Dumbledore: A Vision Based Attendance Analyser

Harry Lynas
pm002501@reading.ac.uk
BSc Computer Science

Supervisor: M. Manjunathaiah
Date of Submission: 24/04/2015

Abstract
This paper presents the proof of concept for a vision based attendance analyser called Dumbledore. The concept allows for a webcam to be placed at the front of a lecture setting and with this lecture attendance can be determined. PCA and the extension WMPCA are the two primary facial verification algorihms that are used in this application to determine which students are present or absent. The tests show how WMPCA adaquately deals with the problems of illumination and pose and how the GUI developed provides a rich and intuitive interface. This proof of concept can be easily developed into a full appplication due to modular implementations and successful tests.

Acknowledgements
M. Manjunathaiah for project supervision and guidance.

Contents

1. Glossary of Terms and Abbreviations

Term / Abbreviation	Meaning
API	Application Program Interface
Dumbledore	Refers to the main application developed that acts as a proof of concept for a vision based attendance analyser.
Face Recognition	Detecting the location of a face within a given image.
Face Verification	Detecting whether a given face matches any known faces.
GUI	Graphical User Interface
HSV	Hue, Saturation, Value
PCA	Principle Component Analysis.
PID	Refers to the Project Initiation Document, which is attached with this report.
RGB	Red, Green, Blue.
SLEASE	Social, Legal, Ethical, and Health concerns within Systems Engineering.
SSE	School of Systems Engineering
WMPCA	Weighted Modular Principle Component Analysis.

2. Introduction

This report describes the entire development cycle of the Dumbledore application. Dumbledore is a prototype that acts as a proof of concept for a vision based attendance analyser. The application aims to operate such that a webcam can be placed at the front of a classroom and the software will use it in order to determine which students are present or absent.

Dumbledore is motivated by facial recognition and verification algorithms becoming more prevalent within the last decade as better algorithms have been developed and hardware has become more powerful [1]. Notably a facial verification algorithm called GaussianFace earned the achievement of outperforming humans for the first time only a few months before this project was initiated [2]. Using these advancements it is feasible to have adequate face recognition and verification in a classroom setting. The PID describes in more detail the steps taken to initiate this project [3].

The two primary algorithms investigated for this project were PCA and WMPCA. WMPCA is an extension of PCA and helps deal with the problem of illumination, as described later in this report.

The rest of this document describes in detail the development process taken and the resulting proof of concept that is Dumbledore.

3. Problem Articulation / Technical Specification

The primary problem to be solved by Dumbledore is that of face verification in a classroom or lecture setting. It is necessary to achieve an adequate success rate in an environment which will have varying levels of illumination and each person in different poses. It is proposed that the algorithms PCA and WMPCA be used to achieve facial verification.

PCA is the most commonly used facial verification algorithm as part of the Eigenfaces approach [4]. PCA is used to reduce the amount of data which reduces computation time and removes data of little importance. In this application the amount of eigenfaces is reduced. An eigenface represents a distance (the variation) from the mean face in a set of faces.

WMPCA is an extension of the PCA implementation such that each face is split into a set number of horizontal regions and PCA is performed on each region individually, an example of which is shown in Figure 1. After each region has been processed a net error function is used to retrieve whether a match has occurred.

Figure 1. A face split into three regions for WMPCA **[5]**.

Other facial verification algorithms may be investigated and possibly implemented but these are of a secondary concern since WMPCA should be adequate in helping to deal with the problems of illumination and pose.

To achieve facial recognition an existing library called OpenCV [6] will be utilised. OpenCV is programmed in C++ but provides interfaces for many other languages including Java. OpenCV is optimised, multithreaded, well tested, and specialises in computer vision thus making it ideal for reliable face recognition.

To support these algorithms a graphical user interface (GUI) is required. The GUI should provide the necessary functionality for a teacher or lecturer to detect and receive the results of attendance. The results should be rendered directly onto the GUI to provide immediate feedback. The GUI should provide audit functionality, reporting the results back to an external entity or by email to the lecturer/teacher.

It is assumed that the machines used to run Dumbledore will be powerful enough as to perform facial recognition and verification in a suitable time frame, as described in the assumptions set out in the PID. The training of images can be conducted on a central more powerful computer, which is the most time consuming process, and then the results saved to a database which can be loaded in the classroom or lecture settings.

It is important that the GUI is suitable for use by a lecturer/teacher and that it displays the results in a coherent and clear view since they are the primary user of this software and thus the primary stakeholder. Other stakeholders may include the entity employing Dumbledore as a solution, as they will be wanting to replace an old and less efficient system, as well as the actual students who will want a system that functions correctly and does not report them as being absent when they are present.

3.1. Solution A

A solution is proposed such that the specification can be split into three key components: the GUI, the facial recognition and verification functionality (API), and the audit functionality. Each component can be treated individually in the design and implementation. These then will link together to provide the complete Dumbledore prototype. This is illustrated in Figure 2, and each component can be thought to satisfy each of the technical products defined in the PID. This solution subsequently links directly with the PID.

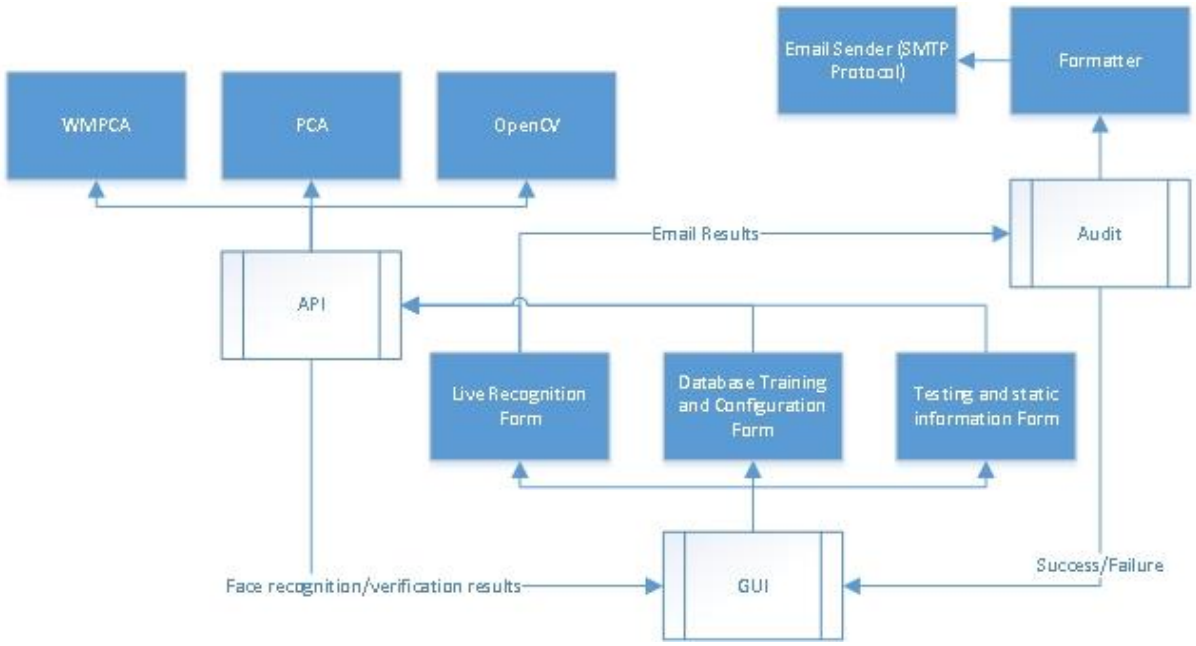

Figure 2. A diagram for the proposed **Solution A**.

The GUI should provide a rich and powerful interface for the user. This component will handle all of this behaviour and call the API and audit components as the GUI is operated. For the GUI three primary forms are required. The live recognition form will display the webcam feed in real time and display matches found as they occur. This form can be used to send the results to an external entity. The database training and configuration form allows for the saving and load of databases, as well as the training of new databases. It contains all the settings used. The testing and static information form is primarily used for debugging and lets the user manually confirm all is in order if something is not working as intended.

The API will provide functions that handle all the heavy processing and return simple results from simple inputs, such as an image being input and the detected people in the image returned. Similarly the audit functionality will allow the results of the API to immediately be parsed into a formatted email and dispatched.

3.2. Solution B

A solution is proposed whereby two primary components exist: a database interface and a main application layer. This system would primarily work through lots of database communication where most of the logic can be processed through structured language queries (SQL), and data stored and retrieved this way. The GUI would include functionality for all that is required and contain classes that handle the facial recognition and verification, as well as the auditing features. This focus on querying a database creates a higher level abstract layer that would take away a lot of the low level processing implementations required. This is illustrated in Figure 3.

The database application used could depend on licencing and the hardware requirements. A likely candidate would be MySQL which provides a free database server with great flexibility on the table and data types used as well as providing support the

Structured Query Language (SQL) queries [7]. It is licenced under the GPL licence which means that it would be free to use.

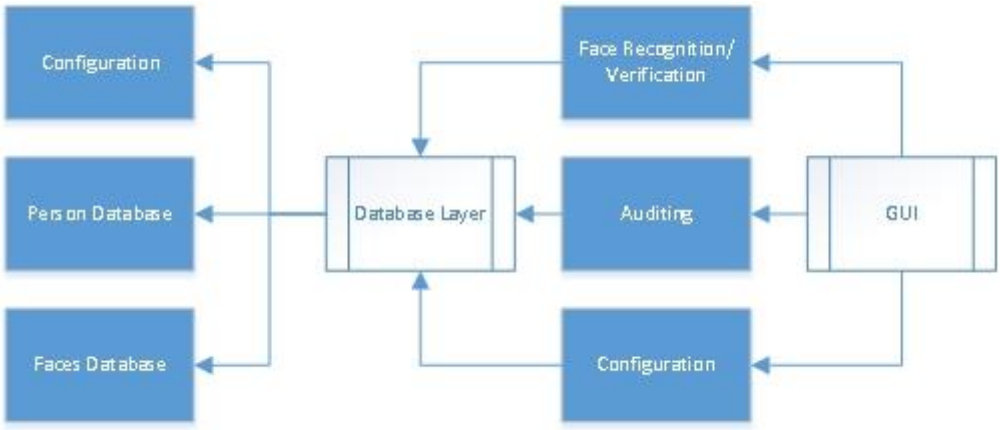

Figure 3. A diagram for the proposed **Solution B**.

3.3. Solution C

A solution was proposed that stresses the safety and privacy of data. The Dumbledore application is required to collect biometric data about individuals as well as their names. The biometric data consists of images containing a person. This data cannot be one-way encrypted because biometric data is noisy. A public private key system is proposed such that only one entity knows the password to access the data. The password can be input at application start up so that it is only in temporary memory. The Advanced Encryption Standard (AES) is the most likely encryption algorithm to be used due to it being a standard and well tested [8].

The username and password would be used to encrypt and decrypt the data but after between these events the actual Dumbledore program will need to function. For this Solution A is suggested because Solution B relies on the database being handled by a foreign entity. As such, this solution can be thought of as an extension to Solution A that would handle the privacy concerns with the data in an appropriate manner. This is illustrated in Figure 4.

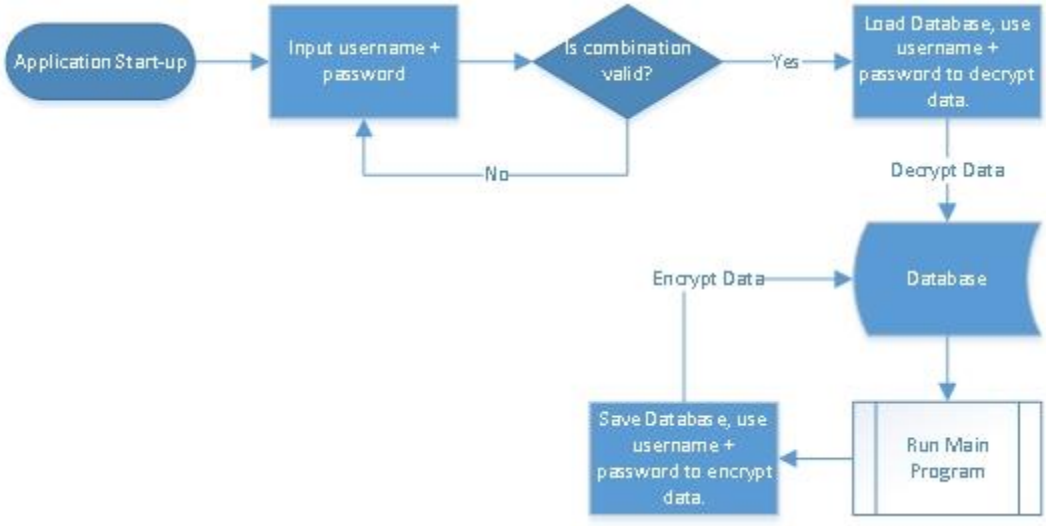

Figure 4. A diagram for the proposed **Solution C**.

3.4. Acceptance Requirements

To classify whether or not a solution has solved the problems and created an adaquate proof of concept a set of critierias have been defined that can be seen in Table 1. Each crieria must be met in order to deem a solution successful.

ID	Criteria	Failed / Met / Exceeded	Comments
1	The GUI must be suitable for use by a normal user. This can be tested against peron(s) not in the SSE department and is a qualiative measure.		
2	The GUI must provide functionality to report the presence or absence of each student in a lesson/lecture to an external entity.		
3	The face verification must perform at an adaquate success rate (>= 70% is ideal).		
4	The system must be able to detect multiple people from a given image if multiple people exist.		
5	The system must perform in real time. (Able to process at least 1 frame per second.)		
6	The system must be able to cope with changing illumination and pose.		

Table 1. The criteria's required to deem a solution successful.

4. Literature Review

The primary problem to solve in order to create the Dumbledore application is that of facial verification within a classroom/lecture setting. A study on humans ability to recognise and verify faces has determined that "humans can recognize familiar faces in very low-resolution images" and can "handle significant degradations in face images" [9], as illustrated in Figure 5 where over 50% of the images were recognised and verified by humans.

Figure 5. "Human observers are able to handle significant degradations in face images. For instance, subjects are able to recognize more than half of all familiar faces shown to them at the resolution depicted here." **[9]**

Only within the last year have machines been able to outperform humans in facial verification [2] which means matching or beating human facial verification rates is a great technical feat and subsequently not in the scope of the Dumbledore application.

AurorA Security boasts the ability to verify faces in real time for attendance and security purposes using custom-built hardware and specialist software [10]. This is a commercial product that does not give a demonstration. Subsequently there is no way to verify their claims but it is reasonable to assume that the solutions they offer are adequate for their customers and as such aids in the idea that Dumbledore's aim of analysing attendance is feasible.

Amazon offers a FaceIN Facial Recognition Attendance System that offers the ability to face recognise and perform attendance analysis with databases made up of up to 70,000 images [11]. The reviews for this product say that there is "trouble with recognition, especially if an employee wears glasses" which suggests that the product is not very good and would not be suitable to use in a classroom situation. Similarly the Time Attendance FK800 boasts being able to handle "time attendance" and "identity management" but only has a technical specification of being able to handle 300 face users [12]. Both products are not competitive with what Dumbledore aims to achieve but suggest there is demand for such systems. It is likely that organisations seeking to use a facial recognition based attendance system would seek a custom solution for their needs from a specialist consultancy firm, such as AurorA.

A face recognition based lecture attendance system is proposed by the Kyoto University that notes how facial verification algorithms will not achieve a sufficiently high success rate [13]. However it does suggest that continued observation and determining if the results are consistent will solve this issue, and this suits the classroom/lecture setting since a lesson will typically last at least half an hour to an hour. The paper describes a test on one image capture resulted in only a 37.5% detection rate being achieved, however when testing over a 79 minute period of time a 80% success rate was achieved which is a much better result.

A face detection system for the attendance of class students is proposed by The University of Engineering and Technology, Lahore, Pakistan [14]. Their system uses the

eigenface approach and obtains good results, however in their test conditions they only use forward-facing frontal images which does not represent the environment of a classroom setting. The main advantage they draw from using eigenfaces is that they do not need multiple source images for each student, but can add more as the student changes each year easily. The authors raise the important issue of privacy concerns, highlighting how one-way encryption cannot be used on the stored images due to biometric data being noisy, and how EU privacy regulations are strong. They suggest using a private/public key approach where only one entity would be in possession of the correct key to decrypt the data. This led to the development of Solution C.

Another similar implementation is described in a publication by the Symbiosis of Technology, Pune, Maharashtra, India [15]. A hybrid algorithm of LCA and PCA is used in order to verify faces in this system which they describe as "reliable and efficient". No testing results are provided, however the implementation is done with a Raspberry Pi which has very small computational power [16] which suggests that the implementation of Dumbledore will be able to achieve real time recognition even without powerful hardware.

A publication in the IJCSI makes the important distinction of how a facial based attendance system is infinitely faster than student's queueing to scan fingerprints or manual attendance analysis [17]. The system proposed uses the eigenfaces approach also, and notes how no specialist hardware is required. It also notes how better algorithms are available that will increase system performance, however the paper states how this method of classroom attendance analysis is "secure enough, reliable and available to use".

A case study on using facial recognition for analysing attendance gives a detailed analysis on how a system would function such that it monitors student's entering a classroom [18]. It also suggests the PCA approach but notes how the main problem is having to compare each student to all entries in a database. It also makes a comparison to many other metrics of analysing attendance, reviewing face recognition as the most efficient proposition.

These existing implementations and research into facial recognition based attendance analysis all use some variation of eigenfaces (PCA) and all deem such an implementation viable with further improvements possible. The original publication of the eigenfaces approach for race recognition [4] gives a high level breakdown of the steps used to recognise new faces and to calculate the first set of known faces. A more thorough and detailed breakdown is given in a tutorial on PCA [19] which gives test input data and the output expected, which will allow for Dumbledore to be tested against known data during its development.

The WMPCA algorithm [5] provides an algorithm based on PCA that should achieve greater success rates. Figure 4 on page 6 of this literature demonstrates how WMPCA is able to better deal with illumination and pose variations than PCA and modular-PCA. This is backed up by the findings of the study on humans ability to verify faces which found "of the different facial features, eyebrows are among the most important for recognition" [9] because WMPCA splits the face horizontally into a set of regions, such as the eyes, nose, and subsequently would be able to match very well on the eyebrows.

Another publication on WMPCA confirms that WMPCA achieves greater success rates than PCA where illumination and/or pose is not continuous [20]. The implementation suggested proposes a hardware solution resulting in a quite fast facial verification rate, as shown in Table 2. This hardware solution achieves suitable speeds for real-time application, as highlighted in the conclusions of the publication. Modern hardware is a lot faster than the hardware solution given, and as such it is expected that Dumbledore will be able to implement WMPCA at a software level and still achieve suitable speeds for real-time application. Furthermore the hardware implementation did not implement the functionality for pre-processing and face-detection, which are both parts necessary for a full system.

Device	EP20K200EFC484-2X
Tool	Quartus-II
Total Logic Elements	7760
Total Memory Elements	26,496
Worst case Tco	12.108 nsec
Clock	33.33Mhz
Time for recognition	38 msec

Table 2. Results of the hardware implementation experiment **[20]**.

The literature given here has led to the conclusion that Dumbledore is both feasible and the concept highly researched into. The objectives given in the PID are subsequently achievable.

5. The Solution Approach

Solution A logically breaks down the problem into its fundamental three components: the API, the GUI, and the auditing features. Having the GUI in its own separate module allows for the use of JavaFX [21] which provides a model viewer controller (MVC) framework. The MVC approach can allow for all the GUI logic to happen within its own module while populating its data by calling the API and audit modules as required. The database in this solution is a serializable class that forms a database, which means that most operations are performed in memory. Due to the simplicity of the database it can easily be extended or stored in another format as a future improvement.

Solution B provides a more robust and stable database with communication layers abstracted away such that SQL queries can simply be used to manipulate, read, and write data. This however complicates the solution in that there are more large dependencies. Also a lot more file I/O will be performed which could lead to a slowdown in the system. This solution would be scalable in that the database server could be hosted on a separate machine to the front end GUI for faster processing if desired.

Solution C is a desirable solution and would be structurally easy to integrate into Solution A. However it overcomplicates the project significantly with cryptography functionality that is not necessary. Photos and names are the only personal data stored in this solution and they can be obtained and kept unencrypted if clear consent is given. As an advanced goal this could be feasible but it would depend on time constraints and the complexity of integrating the cryptography necessary.

Subsequently Solution A is chosen to be used, with the advanced goal of Solution C to be completed if it is deemed feasible within the time constraints of the project later in development. This project will be developed by first implementing much of the API module, then integrating it into a GUI module, and then after these two modules are complete the auditing module can be implemented. This is because the API contains the core functionality to achieve face recognition and verification, and the GUI will be used to access this functionality and display results in a user friendly view. The audit functionality is only necessary after these first two steps are complete and there are results to report.

The criteria used to determine if this solution approach was successful is the same as that given in section 3.4.

6. Implementation

6.1. Facial Recognition

To achieve facial recognition (identifying faces within an image) the image processing library OpenCV was used which has native bindings for Java. Java is the language being used to create the implementation.

The facial recognition API needs to function similarly to that shown in Figure 6. OpenCV has its own image format called a 'Mat'. To retrieve the faces from a given image the image file will need to be loaded by OpenCV, facial recognition called, and the faces found returned in a format that native Java can function with.

```
public Image[] getFacesFromImage(String fileName) {
    Mat file = OpenCV.readFile(fileName);
    Image[] returnFaces = {};

    for (Mat face : OpenCV.getFaces(file)) {
        returnFaces.add( face );
    }

    return returnFaces;
}
```

Figure 6. Pseudo-code for facial recognition.

Upon attempting to implement this function it was discovered that a classifier is needed to recognise a face. OpenCV provides a frontal face LBPCascade classifier on its repository [22] which shall be used to achieve facial recognition. The resulting function correctly identified the top left and bottom right coordinate of each face in an image allowing for all the pixels to be extracted.

6.2. PCA

Many matrix calculations will be required during the implementation of both PCA and WMPCA due to the nature of how an image is usually represented as a matrix – the rows and columns being the width and height of the image with each pixel value stored in each cell. To aid with these matrix calculations the Apache Commons Library [23] was used.

The idea behind PCA is that a set of images is trained and thus the weights and eigenfaces for this data obtained. New unknown images can be projected onto the eigenspace and then whether it is a known face or not measured using the Euclidean distance function. It is determined to be a match if the distance is under a defined threshold. If it is an unknown face then the face can be incorporated with the trained data as a new person.

When researching into PCA a tutorial was found that gave example data and what the output should be when each stage of PCA is formed on that example data [19]. Using this it possible to confirm that each stage of PCA was producing the correct results.

The first step in order to perform PCA is the pre-processing of the faces. All input faces must be of the size and greyscale. It was deemed best to integrate this stage into the face recognition function so that these factors could be applied as each face is identified. To greyscale the image each pixels red, green, and blue (RGB) value was averaged using the mean method. As each face region was generated as a new image it was then rescaled to a standard defined size. To resize the image a library was used called ImgScalr [24] due to its simplicity and ease to integrate. The finalised function is shown in Figure 7, and example normalised faces retrieved using this function are shown in Figure 8. It should be noted that the function is synchronized to prevent the function running while another instance of it is already running. This is because the OpenCV functions will fail when called concurrently. Furthermore, OpenCV uses inversed columns to that of Java, as it is height against width rather than width against height. This caused confusion initially since the images obtained were always inversed, and was fixed by switching row with column in the get function.

```
/**
 * This function returns each face region detected as a new buffered image
 * that is resized to a standard size and greyscaled.
 *
 * @param file The image to process.
 * @return Each face region greyscaled and resized.
 */
public synchronized static BufferedImage[] getProcessedFaceRegions(
        String file) {
    try {
        String path = FaceRecognition.class
                .getResource("lbpcascade_frontalface.xml").getPath()
                .substring(1);
        // Set the classifier
        CascadeClassifier faceDetector = new CascadeClassifier(path);
        // Read the image into a OpenCV format
        Mat image = Highgui.imread(file);
        MatOfRect faceDetections = new MatOfRect();
        // Attempt to determine all face regions
        faceDetector.detectMultiScale(image, faceDetections);
        // Get the pixels for a buffered image for each face
        int numRegions = faceDetections.toArray().length;
        BufferedImage[] images = new BufferedImage[numRegions];
        int count = 0;

        for (Rect rect : faceDetections.toArray()) {
            images[count] = new BufferedImage(rect.width, rect.height,
                    BufferedImage.TYPE_INT_RGB);
            // Obtain the pixels
            for (int i = rect.x; i < rect.x + rect.width; ++i) {
```

```
                for (int j = rect.y; j < rect.y + rect.height; ++j) {
                    // OpenCV has opposite axis
                    double[] data = image.get(j, i);
                    // Greyscale the pixel value by using the mean value of
RGB
                    int pixelVal = (int) ((data[0] + data[1] + data[2]) /
3);
                    images[count].setRGB(i - rect.x, j - rect.y,
                            pixelVal << 16 | pixelVal << 8 | pixelVal);
                }
            }
            // Resize image to the standard size
            images[count] = Scalr.resize(images[count], Scalr.Method.SPEED,
                    Scalr.Mode.FIT_EXACT, standardSize, standardSize,
                    Scalr.OP_ANTIALIAS);
            ++count;
        }

        return images;
    } catch (Exception ex) {
        ex.printStackTrace();
        return null;
    }
}
```

Figure 7. The final face recognition core function.

Figure 8. Example extracted faces that have been converted to greyscale and resized.

This pre-processing stage is applied all faces found within each input image. Each face is also converted into a one dimensional array by concatenating on each row. This illustrated in Figure 9.

A	B	C
D	E	F
G	H	I

A	B	C	D	E	F	G	H	I

Figure 9. A two-dimensional array is concatenated into a one-dimensional array.

Each resulting one-dimensional array representing a face is then placed as a position inside a two-dimensional array. This is because then the entire set of faces data can be represented in a single matrix. This is illustrated in Figure 10 where each row of the matrix represents a different face. All the rows remain the same size due to the faces being resized to a standard size during the pre-processing stage.

```
Face Matrix
{
        {1, 2, 3, 4, 5},
        {5, 4, 1, 2, 4},
        {0, 1, 1, 2, 1}
}
```

Figure 10. An example face matrix where each row represents the pixel data for a face image.

In order for the calculations using this data to be successful each row of the matrix is normalised so that the values are between zero and one. This is because each data value within the matrix represents a pixel RGB value, and a RGB value is inserted into an integer such that all the bits are used, as illustrated in Figure 11**.** Having such a large integer value causes overflow errors when calculations are made, because it is already near the maximum value an integer can hold. To normalise each row between zero and one the pseudo-code given in Figure 12 is used.

Integer (32 bits), Black (value = 16777215), Integer Max Value = 2147483647			
Alpha (8 bits) (unused)	Red (8 bits)	Green (8 bits)	Blue (8 bits)
255	255	255	255

Figure 11. Represents why a pixel value needs to be normalised to prevent overflow errors when performing calculations against other pixel values.

```
double[] pixelData = faceMatrix[currentRow];
double minValue = getMinimumValueInArray(pixelData);
double maxValue = getMaximumValueInArray(pixelData);
for (int i = 0; i < pixelData.length; ++i) {
    pixelData[i] = (pixelData[i] - minValue) / (maxValue - minValue);
}
```

Figure 12. Pseudo-code to normalise a faces pixel data between 0 and 1.

Now the data is in the correct data structure in order to perform PCA. The pre-processing steps have been completed and the data has been formatted as required. Taking the matrix of face data, the mean of each row is found. That mean is subtracted from each piece of data in each row within the matrix. The pseudo-code for this is given in Figure 13.

```
double[] meanForEachRow = new double[facesMatrix.length];
for (int i = 0; i < facesMatrix.length; ++i) {
    double mean = findMean( facesMatrix[i] );
    for (int j = 0; j < facesMatrix[i].length; ++j) {
        facesMatrix[i][j] -= mean;
    }
    meanForEachRow[i] = mean;
}
```

Figure 13. Pseudo-code to find and subtract the mean of each row.

The resulting matrix with the mean subtracted from each row is now used. With this new matrix the covariance matrix is calculated. The Apache Commons Library is used to

calculate the covariance matrix. The eigendecomposition of the covariance matrix is obtained also using the Apache Commons Library which gives all the eigenvalues and eigenvectors. The eigenvectors describe the variance from the mean face. The mean face is the mean average of all of the face matrix data. The eigenvalues describe the amount of variance, whereby a higher value describes more variance. Figure 14 shows some eigenvectors output to images where the pixel data is normalised between 0 and 255 (greyscale intensity).

Figure 14. Eigenvectors normalised between 0 and 255 and output as images for debugging purposes.

The calculation of the covariance matrix and the eigendecomposition is the part of the process that takes the longest due to the typical size of the matrices being processed. If a 50 pixel high and 50 pixel wide standard is used for each face, and there are 100 faces found in a training set, then that is immediately a 100 row 2500 column sized matrix.

The principle components are chosen based on these eigenvalues and eigenvectors. Not all the eigenvectors are required to be kept because some eigenvectors describe negligible variance. Subsequently the eigenvalues and eigenvectors are sorted by eigenvalue descending. Each pair is a principle component. A predefined number of values are kept – this is determined by the user; it is a parameter required to start PCA. In addition, the first three principle components are discarded. This is because the first three components describe the most variance from the mean face, which will primarily be illumination changes.

Once the principle components have been obtained the eigenfaces and weights can be calculated. To calculate the eigenfaces each mean subtracted matrix cell is multiplied by the equivalent sorted eigenvector cell, and then normalised. To calculate the weights the mean subtracted matrix is multiplied with the eigenfaces matrix transposed. The function used to achieve this is given in Figure 15. To further aid with Matrix calculations the Jama library is used [25].

```
private void calculateWeightsAndEigenFaces() {
    int pixelCount = dataWithAverageSubtracted[0].length;
    int imageCount = dataWithAverageSubtracted.length;
    int vectorsCount = vectors.length;
    // Calculate and normalise eigenFaces
    double[][] eigenFaces = new double[vectorsCount][pixelCount];
    for (int i = 0; i < vectorsCount; ++i) {
        double sumSquare = 0;
        for (int j = 0; j < pixelCount; ++j) {
            for (int k = 0; k < imageCount; ++k) {
                eigenFaces[i][j] += dataWithAverageSubtracted[k][j]
                        * vectors[i][k];
            }
            sumSquare += eigenFaces[i][j] * eigenFaces[i][j];
        }
        double norm = Math.sqrt(sumSquare);
        for (int j = 0; j < pixelCount; ++j) {
            eigenFaces[i][j] /= norm;
        }
```

```
    }
    // Get only the selected amount of eigenFaces
    this.eigenFaces = new Matrix(eigenFaces).getMatrix(
            0,
            eigenFaces.length <= numEigenFacesToUse ? eigenFaces.length - 1
                    : numEigenFacesToUse, 0, eigenFaces[0].length - 1)
            .getArray();
    // Calculate weights
    this.weights = new Matrix(dataWithAverageSubtracted).times(
            new Matrix(eigenFaces).transpose()).getArray();
}
```

Figure 15. The function used to calculate the eigenfaces and weights.

When the resulting eigenfaces are normalised between 0 and 255 (greyscale intensity) and output as images it is clear that the first few eigenfaces are describing the most variation and the eigenfaces towards the end are describing very little variation from the facial features that can be identified in the first few eigenfaces. This output is shown in Figure 16.

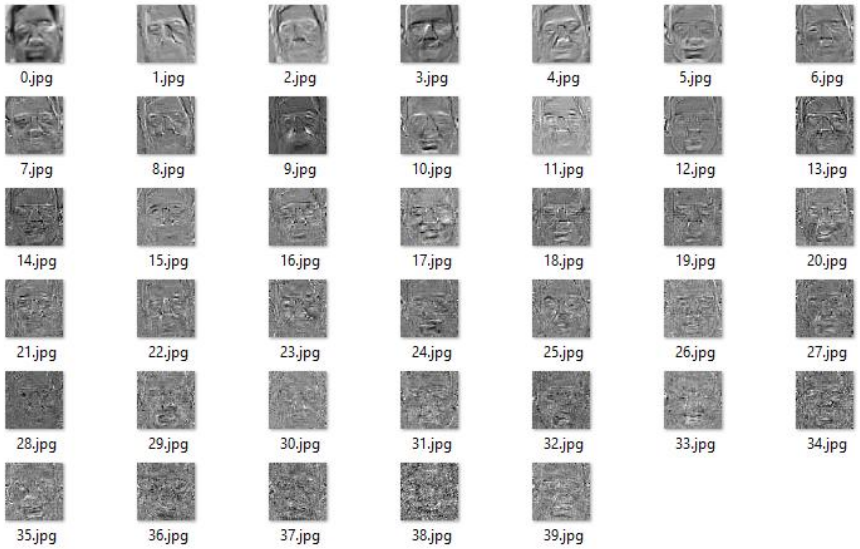

Figure 16. The eigenfaces normalised between 0 and 255 output for debugging purposes.

Now that the eigenfaces and weights have been obtained the PCA algorithm is complete and the data has been trained. In order to determine if an unknown image contains known faces it needs to be projected onto the eigenspace. To achieve this first the unknown image needs to be pre-processed with the same steps taken to train the data: the face(s) are extracted from the image using facial recognition algorithm, set to greyscale, and set to the standard size. The image is flattened into a one dimensional array and normalised between zero and one.

The unknown face is placed into a matrix data structure similar to that when training the data, however only one row exists this time because there is only one face. The average faces (the mean average of each face from the training data) is also turned into a matrix with a single row. The unknown face in matrix form is then subtracted from this matrix of average faces. The resulting matrix is multiplied with the transposed eigenfaces to obtain the weights for unknown input face. Eigenfaces need to be transposed because it was ordered by column rather than row. The pseudo-code for this is given in Figure 17.

```
// Construct the unknown face matrix, 1 row
Matrix in = new Matrix( unknownFace, 1 );

// Construct the average face matrix, 1 row
Matrix averageFaceMatrix = new Matrix ( averageFaces, 1 );

// Get the matrix of eigenFaces from the training set
Matrix eigenFaces = getEigenFaces();

// Subtract the input face from average faces
in = in.minus( averageFaceMatrix );

// Obtain the weights of the unknown face
Matrix inputWeights = in.times( eigenFaces.transpose() );
```

Figure 17. Pseudo-code for obtaining the weights for the unknown input face.

Once the input face weights have been obtained it is possible to use the Euclidean distance function to determine which training set face weights closest match those of the input face weights. The lowest distance is determined to be the closest match, and if it is under a predefined threshold then it is returned as correct match.

6.3. GUI Prototype

To support PCA a GUI prototype was constructed using JavaFX. To confirm that facial recognition (OpenCV) was working across a variety of images a simple frame was created that contained an image control that rendered the results of the OpenCV face coordinate detection. This is shown in Figure 18 working as intended.

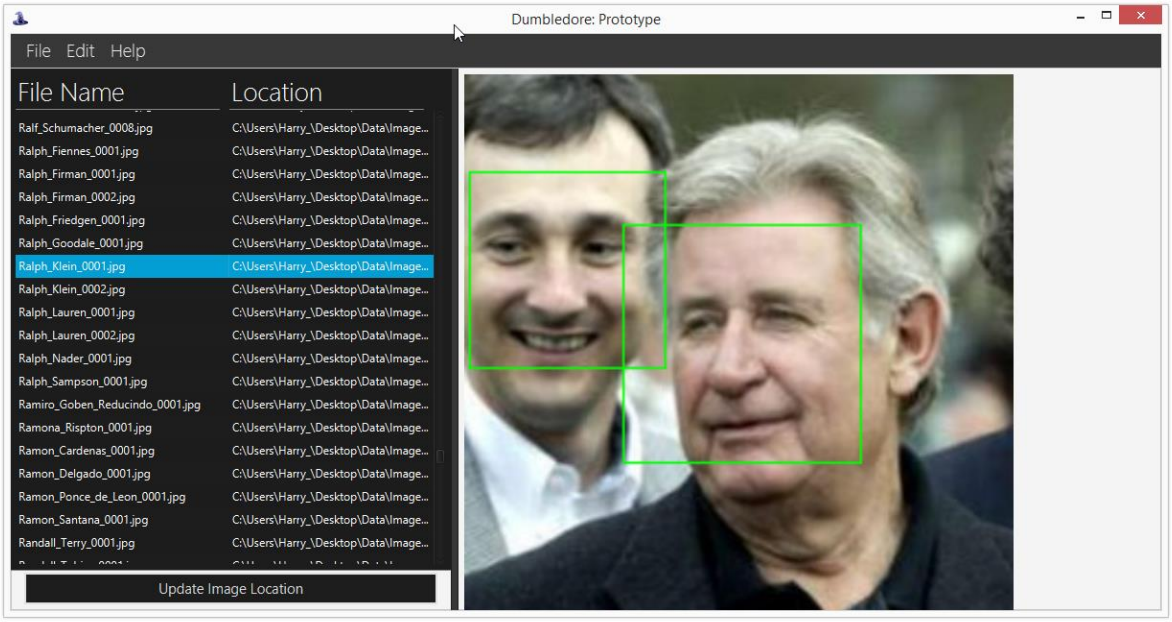

Figure 18. A Prototype GUI for testing OpenCV's facial recognition.

The frame was adapted to be able to debug and test the PCA algorithm. A new image view was added to display the closest matching image and a threshold label added. Buttons were added to train an image set and to lookup an unknown image, and templates for loading

and saving a database were implemented. The GUI at this point in the development process was only used to debug that the algorithms worked as intended. The results of the first PCA test on facial data is presented in Figure 19, and it can be observed that a match was made successfully.

Testing the time taken to train data sets with PCA at this point in development provided concerning results. As documented in Table 3, a long period of time is required to train any significant data set. This is because the covariance matrix and eigendecomposition are both very long calculations when the matrices are so large. For each test in Table 3 a standard face size of 50 by 50 was used, and 80% of eigenfaces were retained. This meant for the data set with 13324 images there was a ~13324 by 2500 matrix.

Images in Data Set	**Time to Train (ms / minutes)**		**Time per Image (ms)**
13324	1102779	18	82.8
530	53318	0.89	106

Table 3. Testing the time taken to train data sets with PCA.

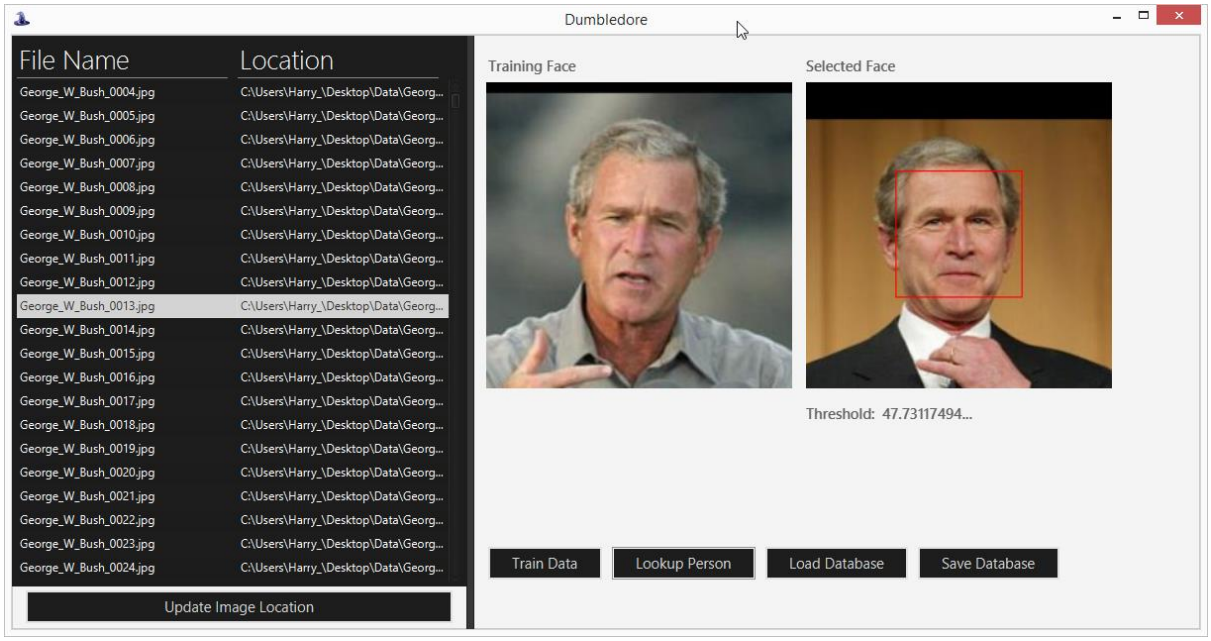

Figure 19. The first successful run of the PCA algorithm on actual face data with an unknown input image.

To give feedback to the user operating the GUI it was deemed appropriate to add a text area that displays debug information. While the text output to this text area may not be readable to the main potential user (a lecturer) it will contain valuable information for the developer to fix any issues that might occur and it will give feedback on what the status of the application is.

The API module is the main source of print statements and thus needs to be able to write to the GUI. However, the GUI is dependent on the API and thus if the API was also dependant on the GUI it would create a circular linking error. Furthermore, the API was

designed with the intention of being able to be used as a standalone module – having it dependant on the GUI would break this. As a solution a class was designed in the API that would store all messages to be printed. An external entity can retrieve messages from this class and handle them as desired. Each message added to this class is automatically given a header containing the time of the message and the thread name that added this message. Furthermore the class is implemented such that concurrency issues cannot occur. The GUI then used the JavaFX timeline feature to automatically update the text area with any new messages each second. The code used to update the text area is given in Figure 20. It was important to use the timeline implementation because JavaFX control objects can only be updated by the GUI thread.

```
Timeline logUpdater = new Timeline(new KeyFrame(Duration.seconds(1), new
EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        String text = "";
        for (String s : Log.getMessages())
            text += s + "\n";
        if (text.length() > 0)
            logArea.appendText(text);
    }
}));
logUpdater.setCycleCount(Timeline.INDEFINITE);
logUpdater.play();
```

Figure 20. The code used that updates the GUI with any new API messages.

The current design provides the functionality in order to train, load, and save databases. In order to work with live recognition it was designed such that a live input feed from a webcam device will appear on the left of a new form, and the results will be displayed on the right of the form. The facial verification algorithm used will be the same as the one used to train the data of the database loaded. To implement this a TilePane and ImageView were used – the tile pane to display the results, and the ImageView is updated with a new capture of the webcam in real time.

To retrieve a webcam's input webcam-capture library was used [26]. Using this library a timeline was implemented, similar to that of the log text area updater timeline, which retrieves the camera input, processes it, and saves any matches found. The pseudo-code for this is given in Figure 21.

```
cam = WebcamCapture.getWebcam();
cam.open();
while (true) {
    // Get a still image from the webcam
    BufferedImage inputImage = cam.getImage();
    // Draw the still image in the live input feed
    // Get OpenCV to draw the face bounds detected on the image
    renderImage(FaceRecognition.drawFaceBounds(inputImage));
    // Lookup which people can be found in the image
    Results[] results = LookupPeople(inputImage);
    // Render each result on the tile pane
    for (Result result : results) {
```

```
            tilePane.renderResult(result.getImage());
        }
        // Wait a period between each frame to prevent
        // this task consuming all available resources.
        Thread.sleep(timeBetweenFrames);
    }
```

Figure 21. The pseudo-code for the webcam task.

6.4. WMPCA

Once PCA was implemented and the GUI was at a point where the algorithms could be tested in real time WMPCA was implemented. WMPCA operates such that each face is split horizontally into a number of regions and PCA is performed on each region individually. When it is necessary to determine the closest match to an unknown input face, a weighted sum of errors function is used to determine the closest match. The same pre-processing stages of PCA and the actual PCA algorithm is applied to each region of the face. The code used to split the face into a number of regions is given in Figure 22. It can be seen how the data is stored in a three dimensional array. This data structure exists for each image and is indexed by the face index in the image, the region index for that face, and then the pixel data for that region (flattened).

```
public static void addWMPCATrainData(Image image, int numRegions) throws
Exception {
    // Get each region
    BufferedImage[] faces = FaceRecognition.getProcessedFaceRegions(image
            .getPath());
    // Check some faces were found
    if (faces.length == 0)
        return;
    // Split each face into regions
    int regionHeight = (int) Math.ceil(faces[0].getHeight() / numRegions);
    double[][][] regions = new
double[faces.length][numRegions][regionHeight * faces[0].getWidth()];
    // For each face
    for (int f = 0; f < faces.length; ++f) {
        // For each region
        for (int r = 0; r < numRegions; ++r) {
            int count = 0;
            // For each pixel
            for (int x = 0; x < faces[f].getWidth(); ++x) {
                for (int y = regionHeight * r; y < regionHeight * (r + 1);
++ y) {
                    if (y > faces[f].getHeight())
                        break;
                    regions[f][r][count++] = faces[f].getRGB(x, y);
                }
            }
        }
    }

    // Update the store
    image.setData(regions);
}
```

Figure 22. The code in order to split an image into a number of faces and regions.

Once all the regions are acquired and the data has been pre-processed it is possible to perform PCA on each region. To achieve this the three dimensional data structure indexed by face, then region, then containing pixel data, is transformed such that the region is the first index and the face the second index. This way each region can be indexed and the two dimensional data within can be sent to the PCA algorithm and trained as normal.

Once the data has been trained then it can be used to match faces in the same manner as the PCA algorithm functioned. The GUI at this stage matching webcam input in real time with WMPCA is shown in Figure 23.

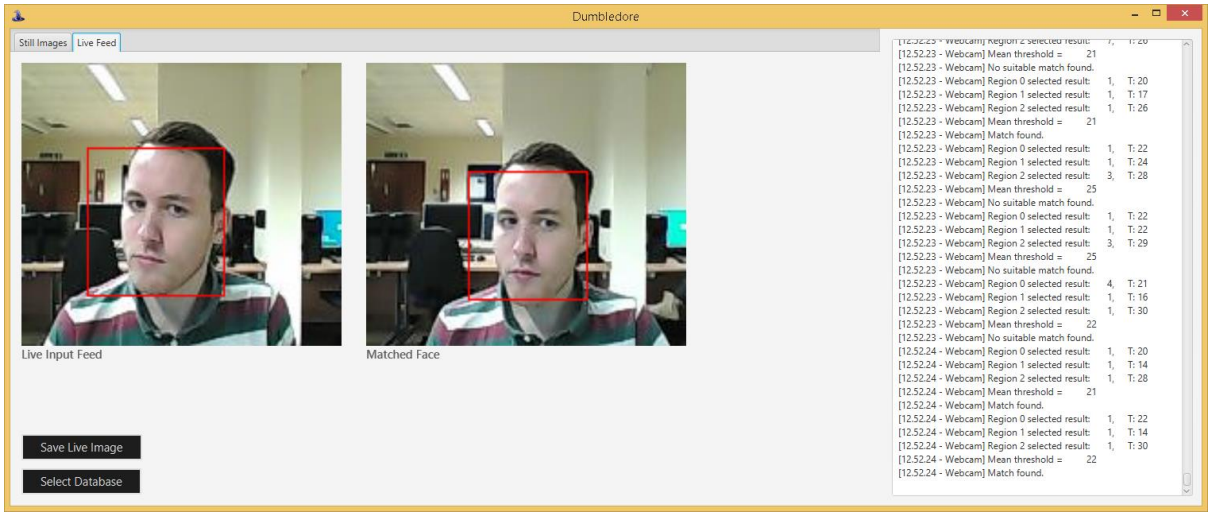

Figure 23. The GUI matching in real time with webcam input using WMPCA.

It was noticed that after running over a period of time the algorithm started performing worse and worse. After tracing through the data flow and the multithreading it was discovered that a lot of the matrices were being passed by reference instead of passed by value. Subsequently a deep copy class was implemented that provides static functions to copy multiple dimension arrays.

Because the algorithm is naturally split into a number of separate logical PCA tasks each task can be run on a separate thread. This significantly improves performance because it allows multi-core computers to simultaneously process multiple regions. The performance is also indirectly improved dramatically because by splitting the face into regions it reduces the number of pixels in each region, thus reducing the size of the matrices, thus speeding up the matrix calculations. This means that the WMPCA algorithm is a lot faster than the PCA algorithm at training data.

The University of Essex provides four face databases (faces94, faces95, faces96, and grimace) in order of increasing difficulty [27]. They contain a variety of individuals under a variety of different conditions but all facing towards the camera. When training the faces94 database with the PCA algorithm it took 905478ms (~15 minutes). When training the same database with the WMPCA algorithm it only took 336076ms (~6 minutes) which is a 269% speedup, a significant improvement. Furthermore this was only when there were three regions – if the face was split up into more regions then the speedup would be even greater.

6.5. GUI Completion

Now that the two core algorithms were complete it was possible to complete the GUI. The GUI needed to have options and functionality in order to achieve the functionality described in the project objectives. To achieve this text fields and labels were added to contain various options. This is illustrated in Figure 24.

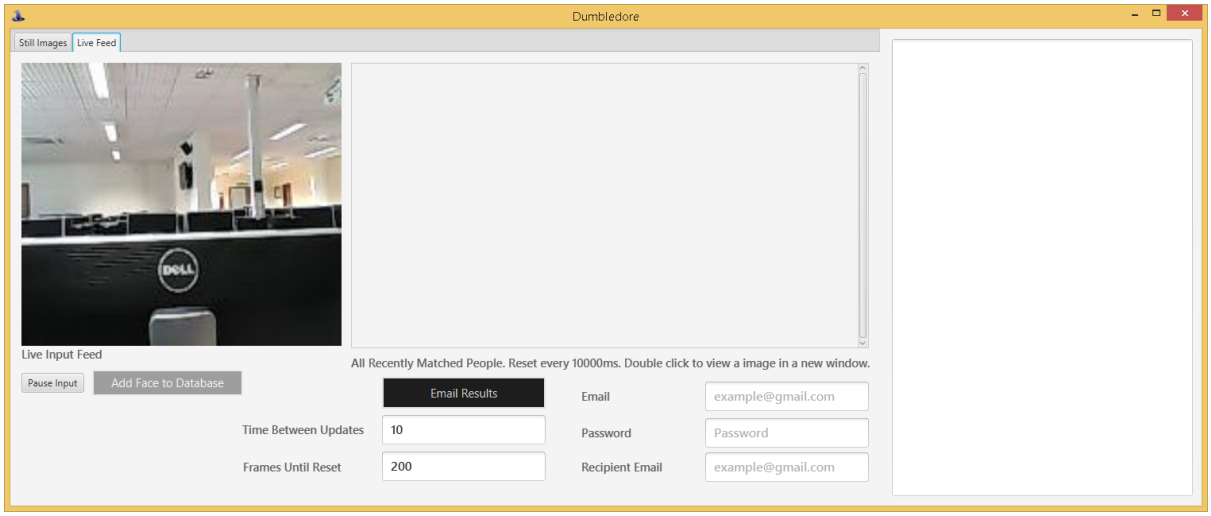

Figure 24. The main GUI form for live recognition.

6.6. Audit Features

To implement the auditing feature it was decided that the best course of action would be to email the results to a recipient. Since emailing is relatively simple to implement within Java it was not necessary to implement a dedicated module as planned in the solutions design. Instead it was integrated as a new package into the API module.

Java includes a native library implementing support for SMTP. This, combined with example code [28] allowed for the easy sending of emails. A Google Mail account was used for sending the emails during testing. The account needed to be configured specifically such that emails could be sent from an application with IMAP.

6.7. The Final Implementation

With all the required functionality for the solution now implemented two modules existed. These can be extended easily due to their logical and clear structure. The API module is documented in Table 4 and the GUI module is documented in Table 5.

API		
File Name	**Package**	**Description**
Image.java	API	Implements serializable, cloneable. Contains the string path to an image and the wmpca/pca data for an image, storing faces, regions, and pixel data.
LookupResult.java	API	Stores a result integer and a double distance. The result refers to a face index, and the distance refers to the distance from the mean face.
GoogleMail.java	API.audit	A class used for handling emailing the results of the webcam task.
EigenfaceCache.java	API.database	Implements serializable, cloneable. Contains all the data necessary to perform PCA or WMPCA, including the original image data (or locations).
Person.java	API.database	Implements serializable. Contains a string name and file location – the name refers to the name of a person and the file refers to the image for that person.
PersonDatabase.java	API.database	Implements serializable. Contains a work directory, a list of people in the database, and the WMPCA / PCA data.
AlgorithmHelper.java	API.helpers	Abstract class. Contains helper functions for during the calculation of the PCA or WMPCA algorithms.
DeepCopy.java	API.helpers	Contains functions necessary to deep copy multi-dimensional arrays.
TrainingThread.java	API.helpers	Implements runnable. Used to perform much of the pre-processing necessary for facial verification on a part of a set or a complete set of images.
Log.java	API.logger	Contains the functions necessary for the API to log messages and for an external entity to retrieve messages.
LookupPerson.java	API.PCA	Extends AlgorithmHelper. Used to lookup an unknown image and return what people have been found within it for the PCA algorithm.
PCA.java	API.PCA	Extends AlgorithmHelper. Contains all the core processing for the PCA algorithm.
PCAPreProcess.java	API.PCA	Extends PCA, implements runnable. Contains the pre-processing code that sets up the data for the main PCA algorithm.
FaceRecognition.java	API.recognition	Contains the functions necessary to detect face coordinates within an image, calls OpenCV to achieve this.
LookupPerson.java	API.WMPCA	Extends AlgorithmHelper. Used to lookup an unknown image and return what people have been found within it for the WMPCA algorithm.
WMPCA_Worker.java	API.WMPCA	Extends PCA implements Runnable. Acts as a worker thread for the WMPCA algorithm, passes a set amount of data into the PCA algorithm and runs it.
WMPCA.java	API.WMPCA	Extends AlgorithmHelper, implements Runnable, Serializable. Contains most of the code for the WMPCA algorithm.

Table 4. The class definitions for the API component.

GUI		
File Name	**Package**	**Description**
MainApp.java	GUI	Extends Application. Contains the main function, sets up and launches the GUI.
ImageController.java	GUI.model	Contains the vast majority of the GUI code. It is the main frame that is displayed.
ProjectImage.java	GUI.model	A class used for storing a file name and path.
TileViewResults.java	GUI.model	Implements EventHandler<ActionEvent>. Populates the TilePane control with match results from the live recognition input feed.
WebcamResult.java	GUI.model	Contains an image, string name, and id. Used as the data type returned from the webcam task.
WebcamTask.java	GUI.model	Extends Task<Void>. Handles the webcam live input feed and the matching of results.

Table 5. The class definitions for the GUI component.

7. Testing: Verification and Validation

7.1. Facial Verification

The main feature of the Dumbledore application is the WMPCA algorithm implemented at a software level in order to achieve suitable rates of facial verification. To test whether this was achieved a number of tests are required. The University of Essex provides four facial databases an increasing level of difficulty to learn (faces94, faces95, faces96, and grimace) [27]. Each database has a variety of images all containing frontal-facing people. Each facial verification algorithm can be tested against these databases by manually determining an appropriate threshold and then running the entire database with the algorithm at that threshold.

To run tests against each database at a set threshold it does not require the GUI, and thus a testing module was devised. The testing module contains code that loads in a facial database and then tries to determine whether the face in each image meets the threshold set. The images rejected as invalid matches are output so that they can be manually checked to see if the threshold is correct or incorrect.

The first set of tests conducted used a standard face size of 50 by 50 pixels. For the WMPCA algorithm three regions were used. 80% of eigenvectors were retained each time. The results of this against the University of Essex's facial databases are shown in Figure 25 and Table 6. From these results it can be observed how WMPCA generally performs marginally better against PCA in all tests. Both algorithms performed worst with the faces96 database which is likely due to the dramatic variation in pose, illumination, and background. The grimace database had more consistent backgrounds and facial sizes within each image. With the faces94 database the results achieved were much higher than ever anticipated, however this was the easiest facial database and is not representative of a classroom situation. This method of testing suffers from false positives and false negatives due to threshold being highly variable on a person to person basis. However it provides a general overview and is the likely method that would be used on a classroom when automating recognition.

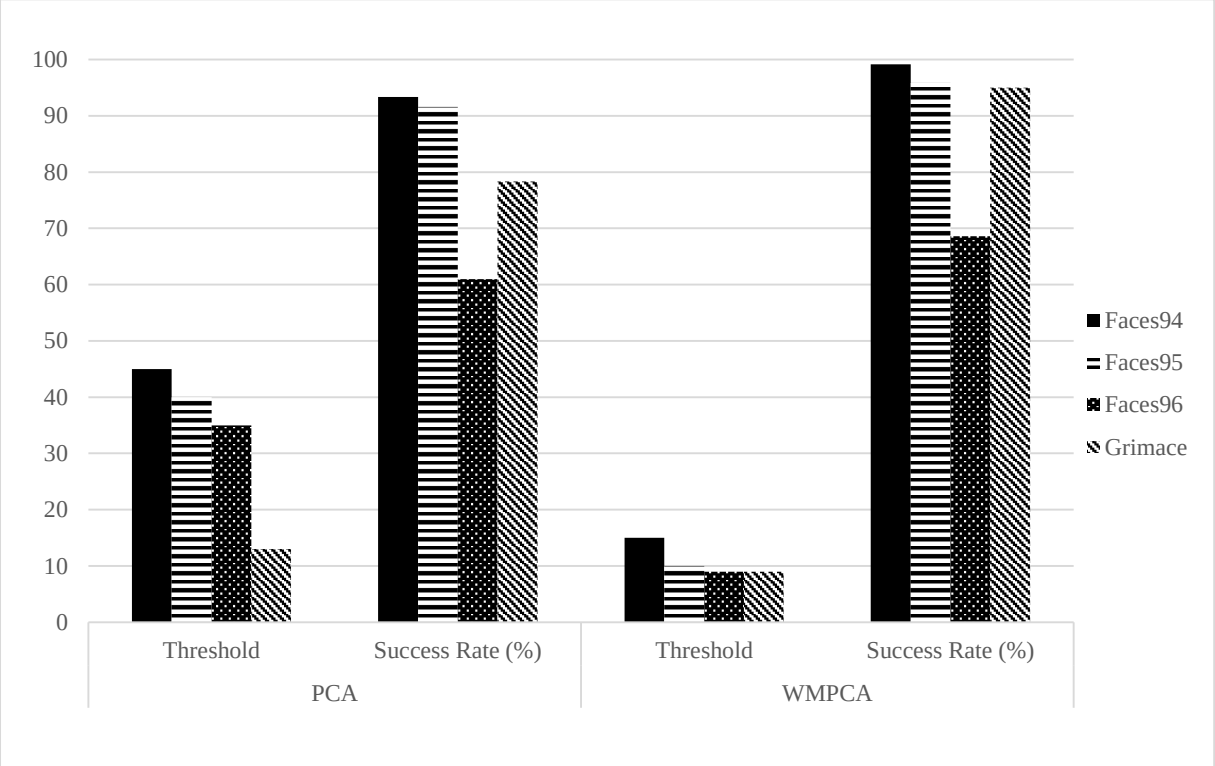

Figure 25. The results of the first set of tests.

Database	Algorithm	Threshold	Success Rate
Faces94	PCA	45	93.37%
Faces94	WMPCA	15	99.16%
Faces95	PCA	40	91.6%
Faces95	WMPCA	10	95.83%
Faces96	PCA	35	60.99%
Faces96	WMPCA	9	68.59%
Grimace	PCA	13	78.33%
Grimace	WMPCA	9	95%

Table 6. The results for the first set of tests.

These tests on WMPCA were done with number of regions set to three. This is because three was deemed to be the ideal number of regions in the test data used. This was proved with a test on the faces95 database where nine images of different individuals were removed from the database at random and attempted to be verified as unknown input images when the database with these images was trained. The face size was set to fifty by fifty pixels and 80% of eigenvectors were retained. The results are given in **Table 7**. Three being the ideal number of regions is probably because any more regions results in the amount of data available (pixels) being far too small to make much meaningful analysis. A fifty by fifty pixel face split ten times horizontally results in each region being five pixels tall.

Number of Regions	Unknown Image (✘ = fail, ✓ = pass)									Success Rate
	1	2	3	4	5	6	7	8	9	
2	✓	✘	✓	✓	✘	✓	✓	✘	✘	55.6%
3	✓	✘	✓	✓	✓	✓	✓	✘	✘	66.7%
4	✓	✘	✘	✘	✘	✘	✓	✘	✘	22.2%
5	✓	✘	✘	✘	✘	✓	✘	✓	✘	33.3%
6	✓	✓	✓	✘	✘	✘	✘	✘	✘	33.3%
7	✘	✘	✓	✘	✘	✘	✓	✘	✘	22.2%
8	✓	✘	✓	✓	✘	✘	✓	✘	✘	44.4%
9	✘	✓	✘	✘	✓	✘	✘	✘	✘	22.2%
10	✘	✓	✘	✓	✘	✘	✘	✘	✘	22.2%

Table 7. Table showing different number of WMPCA regions success rate.

7.2. Real-time Support

It was necessary for the Dumbledore application to be able to achieve facial recognition and verification in real-time. The live input feed successfully ran at ~20 frames per second, and could easily run faster if set to. Furthermore the application was able to handle multiple faces in a single image as shown in Figure 26. It is clearly shown how the two people are being recognised by the system and the closest match displayed in the tile view.

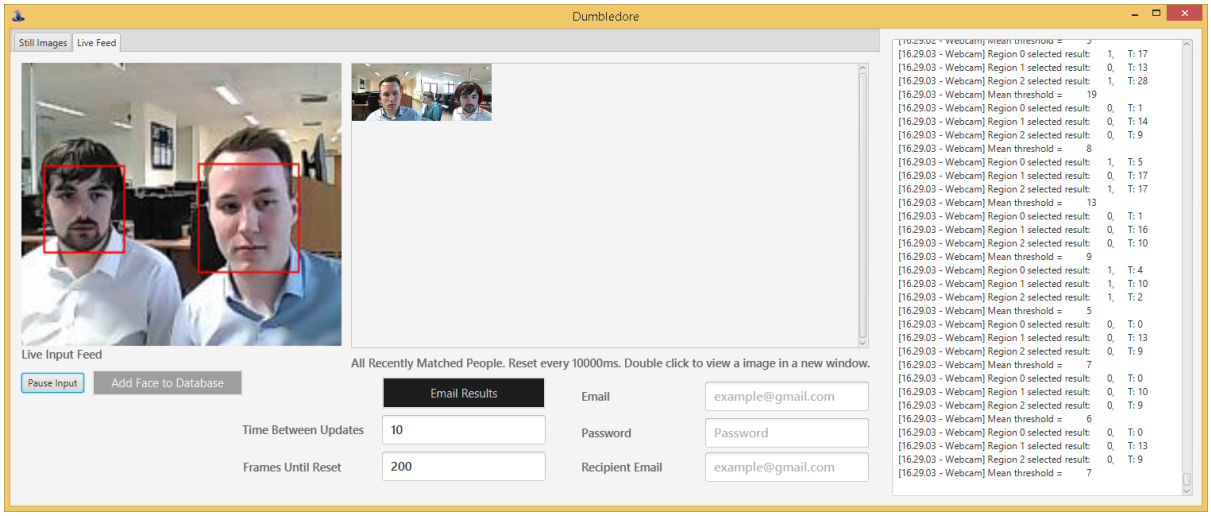

Figure 26. Multiple faces being detected and processed in the same frame.

7.3. Acceptance Requirements

The acceptance requirements given in 3.4 were required to be fulfilled in order to pass testing and deem the project a success. This has been completed within Table 8.

ID	Criteria	Failed / Met / Exceeded	Comments
1	The GUI must be suitable for use by a normal user. This can be tested against peron(s) not in the SSE department and is a qualiative measure.	Met	This was informally tested with willing participants. The controls needed explaining in order to describe what each button precisely did and the overall aim, but it was not deemed confusing once explained.
2	The GUI must provide functionality to report the presence or absence of each student in a lesson/lecture to an external entity.	Met	The GUI can be used to email the results of each student in a database being detected absent or present. This information is also rendered in a tile view.
3	The face verification must perform at an adaquate success rate (>= 70% is ideal).	Met / Exceeded	With most databases WMPCA achieved extremely high success rates at >= 70%, however the most realistic database (Faces96) achieved only a ~60% success rate, which is still okay.
4	The system must be able to detect multiple people from a given image if multiple people exist.	Exceeded	Multiple people can be detected each frame and are all rendered appropiately, as illistrated in Figure 26.
5	The system must perform in real time. (Able to process at least 1 frame per second.)	Exceeded	This was achieved easily with ~20 frames per second. However, given the literature review this requirement was not expected to be difficult. Facial verification has been achieved with the rasberry pi [16] [20].
6	The system must be able to cope with changing illumination and pose.	Met	This is difficult to measure. The faces96 database has the most variation in illumination and pose and the success rate for this database was significantly lower because of this. However Table 6 and Figure 25 does show that a moderate success rate can be achieved across all the databases showing WMPCA can somewhat deal with these problems.

Table 8. Acceptance requirements realised.

Furthermore it was necessary for the objectives set out in the PID to be met.

Objective 1 - Facial verification must be able to be applied to identify faces when compared against a database in a classroom setting.

This objective has been completed in that databases are created and trained for each classroom situation. When a database is loaded the live recognition input can match what is being captured to the database in order to identify students.

Objective 2 - The Dumbledore project must be presented in a rich and intuitive graphical user interface that renders the results of the facial verification.

This objective has been completed in that the GUI provides the functionality to achieve everything the proof of concept set out to do. The webcam capture and results are rendered using a combination of JavaFX controls. Furthermore the controls are styled in CSS such that is very easy to change the colour schemes of the entire GUI.

Objective 3 - The results must be able to be audited properly by automatically sending them through some means to another external entity.

This objective has been completed in that the results for live recognition over a period of time can be emailed. The username and password for the account to send from must be input at runtime because it is not safe to keep this information in a plaintext database. A password field is used so that it cannot be seen visually and is only kept in memory.

Objective 4 - The facial recognition and verification must be accurate enough as to be practical use.

This is quite a qualitative objective and as such it is hard to measure a precise pass rate. Given Table 6 and Figure 25 it is clear a moderate success rate is achieved. The only database it struggled with was faces96, however this also had a great variance between each image in terms of pose and illimination. Over a long period of time results can be analysed to determine if the same people are being matched and thus this allows for a much higher success rate over a longer period of time.

8. Discussion

The results achieved by the WMPCA algorithm tests determine that the algorithm is able to work with a wide variety of data and still achieve okay success rates. It is important to distinguish that these tests were done with large facial databases consisting of hundreds of images. The Dumbledore application is likely to only use data sets consisting of between 10 and 200 images in its real use due to this being the typical size of lectures. With a smaller data set it is a lot easier to determine matches since there are less values within the same eigenspace. This may be offset by a lecture possibly having even greater variations in brightness and pose than the facial databases tested here. The literature review determined that PCA was used by other similar concepts to Dumbledore and achieved adequate levels of facial verification. This means that it is likely that the results gained here are accurate in coming to a similar conclusion.

A possible improvement to the facial verification results would be to operate in the HSV colour space rather than the RGB colour space because HSV is less affected by lighting conditions, as RGB includes luminance within its data [29]. This would require a significant redesign on the way pre-processing is achieved however.

It was determined the ideal number of regions to use given the faces95 database, however this is going to be variable depending on the standard size of a face used and the actual data itself. The threshold to use to determine a match is also highly variable on a person to person basis and depending on the database. Thus the logical solution to this is to have these values configurable per classroom database which has been implemented. As a possible future implementation a configuration can be implemented for each person which would allow for better flexibility with threshold.

The application was successfully able to achieve real-time facial recognition and verification with ease. It could do this with multiple people detected in each frame. The results were rendered well and more details could be retrieved about a specific match by double clicking it in the tile pane. Real-time was tested both with large databases and small databases to confirm that it functioned well across a variety of data. These tests were conducted on modern hardware that is significantly faster than previous studies have used but previous studies also suggest that powerful hardware is not required as detailed in the literature review.

The training of the data is the only part of the developed prototype that takes a noticeable time to execute. Using WMPCA the training of data is very fast because each region is processed on a separate thread and modern hardware is able to process these threads in parallel. However with PCA the matrixes created during training are vast and require significant processing time to learn. This process only needs to be conducted on the initial training of the data, and then the results can be saved to a database. The database can be loaded in a matter of seconds so that attendance analysis can begin immediately at the start of each lecture.

All the original objectives were met or exceeded which means that the developed solution acts as a successful prototype for the Dumbledore application. The GUI developed is easily extended due to how the controls are styled in CSS and the MVC approach used. It is also aesthetically pleasing and features a wide variety of controls that have JavaFX provides. Subsequently this design used provides a rich and intuitive GUI. The only other major GUI library for Java is Swing [30] which does not provide the same features natively.

9. Conclusion

Based on the original objectives the project was a complete success. WMPCA was an advanced goal for the project and was implemented successfully and achieved the results expected from the design and literature review. The facial recognition and verification both function as required and work for a proof of concept but have not been tested in an actual lecture environment, which is where this application would need to be developed for in the future if extended. WMPCA proved more successful than PCA as theorised which is shown throughout the testing.

Solution C was not implemented due to time limitations and it was only a possible extension. It could be implemented in a fully developed version of Dumbledore where more sensitive information about each student may be stored. In the current proof of concept only names and image data is stored which is not that sensitive.

The PID stated as an advanced goal to integrate avatar technology. This would have allowed for the rendering of the lecture environment with pupils in a virtual world using software like Second Life [31]. This would have voided the problems of illumination and pose and would have allowed for an impressive demonstration of the proof of concept. It was chosen not to develop this solution early in the project due to OpenCV struggling to recognise virtual faces within the Second Life application.

The proof of concept is easily extendable, both in terms of the API and GUI. The GUI uses the MVC framework provided by JavaFX which is highly modular and thus very easy to extend. Furthermore the control layout is done using XML which is straightforward to modify. The API is designed such that each key piece of functionality is separated into its own package and thus it is both easy to navigate and highly modifiable. Because of this the developed solution provides a strong base that could be developed into a full application at a later date.

10. Project Commentary

The plan created in the PID that set out the milestones and roadmap was created at a time when the author had very little knowledge on how facial recognition and verification technology worked and the way the technologies would integrate into the Dumbledore application. As such it took much longer to implement the features and to test them than expected.

The main problems that occurred during the development of the project occurred during the implementation of PCA and WMPCA. There was a lot of mathematical concepts introduced that the author had no prior knowledge off and subsequently when data did not match exactly that which was expected it was difficult to debug. As such the implementation took a very long time and took many revisions to be completed. The concepts are now fully understood by the author and the algorithms functioning as expected.

Git [32] is a version control software that was used throughout the projects development to control the progression of the software. It allowed for the development of the code incrementally and allowed for rollbacks or revisions to be made as deemed necessary. As such the entire development lifecycle of the projects code can be viewed from its inception to its final form.

11. Social, Legal, Health & Safety, and Ethical Issues

The primary concern of Dumbledore's concept is that of the legality and ethicality of collecting image data of students. During the development of this application the consent of each subject was obtained for it to be used in this project. The literature review highlights how an attendance analyser has been implemented within the UK before. As a proof of concept and due to the precautions taken here the application has not had any SLEASE issues.

If this project was developed into a full application then further SLEASE analysis would need to be done. Solution C outlines an extended version of Solution A whereby all

sensitive data is encrypted. However such an implementation does not necessarily remove all SLEASE concerns.

12. Reflection

The project has been a valuable learning experience and exposed me to many subjects that I have never encountered and it helped develop my existing skill asset. It is a very large code base and as such was difficult to manage at times. Designing and structuring this logically and incrementally was a challenge but provided valuable lessons. It also further developed my time management skills because it was not possible to conform to the original PID's timeline and as such management of the projects progress was continuously required. Achieving the goal of implementing WMPCA at a software level was a significant technical achievement and I am happy with the project in its current state.

13. References

[1] W. Zhao, R. Chellappa, A. Rosenfeld and P. J. Phillips, "Face Recognition: A Literature Survey," National Institute of Standards and Technology, 2003, p. 1.

[2] C. Lu and X. Tang, "Surpassing Human-Level Face Verification Performance on LFW with GaussianFace," 2014. [Online]. Available: http://arxiv.org/pdf/1404.3840v2.pdf. [Accessed 20 04 2014].

[3] H. Lynas, *PID - Project Initiation Document.*

[4] M. Turk and A. Pentland, "Eigenfaces for Recognition," Massachusetts Institute of Technology, [Online]. Available: http://www.face-rec.org/algorithms/PCA/jcn.pdf. [Accessed 9 4 2015].

[5] A. P. Kumar, S. Das and V. Kamakoti, Face Recognition using Weighted Modular Principle Component Analysis, Indian Institute of Technology Madras.

[6] "OpenCV - Open Source Computing Vision," [Online]. Available: http://opencv.org/. [Accessed 08 01 2015].

[7] "MySQL, The world's most popular open source database," Oracle, [Online]. Available: http://dev.mysql.com/downloads/mysql/. [Accessed 13 4 2015].

[8] F. Information, "Announcing the Advanced Encryption Standard (AES)," Processing Standards Publication 197, 26 11 2001. [Online]. Available: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf. [Accessed 13 04 2015].

[9] P. Sinha, B. Balas, Y. Ostrovsky and R. Russell, "Face Recognition by Humans: 19 Results All Computer Vision Researchers Should Know About," Proceedings of the IEEE Vol. 94 No. 11, 2006, pp. 1948-1962.

[10] "AurorA - The Face of Biometrics," [Online]. Available: http://www.facerec.com/index.html. [Accessed 11 4 2015].

[11] Lathem, "FaceIN Facial Recognition Attendance System," Amazon, [Online]. Available: http://www.amazon.com/FaceIN-Facial-Recognition-Attendance-System/dp/B004HZ50P4. [Accessed 13 04 2015].

[12] Hanvon, "Time attendance FK800," [Online]. Available: http://www.hanvon.com/en/products/Faceid/products/fk800.html. [Accessed 13 4 2015].

[13] Y. Kawaguchi, T. Shoji, W. Lin, K. Kakusho and M. Minoh, "Face Recognition-based Lecture Attendance System," Kyoto University, [Online]. Available: http://www.mm.media.kyoto-u.ac.jp/old/research/doc/682/FRLASinAEARU.pdf. [Accessed 11 04 2015].

[14] M. Fuzail, H. M. F. Nouman, M. O. Mushtaq, B. Raza, A. Tayyab and M. W. Talib, "Face Detection System for Attendance of Class' Students," in *International Journal of Multidisciplinary Sciences and Engineering, Vol. 5*, University of Engineering and Technology, Lahore, Pakistan, 2014, pp. 6-10.

[15] A. Patil and M. Shukla, "Implementation of Classroom Attendance System Based on Face Recognition In Class," in *International Journal of Advances in Engineering & Technology, Vol. 7, Issue 3*, IJAET, 2014, pp. 974-979.

[16] "Raspberry Pi FAQS," Raspberry Pi Foundation, [Online]. Available: https://www.raspberrypi.org/help/faqs/#performanceSpeed. [Accessed 11 4 2015].

[17] N. K. Balcoh, M. H. Yousaf, W. Ahmad and M. I. Baig, "Algorithm for Efficient Attendance Management: Face Recognition based approach," in *International Journal of Computer Science Issues, Vol. 9, Issue 4, No 1*, IJCSI, 2012, pp. 146-150.

[18] U. A. Patel and S. P. R, "Development of a Student Attendance Management System Using RDIF and Face Recognition: A Review," in *International Journal of Advance Research in Computer Science and Management Studies, Vol. 2, Issue 8*, ISSN 2321-7782, 2014, pp. 109-119.

[19] L. I. Smith, "A tutorial on Principle Components Analysis," 26 2 2002. [Online]. Available: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf. [Accessed 13 4 2015].

[20] A. P. Kumar, V. Kamakoti and S. Das, "An Architecture For Real Time Face Recognition Using WMPCA," Indian Institute of Technology Madras, Chennai - 600036, [Online]. Available: http://www.cse.iitm.ac.in/~vplab/publi_journal/conference/frarc.pdf. [Accessed 13 4 2015].

[21] "JavaFX Overview," Oracle, [Online]. Available: http://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm#A1131418. [Accessed 14 4 2015].

[22] "GitHub - OpenCV," [Online]. Available: https://github.com/Itseez/opencv/blob/master/data/lbpcascades/lbpcascade_frontalface.xml. [Accessed 16 4 2015].

[23] "Apache Commons," 8 1 2015. [Online]. Available: http://commons.apache.org/proper/commons-math/. [Accessed 16 4 2015].

[24] "imgscalr - Java Image Scaling Library," The Buzz Media, [Online]. Available: http://www.thebuzzmedia.com/software/imgscalr-java-image-scaling-library/. [Accessed 16 4 2015].

[25] "JAMA: A Java Matrix Package," [Online]. Available:

http://math.nist.gov/javanumerics/jama/. [Accessed 17 4 2015].

[26] "Webcam-Capture," [Online]. Available: https://github.com/sarxos/webcam-capture. [Accessed 17 4 2015].

[27] L. Spacek, "Face Recognition Data," University of Essex, 20 6 2008. [Online]. Available: http://cswww.essex.ac.uk/mv/allfaces/index.html. [Accessed 17 4 2015].

[28] "StackOverflow - Send email using Java," [Online]. Available: http://stackoverflow.com/questions/3649014/send-email-using-java. [Accessed 21 4 2015].

[29] M. Sedlacek, "Evaluation of RGB and HSV models in Human Faces Detection," Slovak University of Technology, [Online]. Available: http://www.cescg.org/CESCG-2004/web/Sedlacek-Marian/. [Accessed 23 4 2015].

[30] "Trail: Creating a GUI with JFC/Swing," Oracle, [Online]. Available: http://docs.oracle.com/javase/tutorial/uiswing/. [Accessed 23 4 2015].

[31] "Second Life," [Online]. Available: http://secondlife.com/. [Accessed 24 4 2015].

[32] "git," [Online]. Available: http://git-scm.com/. [Accessed 24 4 2015].

[33] L. I. Smith, "A tutorial on Principle Components Analysis," 26 2 2002. [Online]. Available: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf. [Accessed 16 4 2015].

14. Appendices

14.1. Appendix 1: Project Mandate

The aim of this project is to develop (primarily) a software system to automate the task of attendance verification, in a classroom scenario, using computer vision concepts --- face recognition. The project, overall, involves designing, implementing and "field testing" a complete hardware/software system. The input to the system will be fed through a standard web-cam of sufficient quality to demonstrate the system's online operational capability. A high-quality implementation has the potential for the system to be used in realistic classroom context in educational places. If successful, attendance verification will never be the same again! The core component of the system will be a real-time face recognition software based on the well-known and successful Eigenface approach. A range of techniques will be implemented and evaluated in the project in stages as follows:

• Design and implement a face recognition base system with the standard Eigenface /PCA technique. Evaluate the system for its performance --- in particular it's robustness.

• Enhance the robustness of recognition by implementing the weighted modular PCA (WMPCA) technique. Perform experimental validation to verify the robustness claim!

~~• Implement the WMPCA on the latest NVidia GPU massively parallel multi-core supercomputer with 128 processors to realise real-time recognition performance.~~

• Incorporate possible enhancements to the system to improve robustness (pose, expression, colour input, etc.)

• Incorporate "audit" functionality in the software (automatic email to non-attendees, update records etc.).

In addition to the core recognition software component, there is a requirement to design and implement a GUI for displaying the attendance visually (perhaps blending with "avatar" technology) and to allow for manual updates to the output of the verifier. This gives you the opportunity to unleash your creative potential in designing a slick, clean and artistic interface to portray in a rich form the mystical powers of a Dumbledore!

14.2. Meeting Summaries

Date	**Comments**
20/06/2014	Discussed the project outlines, what to do initially and the possible outcomes of the project.
06/10/2014	Discussed what to focus on, initial thoughts, will need to revise PID.
20/10/2014	Emailed about status of project.
27/10/2014	Discussed PCA and how it extends into WMPCA.
10/11/2014	Discussed how covariance matrix will take a long time for large sets of data, might be possible to add to existing matrix rather than recreating each time. Compare new images being measuring the eigenspace.
08/12/2014	Discussed project progress – PCA is implemented but low success rate on currently tested data. Next step is bug testing and WMPCA. Progress is good.
19/01/2015	Discussed the issue of PCA requiring an input with the same sized dimensions. WMPCA splits the face up horizontally such that it would be hard to obtain the same dimensions for the inputs. Unsure of the best solution currently.
09/02/2015	Organised to do the project demo on 11/02/2015 at 14:00, lots to prepare before then.
11/02/2015	Project demo. WMPCA and PCA needs to be compared more clearly with exact percentages on success rate and better test data. Discard first three eigenfaces to help with brightness issues. Add unknown faces to training set. Graph results, add basic auditing features, and consider reducing debug info to interface. Can begin the write up realistically now.
20/03/2015	Discussed a lot – see notes on 20/03/2015.

14.3. PID

Attached at the end of this document.

Project Initiation Document

Dumbledore: A Vision Based Attendance Analyser

Author: Harry Lynas

Module: SE3IP11
Project ID: 220

Date: 02/07/2014
Document Version: 1
The current status of this document is: Draft

Primary Supervisor	M. Manjunathaiah	
Secondary supervisor		
Supervisor's check: For each item below the supervisor should circle ✓ if it is satisfactory for this stage of the project; if it is not satisfactory then the supervisor should circle ✖ and add a suitable comment to indicate the deficiency. The supervisor should then sign below to confirm the ✓,✖ and comments.		
CHECKLIST		Comments, concerns and recommendations
Background	✓ ✖	
Problem statement	✓ ✖	
Technical Products	✓ ✖	
Crosscheck	✓ ✖	
Purchases	✓ ✖	
Health & Safety	✓ ✖	
Social, Legal & Ethical	✓ ✖	
References	✓ ✖	
Project Plan	✓ ✖	
Supervisor's Signature		**Date:**
Submission rules: The PID, with this form completed and signed by the supervisor, must be handed in to the student information centre (G47) AND submitted online on Blackboard by 1030 the appropriate deadline (1030 on Friday 4th July 2014 for students on SE3IP11, and 1030 on Friday 10th October for students on SE3GP11, SE4IP11, SE4RP11 and those students on SE3IP11 that are returning from placement or suspension). If the deadline is not met, the student will face a penalty of 5 marks being deducted from their overall module mark. It is expected that students will address any comments noted by their supervisor on this form in due course.		

School of Systems Engineering, The University of Reading

1 Introduction

This document initialises the project originally mandated in [1]. It clarifies *what* is expected and *how* it is to be done. As in any project initiation there may be uncertainties and they are identified where possible using the term TBD – to be determined.

The background motivating this project is summarised below in §2. A concise statement of the objectives of this project is presented in §2. How these objectives are expected to be satisfied is summarised with a headline list of *products*[1] listed in §4.

The project is undertaken under the auspices of the final year project module SE3IP11 and this mandates that the examination products listed in §9 are delivered as well as the technical products listed in §4.

An assurance that the intended technical products are likely to be adequate is given in §5 where the objectives and technical products are crosschecked. §9 provides an outline plan and shows the plausibility that delivering the products (both technical and examination) is feasible.

The purchases expected in the plan are listed in §6. Mandatory safety, social, legal and ethical concerns are addressed in §7 and §8 respectively. All references referred to in this document are listed in §10.

2 Background

Facial recognition and verification has become more prevalent within the last decade as better algorithms have been developed and hardware has become faster. Traditionally humans have always been able to outperform machines in face verification but recently machines have been able to outperform humans for the first time[3].

Using these algorithms it would be feasible to implement a real-time classroom attendance system that is able to automatically capture which students are in attendance, render this data in a rich user interface, and report this data back to a foreign entity. This would mean that lecturers and teachers would no longer need to spend time taking registration as the entire system would be automated and effective.

The primary concern of this system would be the ethicality and legality of it as a growing number of privacy concerns are being presented[4].

3 Problem Statement

Faces in a classroom setting must be recognised. There could be many faces visible at any given moment so the algorithms will need to be able to identify each face. Each face will need to be verified against a set of known faces within a database to see if that person is known, and if they are, mark that that person is in attendance.

[1] A *product* in this context is something completed by the activity of the project. This is terminology of PRINCE2 project management. Some other PM regimes refer to *deliverables*.

There are many different algorithms that could be used for facial verification. WMPCA (weighted modular principle component analysis) will be researched and focused on initially and then further algorithms will be tested against it to measure performance.

Upon students having being identified, this should be marked down and stored internally. After a set amount of time the system should report back to a foreign entity the attendance results it has measure, such as via email or a prepared structured query language statement into a database.

Each process should be rendered with an intuitive graphical user interface that demonstrates the capabilities of the Dumbledore project. The interface should provide both information regarding the state of the system and live results.

This project assumes that permission is gained to test the system on a real life audience and that the data can be collected to verify each tester against the database.

It also assume that the facial recognition will work on virtual reality avatar software such as second life[5] for testing purposes before testing on a live audience. The ‘avatars' must have enough human characteristics as to be detected and recognised properly.

On the basis of the soundness of the following assumptions, §3.3, it is deemed worthwhile to satisfy these objectives, §3.1, by delivering technical products subject to these constraints, §3.2, on how they may be developed and/or delivered.

3.1 Objectives

1 - Facial verification must be able to be applied to identify faces when compared against a database in a classroom setting.

2 - The Dumbledore project must be presented in a rich and intuitive graphical user interface that renders the results of the facial verification.

3 - The results must be able to be audited properly by automatically sending them through some means to another external entity.

4 - The facial recognition and verification must be accurate enough as to be practical use.

3.2 Constraints

The quality of the input – being a webcam or a virtual reality – directly impacts upon the system’s ability to recognise and verify faces.

The virtual reality technology and/or input device must have sufficient accessible functionality as to achieve the functionality required.

The lighting conditions must be sufficient for the algorithms to be able to detect and verify faces.

3.3 Assumptions

It is assumed that lighting will be sufficient and that the input device will be located in an adequate position.

It is also assumed that the machines used to run the algorithms will be powerful enough as to process them in a suitable time frame.

4 Technical Products

It is clear that there are a few primary technical products that will be delivered.

The first is ‘Product 1’ which consists of the facial recognition and verification algorithms. This product will allow for the recognition of faces from the input and the ability to match that against a known set of faces within a database. For this product to be accepted it must function as expected, and the main risk is the input not being of sufficient quality for the algorithms to use properly.

The second technical product is ‘Product 2’ which consists of the graphical user interface. This needs to be rich and intuitive, perhaps making use of virtual reality avatar technology. For this product to be accepted it must be original and function as required. The main risk is that the virtual reality technology is difficult to interface with if this route is taken.

The third and final technical product is ‘Product 3’ which consists of the audit functionality. This must be able to take the verification results and transfer them to an external entity. For this to be accepted at least one method of transferring the result must be implemented and working. There is little risk of this product not being completed.

5 Crosscheck of approach

	Product 1	Product 2	Product 3
Objective 1	X		
Objective 2		X	
Objective 3			X
Objective 4	X		

In order for the first product of facial detection and verification to be completed it is necessary for objective 1 to be complete. Objective 4 is needed to be completed in order for this product to be delivered.

Similarly objective 2’s primary aim is to provide a rich and intuitive graphical user interface which will deliver the second product.

Objective 3 is directly dependent on the ability to deliver the third product.

Given §9 it is clear that each product should be able to be achieved within the given time constraints unless something catastrophic happens. Each product is able to be extended, improved, and polished as more time becomes available due to potentially finishing deadlines early. If any deadlines overrun then any time left near to product delivery that previously would have been used for improving the implementations can instead be used instead to make sure the core functionality works as expected.

6 Purchases

There are no necessary purchases.

7 Health and Safety

The only potential health and safety issue that is worth taking into account is working alone in possible out of hour's conditions. To be safe in these conditions, somebody will always be nearby in the building and access will be logged within the buildings protocols.

8 Legal and Ethical

There are some ethical and legal potential issues with this project. Should it ever reach a point in time where the system is deemed suitable to test on live subjects, full consent will need to be obtained that the persons can be electronically captured and the data destroyed after the tests have been completed.

9 Examination Products

The mandated examination products are as follows. Risks identified that may obstruct their completion are listed in[2] with cross-reference to this section.

EX1. Name: PID

- **Short Description**: The document that initiates this project and guides it thereafter.
- **Acceptance Criteria**: The PID must be written as a Microsoft Word or PDF document in compliance with the format and content instructions indicated in this document (the PID Master). All sections should be complete to a reasonable depth and quality, consistent with the stage of the project, that satisfies the technical judgement of the project supervisor and, if appropriate, the Company Partner. The sign offs on the cover page will be completed.

EX2. Name: Autumn Term Week 6 Logbook Check

- **Short Description**: A formal check of project progress based on the logbook.
- **Acceptance Criteria**: The Student Information Centre staff, and Project Co-Ordinator if necessary, must be convinced the project progress has been sufficient to date and that this progress has been adequately recorded in the logbook. The logbook must include a sign-off sheet that indicates that the logbook has been assessed by the supervisor on a weekly basis.

EX3. Name: Project Progress Review

- **Short Description**: A formal check of project progress based on a checklist form.
- **Acceptance Criteria**: The 'Project Progress Review Individual Form' must be completed by the student and supervisor and signed and submitted by the appropriate deadline.

EX4. Name: Spring Term Week 6 Demonstration

- **Short Description**: A formal presentation of the executable technical products of the project to the project supervisor and internal moderator.
- **Acceptance Criteria**: During Week 6 of Spring Term, the student must demonstrate their project to their supervisor and internal moderator. The demonstration should show that the project's technical products are on track to be completed by the end of the project.

EX5. Name: Poster

- **Short Description**: A poster summarising a significant aspect of your work.
- **Acceptance Criteria**: A poster is constructed based on the template that will be provided by the university. It will be of a quality typical of an academic conference poster presentation. The content will introduce and explain some aspect, claim of other facet of the project that the student deems most interesting and which is agreed by the supervisor.

EX6. Name: SCARP Abstract

- **Short Description**: An abstract in a form typical of an academic conference.
- **Acceptance Criteria**: The abstract is written according to the academic template provided by the school of systems engineering. It will summarise the project with a focus on its achievements with respect to the objectives and technical products and what was learned.

EX7. Name: SCARP Paper

- **Short Description**: A short paper in a form typical of an academic conference.
- **Acceptance Criteria**: The paper is written according to the academic template (TBD). It will describe the project with a focus on its achievements with respect to the objectives and technical products and what was learned.

EX8. Name: Final Report

- **Short Description**: The academic write up of the work achieved by the project in addressing the technical problem.
- **Acceptance Criteria**: The report shall be formatted according to provided rules by the school of systems engineering. It shall have content assessable according to the criteria (TBD). It will include sections such as Introduction, Literature Survey, Problem Analysis, Solution Analysis, Implementation, Evaluation (detail TBD).

EX9. Name: Demonstration to Internal Examiners

- **Short Description**: A formal presentation of the executable technical products of the project.
- **Acceptance Criteria**: The working demonstration must be convincing to the examiners in terms of: showing a significant satisfaction of the objectives through the operation of the developed technical products; showing that the work is substantially the product of the student's efforts.

EX10. Name: SCARP Presentation

- **Short Description**: A 10 minute presentation of your work in a conference setting.
- **Acceptance Criteria**: The paper is presented in a clear, coherent manner in the allotted time and plausible answers are given to such questions that are possible in the conference schedule (typically two or three questions).

EX11. Name: Project Archive

- **Short Description**: A CD containing all project documents.
- **Acceptance Criteria**: The CD must contain all the files and folders specified in the guidelines given by the school of systems engineering.

10 References

[1] *Dumbledore: A Vision Based Attendance Analyser*, ID 220, SSE Staff Proposed Project List, Blackboard.

[2] RA1 Risk Document, attached to this document.

[3] Lu, C.L. and Tang, X.T. (2014). *Surpassing Human-Level Face Verification Performance on LFW with GaussianFace*, http://arxiv.org/pdf/1404.3840v2.pdf [28/06/2014]

[4] (2014), *Privacy fears over FBI facial recognition database*, http://www.bbc.co.uk/news/technology-27037009 [01/07/2014]

[5] *Second Life*, http://secondlife.com/, [30/06/2014]

11 Project Plan

11.1 Technical Products

Table 11-1 Technical Products

Product	Task ID	Task Description	Effort (weeks)
Facial Product	T1		
	T1.1	Face Recognition	1
	T1.2	Face Verification	2
	T1.3	Testing & Improving	2
GUI Product	T2		
	T2.1	Researching	1
	T2.2	Implementing	1
	T2.3	Testing	1
	T2.4	Polishing	1
Audit Product	T3		
	T3.1	Implementing	1
	T3.2	Testing	1

11.2 Examination Products

Table 11-2 Examination Products

Product	Task ID	Task Description	Effort (weeks)
Product Initiation Document	EX1	A product initiation document (PID).	1
Logbook Check	EX2	A formal check of project progress based on the logbook.	1
Project Progress Review	EX3	A formal check of project progress based on a checklist form.	1
Spring Term Week 6 Demonstration	EX4	A formal presentation of the executable technical projects.	10
Poster	EX5	A poster summarising significant aspect of the work.	1
SCARP Abstract	EX6	A the abstract in a form typical of an academic conference.	1
SCARP Paper	EX7	A short paper in a form typical of an academic conference.	1
Final Report	EX8	The main write up of the work achieved by the project.	4
Demonstration to Internal Examiners	EX9	A formal presentation.	1
SCARP Presentation	EX10	A 10 minute presentation of the work.	1
Project Archive	EX11	A CD containing all the projects documents.	1

11.3 Time Plan for the proposed Project work

Table 11-3 Time Plan Technical Products

START DATE: 01/07/2014

Product / Product phase	Pre-Term	AUTUMN TERM (weeks)						Break	SPRING TERM (weeks)						Break	SUMMER TERM (exams)
		1-2	3-4	5-6	7-8	9-10	11		1-2	3-4	5-6	7-8	9-10	11		
Facial Product (1)																
GUI Product (2)																
Audit Product (3)																

Table 11-4 Time Plan Examination Products

START DATE: 01/07/2014

Product / Product phase	Pre-Term	AUTUMN TERM (weeks)						Break	SPRING TERM (weeks)						Break	SUMMER TERM (exams)
		1-2	3-4	5-6	7-8	9-10	11		1-2	3-4	5-6	7-8	9-10	11		
PID	X															
Autumn Term Week 6 Logbook Check				X												
Project Progress Review							X									
Spring Term Week 6 Demonstration											X					
Poster														X		
SCARP Abstract														X		
SCARP Paper														X		
Final Report																X
Demonstration to Internal Examiners																X
SCARP Presentation																X
Project Archive CD																X