A Face Authentication System

Tom Bedford

BSc Computer Science, xw009807@reading.ac.uk

ABSTRACT

Facial recognition is the identification and recognition of noticeable characteristics of a human face. In the field of image analysis it is leading the race in research as its speed and versatile application out weights previous methods. It is less invasive than other biometrical analysis methods like retinal and finger print recognition with less hardware requirements. Human to computer authentication is an integral functionality of many software systems as it manages data or location accessibility.

This report explores the application of face recognition and its effectiveness as an authentication system using the open vision library developed by IBM. The recognition algorithm uses Viola-Jones methodology of classification to detect the users face using a trained Haar classifier. PCA (Principal Component Analysis) is then used to train image sets to reduce data representation and extract a given range of Eigen values. Derived Eigen faces are compared and accepted when falling within a given threshold of deviation from the trained data’s average Eigen face.

# Introduction

The general operation of a face authentication system is the identification and recognition of a user through biometric analysis to grant them access to secure data or location. With the advances of more affordable and powerful computer processing hardware providing a platform for more sophisticated computation and innovative methods of Human Computer interaction, facial interpretation as a means of authentication and computer interaction has become a focused area of research.

The Facial authentication system developed in this project uses a derived library of ‘OpenCV’ called ‘JavaCV’ []. This allows the application to be developed using Java programming language whist utilizing functionality of the natively C++ ‘OpenCV’ library. Potentially the Java virtual machine provides a means to develop a cross platform face authentication application. This application could potentially compliment an existing authentication system by providing an extra layer of security or run as a stand-alone security system.

# Application

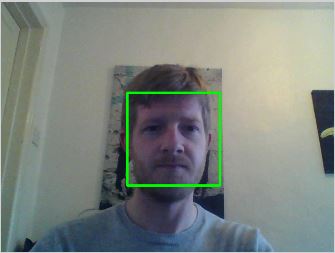
Images are read and normalized from a target database. The normalization process centralizes the image and converts the images RGB colour model to greyscale two channel. This reduces the amount of data that will need to be processed and focuses on most relevant regions of interest. By standardizing all images data comparison during recognition can be performed efficiently by referencing the location of an images features.

The image database is then trained to learn and calculate Eigen faces [2]. To perform recognition either an image is selected from an existing database or created. It is added to the targeted database image set and retrained. Trained Eigen faces are scored by determining the ratios of entity image present within the Eigen Face. This way the input or image for recognition score can be observed and associated to its highest matching entity image.

## Cascade Classification (Haar Classifier)

The OpenCV library offers the use of trained Classifiers that extract Haar like features from the subject image using Viola-Jones methodology of object detection [3]. These classifiers have been taught to detect a specific feature of the human face with the use of linear mathematical observation which recognize objects.

### Face Detection through live camera feed



**Figure 1.** Detected face



**Figure 2.** Captured Image

Figure 1 illustrates the detected face from the live camera feed. A rectangle is drawn over the image to represent the region of interest that has been located. The captured image is normalized and converted to grayscale as seen in figure 2.

## PCA (Principal Component Analysis)

PCA (Principal Component Analysis) methodology is used to reduce the representation of image data which will allow comparing data during recognition to be executed much faster and more efficiently. PCA and recognition is used in other technical analysis fields such as voice recognition and medical image analysis [4]. Essentially PCA derives the most dominant principal components of an images data within in given range which construct Eigen faces. Figure 1 illustrates a sample set of images drawn from the xm2vts face database [5]. The images sourced from this database have already be normalized.

### Sample image set from XM2VTS database

**Figure 3.** Sample image set from XM2VTS



### Eigen faces of sample image set

**Figure 4.** Calculated Eigen faces

Each image from the set is read into a vector and added to a face matrix array known as projecting data into an Eigen Space [6]. Here observations can be made on the Eigen Space to calculate a mean set of component data displayed below as figure 5.

### Eigen Faces



**Figure 5.** ReconstructedMean Eigen Face

By restructuring the data and essentially rotating the data space the algorithm can then search for the highest variances in data and sort them into descending order. These are known as the principal components. A sample of N size components are taken to represent the Eigen vectors. Similarly this shifting data method is commonly used in image analysis for operations such as reducing image noise and can be performed with alternative methods such as DCT (Discrete Cosine Transformation). These Eigen vectors signify the Eigen Faces. It is a commonly known practice to drop a proportion of the highest variance components of the generated Eigen faces to reduce recognizable illumination changes.

## Recognition

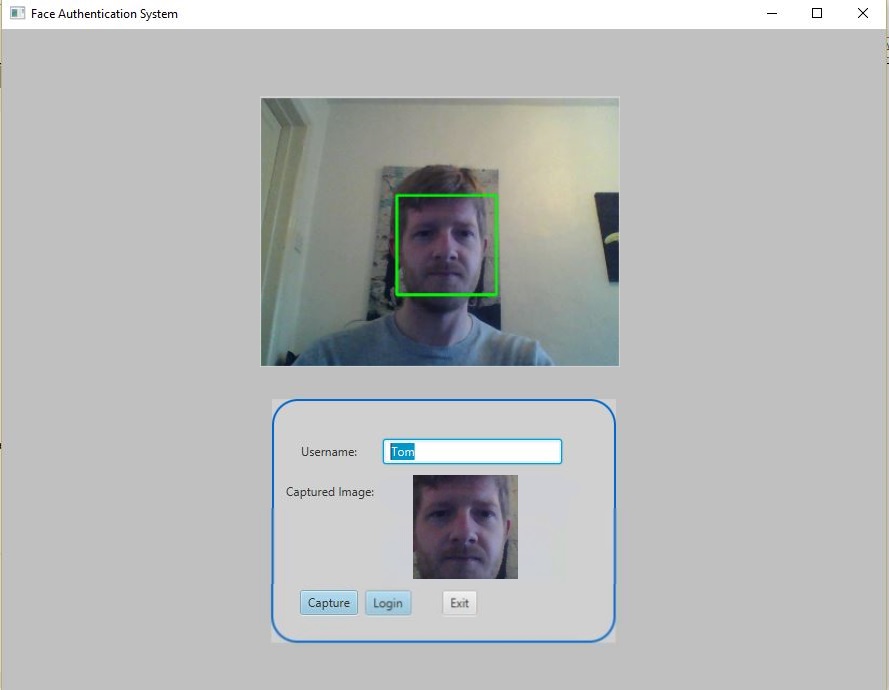
One of the most inflicting environmental aspects of face recognition is the challenges of illumination levels [7].

Once the input image has been added to the training data and re-trained the input images Eigen Face can be scored by calculating the Euclidean distance [8]. The smaller the distance the higher the similarity in facial features. This score is compared to the scores of the mean Eigen face and multiplied by the number of Eigen faces. If the comparison results fall between set thresholds the associated image is return as a successful match.

## Authentication Application

Application functionality will allow a user to create a custom database or add a captured image to an existing database with the face detection algorithm featured in the software. It features a login interface that the user can utilize as a facial authentication system providing the user has added a trained an image of themselves. Figure 6 illustrates a snap shot of the user login sceen. Here the user can capture an image and present it as the input image to be added to the training set.

### Screen shot of login screen



**Figure 6**. Screen shot of application login screen

The application also features an interface which allows the user to load and modify a specific library of images. Alternatively they can create a new database of images from here that are stored to the machines local memory.

# Conclusion

Computationally the system will be demanding and could take long periods of time to train large data sets. This can be helped by pre-training image sets prior to recognition.

Through testing the thresholds will need to be adjusted to provide the most optimal results for the environment where the recognition will take place.

The system is limited to frontal face recognition. The frontal face classifier supplied by the OpenCV library can 90% successfully identify faces

There are many potential avenues of application for the system. It could potentially source its images from a compiled face database of future students visiting the university on an open day. A camera would provide a feed to the system where face detection can be performed in real time and deliver a recognized profile and welcome message to a visible display near the entrance of a computer lab.

Other image analysis methods such as edge detection or converting images to a binary representation could provide more accurate results for the recognition. The system could be expanded to feature functionality to upload and store created and/or trained image sets to an online repository or storage facility. This would allow the application to be easily integrated into a web based application.

Further LDA’s (Linear detection Algorithm) methods such as DCT (Discrete Cosine Transformation) [9] could be used for reducing principal components. Likewise alternative component calculation methods such as FisherFace evaluation can deliver an effect means of facial recognition.

**Acknowledgement:** Hong Wei for overseeing the project and support.

# references

1. JavaCV Documentation, <http://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html>
2. OpenCV Documentation, “Haar like Features”, <http://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html>
3. Drexel University, Eigen faces. Available at: <http://www.pages.drexel.edu/~sis26/Eigenface%20Tutorial.htm>
4. Wikipedia. Eigen Faces. Available at: <https://en.wikipedia.org/wiki/Eigenface>
5. XM2VTS Face Database. [online] <http://www.idiap.ch/resource/faceverif/>
6. School-Wikipedia, Eigenvalues, Eigenvectors and Eigen Space. Available at: <http://schools-wikipedia.org/wp/e/Eigenvalue%252C_eigenvector_and_eigenspace.htm>
7. Science Direct, Illumination Challenges of face detection. Available at: <http://www.sciencedirect.com/science/article/pii/S0031320305001378>
8. EHow, Calculating the Euclidian distance. Available at: <http://www.ehow.com/how_7829754_euclidean-distance.html>
9. DCT,