeThreshold',1);

```
375 - FaceDtect = vision.CascadeObjectDetector('FrontalFaceCART','MergeThreshold',1); %algorithm that is used to detect the faces in the image.
376   %the visionCasdeObjectDetector based on the Viola-Jones
377 - FaceDtect.MinSize = [20,20];
378 - FaceDtect.ScaleFactor = 1.05;
379 - images = rgb2gray(image);
380   % Returns Bounding Box values based on number of objects
381 - BBox = step(FaceDtect,images);
```

Figure 5.3: The FaceDtect declared to the vision.CascadeObjectDetector and parameters to influence detection rate.

On line 381, we implement the vision library of MATLAB to run the Viola-Jones algorithm to detect faces by calling "BBox = step (FaceDtect, images)" to see if any object is being detected. The rectangular search region of interest is denoted by the variable "images" and it is within FaceDtect, specified as a four-element vector [x y width height] specified as pixels in the upper left corner and size of bounding box (Matlab R2018a Documentation). This is also implemented in the real-time camera mode described on figure 5.2 above.

FrontalFaceCART: This is an example of a classification model character vector. This character vector

will detect upright forward-facing faces. It is a default parameter of the algorithm. (MATLAB documentation R2018a).

MergeThreshold: is the criterion needed to define the final detection in an area where there are multiple detections around an object (MATLAB Documentation R2018a). With the threshold specified by an integer, varying the integer targets a large area and influences the false detection rate during a multiscale detection.

ScaleFactor: This takes care of the detection resolution between the MinSize and MaxSize. The search region is scaled by the detector between MinSize and MaxSize, where the MinSize is the size of the smallest detectable object in a vector format with two elements, set in pixels. Similarly, the MaxSize is the largest detectable object in the same format as MinSize set in pixels.

Adjusting the Size of the Detections:

BBox Detections return an M-by-4 element matrix. The bounding box position can be adjusted as shown on line 385 in figure 5.4 to return the bounding box values based on the number of faces. The values can be changed accordingly, to adjust the rectangle dimensions of width and height as on line 387 in figure5.4. The number of BBox can also be counted with the MATLAB command 'size' and annotations inserted with the 'insertObjectAnnotation' as shown on line 388 on figure 5.4 below. Adjusting the size of the BBox was an additional implementation spotted iteratively during testing to crop the face image detect to a size that will not lose many pixels during resizing. This section of the code will be adjusted accordingly depending on the number of subjects in a group as we do not know their positions yet.

```
382      % Scale the rectangle to 1.1 times its original size
383 -    scale = 2;
384      % Adjust the lower left corner of the rectangles
385 -    BBox(:,1:2) = BBox(:,1:2) - BBox(:,3:4)*0.5*(scale - 1);
386      % Adjust the width and height of the rectangles
387 -    BBox(:,3:4) = BBox(:,3:4)*scale;
388 -    detectedImg = insertObjectAnnotation(images,'rectangle',BBox,1:size(BBox,1));
```

Figure 5.4: Adjusting the Size of BBox.

As mentioned earlier, the image resolution (pixels) matter very much in face recognition and every pixel counts towards the performance of the system (Marciniak et al 2015, p.4330). However, converting an image to grayscale using the command **rgb2gray** is important in face detection. This will enable the resulting image to have one image plane (single colour channel) instead of the RGB colour model with extra alpha channel (RGBA) image planes which is easier than applying the operation on all four channels of the RGBA image. Data size and computational time is also important as only a quarter of the grayscale image will be processed. The line of code that converts the image into grayscale is 379 in figure 5.3. The image is displayed on axis 2 using the GUIDE command handles which outputs functionalities to the user interface.

Crop and Write Images to a Folder:

The figure 5.5 below shows the implementation of the code to crop the faces in the bounding boxes using the MATLAB command "imcrop" implemented on line 334 on figure 5.5. The faces are cropped and resized with the MATLAB command 'imresize'. The files are written by the command 'imwrite'. The basepath on line 328 in figure 5.5 indicates where the images will be saved. The "imwrite" function implemented on line 346 in figure 5.5 calls the basepath to write the images in number order and stored in jpg format. All the faces detected in the bounding boxes will be checked and cropped by use of the for-loop will iterate over each bounding box. The for loop starts from line 330 on the figure 5.5 to line 347 on the same figure.

```
327      %%
328 -    basepath = '/Users/macbookair/Desktop/FYPMATLAB/Face Detection/Images/'; % folder in which i want to save images
329
330 -  for i = 1:size(BBox,1) %number of faces in bounding boxes
331
332 -    face=cell(1,size(BBox,1));
333      %axes(handles.axes2);
334 -    face{i}=imcrop(im2,BBox(i,:));
335      %cropped_image = image(RowStart:RowEnd,ColStart:ColEnd,:);
336      %face{i} = imcrop(im2,[45 30 92 112]);
337      %axes(handles.axes2);
338      %figure(4);imshow(face{i});
339 -    faceImage = imresize(face{i},[112 92]);
340 -    faces = (faceImage);
341      %faces = face{i};
342      % try
343      %     faces = rgb2gray(faces);
344      % catch
345      % end
346 -    imwrite(faces, strcat(basepath, sprintf('%d.jpg',i)),'jpg');
347 -  end
```

Figure 5.5: Crop and write files.

The "insertObjectAnnotation" as described above inserts a number in order of detection (1 to5 for five faces detected) and annotation on each bounding box detected with a face. For each bounding box, as of line 392 on figure 5.6, we go each box and the total number of boxes is known by calling the "size" function on line 396 and displayed by a dialogue box shown by the implementation on line 398.

```
388 -    detectedImg = insertObjectAnnotation(images,'rectangle',BBox,1:size(BBox,1));
389 -    axes(handles.axes2)
390 -    imshow(detectedImg); hold on
391      %imshow(detectedImg,I_faces); hold on
392 -    for i = 1:size(BBox,1)
393
394 -        rectangle('Position',BBox(i,:),'LineWidth',2,'LineStyle','-','EdgeColor','y');
395 -    end
396 -    numberOfBBs = size(BBox,1);  %length(BBox);
397 -    message = sprintf('Attendance.\nThe number of students in lecture today = %d', numberOfBBs);
398 -    uiwait(helpdlg(message));
```

Figure 5.6: Counting Number of Bounding Boxes to equate numeric attendance.

The numeric attendance of the implementation in figure 5.6 output to the user by the line 398.

Important Code Details of Face Recognition:

The recognition part of the system has been implemented using Hidden Markov Model with Singular Value Decomposition (SVD). The use of this has been based on the information gathered during the literature review. This has been explained in the previous chapter of this report.

Generating and Loading Dataset:

The dataset is a folder of image folders with image folder assigned to a subject on the dataset. In figure 5.7 below, I have assigned parameters that I will use later in the implementation to generate the indices needed for the training and recognition processes respectively.

```
2      % Jireh Robert Jam
3      %k1557901@kingston.ac.uk
4      % Ref: Matlab Face Recognition Software based on Hidden Markov Model (HMM) Omid Sahki.
5    function [studentDatabase,p] = gendata(p) % P is the parameter value which can be in another way params.
6    % This module reads all the folders inside data folder. Each folder belongs
7    % to one person and the system assumes that the name of the person is equal
8    % to the name of its folder.
9    % In each folder there are 10 images of one person. The system use 5 of
10   % those images to generate its database. If there are less than 5 images
11   % inside a folder, the system use all of them for training. Even if there
12   % is one face photo.
13
14     %% These parameters can be changed with care for Experimental purposes
15 -   p.used_faces_for_training= [1 3 4 7 9]; % [3 5 7 9 10]
16 -   p.used_faces_for_testing =[2 7 8 9 10] ;%[1 2 4 6 8]
17 -   p.block_height = 5;%8 n 10
18 -   p.block_overlap = 4;
19     %these numbers are in the third row of myDatabse and must be discrete values between 0 and 17
20     %for the first value,the second value 0 and 9 and third between 0 and 6
21 -   p.coeff1_quant = 18;%18 first parameter value
22 -   p.coeff2_quant = 10;%10  second discrete value
23 -   p.coeff3_quant = 7;%7 third discrete value
24 -   p.number_of_states = 7;
25 -   p.face_height = 56;%112 84 56 28
26 -   p.face_width = 46;%92 69 46 23
27
28 -   p.eps=.000001;
29 -   p.trained = 0;
```

Figure 5.7: Declared Parameters to be used during the Training and Recognition

The figure 5.8 below shows how a folder of images of each student can be accessed from the dataset. The dataset content is declared by the variable "folderContents" as a cell matrix on line 33. A matrix in

MATLAB is an array of single data type while a cell matrix in MATLAB is a matrix that holds matrices of different types and formats. With a simple MATLAB command "dir" contents from the dataset(database) can be retrieved. Before going through each folder on our dataset, we initialise an empty cell where this information will be stored. This is done on line 35. In order to have a valid index for each subject on the dataset, I have initialised the subject(studentIndex) to zero, as on line 36. The size command in MATLAB is used on line 34 to know the number of folders in the dataset and is declared as "numbofFolders". The for-loop from line 39 to 45 goes through all the folders of the dataset. The waitbar on line 40 will read the names of the folders while the loops go through each folder and will be output the names to a pop-up box to the user. The first two rows in a directory is given by "." And ".." refers to the parent directory (Matlab R2018 Documentation). Inside the loop, the folder for each subject can be accessed to read the list of files ending image format of which it has been saved(e.1.jpg). This is done with the same function "dir" but this time with the name of the subject and the file format at the end as shown on line 50.

In each folder, the name can be accessed by the code "folderContents(student,1).name)" as shown as a parameter of the waitbar. This functionality is important as it will be used later in the implementation to output the name of the input subject to the match subject and will be displayed on the same plot. It is important to output the name to the view of the user to show the code is being executed. A vector is declared by variable "uvt" to contain 5 integers that will match the integer values of file name, stored in the folder of each subject. With these integer values, only the images that will be used for training and recognition to test the system will be accessed and processed respectively.

```
32 -  p.number_of_blocks = (p.face_height-p.block_height)/(p.block_height-p.block_overlap)+1;
33 -  folderContents = dir ('./database'); % loads the dataset, reads all the faces and store them in a memory. dir is invoked to load the datas
34 -  numbOfFolders = size(folderContents,1); % there are folders in directory and each folder holds a number of pictures be student
35 -  studentDatabase = cell(0,0); % first we declare an array of empty cell to store all the information we will process.
36 -  studentIndex = 0; % declaring index for each student to zero gives a valid index because some folder names like "." and ".." will be ignor
37    %% Access folder of each student in database folder individually. first we declare the index to be zero
38 -  h = waitbar(0,'Please wait. Database is underprocess...!!','name','DATABASE LOADING IN PROGRESS');
39 -  for student=1:numbOfFolders
40 -      waitbar(student/size(folderContents,1),h,['Loading '  folderContents(student,1).name ' of ' num2str(length(folderContents)) ' images.'
41 -  if (strcmp(folderContents(student,1).name,'.') ||...      % is not a folder -> skip. this refers to the first directory % and checks to s
42              strcmp(folderContents(student,1).name,'..') ||...  % is not a folder -> skip. this refers to the second directory
43              (folderContents(student,1).isdir == 0))            % is a file -> skip
44 -        continue;
45 -  end
46 -      studentIndex = studentIndex+1; % increments student index after finding a new student in the for-loop
47 -      studentName = folderContents(student,1).name;
48 -      studentDatabase{1,studentIndex} = studentName;
49 -      fprintf([studentName,' ']);
50 -      studentFolderContents = dir(['./database/',studentName,'/*.jpg']);
51 -      nImageStudentFolder = size(studentFolderContents,1);
52 -      blockCell = cell(0,0);
53 -      if (nImageStudentFolder==10)
54          % uvt = [2 3 4 7 9]; %uvt is vector of integer values between 1 and 10 (Dataset2)
55 -         uvt = 9:-2:2; % Database1
56          % uvt = [1 5 6 8 10];
57 -      else
58 -          uvt = 1:nImageStudentFolder; %ufft is a vector that contains 5 integer values between 1 and 10
59 -      end
```

Figure 5.8. Reading and loading dataset.

The code in the figure 5.8 above loads the dataset and reads all the faces and store them in a form on a memory Database.mat on Matlab workspace that will be used for training.

Feature Extraction:

The feature extraction has been implemented as shown in figure 5.9 below. First use the MATLAB function "imresize" to make sure all the images in the folder are of the same size. Then we convert the image to gray. This will improve the computational speed and less information will be lost during extraction. This can be done in two separate lines, but I have chosen to do it on one line as on line 62 in figure 5.9. The images (defined by the vector) are read in a for-loop as implemented from line 61 to 88.

Image Filtering:

```
69 -          I = ordfilt2(I,1,true(3));
```

A minimum order-static filter (non-linear spatial filter) is added to the grayscale image. This filter reduces the amount of intensity variation in pixel order where the number of observations are the order statistics. It is an image restoration technique that make corrupted images near similar as the original image (Kaur and Singh 2014). It has been implemented as shown on line 69 in the figure 5.9. Also, the results from the paper in the literature review shows they were good due to the implementation of this filter.

```
60 -    nfacesTotrain = size(uvt,2);
61 -    for faceIndex=1:nfacesTotrain % we want to extract the features for n training faces.
62 -        I = imresize(double(rgb2gray(imread(['./database/',studentName,'/',studentFolderContents(uvt(faceIndex),1).name]))),[112 92]);
63          % I = imread(['./database/',studentName,'/',studentFolderContents(uvt(faceIndex),1).name]);
64 %          try
65 %              I = rgb2gray(I);
66 %          catch
67 %              %  warning('Check if everything is Ok !');
68 %          end
69 -        I = imresize(I,[p.face_height p.face_width]);
70 -        I = ordfilt2(I,1,true(3));
71          %      figure(1),imshow(I);
72 -        blockStart = 1;
73 -        blockIndex = 0;
74 -        for blockEnd=p.block_height:p.block_height-p.block_overlap:p.face_height
75 -            block = I(blockStart:blockEnd,:);
76              %block_double = double(block);
77 -            [U,S,V] = svd(double(block)); % converts a block matrix into double so it can represent a data matrix
78 -            coeffs = [U(1,1) S(1,1) S(2,2)];
79 -            blockIndex=blockIndex+1;
80 -            blockCell{blockIndex,faceIndex} = coeffs;
81 -            blockStart = blockStart + (p.block_height-p.block_overlap);
82 -        end
83 -    end
84 -    studentDatabase{2,studentIndex} = blockCell;
85 -    if (mod(studentIndex,10)==0) %after 10 rows, move to a newline for display
86 -        fprintf('\n');
87 -    end
88 - end
```

Figure 5.9: Feature Extraction

Proceeding to the feature extraction for training, we use SVD. First, we convert the image into grayscale and convert the block matrix into double to represent a data matrix with gray scale values to each pixel. Line 77 to line 78 computes the SVD from the block matrix into double within the loop. The "blockcell" is a 5x52 matrix with 52 rows corresponding to the number of blocks extracted for each image and 5 columns correspond to the number of images to be trained. These values are obtained due to the image resized to half of its size to give [56 46]. This increases computational speed and the observation sequence generated as explained in the design chapter. Each face of width=46 and height = 56 are divided into overlapping blocks of newHeight =5 and width = 46. The overlapping size given by OL = (newHeight-1). The number of blocks as seen previously is now given by #Block = (H-newH)/(newH-OL) +1. This leaves us with #Block = (56-5)/ (5-4) +1= 52.

The block extraction starts with the parameters of the blocks are declared in figure 5.7. With these parameters already declared, it can be implemented by the code from line 74 to line 82 of figure 5.9 above. This will now convert each face into a sequence of 52 elements as mention earlier represented by a 5-by-52 element matrix. We should note here that we are using 5 images for the training as declared by the vector integers. How these are output to the command window is as shown in Figure 5.10 below.

```
Command Window
  Loading Faces ...
  Adrien_Brody Andre_Agassi AngelaMerkel Ann_Veneman Ari_Fleischer Atal_Bihari_Vajpayee Bill_Clinton Bill_Gates Bill_McBride Bill_Simon
  Britney_Spears Carlos_Moya Catherine_Zeta-Jones Charles_Moose Charles_Taylor Dominique_de_Villepin GeorgeHWBush George_W_Bush Gonzalo_Sanchez_de_L
  Gray_Davis Halle_Berry Hillary_Clinton Ian_Thorpe Jack_Straw Jacques_Chirac Jacques_Rogge Jean-David_Levitte Jeremy_Greenstock Jimmy_Carter
  Leonardo_DiCaprio Mohammad_Khatami Paul_Wolfowitz Ray_Romano Salma_Hayek Sergey_Lavrov Sergio_Vieira_De_Mello Silvio_Berlusconi Sylvester_Stallone
  Tom_Cruise Tom_Daschle Tom_Ridge Tommy_Thompson Tony_Blair Venus_Williams Vicente_Fox Vladimir_Putin Walter_Mondale Wen_Jiabao
  Winona_Ryder Yoriko_Kawaguchi
   Database generated.

  ans =

    5×52 cell array

    Columns 1 through 7

      {'Adrien_Brody'}    {'Andre_Agassi'}    {'AngelaMerkel'}    {'Ann_Veneman'}    {'Ari_Fleischer'}    {'Atal_Bihari_Vaj…'}    {'Bill_Clinton'}
      {52×5 cell     }    {52×5 cell     }    {52×5 cell     }    {52×5 cell    }    {52×5 cell      }    {52×5 cell          }    {52×5 cell     }
      {52×5 cell     }    {52×5 cell     }    {52×5 cell     }    {52×5 cell    }    {52×5 cell      }    {52×5 cell          }    {52×5 cell     }
      {52×5 cell     }    {52×5 cell     }    {52×5 cell     }    {52×5 cell    }    {52×5 cell      }    {52×5 cell          }    {52×5 cell     }
      { 1×5 cell     }    { 1×5 cell     }    { 1×5 cell     }    { 1×5 cell    }    { 1×5 cell      }    { 1×5 cell          }    { 1×5 cell     }

    Columns 8 through 14

      {'Bill_Gates'}    {'Bill_McBride'}    {'Bill_Simon'}    {'Britney_Spears'}    {'Carlos_Moya'}    {'Catherine_Zeta-…'}    {'Charles_Moose'}
      {52×5 cell   }    {52×5 cell     }    {52×5 cell   }    {52×5 cell       }    {52×5 cell    }    {52×5 cell          }    {52×5 cell      }
      {52×5 cell   }    {52×5 cell     }    {52×5 cell   }    {52×5 cell       }    {52×5 cell    }    {52×5 cell          }    {52×5 cell      }
      {52×5 cell   }    {52×5 cell     }    {52×5 cell   }    {52×5 cell       }    {52×5 cell    }    {52×5 cell          }    {52×5 cell      }
      { 1×5 cell   }    { 1×5 cell     }    { 1×5 cell   }    { 1×5 cell       }    { 1×5 cell    }    { 1×5 cell          }    { 1×5 cell      }

    Columns 15 through 20

      {'Charles_Taylor'}    {'Dominique_de_Vi…'}    {'GeorgeHWBush'}    {'George_W_Bush'}    {'Gonzalo_Sanchez…'}    {'Gordon_Brown'}
      {52×5 cell       }    {52×5 cell         }    {52×5 cell     }    {52×5 cell      }    {52×5 cell          }    {52×5 cell     }
```

Figure 5.10: Showing "blockcell" in 5x52 array with each subject.

Quantization:

As defined in the previous chapter, we have to round up large values with quantization which is an irreversible step as some of the information is lost during the process. However, it is necessary for this application as we are using HMM. The values for the coefficients on line 78 in figure 5.9 are quantized into discrete values declared on line 21 to 23 in figure 5.7.

```
89    %% finds the maximum and minimum of all three coefficients for all coefficients in the entire observation vector.
90    % so we fine the minimum of U(1,1) and the maximum of Sumation(1,1) and Sumation(2,2) and
91    % U(1,1).
92 -  coeffs1 = [];
93 -  coeffs2 = [];
94 -  coeffs3 = [];
95 -  nStudents = size(studentDatabase,2);
96 -  for studentIndex=1:nStudents
97 -      [nBlocks,nImages] = size(studentDatabase{2,studentIndex});
98 -      for imageIndex=1:nImages
99 -          for blockIndex=1:nBlocks
100 -             if (isempty(studentDatabase{2,studentIndex}{blockIndex,imageIndex}))
101 -                 continue;
102 -             end
103
104 -             coeffs1(:,end+1) = studentDatabase{2,studentIndex}{blockIndex,imageIndex}(1,1);
105 -             coeffs2(:,end+1) = studentDatabase{2,studentIndex}{blockIndex,imageIndex}(1,2);
106 -             coeffs3(:,end+1) = studentDatabase{2,studentIndex}{blockIndex,imageIndex}(1,3);
107 -         end
108 -     end
109 - end
110 -  CoefMax1 = max(coeffs1(:)); %maxCoeff1
111 -  CoefMax2 = max(coeffs2(:)); %maxCoeff2
112 -  CoefMax3 = max(coeffs3(:)); %maxCoeff3
113 -  CoefMin1 = min(coeffs1(:));
```

Figure 5.11: Quantization of observation vectors

The next step is block quantization. We will find the maximum and minimum coefficients of the observation vector. First, we declare the coefficients and use a for-loop to go through the second row of the data generated in put them into separate matrices as shown on line 92 to 93 in the figure 5.11 above. The values of U(1 ,1), S(1,1), S(2,2) which were stored on line 84 in figure 5.9 are quantized into the discrete values mentioned above. These coefficients have maximum and minimum values and can be computed from all possible observed coefficients. First, we used the values stored in the second row of the database(studentDatabase) as on the code in line 84 of figure 5.9 to compute the minimum and maximum values of all observation vectors using a for-loop as in line 96 to line 112, gathering them into separate matrices as shown. The quantized values are calculated with delta coefficients and stored in the third row of the database(studentDatabase). This is done as shown on line 117 to 138 as in figure 5.12. A study by Miar-Naimi and Davari (2008p.6) has shown that each quantized vector is associated with a label that here is an integer" Each block needs to store a discrete value(label). Each block of image is assigned an integer. These integers are the discrete values declared in figure 5.7 line 21-23 (18,10,7). and this is stored on line 137 on the 4 row of the database (studentDatabase). With all the labels stored as a cell matrix in the 4 row of the database, they can be converted to a regular matrix using the MATLAB command “cell2mat” as shown on line 142 in figure 5.12.

```
117 -  deltaCoeff1 = (CoefMax1-CoefMin1)/(p.coeff1_quant-p.eps);
118 -  deltaCoeff2 = (CoefMax2-CoefMin2)/(p.coeff2_quant-p.eps);
119 -  deltaCoeff3 = (CoefMax3-CoefMin3)/(p.coeff3_quant-p.eps);
120 -  p.coeff_stats = [CoefMin1 CoefMin2 CoefMin3;CoefMax1 CoefMax2 CoefMax3;deltaCoeff1 deltaCoeff2 deltaCoeff3];
121 -  minLabel = Inf; %minLabel
122 -  maxLabel = -Inf;%maxLabel
123 -  for studentIndex=1:nStudents
124 -      for imageIndex=1:nImages
125 -          for blockIndex=1:nBlocks
126 -              if (isempty(studentDatabase{2,studentIndex}{blockIndex,imageIndex}))
127 -                  continue;
128 -              end
129 -              blockCoeffs = studentDatabase{2,studentIndex}{blockIndex,imageIndex};
130 -              minCoeffs = p.coeff_stats(1,:);
131 -              deltaCoeffs = p.coeff_stats(3,:);
132 -              quant = floor((blockCoeffs-minCoeffs)./deltaCoeffs); % rounds to the nearest integer.
133 -              studentDatabase{3,studentIndex}{blockIndex,imageIndex} = quant;
134 -              label = quant(1)* p.coeff2_quant*p.coeff3_quant + quant(2) * p.coeff3_quant + quant(3)+1;
135 -              minLabel = min(label, minLabel); % returns smallest element in each row
136 -              maxLabel = max(label, maxLabel); % can be represented as maxLabel = max([label maxLabel]);
137 -              studentDatabase{4,studentIndex}{blockIndex,imageIndex} = label;
138 -          end
139           % fifth row cell matrix converted into regular matrix with integer value that holds observation for each image.
140           % i.e after computing the labels for each image. this is done using
141           % cell2mat. we now have a matrix of 5x52
142 -         studentDatabase{5,studentIndex}{1,imageIndex} = cell2mat(studentDatabase{4,studentIndex}(:,imageIndex));
143 -     end
144 - end
```

Figure 5.12: Quantization data stored on the third row of database as in line 133.

On the figure 5.12 above, on line 134 then the maximum value for one discrete block is 1260. The values computed for each subject are stored on the fourth row of the matrix as on line 137 in the figure above. The fifth row is a cell matrix converted into a regular matrix with integer values that holds observation for each image.

Training the Dataset:

The training of the HMM is done by the MATLAB code hmmtrain as shown below.

```
[estTRANS,estEMIS]=hmmtrain(seqs,TRANSGUES,EMISGUES,'Tolerance',.01,'Maxiterations',10,'Algorithm', 'BaumWelch');
```

This will give an estimation of the emission and transition probabilities for a Hidden Markov Model.

First, create initial probabilities as shown on line 13 below. TRANSGUES is a good guess for transition probabilities and EMISGUES is a good guess for emission probabilities. Miar-Naimi and Davari (2008, p.7) initial estimates of observation probability matrix(TRANSGUES) has to be obtained. This is computed with the number of states of our model and all possible observation symbols obtained during quantization. This is the probability matrix that will be used to estimate the final probability states for the HMM. On line 13 in the figure below, the return values are an n-by-n matrix of ones representing the number of states (7) since we are using the 7-state model. estTRANS and estEMIS are final estimated probabilities. These are stored on a sixth row on the database as shown on line 32 and 33. These processes are iterated for all training images corresponding to the integer declared by the vector "uvt" of the subjects in the dataset.

```
8 -  load DATABASE.mat;
9 -  p.trained = 1;
10 -  p.number_of_labels = ...
11        p.coeff1_quant* p.coeff2_quant* p.coeff3_quant;
12    %% TRANGUES is good guess for transmission probabilities and EMISGUES is good guess for emission probabilities
13 -  TRANSGUES = ones(p.number_of_states,p.number_of_states) * p.eps;% params.eps; returns an n-by-n matrix of ones i.e p.numOfStates-by-p.numOf
14 -  TRANSGUES(p.number_of_states,p.number_of_states) = 1;
15 -  for r=1:p.number_of_states-1
16 -      for c=2:p.number_of_states
17 -          TRANSGUES(r,c) = 0.5;
18 -          TRANSGUES(r,c-1) = 0.5;
19 -      end
20 -  end
21
22 -  EMISGUES = (1/p.number_of_labels)*ones(p.number_of_states,p.number_of_labels);
23
24 -  fprintf('Training Images, Please wait ...\n');
25 -  nStudents = size(studentDatabase,2);
26 -  h = waitbar(0,'Please wait. Database is underprocess...!!','name','TRAINING IN PROGRESS');
27 -  for studentIndex=1:nStudents
28 -      waitbar(studentIndex/nStudents,h,['Training '  [studentDatabase{1,studentIndex},' '] ' of ' num2str(length(nStudents)) ' images.']);
29 -      %fprintf([studentDatabase{1,studentIndex},' ']);
30 -      seqs = cell2mat(studentDatabase{5,studentIndex})'; % matrix (seq) of all the observation sequence matrix to be used converted to a regu
31 -      [estTRANS,estEMIS]=hmmtrain(seqs,TRANSGUES,EMISGUES,'Tolerance',.01,'Maxiterations',10,'Algorithm', 'BaumWelch'); % estimates the trans
32 -      estTRANS = max(estTRANS,p.eps);
33 -      estEMIS = max(estEMIS,p.eps);
34 -      studentDatabase{6,studentIndex}{1,1} = estTRANS;
35 -      studentDatabase{6,studentIndex}{1,2} = estEMIS;
36 -      if (mod(studentIndex,10)==0)
37 -          fprintf('\n');
38 -      end
```

Figure 5.13: Training the Dataset.

Recognition:

The recognition done by associating each model(face) to a HMM. Each input image is associated with its own observation vector. In this case, each test image should undergo block extraction and quantization process, as we have already seen this above.

```
70 -  number_of_student_in_database = size(studentDatabase,2);
71 -  results = zeros(1,number_of_student_in_database);
72 -  for faceIndex=1:number_of_student_in_database
73 -      TRANS = studentDatabase{6,faceIndex}{1,1}; %transition probabilities of the trained model
74 -      EMIS = studentDatabase{6,faceIndex}{1,2}; %emission probabilities of the trained model
75 -      [~,logpseq] = hmmdecode(seq,TRANS,EMIS);
76 -      P=exp(logpseq);
77 -      results(1,faceIndex) = P;
78 -  end
79 -  [maxlogpseq,studentIndex] = max(results);
80    %     [maxlogpseq,states] = hmmgenerate(1000,TRANS,EMIS);
81    %     likelystates = hmmviterbi(maxlogpseq, TRANS, EMIS);
82    %     disp(sum(states==likelystates)/100);
83    %disp(P);
84    %disp(maxlogpseq);
85    %disp(logpseq);
86    % TRANS = studentDatabase{6,faceIndex}{1,1};% disp(TRANS);
87    % EMIS = studentDatabase{6,faceIndex}{1,2}; %disp(EMIS);
88 -  for faceIndex=1:number_of_student_in_database
89 -      TRANS = studentDatabase{6,faceIndex}{1,1}; %transition probabilities of the trained model
90 -      EMIS = studentDatabase{6,faceIndex}{1,2}; %emission probabilities of the trained model
91 -      [maxlogpseq,states] = hmmgenerate(1000,TRANS,EMIS);
92 -      likelystates = hmmviterbi(maxlogpseq, TRANS, EMIS);
93       % disp(sum(states==likelystates)/100);
94 -  end
```

Figure 5.14: Face Recognition.

Using the trained model, we have final estimated transition and emission of the trained model and we store these to TRANS and EMIS respectively as shown on line 73 and 74 the figure 5.14 above. This process is iterated for all images as implemented from line 72 to 78. The maximum logarithm of probability of sequence given the transition and emission matrix stored on line 73 and 74 in figure 5.14 can be computed with the MATLAB algorithm "hmmdecode". The results are stored for each face index. To obtain the percentage at which the recognised subject will agree with the random sequence, iterate through all images as shown on lines89-94. The MATLAB command hmmgenerate is used to generate the random sequence of states from the model from the transition and emission probabilities stored on row 6 of the database(studentDatabase). Given these probabilities, the function hmmviterbi

uses the Viterbi algorithm to calculate the most likely sequence of states that will generate a given sequence of emissions. The actual percentage now is computed on the line 93 of figure 5.14. This is commented out but is used on line 97 in figure 5.15 to display the results to the user depending on which image is tested.

From the requirements, the client wanted to see the matched face displayed alongside the searched image. The name of the subject the image is matched to is to be displayed on top of the image found on the database. This was implemented as shown on the figure below and displayed using the "imshow" command in MATLAB. The use of a for-loop from line 104 to 112 is to go through the database and select the folder set that is equal to the match equivalent to the index of the maximum probability in the displayed results.

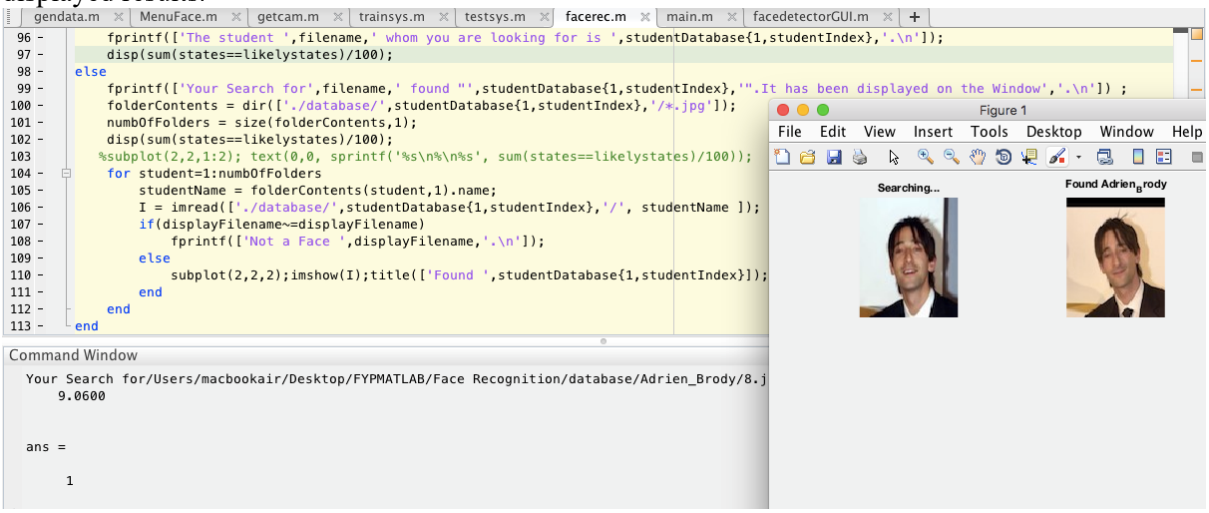


Figure 5.15: Display of Image Match and Implementation of functionality.

Data Capture Implementation:

The webcam or external camera has been implemented with the MATLAB command shown on line 4 on the figure 5.17 below. This creates the webcam object cam that connects a single webcam on the system. (MATLAB R2018a Documentation). With this function a menu choice can be created to capture or exit when the camera is initiated. The face detector algorithm has been embedded here to detect faces straight away by way of choice from an input command by the user.

```
1  function I = getcam()
2  %vid = videoinput('macvideo',1);
3  %%,'YCbCr422_1280x720'
4  cam = webcam();
5  preview(cam);
6  %I = snapshot(cam);
7  %imshow(I);
8
9  %preview(vid);
10 choice = menu(' Capture Frame  ','  Capture  ','  Exit  ');
11 I = [];
12 if (choice == 1)
13     %I = getsnapshot(vid);
14     %I = snapshot(vid);
15     I = snapshot(cam);
16     % figure,imshow(I);
17     try
18         I = rgb2gray(I);
19     catch
20         warning('Problem using getcam(). Please try again');
21     end
22     faceDetector = vision.CascadeObjectDetector();
23
24     bbox            = step(faceDetector, I);
25
26     videoOut = insertObjectAnnotation(I,'rectangle',bbox,'Face');
27     figure, imshow(videoOut), title('Detected face');
28
29     I = I(8:231,68:251);
30     I = imresize(I,[112 92]);
31
32 end
33 closePreview(cam);
34 clear('cam');
```

Figure 5.16: Accessing a web cam or external camera. (MATLAB R2018a).

The system can be evaluated by passing the face recognition function. The face recognition function returns the index of the recognised face on the database. The total index of correctly recognised faces can be computed as a percentage against the total number of faces. First, the function is implemented

to go through the folder and pick images that were not trained declared by a vector “uvt”. The vector has integer values that match the indices of images that were trained. The total number of recognised subjects is held in “recSubjects” of the function.

```
if (p.trained==0)
    fprintf('System is not trained. Please train your system first.\n');
    return;
end
total = 0;
recSubjects = 0;
fprintf('Please Wait...\n');
folderContents = dir ('./database');
nImageStudentFolder = size(folderContents,1);
studentIndex = 0;
for student=1:nImageStudentFolder
    if (strcmp(folderContents(student,1).name,'.') ||... % is not a folder -> skip
        strcmp(folderContents(student,1).name,'..') ||...
        (folderContents(student,1).isdir == 0))
          continue;
    end
    studentIndex = studentIndex+1;
    studentName = folderContents(student,1).name; %fprintf([studentName,'\n']);
    studentFolderContents = dir(['./database/',studentName,'/*.jpg']);
    nImageStudentFolder = size(studentFolderContents,1);
    if(nImageStudentFolder==10)
    %uvt = [2 3 4 7 9];   % uvt = [1 3 4 7 10];
    uvt = 9:-2:1;
    else
        uvt = 1:nImageStudentFolder;
    end
    for faceIndex=1:size(uvt,2)
        total = total + 1;
        filename = ['./database/',studentName,'/',studentFolderContents(uvt(faceIndex),1).name];
        answerStudentIndex = facerec(filename,studentDatabase,p,1);
        %disp(answerStudentIndex);  %disp(studentIndex);
        if (answerStudentIndex==studentIndex)
            recSubjects = recSubjects + 1;
        end
    end
end
recSubjects = recSubjects/total*100;
fprintf(['\nRecognition Rate is ',num2str(recSubjects),'%% for a total of ',num2str(total),' unseen faces.\n']);
```

Figure 5.17: Recognition Rate Calculations.

.

Structure of the GUI and Implementation:

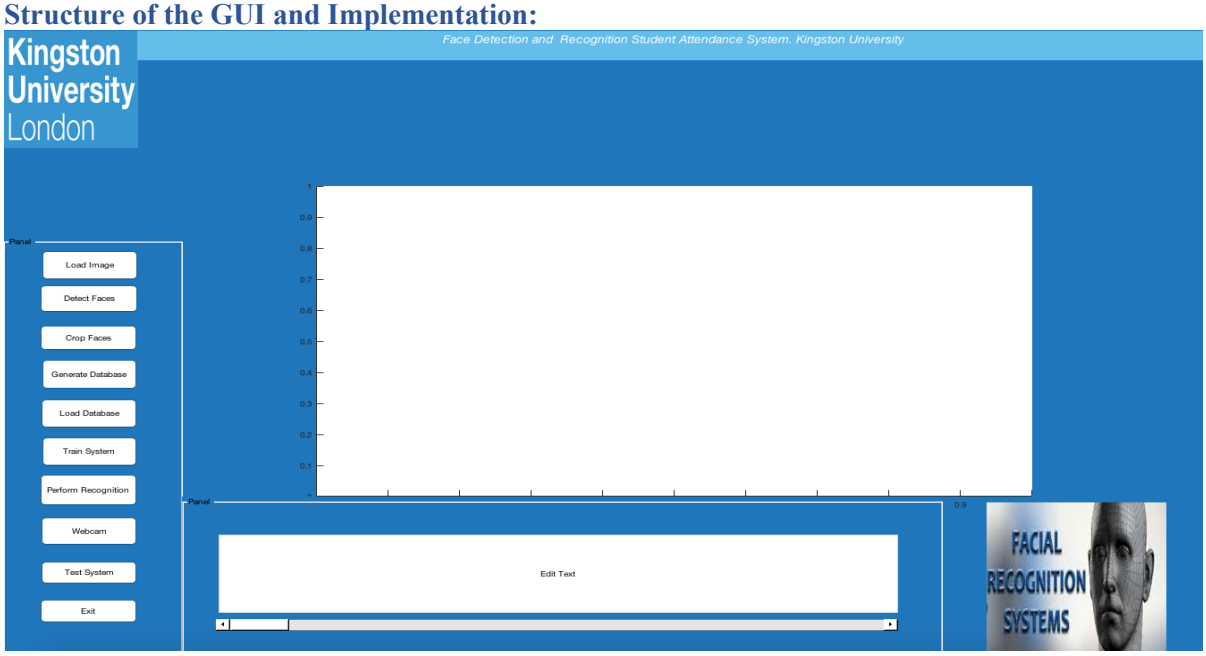


Figure 5.18: Structure of GUI.

Load button callback:

The load button callback has been implemented in a way to select images stored in all format type by the “uigetfile” command. This has been shown on the figure 5. 20 below.

```
97    function LoadBtn_Callback(hObject, eventdata, handles)
98    % hObject    handle to LoadBtn (see GCBO)
99    % eventdata  reserved - to be defined in a future version of MATLAB
100   % handles    structure with handles and user data (see GUIDATA)
101 -  global image im2 %variable image
102 -  if true
103 -  [filename,pathname]=uigetfile({'*.jpeg;*.jpg;*.gif;*.tif;*.tiff;*.bmp;*.png',...
104       'all  image file';'*.jpg;*.jpeg','JPEG Files(*.jpg,*.jpeg)';...   % Select the correct color for test
105       '*.gif','GIF Files(*.gif)';'*.tif;*.tiff','TIFF Files(*.tif,*.tiff)';...
106      '*.bmp','Bitmap Files(*.bmp)';'*.png','PNG Files(*.png)';'MultiSelect', 'on'});
107 -  end
108 -  ab =strcat(pathname, filename); %Adds path to filename and reads the image using Imread function below.
109 -  image =imread(ab); %Imread function reads the image (ab) extracted from the path declared above.
110 -  axes(handles.axes2) % Handles the pictures in the axis1.
111 -  imshow(image)  % this function shows the image in the axis declared above.
112
```

Figure 5.20: Implementation of Load button function.

The recognition callback has been embedded by the same command. However, because the requirement requested to display the input image side by side with the matched image, it has been done using the subplot and imshow command. This is as shown in the figure 5.21 below.

```
function recognition_Callback(hObject, eventdata, handles)
% hObject    handle to recognition (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load DATABASE.mat;
guidata(axes,handles)
% results = set(handles.text3,'String',handles.recognition);
%handles = guihandles();
%handles.mydata = facerec;
% guidata(hObject, handles);
        if (~exist('studentDatabase','var'))
            fprintf('Please load database first!\n');
        else
            pause(0.001);
            % show file dialog box and get one image from the user
            [file_name, file_path] = uigetfile ({'*.jpg';'*.bmp';'*.pgm';'*.png'});
            if file_path ~= 0
                filename = [file_path,file_name];
                %subplot(2,2,1)
                axes(handles.axes2)
                subplot(2,2,1);imshow(filename);title('Searching...');
                facerec (filename,studentDatabase,p,0)
```

Figure 5.21: Implementation of recognition callback button.

The implementation of the other buttons was done using the “handles” command using GUIDE in Matlab. The figure below is an example of the implementation of the training button.

```
269    function TrainData_Callback(hObject, eventdata, handles)
270    % hObject    handle to TrainData (see GCBO)
271    % eventdata  reserved - to be defined in a future version of MATLAB
272    % handles    structure with handles and user data (see GUIDATA)
273 -  handles.mydata = trainsys;
274 -  guidata(hObject, handles);
```

Figure 5.22: Implementation of training callback button

The implementation of the exit button is set to destroy the GUI with the command delete(handles.MenuFace).

```
232    function ExitBtn_Callback(hObject, eventdata, handles)
233    % hObject    handle to ExitBtn (see GCBO)
234    % eventdata  reserved - to be defined in a future version of MATLAB
235    % handles    structure with handles and user data (see GUIDATA)
236 -  clc;
237 -  close All;
238    % Clear the variables
239    % clearStr = 'clear all';
240 -  clearStr = 'clear All';
241 -  evalin('base',clearStr);
242    % Delete the figure. Use the tag on the main GUI to add to the handle. You
243    % can rename the tag or use the default name.
244 -  delete(handles.MenuFace);
```

Figure 5.23: Implementation of exit button.

Summary:

In summary, the implementation of this system has been carried out incrementally following the methodology and the requirements from the supervisor and client.

The first full working prototype could only detect a few faces and has been discussed in the evaluation chapter below. Another functionality that improved on the recognition accuracy, adjusted the rectangle of the bounding box scaled to a reasonable size that will not lose so much information during resizing. This has been discussed in the chapter below and the sizes stated. Furthermore, the percentage probability of the image that is matched to a subject on the database was implemented. The results were further implemented to display the input image and output image for visible confirmation.

CHAPTER SIX

Evaluation and Analysis of the System.

Analysis of the Face Detection part of the system.

As already stated in previous chapter of this paper, in face detection, the algorithm returns the location of image patches of any size that contains a face defined as two eyes, a nose and a mouth. From this, there is always a risk of detecting false positives in a face detection system. Yi-Qing Wang (2014 p.1), in his analysis of the Viola Jones algorithm highlights that "no objective distribution can describe the actual probability for a given image to have a face". Therefore, in order to achieve an acceptable performance, any face detection algorithm used must minimize false positive rate and false negative rates respectively.

This system uses the state of the art Viola Jones algorithm framework for face detection which has been mentioned in the literature review section on this paper. The Viola-Jones algorithm becomes highly efficient with the use of image scaling approaches and techniques to detect faces of different dimensions which are at different angles to the viewer. The quality of the image was also taken into consideration whilst analysing the system.

The considered in this analysis are additional cascade e.g. FrontalFaceCART alongside three parameters of the vision.CascadeObjectDetector (MergeThreshold level, scale factor and window size) An integral image with low quality scaling of this algorithm will lead to a loss in features which can directly affect the performance of the system.

Window Size: The window size consists of the MinSize and MaxSize. This sets the size a face detection can be. For this system, to maximise the accuracy, I have decided to use MinSize only which sets the minimum size for a face in order to include very tiny faces at the back of the class, as the faces are of different sizes. The MinSize [height width] is greater than or equal to [20 20] for this system. Other sizes have been tested during the implementation and iteration testing of the system before settling for this size.

ScaleFactor: The ScaleFactor determines the scale for the detection resolution between successive increments of the window size during scanning. This parameter will help to decrease the number of false positives. (decrease in false positive is as a result of increase in scale factor).

MergeThreshold: This parameter will control the number of face detections and declare the final detection of a face area after the combination and rejection of multiple detections around the object. The accuracy of the system depends on the level of MergeThreshold. The higher the value of the MergeThreshold level, the lower the accuracy and vice versa.

The algorithm has been analysed with images of different sizes for Image1, Image2 and Image3 respectively. These images were taken from different websites showing students in a classroom setting with natural sitting positions showing faces of different sizes from (24x24 to 60x60). The textual description of the images has been summarized on the table below and has been classified in order of difficulty.

Key:

nFaces = Number of Faces per image

nRows = Number of Rows per image

	Image Size	nFaces	nRows	Description	Image Classification
Image1	930X620	29	9	• All the students are facing the camera with each face clearly visible. • The sitting position is not relatively spaced out (can be seen as organised). • The students are arranged in 8 rows and 6	Easy

				columns directly facing the camera. • The sitting arrangement is queued in a quadrilateral form. • There is minimal obstruction to student's frontal faces • At least 5 students have facial hair. • Older looking students of age range 30 to 50 • At least 7 students have glasses on. • The face sizes are ranging from 25x25 to 80x80. • This is a clear picture with high illumination and high resolution. • Angle of inclination to the camera is approximately 90 degrees.	
Image2	850x570	59	17	• Not all the students are facing the camera, as their sitting positions is seen to be side by side. • Sitting position has a more naturally looking feel of a lecture hall setting compared to image1. • The sitting position can be in seen in the form of 11 rows with 5 identical columns at the front and 7 columns at the back. • Students faces are at different angles to each other with reference to the camera position. • The gap between the first row and last row is wide, making the face sizes at the back very tiny. • There is a wide variety of face sizes ranging from 16x16 to 60x74. • The image shows increased brightness from the front seats and decreases progressively towards the back of the lecture hall. • This image generally can be described as low illumination and lower resolution compared to image1. • Angle of inclination to the camera is approximately 90 degrees	Difficult
Image3	1200x630	88	13	• Camera in position to capture most of the classroom. • There is sparse distribution of the students in a haphazard manner distant away from the camera. • Majority of the students are sitting at the back of the classroom. • There is a wide gap and empty space in the middle between the first row and the last row. • The faces are at different angles to each other. • There is a wide variety of faces sizes compared to image2 ranging from 14x16 to 40x40. • Some students have their hands on their faces	Very Difficult

				and obstruction to other faces by some other students. • The lecturer is an obstruction to some faces in some of the front rows which could be detected. • There is no clear distinction of facial features such as beards, mouth nose due to the student faces in the image a little blurry. • Angle of inclination to the camera is approximately 45 degrees	

Table 1: Textual Description of Images with Level of Difficulty for Face Detection.

In order to carry out the experiment, all three images were used. The values of the image sizes were generated by resizing each image to ¾, ½ and ¼ of the original image sizes respectively. The MergeThreshold values were obtained by performing the experiment at MergeThreshold levels of 0,1,5,7 and 10. The ScaleFactor used was 1.05 and 1.25. The performance metrics used to analyse the impact of these techniques are True Positive (TP), False Positive (FP) and False Negative(FN).
Where;

TP is the number of faces detected from the algorithm (detected and identified true)
FP is the number of non-faces falsely detected as faces (detected and identified as false)
FN is the number of faces not detected

Precision is the proportion of TP against all positive results.
Precision = TP/TP+FP.
Detection Rate = TP/Total Faces.

Below the first image is considered with students sitting with all their faces facing the front of the classroom.

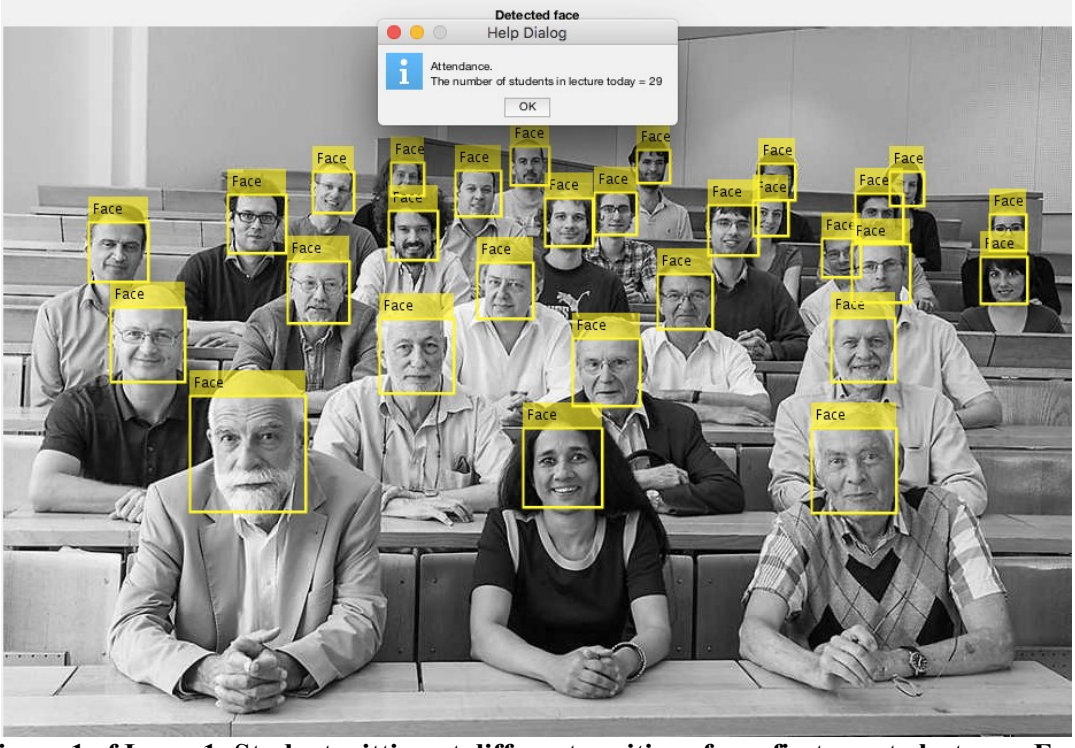


Figure 1 of Image1: Students sitting at different positions from first row to last row. Face

Detection tested with no parameters.

In this image, the size of the image has not been reduced. The default FrontalFaceCART, MergeThreshold of 4, MinSize of [], and Scale factor of 1.1 have been implemented and all the 29 faces of the students detected. The MergeThreshold level, face window size and ScaleFactor have not been taken into consideration. This shows how well the system can perform with a good quality image with faces not too distant from the camera.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image1	4	930X620	[]	1.1	29	0	0	100
Image2	4	850x570	[]	1.1	29	0	0	49.15
Image3	4	1200x630	[]	1.1	20	0	0	22.72

Table 2: Showing default parameters for simple Viola Jones Face Detection

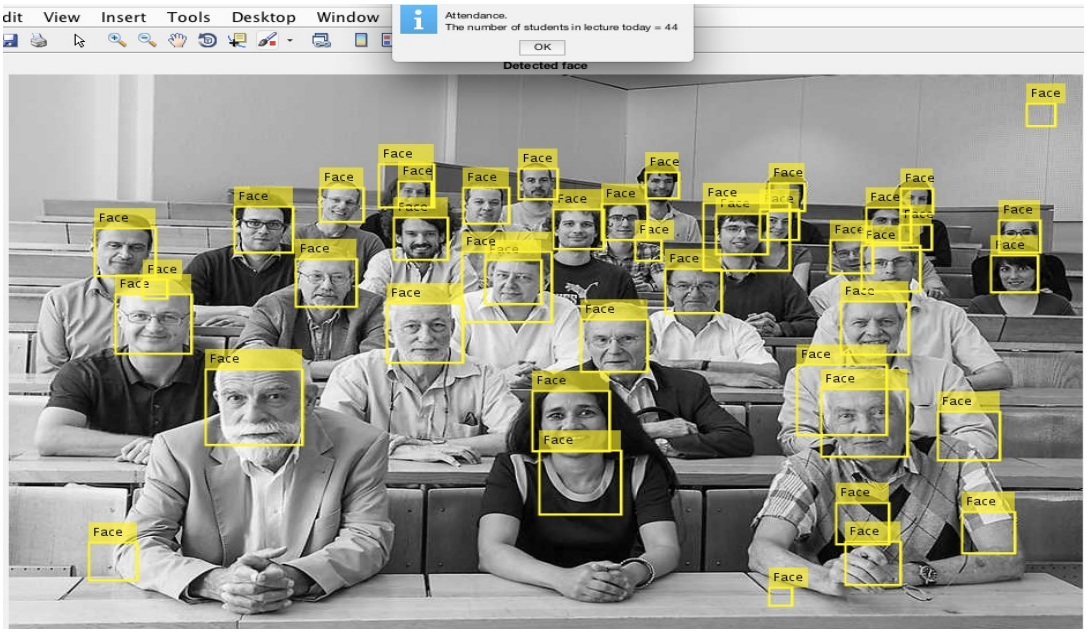


Figure 2 of Image1: Face Detection tested with different parameters.

In figure 2 of image1, the algorithm has been experimented with the following parameters MergeThreshold = 1, window size for face = [20,20] and ScaleFactor = 1.05.
In this approach, the window scale of size [20,20] slides over the image at different directions scanning the image. After one complete scan, the scale factor (1.05) is applied to increasing the detection resolution of the window. This process continues until the size adjust to approximate the size of each face. Also, the image size plays a big role for during this process as it determines the face sizes in the image. Reducing the image size may lead to loss in features etc. However, it still detects some false positives as the algorithm searches for anything in shape with features of the face. Further experiments were carried out to demonstrate the performance of the algorithm. The results of these experiments are as summarized on the tables below.

Table Key:
MergeThreshold = MT.
MATLAB function imresize(I) where I is the image,
ScaleFactor = SF.
True Positive = TP.
False Positive = FP.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image1	4	Imresize(I)	[]	1.1	29	0	0	100
Image1	4	Imresize(I,0.75)	[]	1.1	29	0	0	100
Image1	4	Imresize(I,0.5)	[]	1.1	19	0	0	65.51
Image1	4	Imresize(I,0.25)	[]	1.1	3	0	0	10.34

Table 3A Image 1: Shows Performance Metrics of different Image sizes (using imresize) to ¾, ½ and ¼ of the original image size using default parameters of the Viola Jones Face Detection algorithm.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image1	1	Imresize(I)	[20 20]	1.05	28	15	0	96.55
Image1	1	Imresize(I,0.75)	[20 20]	1.05	28	9	0	96.55
Image1	1	Imresize(I,0.5)	[20 20]	1.05	25	3	4	89.29
Image1	1	Imresize(I,0.25)	[20 20]	1.05	7	0	22	24.13

Table 3B Image 1: Shows Performance Metrics of different Image sizes (using imresize) to ¾, ½ and ¼ of the original image size for Viola Jones Face Detection.

Table 3B above shows the evaluation of Image1 with the original image size, experimented with different image sizes, and the results have been displayed as shown. From this table, the figures, the system performs better with a 96.55 % detection rate. With the image size reduced to different sizes, it can be seen that the detection rate drops. A plot of the Detection rate against image size has been shown in Figure 3 below.

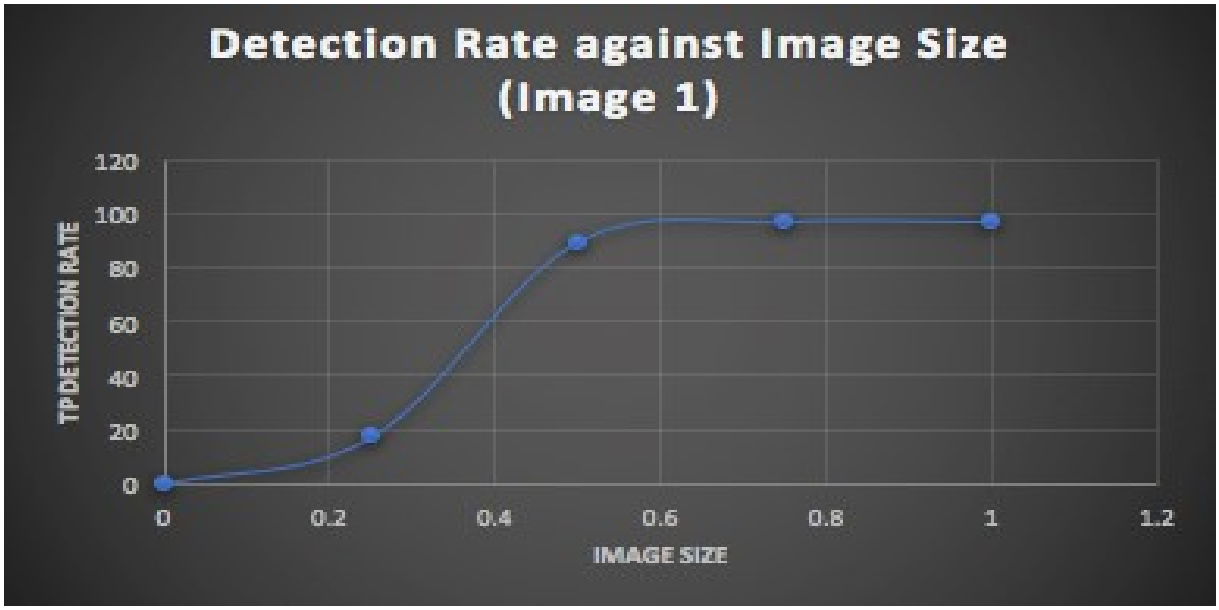


Figure 3: Plot of Detection Rate Against Image Size for Image 1.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image1	0	Imresize(I)	[20 20]	1.25	22	3	0	75.86
Image1	1	Imresize(I)	[20 20]	1.25	28	5	1	96.55
Image1	5	Imresize(I)	[20 20]	1.25	21	0	8	72.41
Image1	7	Imresize(I)	[20 20]	1.25	15	0	14	56.62
Image1	10	Imresize(I)	[20 20]	1.25	11	0	18	37.93

Table 3C Image 1: Shows Performance Metrics of different MergeThrehold Levels for Viola Jones Face Detection ('FrontalFaceCART').

Table 3C above shows the analysis at different mergethreshold levels with a scaleFactor of 1.25 and window size of [20,20] keeping the original size of the image. The detection rate was 78.86% at the threshold level of 0. At this mergethreshold level, the detected face is counted depending on the number of windows clustered on the face. The least number of windows on a face to be considered is three. The smaller windows centre is within the centre of the face and the window with high detection confidence was considered. The number of windows detecting with size is predicted to grow in linearity with the size of the window scanned on a face.

Based on a research by Y-Qing (2014), actual faces will have a high cluster of bounding boxes. As a consequence, faces with less than three bounding boxes are not counted because the system is performed at a MergeThreshold level of zero. Consequently, object with high cluster of windows were counted as a face and used to calculate the detection rate. Figure 4 of image one below shows the face detection performed at a MergeThreshold level of zero.

Figure 4 of Image1 (Gettyimages): Face Detection performed at MergeThreshold level of 0.

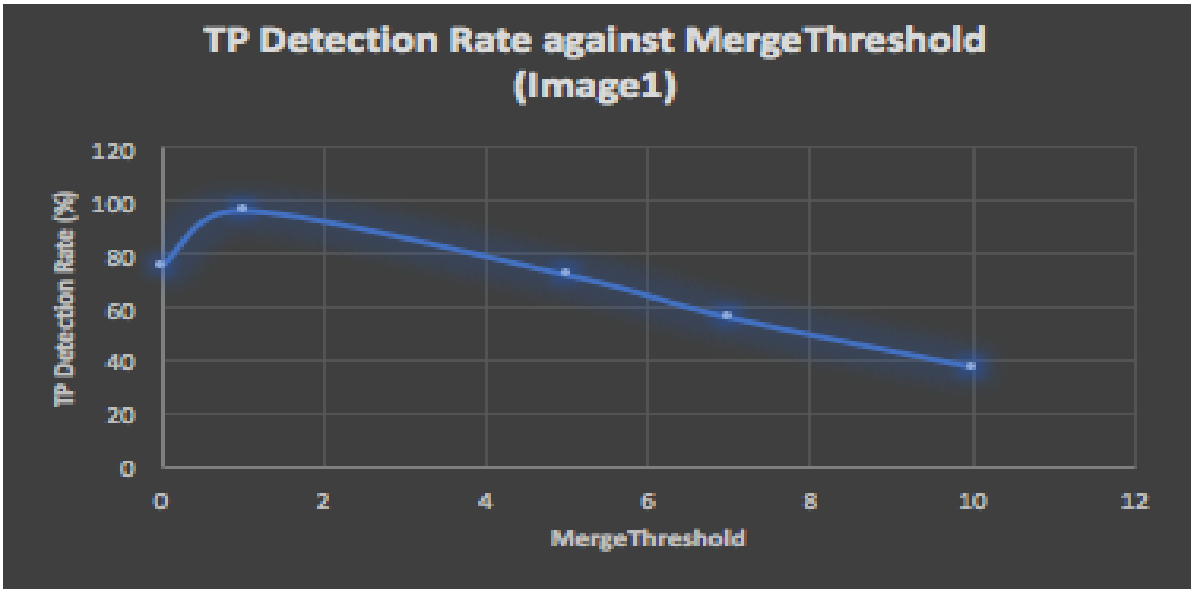


Figure 5: Plot of Detection Rate Against MergeThreshold level for Image 1.

A further experiment was carried out on image 2 and image 3 respectively with the respective parameters used to analyse that of image 1 and the following results were obtained as shown on the tables below figures and tables below.

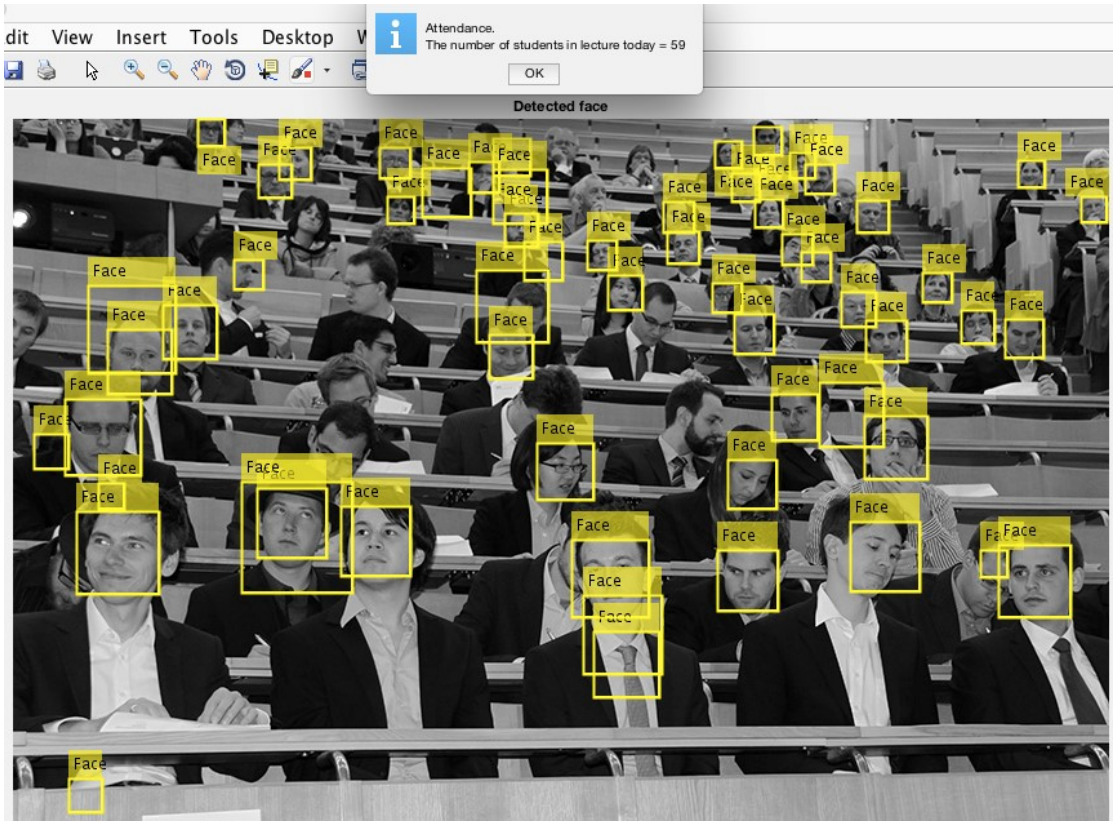


Figure 6 of Image 2 (Gettyimages): Face Detection tested with different parameter

In figure 6 of Image2, the algorithm has been experimented with the following parameters MergeThreshold = 1, window size for face = [20,20] and ScaleFactor = 1.05.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image2	4	Imresize(I)	[]	1.1	29	0	0	49.15
Image2	4	Imresize(I,0.75)	[]	1.1	18	0	0	30.51
Image2	4	Imresize(I,0.5)	[]	1.1	8	0	0	13.55
Image2	4	Imresize(I,0.25)	[]	1.1	0	0	0	0

Table 4A Image 1: Shows Performance Metrics of different Image sizes (using imresize) to ¾, ½ and ¼ of the original image size using default parameters of the Viola Jones Face Detection algorithm.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image2	1	Imresize(I)	[20 20]	1.05	47	12	12	76.6
Image2	1	Imresize(I,0.75)	[20 20]	1.05	33	9	26	55.93
Image2	1	Imresize(I,0.5)	[20 20]	1.05	17	4	42	28.81
Image2	1	Imresize(I,0.25)	[20 20]	1.05	3	1	56	5.08

Table 4B Image 2: Shows Performance Metrics of different Image sizes (using imresize) to ¾, ½ and ¼ of the original image size for Viola Jones Face Detection.

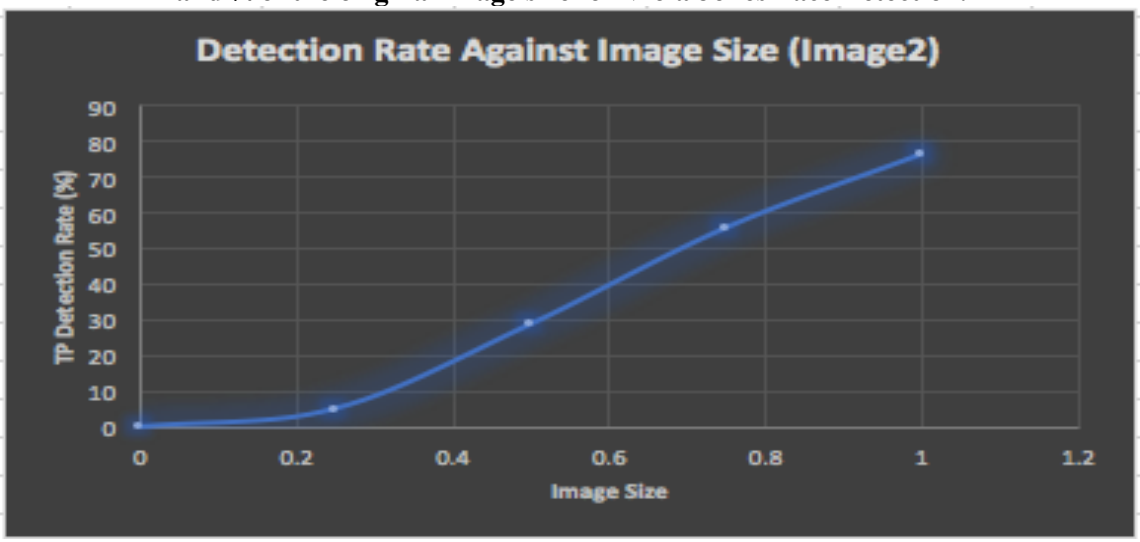


Figure 7: Plot of Detection Rate Against Image Size for Image 2.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image2	0	Imresize(I)	[20 20]	1.25	39	0	20	66.10
Image2	1	Imresize(I)	[20 20]	1.25	40	7	19	67.79
Image2	5	Imresize(I)	[20 20]	1.25	14	0	45	23.72
Image2	7	Imresize(I)	[20 20]	1.25	11	0	48	18.64
Image2	10	Imresize(I)	[20 20]	1.25	8	0	51	13.56

Table 4C Image 2: Shows Performance Metrics of different MergeThreshold Levels for Viola Jones Face Detection ('FrontalFaceCART').

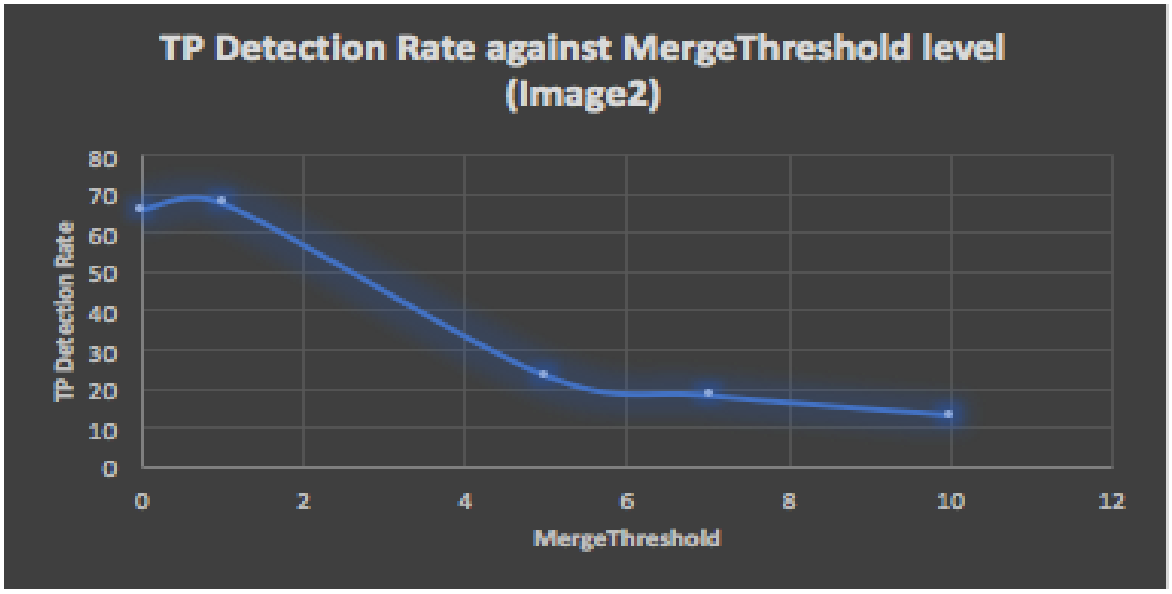


Figure 8: Plot of Detection Rate Against MergeThreshold level for Image 2.

Figure 9 of Image 3 (GettyImages): Face Detection tested with different parameters.

In figure 9 of Image3, the algorithm has been experimented with the following parameters MergeThreshold = 1, window size for face = [20,20] and ScaleFactor = 1.05.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image3	4	Imresize(I)	[]	1.1	20	0	0	22.72
Image3	4	Imresize(I,0.75)	[]	1.1	7	0	0	7.95
Image3	4	Imresize(I,0.5)	[]	1.1	0	0	0	0
Image3	4	Imresize(I,0.25)	[]	1.1	0	0	0	0

Table 5A Image 1: Shows Performance Metrics of different Image sizes (using imresize) to ¾, ½ and ¼ of the original image size using default parameters of the Viola Jones Face Detection algorithm.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image3	1	Imresize(I)	[20 20]	1.05	47	14	41	53.40
Image3	1	Imresize(I,0.75)	[20 20]	1.05	18	4	70	20.45
Image3	1	Imresize(I,0.5)	[20 20]	1.05	5	1	83	5.68
Image3	1	Imresize(I,0.25)	[20 20]	1.05	3	0	85	3.40

Table 5B Image 3: Shows Performance Metrics of different Image sizes (using imresize) to ¾, ½ and ¼ of the original image size for Viola Jones Face Detection.

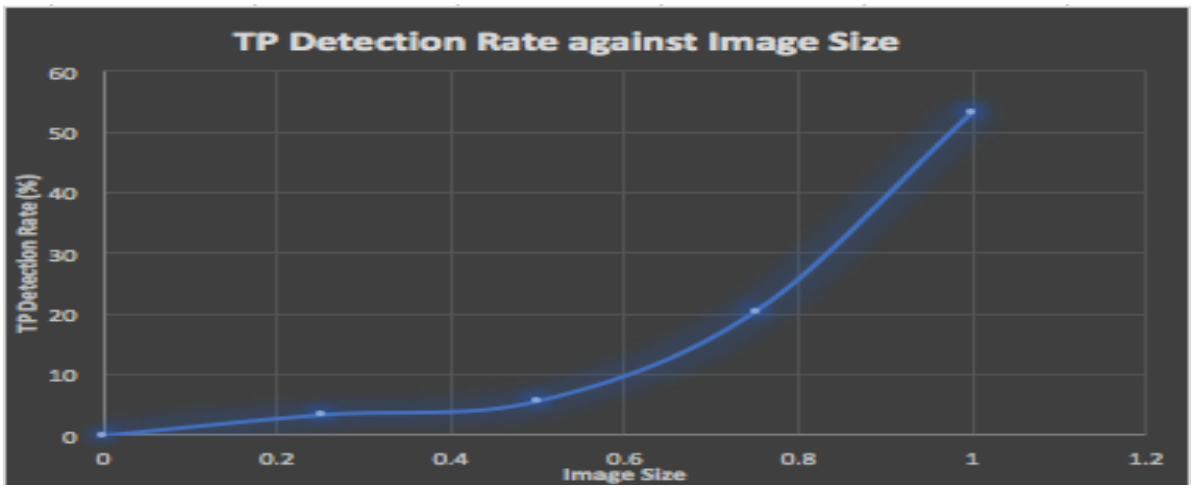


Figure 10: Plot of Detection Rate Against Image Size for Image 3.

Image	MT	ImageSize	Window Size	SF	TP	FP	FN	Detection Rate
Image3	0	Imresize(I)	[20 20]	1.25	20	14	41	22.73
Image3	1	Imresize(I)	[20 20]	1.25	41	4	70	46.59
Image3	5	Imresize(I)	[20 20]	1.25	1	1	83	1.13
Image3	7	Imresize(I)	[20 20]	1.25	1	0	85	1.13
Image3	10	Imresize(I)	[20 20]	1.25	1	0	87	1.13

Table 5C Image 3: Shows Performance Metrics of different MergeThreshold Levels for Viola Jones Face Detection ('FrontalFaceCART').

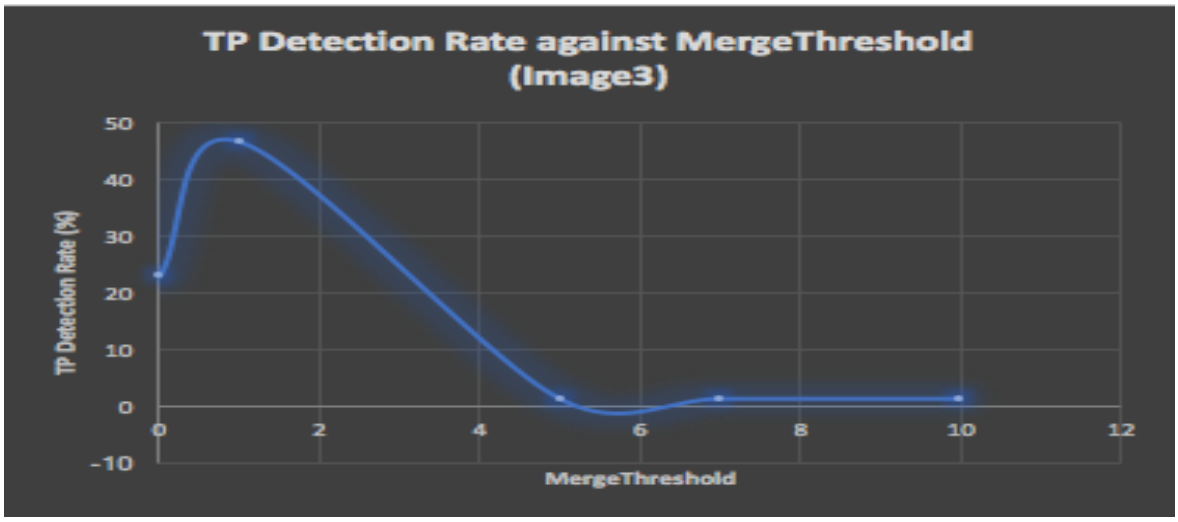


Figure 11: Plot of Detection Rate Against MergeThreshold level for Image 3

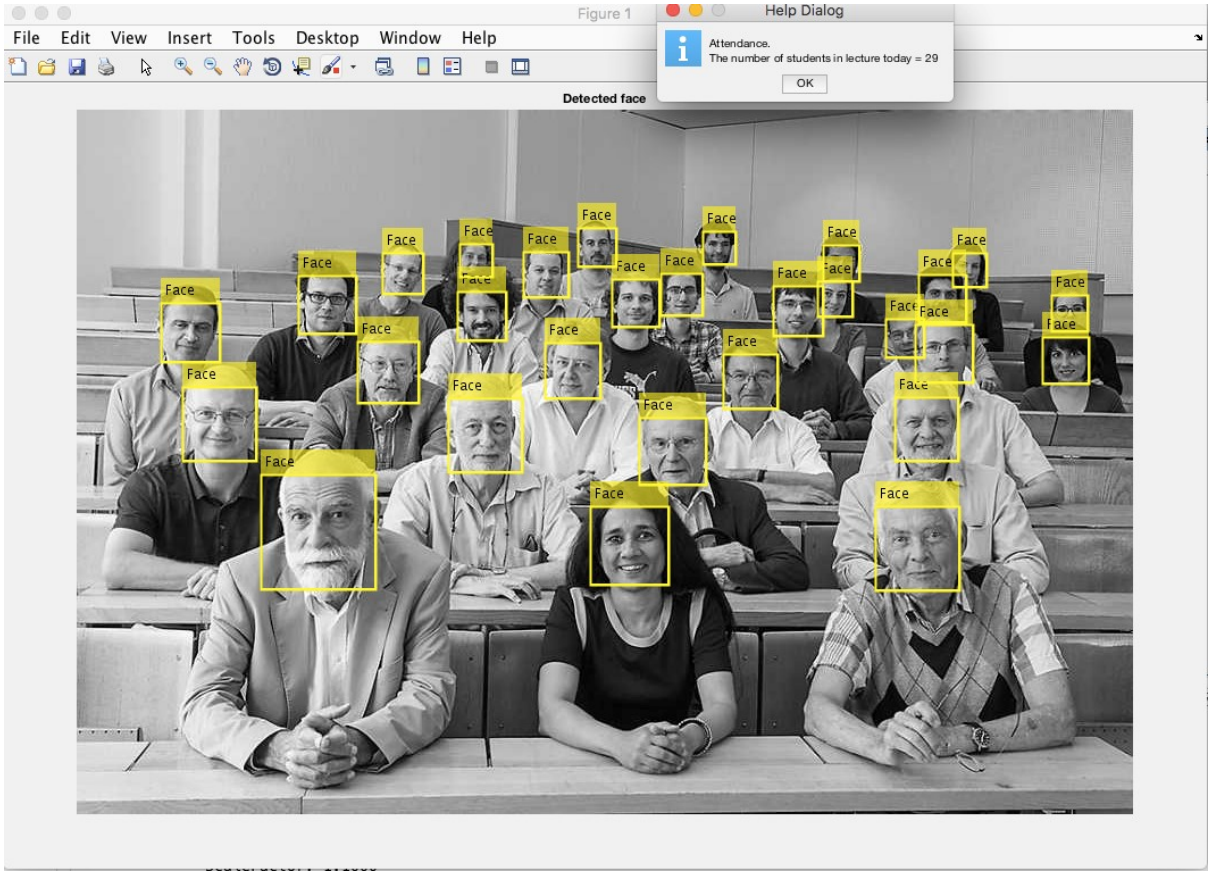


Figure 12: Face Detection of Image1 at original Size.

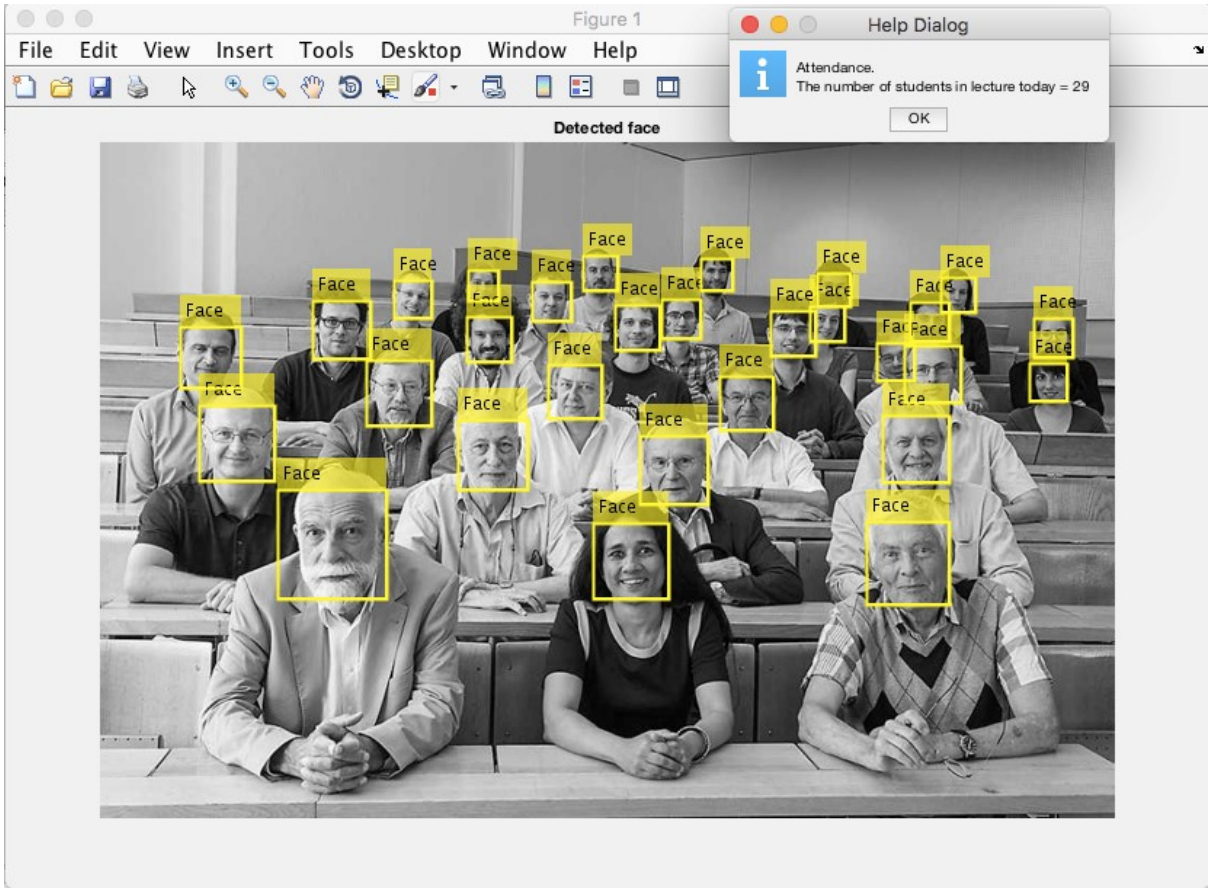


Figure 13: Face Detection of Image1 at ¾ of Original Size

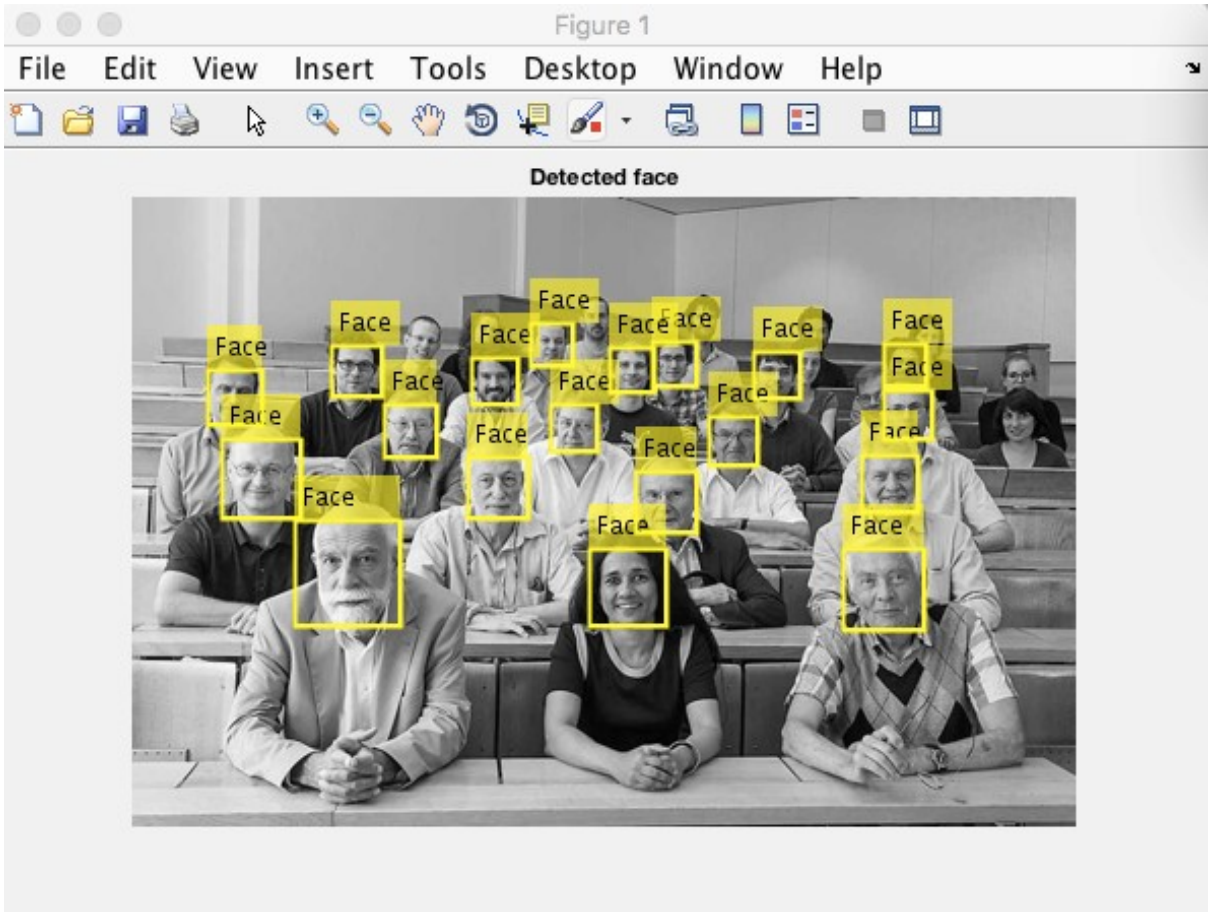


Figure 14: Face Detection of Image1 at ½ of original Size

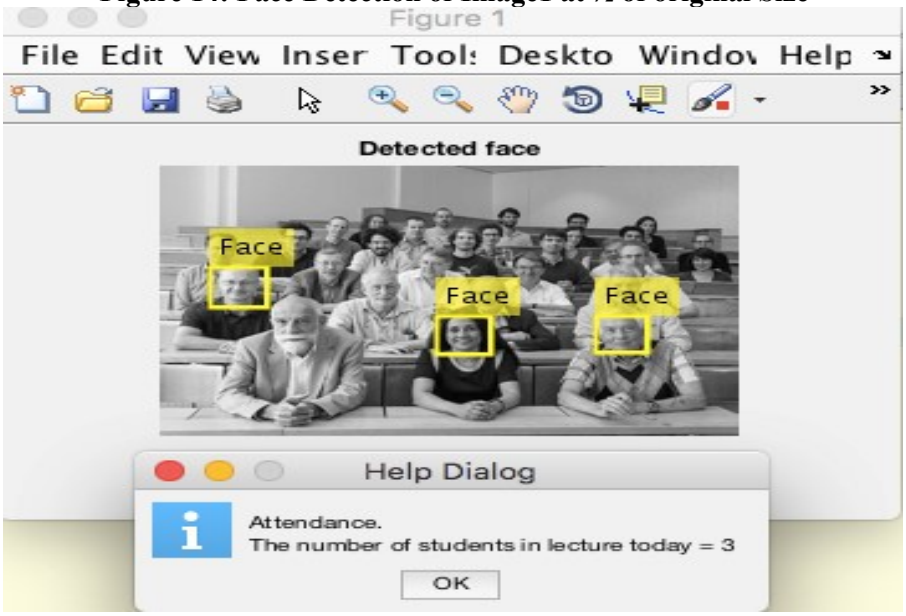


Figure 15: Face Detection of Image1 at ¼ of original Size.

The tables above show three performance metrics obtained from the different parameters carried out during the experiment. Tables 3A,4A and 5A show the performance metrics at different image sizes with respect to the default parameters of the Viola-Jones Face detection algorithm. On Table 3A, the performance of the algorithm with Image1 shows a 100% output. However, this is dependent on the illumination and the frontal pose of each individual in the picture. As the size of the image is halved, a mark drop in the detection rate leaving the TP at total of 19 detected faces and is reduced further as the image size is reduced to ¼. There was no FP and FN obtained. Comparing with Table 3B against the parameters set out in this experiment, the algorithm performs better because it shows a detection rate of 96.55 and this does not reduce drastically as the image size is reduced compared to the values obtained in Table 3B.

The performance of the algorithm on Table 4A shows how weak the algorithm will perform with the default parameters on an Image with a more difficult level. The detection rate is at 49.15% as compared to 76.6 obtained on Table 4B and reduces as the image size decreases. There are no FP and FN obtained using the default parameters. On a more difficult image, as shown on Table 5A, the detection rate is 22.72% against 53.40 using the parameters specified on this paper. Figures 12 to 15 shows the different sizes of the Image1. The size original size of the image is shown in Figure 12. Figure 13 shows the size of the image reduced to ¾ of the original size. From the figure, you can see the reduction in size is not as remarkable as the reduction in size shown in Figure 14 (reduced to half of the original size) and Figure 15 (reduced to ¼ of the original size). This has an impact in the face sizes of the individuals in the images. The faces sizes reduce as the image size reduces leading to a poor detection rate. As a result, the performance of the system depends on the variation of the image resolution.

This shows that the default parameters will only perform better with images of higher illumination, higher resolution and face size of 25x25, enough contrast on the faces, and most of the characteristics outlined for Image1 on Table 1 above.

Tables 3B, 4B and 5B show the performance metrics at different image sizes. The values were obtained by resizing the original image size of Image1, Image2 and Image3 (930X620, 850x570, 1200x630) respectively. Each image was scaled by a factor of 1.05 at window size of [20 20]. On these tables (3B, 4B and 5B), the number of True Positives, False Positives and Detection rates drops as the image size decreases. In contrast, the False Negatives increases as the image size decreases. This has been further illustrated by a plotting the Detection rate against image size as shown in Figures 3, 7 and 10 respectively.

Tables 3C, 4C and 5C show the performance metrics at different MergeThreshold levels. Each threshold level was introduced as the experiment was carried out. The window size [20 20] was scaled at 1.25. From these tables (3C,4C and 5C), we can see that the number of True Positives, False Positives

decreases as the MergeThreshold level is increased at the different levels from 1 to 10. There is no great significant change with a MergeThreshold level of zero. The detection rate decreases with increase in MergeThreshold level. The number of False Negatives increases as MergeThreshold level increases. To further illustrate this, a plot of Detection rate against MergeThreshold levels is shown in figures 5, 8 and 11 respectively

From the analysis, above, based on the figures obtained, the face detection part of this system will take into consideration a MergeThreshold =1, ScaleFactor =1.05, window size of [20 20] with a FrontalFaceCART as the cascade. With the assumption that the lecturer will ask the students to face forward into the camera, and the students being able to cooperate, the performance rate of face detection by the system will be at an approximate of 70%.

The faces that have not been detected were either faces which the ScaleFactor could not correctly determine its resolution as set by the MinSize of the system or did not have enough contrast on them. Some faces are sideways at approximately angles of between 70-90^0 (orientation). Some of the faces appear in half profile which makes it difficult to extract features that makes it easy for the system to detect as a face (Head pose).

Some faces (approximately 18 in image2 and image3 combined), were blocked by their colleagues sitting in front of them or have an obstruction to their face (i.e. hands etc.) which blocks the features for face detection.

Approximately 56 out of 88 faces in image3 and approximately 12 out of 59 faces in image2 are too far at the back of the lecture hall which can also be a resolution problem. The sizes of these faces are approximated at [17 17], [16 16], [19 19], [19x25], which are far below the minimum size specified by the system and MATLAB overall for detection. Moreover, one can conclude that although all the images are of high resolution (930x620, 850x570, 1200x630) it will not meet the requirements for a better performance of the algorithm if reduced remarkably. A reduction in the image size leads to a reduction in the face sizes which in turn leads to poor detection rates. In summary for better performance the image size should not be reduced more than ¾ of its original size.

Furthermore, the difficulty of image3 which makes it difficult to see facial features even from the human point of view could be a more reason why the system could not detect more faces. However, it can be very difficult to decide image quality as it is a contributing factor that determines the evaluation time to detect a face (illumination).

This part of the system has reached a conclusion, based on the findings from the investigation of the different parameters of the Viola-Jones algorithm. The purpose of this part of the system is to detect faces and count the number of faces detected for attendance. The system automatically counts the total number of bounding boxes the system detects as a face. This includes both True positive and False positive respectively. In this regard, the numerical total of the lecture room will be obtained from the total number of bounding boxes detected by the system. This can however, be more or less due to the detection of TP and FP respectively. The lecturer will have to confirm by counting the total number of students to compare with that obtained by the system.

To achieve a good register, the images of the lecture room will have to be taken in rows. By doing this, the first front three rows will be taken and followed in rows of three for detection and subsequently for the recognition phase of the system. Recognition cannot be achieved with tiny face detection as the size will be too small.

To achieve this on a class size of 80, a 16MP phone camera is recommended for a maximum image size of 4920X3264 pixels. The minimum requirements for the parameters of the Viola -Jones algorithm will be FrontalFaceCART, MergeThreshold level of 1, ScaleFactor of 1.05 and window size (MinSize) of [20 20]. The assumption here is that the Camera position is directly facing, the students with respect to their sitting position.

As part of this project, the practical conclusion to achieve the recognition phase is to use a professional Camera which can give a high-resolution image with good illumination. With this, we reduce the class size to 8 rows of 40 students. The assumption here is that, images of each row will be taken with students facing the camera. Each student will have a sitting position with no obstruction. This will enable the face detection part of the system to be able to detect faces at a size that is useful for face recognition and also an accurate numerical count for the class register. However, each row of students sitting position will be considered and a photo for each set of students taken at a time for face detection. The face detection part of the system has been implemented to automatically crop faces detected within a

bounding box. The cropped faces are automatically stored to a folder and will be used as test images for recognition.

Analysis of the Face Recognition part of the system.

The face recognition part of this system uses a 7-States Hidden Markov Model for Face Recognition as mentioned in the literature review. The HMM will require seven blocks of the image vector for a face. The seven segments of a face are shown in the figure below.

Figure 16: Seven Segments of the face.

The image above is of size 112X92 which is the size recommended for this part of the system. The different lines represent the sections of the facial features of the face. The face detection part is set to resize cropped images to the recommended size as stated. Therefore, to achieve this, the image must be of good quality with enough illumination clearly showing all the features that will be extracted as individual vector blocks for quantization. As facial features, may vary over time, the choice to use the 7-State HMM was to include chin (including face hair underneath the chin) and eyebrows. However, the datasets used for this project is taken from Faces in the Wild which include images taken without consideration of illumination, size, resolution, contrast etc. The images represent faces in real-life in a true natural environment which are stored in JPG format. The experiment conducted in this section and the analysis will give the reader a realistic picture of how the face recognition system will work within the classroom.

In this section, the investigation of the face recognition part is focused on conducting experiments based on image resolution and it's the impact on the recognition accuracy. These experiments, will investigate the performance of the system with variations in image resolution. The quality of an original image is always different from a compressed image or resized image, as some of image information is always lost during compression or resizing. This will normally bring about a change in the resolution of the image. Because the system discussed on this paper are in different phases (face detection through face recognition) where face size if priority to achieve the purpose, it is good to consider face size for analysis. The first phase will aim to achieve face sizes of approximately 64x80 detected in a bounding box of size 109x109 on an image captured with a professional camera. The second phase will recognise images of size 92x112. To achieve this, the images will be cropped and resized to 92x112. However, the scale of resizing or cropping determines the quality. The dataset of pictures from Faces in the Wild has images of sizes 250x250 and resizing these pictures without cropping will have a different quality to images of the same set that are cropped. Therefore, before moving to the analysis phase, two datasets were created with Preview. Database1 images are resized to [92 112] (width height) and the Database2 the images are cropped to size [92 112]. All of these pictures are taken from the same database (faces in the wild). Both Database1 and Database2 have images with natural background. Database1 has 52 subjects with a total of 520 images and Database2 has 40 subjects with a total of 384 images. Not all the faces are front facing and some of the subjects have 9 images instead of 10. Some of the images have other faces on the background. These are images of celebrities taken in conferences, ceremonies etc. The idea is to give the choice for the user to either resize the pictures or crop the pictures to the size required on this paper. This is because the Face detection part of the system is set to crop pictures at a size [92 112] for recognition. Also, we do not know the sizes of pictures stored on the Kingston University student database.

The two datasets were trained and tested against images of subjects in their respective datasets (Database1 and Database2). During the test, not all the test images of all the subjects tested, were correctly matched. The figures below show images that have been correctly matched and images that have been mismatched. The images are of the same dataset as their respective input images but not the images that have be trained. The algorithm, has been implemented to display the name of the image

found. It doesn't matter if the image is a correct match or a mismatch, the name will tell the user if it is correct.

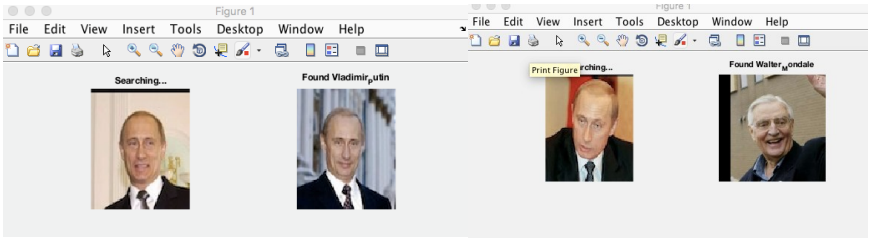


Figure 17: Match of Image from Database1 and Mismatch of the same image on the same Database.

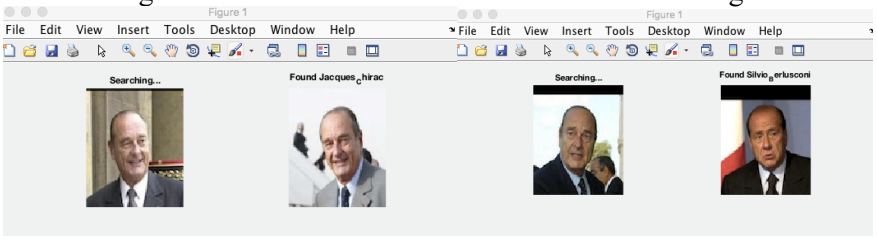


Figure 18: Match of Image from Database1 and Mismatch of the same image on the same Database.

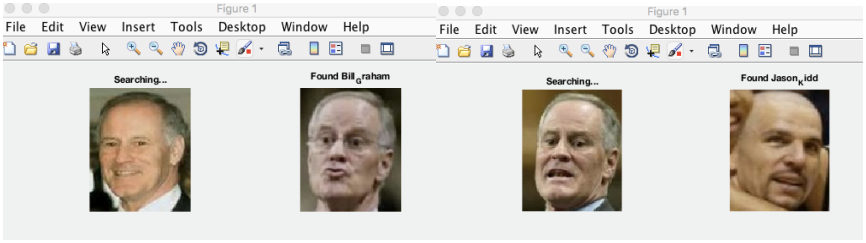


Figure 19: Match of Image from Database2 and Mismatch of the same image on the same Database.

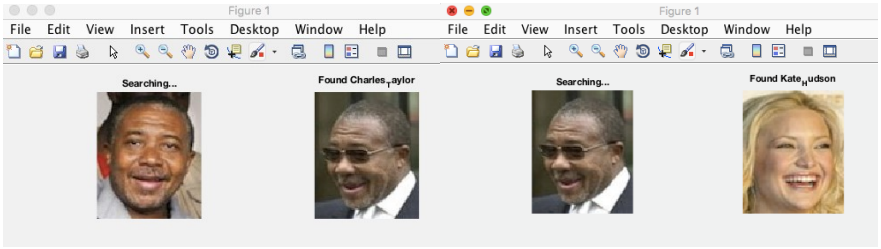


Figure 20: Match of Image from Database2 and Mismatch of the same image on the same Database.

In order to analyse the system, five images are used during the training and the remaining number of images for testing. The following performance metrics have been taken into consideration. True Positive (TP) here means correctly matching individuals whose photos are on the database. False positive (FP) means wrongly matching individuals on the database. The recognition rate is calculated based on the TP and FP obtained from the system. However, because the system extract features in blocks, it has been analysed using the various sizes [92 112], [84 69] and [56 46] respectively. This is because the blocks are extracted in a sequence of overlapping blocks as a patch size of face height (112) and face width (92) is passed through the sliding window from top to bottom. The computational time varies depending on the size of the image. However, this can have an impact on the recognition rate of the system.

Database	Number of Subjects	Number of images tested	Image Size (I,[p.face_height face_width])	Recognition Rate (%)
Database1	52	260	[112 92]	67.3077%
	52	260	[69 84]	69.2308%
	52	260	[46 56]	73.0769%
Database2	40	268	[112 92]	63.6364%

	40	268	[69 84]	64.3939%
	40	268	[46 56]	65.1515%

Table 6. Recognition rate of different databases based on different image sizes used for feature extraction during the training process.

From the table, the recognition rate of the on the two datasets is much better when the size of the image has been halved. We have better recognition rates with the first dataset compared with the second dataset. However, when the image size is reduced, there is less computational complexity and memory consumption which turns to increase recognition rate. This system on this paper is based on Hidden Markov Model and SVD coefficients. The SVD coefficients are implemented for robust feature extraction. The coefficients are used as features during the training process instead of gray values in the sampling blocks. To avoid the curse of dimensionality and influence the recognition rate, we take into consideration the choice on the parameters values (block overlap and window height). The sampling window height has been set to five pixels high, with the overlap size set to 4 pixels. The overlap size of two consecutive blocks influences the recognition rate and this relies mostly on the percentage overlap in the vertical direction. So, with a width of 46 and height of 54, we have an observation vector composed of 54x46 blocks containing 230 pixels and 42 blocks. This shows us how long the observation vectors will be. We should note here that, reducing an image results in great loss of image quality, hence the reason for using SVD coefficients for feature extraction. Therefore, making the right choice to achieve high recognition rate without so much loss in image quality is a bit tricky. It is therefore safe to say that; the size should not be reduced more than 46x54 as this will result in great loss of image quality and there will be a drop in the recognition rate.

The dataset with highest recognition rate will now be used to analyse the system with the performance metrics mentioned above.

The table below shows a count of True Positives and False positives obtained during while testing the accuracy of the system.

Range	Count True Positive	Count False Positive
0-9	0	0
40-49	7	0
50-59	35	7
60-69	77	16
70-79	69	16
80-89	20	0
90-99	10	0
Total	218	39

Table 7: Range Count of True Positive and False Positives.

The values obtained on this table were plotted on a graph to further illustrate the performance of the system. This can be seen on the figure below.

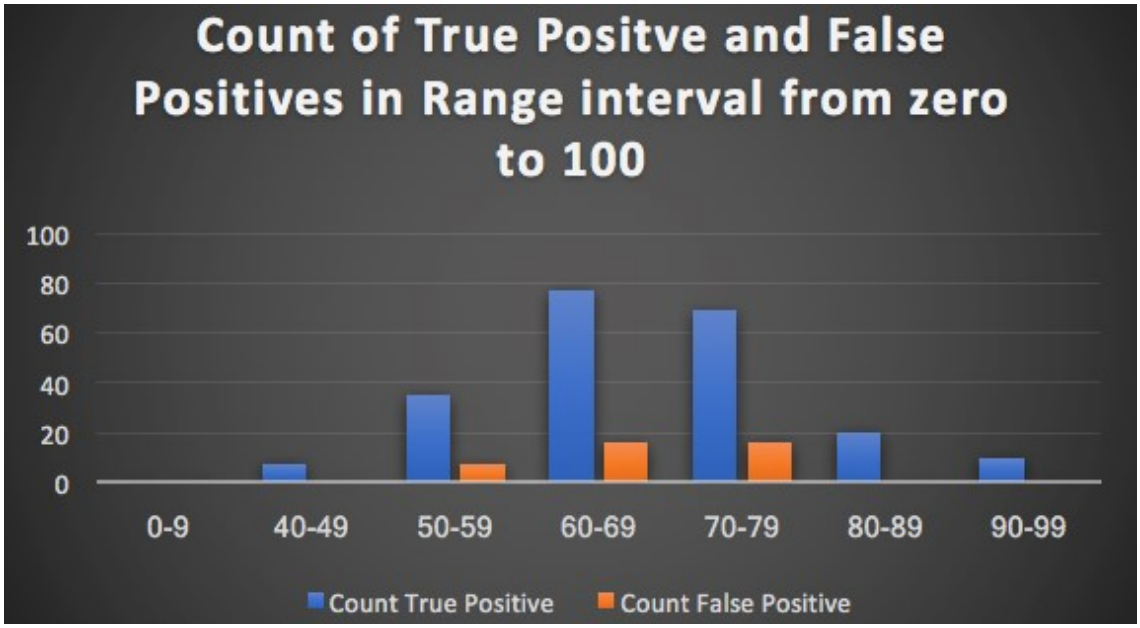


Figure 21: Count of True Positive and False Positive.

From the analysis, the images tested with each subject had at least two recognitions. The number of subjects with at least two recognitions were fewer. Other subjects had three to five recognitions. From the above illustrations, it can be seen that at a percentage range of 60-69, the system will perform at its best. Because the system in this report depends on transmission and emission probability matrix, we use the BaumWelch Algorithm to estimate these probability matrices during the training. The maximum log probability of the transition and emission probabilities of the trained model is generated to obtain a random sequence of states. With these sequence of states, the most likely paths through the model specified by these probabilities is the probable state and calculated in percentage. This explains why carrying out a different test will still obtain the same results but the percentage range at which the system will perform best will differ slightly. However, this will not be very far from a range of 60-69 or 70-79 as it has the highest cluster of recognition per image.

Also, from Table 7 above, we can calculate the precision or positive predictive value (PPV) or False discovery rate (FDR) as a percentage.

Where;

Precision or PPV = (TP/TP+FP)

FDR = FP/FP+TP = 1-PPV

The PPV for this experiment is 83.84% and the FDR is 16.16%. The high positive predictive value (PPV = 83.84%) indicates that many images of each subject tested during the recognition procedure are true positives. However, with the false discovery rate at 16.16%, it suggests that follow up is not very much needed in case a subject has only two or three matches against the five tested.

Therefore, it is safe to say that each subject will score approximately 60% during the test with at least three images recognised against images on the database. With this score and confirmation of the name of the subject of the image and input image side by side, the lecturer will be confident of the presence of a student during the lecture.

Analysis of the complete system (Face Detection followed by Face Recognition Process)

As already mention earlier, the system should be able to recognise images detected by the face detector. In this case, output images from the face detection part of the system are used as inputs for face recognition.

At most five or less images of each subject already stored on a database will be trained for recognition. This is to carry out a rationality check and avoid cheating the system.

The system has been implemented to crop faces of bounding box around the face of size 109x109 with the face size of approximately 64x80. Based on this finding, the image below has been considered to test the system. This image gives a typical example the face sizes we want to achieve in the front row in sets of three or four for detection by the face detection part. Although, we may have 10 or 15 students on the front row, we can only achieve this face size by taking images in the order previously stated.

This will however achieve good quality in resolution. All the faces on the image below are on Database1 of our datasets, with 52 subjects of 10 images each. These faces will be used in the face detection part of the system for detection, cropping and recognition.

Figure 21: Heads of States traditional photo at the G8 Summit. (Gettyimages).

The faces on the image above were detected and cropped by the face detection part of the system. The faces were detected by the system as shown below. The cropped faces were stored in a folder. With the system already trained, the faces are used for recognition.

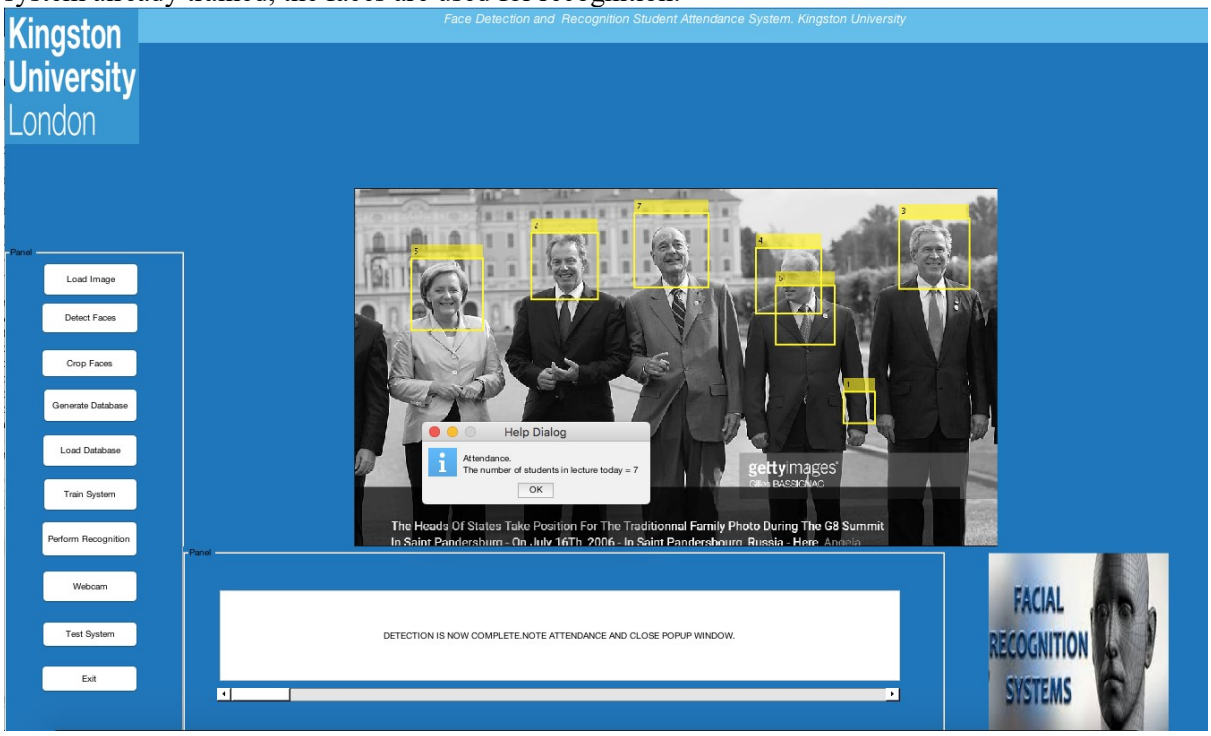


Figure 22: Face Detection of image shown in Figure 21 above.

Three of the five faces were correctly matched and two were mismatched. This has been shown below.

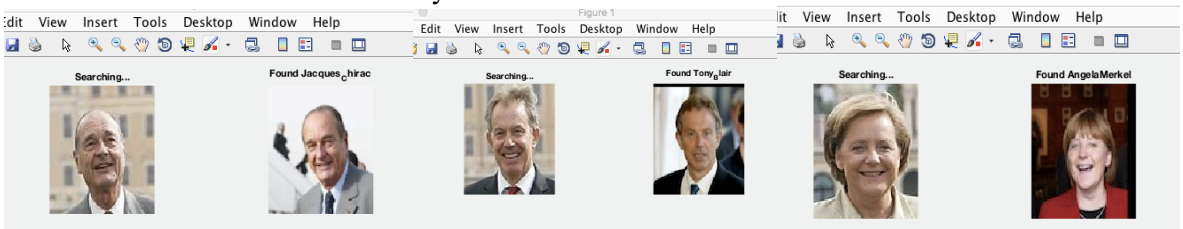


Figure 23: Correct Match of cropped faces against faces of subjects on database.

The faces above show an 80% match for the Jacque Chirac, 70% match for Tony Blair and 50% match for Angela Merkel.

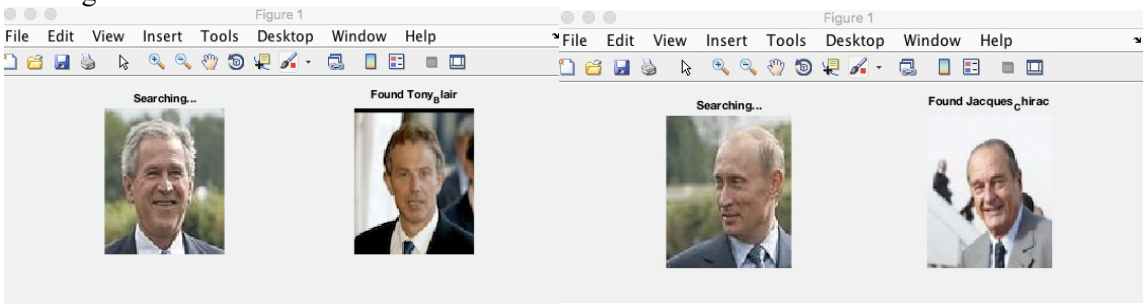


Figure 24: Mismatch of cropped faces against faces of subjects on database.

The faces above show a 30% match for George Bush with Tony Blair and 40% match of Putin with Jacques Chirac.

From this result, there is a potential that approximately more than 60% of the overall students taken in sets of three or four row by row will be correctly matched by the system. However, this is only possible if a face size of 64X80 can be achieved and resized or cropped to fit 92X112 by the face detector.

Conclusion:

The analysis carried out in the various sections has gone through a series of experiments and some conclusions. The experiments were centred mostly on the image size. The aim was to get the reader to understand how pixel value can affect face detection and face recognition as a whole. From this analysis, the utilised approach of image size has yielded satisfactory result. This has been shown on tables 2,3B,4B,5B,6 and 7. From this, the structure to follow to achieve these results has been outlined in both conclusion part of the face detection and face recognition.

Moreover, we can now confidently apply our system based on the following.

- Images will be captured row by row to achieve a face size of 64x80. The overall image size to achieve such a face size should be approximately 1024×570.
- Face detection will be applied, and the detected faces cropped and saved to a folder. The total number of images saved in the folder will actuate the numeric attendance during the lecture.
- The images in the cropped folder will be matched by the recognition part to faces already stored and trained on the database. This will facilitate a confirmation of attendance for a particular student. It will also enable the lecturer to take full control of the class by calling each student by name when need arises.

From this, I can conclude that a class attendance system based on face detection and face recognition can be achieved at this level. Further research and experiments will be carried out on future work to improve the system.

CHAPTER SEVEN

Conclusion

The entire project has been developed from the requirements to a complete system alongside evaluation and testing. The system developed have achieved its aim and objectives. The client was happy with the overall performance of the system. However, though some challenges were encountered during implementation, they were addressed and implemented.

Also, future work and strategies on how to improve the system are further in this section.

The Problems:

During the development of the system, some problems were encountered and have been discussed here. The experiments conducted after the implementation of the system required some changes. During the initial phase of the evaluation, the aim was to change the parameters of the Viola-Jones algorithm to meet the objectives. However, it was proven that, achieving on this objective with these parameters will lead to a big failure on the second part of the system. The conclusion to set the parameters of this part of the system based on very small class size was due to the failures obtained from the recognition part of the system. The size of the image is very important in face recognition as every pixel count. Resizing an image leads to a loss in pixels and highly influenced the performance of the system poorly. The evaluation results showed the face detection part of the system to perform at approximately 70% detection rate which was not as great as compared to that obtained by Viola and Jones (2001). Also, I have learned that face detection algorithms work on changes to illumination, image resolution etc. as already mention and is still an ongoing research to images of uncontrolled backgrounds. However, the next challenge for future work and research is to implement a system that can achieve a high performance on such images.

The evaluation of the face recognition part of the system produced results which as not as expected. The evaluation showed that the recognition part can achieve approximately 60% recognition rate based on the image resolution. With a poor image resolution, there is a high chance the system will fail. Furthermore, in order to achieve this result, the user must make sure image is of the right resolution discussed on this report. Some people are not good at following instructions. This may lead to poor image quality and will make the system perform poorly. However, for the purpose of evaluating the system against images taken in controlled backgrounds (Yale Database), there is a high chance of excellent result. The use of images taken from faces in the wild to evaluate the database was a great idea suggested by my supervisor. This has given us a rough idea on the performance rate at approximately 70%. This is different from that seen in the literature review for a similar system which performed at 97% on one of the datasets used in research.

Also, one of the reasons for the choice of the platform used implementation could have influenced the poor performance as I do not really know the language at my best.

The algorithm used to determine the percentage probability generates different percentage scores each time. This is because, the percentage generated is a percentage to show to what extent the most likely sequence of state agrees with the random sequence. The random sequence will be that of the input image and the most likely sequence is that from the output image. Although it does not change the overall percentage of recognition, it was not possible to tell the user at what percentage they could decide a face is a face. The only way out was to carry out a test on all five input images and at least with three matches, they user can confirm a face, based on the output match displayed side by side.

The performance of the system has impacted the reliability of the system. Because it is still an ongoing research area, the system will not be available for use at the end of the project. However, it can be used

for research purposed by the supervisor and experimented in a lecture room before approval.
It was not possible to deploy the system as a standalone as the Matlab SDK Compiler was not available on a Matlab version with Vision support packages.

Future Work and Improvement:

A more detailed research is needed on a project as such. The methods used could be combined with others to achieve great results. Different methods have been implemented in the past according to the literature review.
The use of HMM with other feature extraction methods can be implemented and tested. This will need more time as it is only a trial that will be made taking into consideration the method that already exist in order to have a complete new idea.

A login functionality would be implemented on the system for security purposes.
The system will be deployed as a standalone which could be used by other schools. This will now be done using the MATLAB App builder.
Data confidentiality is very important. At the start of each school year, the images of new students are taken and stored by the university. Each student will have the right to be informed about the use of their faces for a face recognition attendance system. This must be in line with the government laws on ethical issues and data protection laws and rights. The students will have to consent to their images used for the purpose of attendance.

The system that has been delivered and should only be used for experimental purpose as it is not completely reliable.

Critical Review

Here, I will discuss the challenges faced during the implementation of the project.

Challenges:

The major challenge during the implementation of this project was learning Matlab from scratch. I started learning Matlab from the course Computer Vision, Graphics and Image processing. I had a clash with my core module which stopped me from carrying on with the course and later focused on YouTube tutorial in image processing, face detection and face recognition in MATLAB. Other alternatives like OpenCV and digiKam which is written in C++ could have been exploited but was not due to time constraint and the nature of the project and other courses.
Agreeing on the set of objectives with the client was not easy. Like all other clients, my client was too ambitious but at the end, we settled for objectives that will meet the solution to the system.
Learning how to implement Face Detection in MATLAB was the first challenge but with the help of MATLAB webinars and YouTube tutorials, I was able to overcome this.
Another challenge encountered was to set the bounding box to a size that will not be too small to resize to the size required for face recognition. Moreover, getting these bounding boxes and putting them in the right position relative to the position of subjects in the image was not easy.
Learning how to implement Face Recognition using PCA with SVD was another challenge faced on its own but served as a foundation. This gave me the enthusiasm to carry on with the project and implement it with Hidden Markov Model the discussed on this report.
Also, implementing Hidden Markov Model for face recognition was not as easy as I thought, but I finally did with the help from Mathworks.com. With reference to a similar solution, questions and answers from other researchers, I was able to carry on with the implementation.
Evaluation and testing was not an easy task. I had to manually test the system, calculate the percentage of recognition against each subject. Also, the changing parameters for the face detection part and making a decision to the final parameter for the system was more of a challenge.
Using GUIDE for the GUI has been a challenge as the documentation has changed and some functionalities removed completely with reference to the old versions.
Time management has not been great. This has resulted from the weekly medical appointments due to ill health and disability. Also, there has been other coursework and datelines which carry almost equal amount of work. Furthermore, developing a real-life project as one of my module assessment has contributed to time constraint as I am SCRUM master of the project.

Reflections on the Project:

The entire project has not been managed at its best. This is because the project was more complicated than I initially thought. Although, I have acquired an enormous amount of knowledge during the research and implementation, I cannot be 100% sure the implementation of this project is at its best. This just due to the ongoing research and will have to learn other languages and more time carry out an in-depth research of this project. Also, I believe without the help of MATLAB and other ways of implementing the same project, it would have been a difficult journey overall. Moreover, throughout this project lifecycle, I have developed more interest in face detection and recognition, based on research carried out during this project. However, the recommendation for the system will be to use only for experimental purpose pending future work and improvements.

Finally, I will work more on this project by taking it to the next level, conducting a deep research with implementation using other languages. This will be a complete research on its own and more time will be focused on the project as they will be no other modules and commitments involved.

BIBLIOGRAPHY

Alpaydin, E. (2014) *Introduction to Machine Learning.* 3rd ed edn. Cambridge: Cambridge: The MIT Press.

Anthony, S. (2014) *Facebook's facial recognition software is now as accurate as the human brain, but what now?.* Available at: http://www.extremetech.com/extreme/178777-facebooks-facial-recognition-software-is-now-as-accurate-as-the-human-brain-but-what-now (Accessed: 09/01/2018).

Baseer, K. (2015) 'A Systematic Survey on Waterfall Vs. Agile Vs. Lean Process Paradigms', *I-Manager's Journal on Software Engineering,* 9 (3), pp. 34-59.

Belaroussi, R. and Milgram, M. (2012) 'A comparative study on face detection and tracking algorithms', *Expert Systems with Applications,* 39 (8), pp. 7158-7164.

C, R.,Kavitha and Thomas, S.,Mary (2011) 'Requirement Gathering for small Projects using Agile Methods', *IJCA Special Issue on "Computational Science - New Dimensions & Perspectives,* pp. 122-128.

Carro, R. C., Larios, J. -. A., Huerta, E. B., Caporal, R. M. and Cruz, F. R. (2015) 'Face recognition using SURF', *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* 9225 pp. 316-326.

Castrillón, M., Déniz, O., Hernández, D. and Lorenzo, J. (2011) 'A comparison of face and facial feature detectors based on the Viola–Jones general object detection framework', *Machine Vision and Applications,* 22 (3), pp. 481-494.

Cheng, Y. F., Tang, H. and Chen, X. Q. (2014) 'Research and improvement on HMM-based face recognition', *Applied Mechanics and Materials,* 490-491 pp. 1338-1341.

Da Costa, Daniel M. M., Peres, S. M., Lima, C. A. M. and Mustaro, P. (2015) *Face recognition using Support Vector Machine and multiscale directional image representation methods: A comparative study.* Killarney,Ireland. Neural Networks (IJCNN), 2015 International Joint Conference on: IEEE.

Dagher, I., Hassanieh, J. and Younes, A. (2013) *Face recognition using voting technique for the Gabor and LDP features.* Dallas, TX, USA. Neural Networks (IJCNN), The 2013 International Joint Conference on: IEEE.

Dhanaseely, A. J., Himavathi, S. and Srinivasan, E. (2012) 'Performance comparison of cascade and feed forward neural network for face recognition system', pp. 21.

Elbeheri, A. (2016) *The Dynamic Systems Development Method(DSDM)-Agile Methodology.* Available at: https://www.linkedin.com/pulse/dynamic-systems-development-method-dsdm-agile-alaa (Accessed: 28/03/2018).

Feraud, R., Bernier, O., Viallet, J. E. and Collobert, M. (2000) 'A fast and accurate face detector for indexation of face images', *Automatic Face and Gesture Recognition, 2000.Proceedings.Fourth IEEE International Conference on,* pp. 77-82.

GettyImages (2017) *Lecture Hall.* Available at: https://www.gettyimages.co.uk/photos/lecture-hall?mediatype=photography&page=5&phrase=lecture%20hall&sort=mostpopular (Accessed:

25/01/2018).
Hadizadeh, H. (2015) 'Multi-resolution local Gabor wavelets binary patterns for gray-scale texture description', *Pattern Recognition Letters,* 65 pp. 163-169.

Hiremath, P. S. and Hiremath, M. (2014) '3D Face Recognition based on Radon Transform, PCA, LDA using KNN and SVM', *International Journal of Image,* 6 (7), pp. 36-43.

Hjelmås, E. and Low, B. K. (2001) 'Face Detection: A Survey', *Computer Vision and Image Understanding,* 83 (3), pp. 236-274.

Jadhav, D. V. and Holambe, R. S. (2010) 'Rotation, illumination invariant polynomial kernel Fisher discriminant analysis using Radon and discrete cosine transforms based features for face recognition', *Pattern Recognition Letters,* 31 (9), pp. 1002-1009.

Jafri, R. and Arabnia, H. ((2009).) 'A Survey of Face Recognition Techniques.', *Journal of Information Processing Systems,* 5 (2), pp. 41-68.

Jeong, G. and Choi, S. (2013) 'Performance evaluation of face recognition using feature feedback over a number of Fisherfaces', *IEEJ Transactions on Electrical and Electronic Engineering,* 8 (6), pp. 541-545.

Kashif, M., Deserno, T. M., Haak, D. and Jonas, S. (2016) 'Feature description with SIFT, SURF, BRIEF, BRISK, or FREAK? A general question answered for bone age assessment', *Computers in Biology and Medicine,* 68 pp. 67-75.

Leigh-Pollitt, P. (2001) *The Data Protection Act explained.* 3rd ed. edn. London: London : The Stationery Office.

Lemley, J.Bazrafkan, S. and Corcoran, P. (2017) 'Deep Learning for Consumer Devices and Services: Pushing the limits for machine learning, artificial intelligence, and computer vision' *Consumer Electronics Magazine, IEEE,* 6 (2), pp. 48-56. 10.1109/MCE.2016.2640698.

Lenc, L. and Král, P. (2014) 'Automatic face recognition approaches', *Journal of Theoretical and Applied Information Technology,* 59 (3), pp. 759-769.

Li, C., Tan, Y., Wang, D. and Ma, P. (2017) 'Research on 3D face recognition method in cloud environment based on semi supervised clustering algorithm', *Multimedia Tools and Applications,* 76 (16), pp. 17055-17073.

Li, S. Z.Rong Xiao, S. Z.Li, Z. Y. and Hong, J. Z. (2001) 'Nonlinear mapping from multi-view face patterns to a Gaussian distribution in a low dimensional space' *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001.Proceedings.IEEE ICCV Workshop on,* pp. 47-54. 10.1109/RATFG.2001.938909.

Linna, M., Kannala, J. and Rahtu, E. (2015) 'Online face recognition system based on local binary patterns and facial landmark tracking', *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* 9386 pp. 403-414.

Lowe, D. G. (1999) 'Object recognition from local scale-invariant features', *Computer Vision, 1999.the Proceedings of the Seventh IEEE International Conference on,* 2 pp. 1150-1157.

Marciniak, T., Chmielewska, A., Weychan, R., Parzych, M. and Dabrowski, A. (2015) 'Influence of low resolution of images on reliability of face detection and recognition', *Multimedia Tools and Applications,* 74 (12), pp. 4329-4349.

Mathworks (2017) *Detect objects using the Viola-Jones algorithm.* Available at: https://uk.mathworks.com/help/vision/ref/vision.cascadeobjectdetector-system-object.html?s_tid=srchtitle (Accessed: 05/02/2018).

Mayank Chauhan , Mukesh Sakle. (2014) 'Study & Analysis of Different Face Detection Techniques', *(IJCSIT) International Journal of Computer Science and Information Technologies,* 5 (2), pp. 1615-1618.

Miar-Naimi, H. and Davari, P. (2008) ' A New Fast and Efficient HMM-Based Face Recognition System Using a 7-State HMM Along With SVD Coefficients ', *Iranian Journal of Electrical and Electronic Engineering.IJEEE,* 4 (1), pp. 46-57.

Ming-Hsuan Yang, Kriegman, D. J. and Ahuja, N. (2002) 'Detecting faces in images: a survey', *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* 24 (1), pp. 34-58.

Mittal, Y.Varshney, A.Aggarwal, P.Matani, K. and Mittal, V. K. (2015) 'Fingerprint biometric based Access Control and Classroom Attendance Management System' *India Conference (INDICON), 2015 Annual IEEE,* pp. 1-6. 10.1109/INDICON.2015.7443699.

Modi, M. and Macwan , F. (2014) 'Face Detection Approaches: A Survey.', *International Journal of Innovative Research in Science, Engineering and Technology,* 3 (4), pp. 11107-11116.

Mohamed, A. S. S., Ying Weng, S. S., Ipson, S. S. and Jianmin Jiang, S. S. (2007) 'Face detection based on skin color in image by neural networks', *Intelligent and Advanced Systems, 2007.ICIAS 2007.International Conference on,* pp. 779-783.

Mohamed, A. S. S., Ying Weng, S. S., Ipson, S. S. and Jianmin Jiang, S. S. (2007) *Face detection based on skin color in image by neural networks.* Kuala Lumpur, Malaysia. Kuala Lumpur, Malaysia:

Moutzouris, A.Martinez-Del-Rincon, J.Lewandowski, M.Nebel, J. and Makris, D. (2011) 'Human pose tracking in low dimensional space enhanced by limb correction' *Image Processing (ICIP), 2011 18th IEEE International Conference on,* pp. 2301-2304. 10.1109/ICIP.2011.6116100.

Mundschenk, N. A., Miner, C. A. and Nastally, B. L. (2011) 'Effective Classroom Management: An Air Traffic Control Analogy', *Intervention in School and Clinic,* 47 (2), pp. 98-103.

Ojala, T.Pietikainen, M. and Maenpaa, T. (2002) 'Multiresolution gray-scale and rotation invariant texture classification with local binary patterns' *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* 24 (7), pp. 971-987. 10.1109/TPAMI.2002.1017623.

Orceyre, M. J. (1975) 'Data security', *Journal of Chemical Information and Computer Sciences,* 15 (1), pp. 11.

Parmar, D. N. and Mehta, B. B. (Jan-Feb 2013) 'Face Recognition Methods & Applications', *International Journal of Computer Technology & Applications,* 4 (1), pp. 84-86.

Perveen, N., Ahmad, N., Abdul Qadoos Bilal Khan, M., Khalid, R. and Qadri, S. (2015) 'Facial Expression Recognition Through Machine Learning', *International Journal of Scientific & Technology Research,* 4 (8), pp. 91-97.

Peter, U. E.Joe-Uzuegbu, C.Uzoechi, L. and Opara, F. K. (2013) 'Biometric-based attendance system with remote real-time monitoring for tertiary institutions in developing countries' *Emerging & Sustainable Technologies for Power & ICT in a Developing Society (NIGERCON), 2013 IEEE*

International Conference on, pp. 1-8. 10.1109/NIGERCON.2013.6715633.
Pierre, H. (2016) *Dynamic System Development Method(DSDM).* Available at: https://www.slideshare.net/PierreHervouet/heart-of-agile (Accessed: 23/03/2018).

Postma, E. (2002) 'Review of Dynamic Vision: From Images to Face Recognition: S. Gong, S. McKenna & A. Psarrou; Imperial College Press, 2000 (Book Review)', *Cognitive Systems Research,* 3 (4), pp. 579-581.

Rowley, H. A., Baluja, S. and Kanade, T. (1998) 'Neural network-based face detection', *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* 20 (1), pp. 23-38.

Ryu, H., Chun, S. S. and Sull, S. (2006) 'Multiple classifiers approach for computational efficiency in multi-scale search based face detection', *Advances in Natural Computation, Pt 1,* 4221 pp. 483-492.

Saxena, V.Grover, S. and Joshi, S. (2008) 'A real time face tracking system using rank deficient face detection and motion estimation' *Cybernetic Intelligent Systems, 2008.CIS 2008.7th IEEE International Conference on,* pp. 1-6. 10.1109/UKRICIS.2008.4798956.

Shang-Hung Lin. (2000) 'An Introduction to Face Recognition Technology', *Informing Science the International Journal of an Emerging Transdiscipline,* 3 (1), pp. 1-7.

Sharma, H., Saurav, S., Singh, S., Saini, A. K. and Saini, R. (2015) *Analyzing impact of image scaling algorithms on viola-jones face detection framework.* Kochi, India. Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on: IEEE.

Shiwen, L., Feipeng, D. and Xing, D. (2015) *A 3D face recognition method using region-based extended local binary pattern.* Quebec City, QC, Canada. Image Processing (ICIP), 2015 IEEE International Conference on: IEEE.

Smith, S. M. (1995) 'ASSET-2: real-time motion segmentation and shape tracking' *Computer Vision, 1995.Proceedings., Fifth International Conference on,* pp. 237-244. 10.1109/ICCV.1995.466780.

Soewito, B. and Echo Wahana, M. S. (2014) 'EFFICIENCY OPTIMIZATION OF ATTENDANCE SYSTEM WITH GPS AND BIOMETRIC METHOD USING MOBILE DEVICES', *CommIT Journal,* 8 (1), pp. 5-9.

Sommerville, I. (eds.) (2011) *Software Engineering.* 9th edn. USA: Pearson Education.

S. Sharavanan et al, “LDA Based Face Recognition By Using Hidden Markov Model In Current Trends”, International Journal of Engineering and Technology Vol.1(2), 77- 85, 2009.

Sreenivasan, S. and Kothandaraman, K. (2017) 'Predictability with agility: Achieving excellence in software delivery through SPEED', *Global Business and Organizational Excellence,* 37 (1), pp. 6-15.

Sunghoon, K., Youngjin, K. and Seunghyung, L. (2016) '
An Improved Face Recognition based on Scale Invariant Feature Transform (SIFT): Training for Integrating Multiple Images and Matching by Key Point’s Descriptor-Geometry', *Indian Journal of Science and Technology,* 9 (35), pp. 1-11.

Surayahani, S. and Masnani, M. (2010) 'Modeling Understanding Level Based Student Face Classification' *Mathematical/Analytical Modelling and Computer Simulation (AMS), 2010 Fourth Asia International Conference on,* pp. 271-275. 10.1109/AMS.2010.61.

Taigman, Y., Yang, M., Ranzato, M. and Wolf, L. (2014) 'DeepFace: Closing the gap to human-level performance in face verification', *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* pp. 1701-1708.

Tarwani, S. and Chug, A. (2016) 'Agile Methodologies in Software Maintenance: A Systematic Review', *Informatica,* 40 (4), pp. 415-426.

Thai, L. H., Nguyen, N. D. T. and Hai, T. S. (2011) 'A Facial Expression Classification System Integrating Canny, Principal Component Analysis and Artificial Neural Network', *International Journal of Machine Learning and Computing,* 1 (4), pp. 388-393.

Veerapaneni, E.,Jyothi and K, N.,Rao (2011) 'Effective Implementaton of Agile Practices', *(IJACSA) International Journal of Advanced Computer Science and Applications,* 2 (3), pp. 41-48.

Venugopalan, J.Brown, C.Cheng, C.Stokes, T. H. and Wang, M. D. (2012) 'Activity and school attendance monitoring system for adolescents with sickle cell disease' *Conference Proceedings : ...Annual International Conference of the IEEE Engineering in Medicine and Biology Society.IEEE Engineering in Medicine and Biology Society.Annual Conference,* 2012 pp. 2456. 10.1109/EMBC.2012.6346461.

Venugopalan, J., Brown, C., Cheng, C., Stokes, T. H. and Wang, M. D. (2012) 'Activity and school attendance monitoring system for adolescents with sickle cell disease', *Conference Proceedings : ...Annual International Conference of the IEEE Engineering in Medicine and Biology Society.IEEE Engineering in Medicine and Biology Society.Annual Conference,* 2012 pp. 2456.

Vincent, P.Larochelle, H.Bengio, Y. and Manzagol, P. (2008) 'Extracting and composing robust features with denoising autoencoders' pp. 1096-1103. 10.1145/1390156.1390294.

Viola, P. and Jones, M. (2001) 'Rapid object detection using a boosted cascade of simple features', *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 1 pp. I511-I518.

Wang, K., Song, Z., Sheng, M., He, P. and Tang, Z. (2015) 'Modular Real-Time Face Detection System', *Annals of Data Science,* 2 (3), pp. 317-333.

Wang, K., Song, Z., Sheng, M., He, P. and Tang, Z. (2015) 'Modular Real-Time Face Detection System', *Annals of Data Science,* 2 (3), pp. 317-333.

Wang, X., Cai, Y. and Abdulghafour, M. (2015) *A comprehensive assessment system to optimize the overlap in DCT-HMM for face recognition.* Dubai, United Arab Emirates. Innovations in Information Technology (IIT), 2015 11th International Conference on: IEEE.

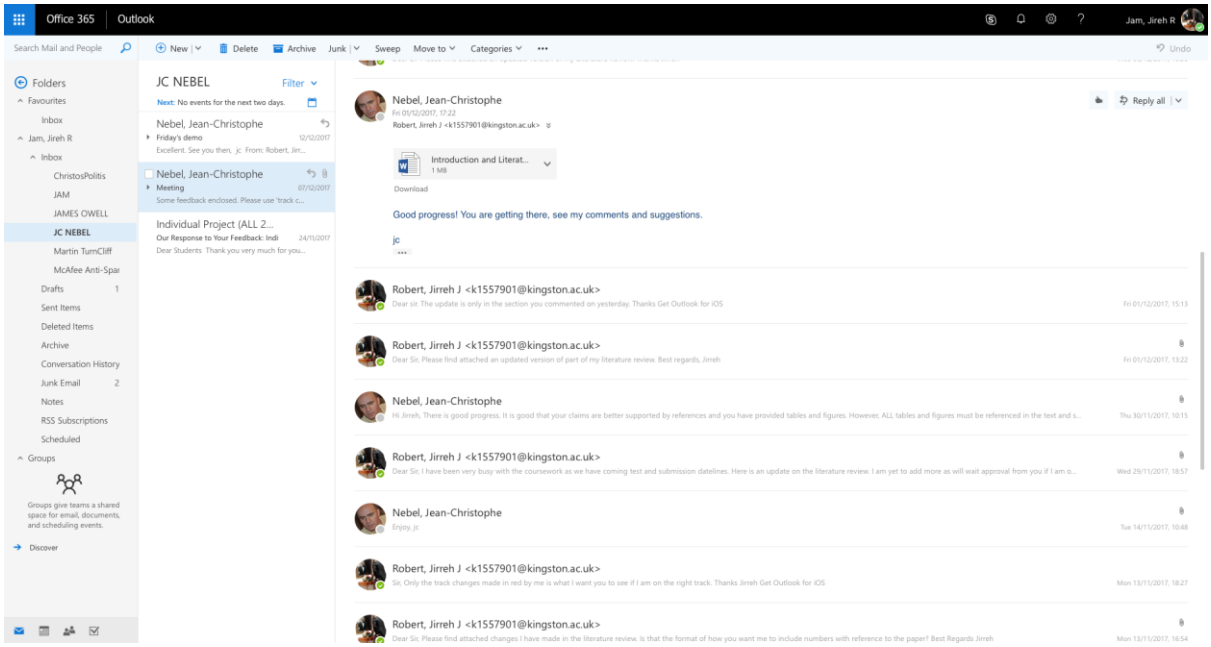
Wenyi Zhao, and Rama Chellappa (2005) *Face Processing: Advanced Modeling and Methods.* Elsevier Science.

Yi, J., Yang, H. and Kim, Y. (2000) 'Enhanced fisherfaces for robust face recognition', *Biologically Motivated Computer Vision, Proceeding,* 1811 pp. 502-511.

Yi-Qing Wang. (2014) 'An Analysis of the Viola-Jones Face Detection Algorithm', *Image Processing on Line,* 4 pp. 128-148.

Appendix:

A1: Client Meeting and Communication.

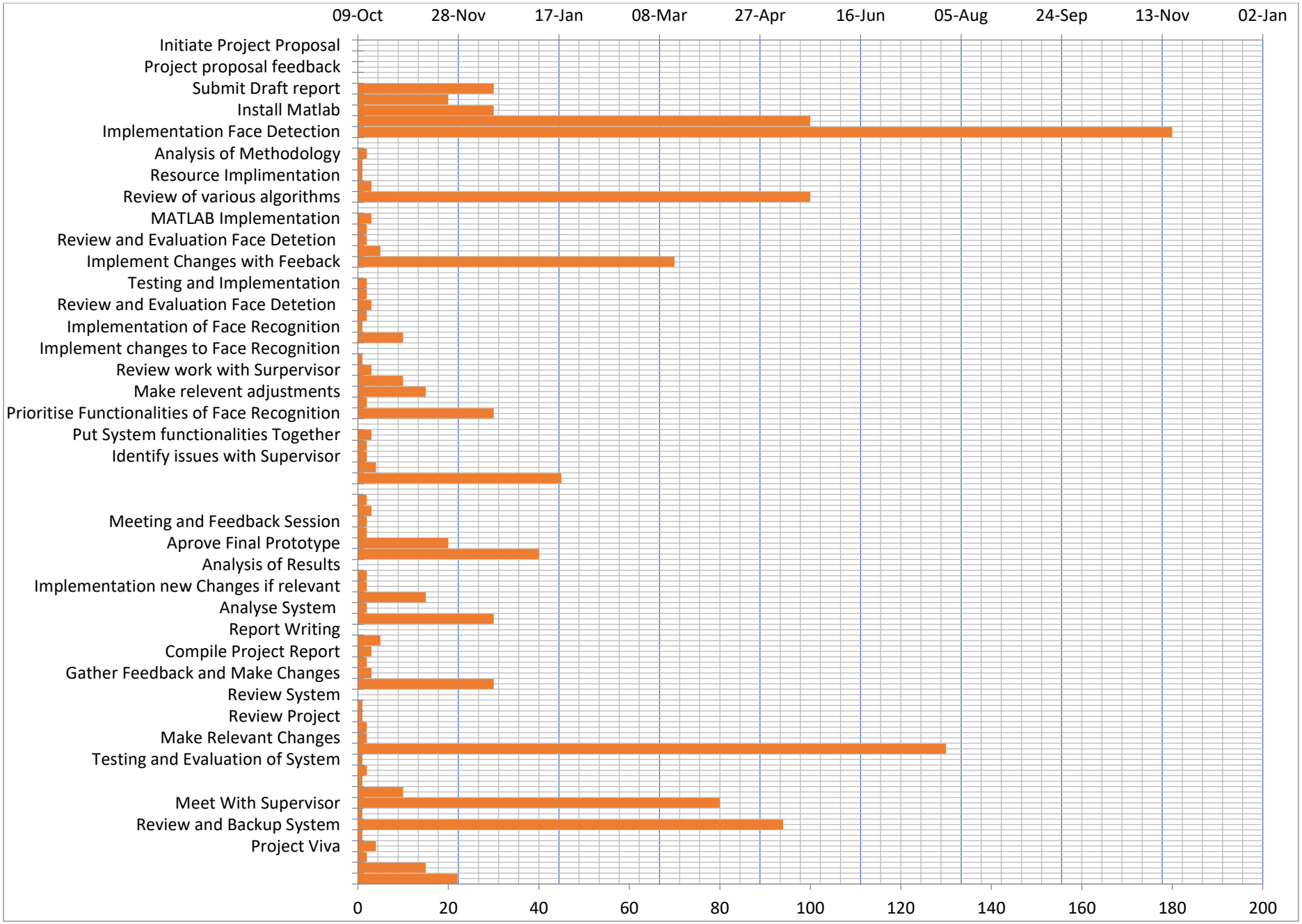


A2. GANNT Chart.

GANNT Chart October 1st To April 23rd 201

Task ID	Task	Start Date (DD-MMM)	Duration (Days)	End Date
1	**Initiate Project Proposal**	**09-Oct**	**22**	
3	Project Proposal Draft	09-Oct	15	
4	Project proposal feedback	10-Oct	2	
5	Review changes with Feedback	11-Oct	4	
6	**Submit Draft report**	15-Oct	1	
7	**Prototyping and Initial Demo**	**05-Nov**	**94**	
9	Install Matlab	12-Nov	1	
10	Research	14-Nov	80	
11	Implementation Face Detection	15-Nov	10	
12	Initial Demo	16-Nov	1	
19	Analysis of Methodology	17-Nov	2	
20	Literature Review completed		1	
21	**Resource Implementation**	**19-Oct**	**130**	
22	Develop and Refine functionalities	19-Oct	2	
23	Review of various algorithms	21-Oct	2	
24	Decide on Design and algorithm	23-Oct	1	
25	MATLAB Implementation	24-Oct	1	
26	First Working Prototype			
27	**Review and Evaluation Face Detection**	**25-Nov**	**30**	
28	Identify Issues with Detection	25-Nov	3	
29	Implement Changes with Feedback	28-Nov	2	
30	Demo and Feedback	30-Nov	3	
31	Testing and Implementation	03-Dec	5	
32	Feedback from Supervisor			
33	**Review and Evaluation Face Detection**	**08-Jan**	**30**	
34	Feedback on Face Detection	08-Jan	2	
35	Implementation of Face Recognition	10-Jan	15	
36	Feedback on Face Recognition	25-Jan	2	
37	Implement changes to Face Recognition	27-Jan	2	
38	Incremental Implementation			
39	**Review work with Supervisor**	**15-Feb**	**40**	
40	Analyse Requirements	15-Feb	20	
41	Make relevant adjustments	07-Mar	2	
42	Review progress with Supervisor	09-Mar	2	
43	Prioritise Functionalities of Face Recognition	11-Mar	3	
44	Face Recognition Testing and Evaluation	14-Mar	2	
	Put System Functionalities Together			
45	**Evaluation and Analysis of System**	**01-Mar**	**45**	
46	Identify issues with Supervisor	01-Mar	4	
47	Implement Changes to System	05-Mar	2	
		07-Mar	2	

48	Approve Final GUI Design	09-Mar	3
49			
50	**Review of Final System(GUI)**	**15-Mar**	**30**
51	Meeting and Feedback Session	15-Mar	2
52	Make relevant adjustments	17-Mar	15
53	Approve Final Prototype	01-Apr	10
54	Testing	11-Apr	3
55	*Analysis of Results*	14-Apr	1
56	Report Writing		
57	**Implementation new Changes if relevant**	**27-Mar**	**10**
58	Writing of Report	27-Mar	1
59	Analyse System	28-Mar	2
60	Make Adjustments	30-Mar	3
61	Report Writing	02-Apr	2
62	*Deployment of System as Standalone*	04-Apr	2
63	Compile Project Report		
64	**Review Project Report**	**06-Apr**	**70**
65	Gather Feedback and Make Changes	06-Apr	5
66	Implement Changes to Report	11-Apr	2
67	Review System	13-Apr	2
68	*Meeting with Supervisor*	15-Apr	3
69	Review Project		
70	**Project Report Alongside Supervisor**	**29-Oct**	**100**
71	Make Relevant Changes	29-Oct	3
72	Submit Project Report(Draft)	01-Nov	1
73	*Testing and Evaluation of System*	02-Nov	1
74	Prepare Viva Power Point	03-Nov	2
75	**Project Viva**	09-Oct	**180**
76	Meet with Supervisor	09-Oct	100
77	Review System and Implement Changes	17-Jan	30
78	Review and Backup System	16-Feb	20
79	Allowance for Further Changes	08-Mar	30
80	**Project Viva**	20-Mar	


09-Oct
28-Nov
17-Jan
08-Mar
27-Apr
16-Jun
05-Aug
24-Sep
13-Nov
02-Jan
Initiate Project Proposal
Project proposal feedback
Submit Draft report
Install Matlab
Implementation Face Detection
Analysis of Methodology
Resource Implimentation
Review of various algorithms
MATLAB Implementation
Review and Evaluation Face Detetion
Implement Changes with Feeback
Testing and Implementation
Review and Evaluation Face Detetion
Implementation of Face Recognition
Implement changes to Face Recognition
Review work with Surpervisor
Make relevent adjustments
Prioritise Functionalities of Face Recognition
Put System functionalities Together
Identify issues with Supervisor
Meeting and Feedback Session
Aprove Final Prototype
Analysis of Results
Implementation new Changes if relevant
Analyse System
Report Writing
Compile Project Report
Gather Feedback and Make Changes
Review System
Review Project
Make Relevant Changes
Testing and Evaluation of System
Meet With Supervisor
Review and Backup System
Project Viva
0
20
40
60
80
100
120
140
160
180
200