

Face Recognition and Voice Controlled Personal Assistant

Metuku Shyamsunder¹, B.L.Bharath babu², Sainath Gorige³

^{1,2,3}Department of Electronics and communication engineering, Osmania University, India.

Abstract - The conventional manual method of drawing attention by knocking the door, pressing door bell, or/and shouting one's presence in order to gain access are inefficient and risky. A better method is automatic personal identification based on face and speech recognition which is the subject of this paper. Eigen face method is used for the face recognition. While the speaker and spoken command recognition are both based on the ALEXA API. Experiments yielded 70% face recognition while recognition rates for speakers and the spoken commands were 87% and 74% for utterances in English respectively. The performance of the algorithms is evaluated through computer simulations and is found to be quite effective.

Key Words: Face recognition, Eigenface method, ALEXA API

I. INTRODUCTION

Biometric identity authentication systems are based on the biological characteristics of a person, such as face, voice, fingerprint, iris, gait, hand geometry or signature. Identity authentication using the face or the voice information is a challenging research area. Face and speech are the two most popular means of personal identification particularly when the person to be identified is physically present. Hence, these two modes become handy in machine effected personal identification using biometrics for automatic access control. Security personnel manning access points most often based their access authorization on recognition of faces.

The face is the primary focus of attention in social intercourse playing a major role in conveying identity and emotions. Although the ability to infer intelligence or character from facial appearance is suspect, the human ability to recognize faces is remarkable .we can recognize thousands of faces learned throughout our lifetime and identity similar faces at a glance even after years of separation. This skill is quite robust, despite large changes in the visual stimulus due to viewing condition expression, aging and distractions such as glasses or changes in hairstyle or facial hair. As a consequence the visual processing of human faces has fascinated philosophers and scientists for centuries, including figures such as Aristotle and Darwin. As security is primary issue now, fingerprint type of systems is easily hackable. We make use of raspberry pi to built face and voice recognition systems, which is more secure than the present systems .Face recognition is obtained by using eigen faces and voice recognition using alexa speech to text API. Our main objective is to give our client more security and the secondary objective to get awareness with raspberry pi and python and to learn all the face recognition techniques work in general and how speech to text conversion takes place. By using Raspberry Pi we also get knowledge about Linux ,as the operation system of raspberry pi is linux.

During 1964 and 1965, Bledsoe, along with Helen Chan and Charles Bisson, worked on using the computer to recognize human faces (Bledsoe 1966a, 1966b; Bledsoe and Chan 1965). He was proud of this work, but because the funding was provided by an unnamed intelligence agency that did not allow much publicity, little of the work was published. Given a large database of images (in effect, a book of mug shots) and a photograph, the problem was to select from the database a small set of records such that one of the image records matched the photograph. The success of the method could be measured in terms of the ratio of the answer list to the number of records in the database. In 2006, the performance of the latest face recognition algorithms were evaluated in the Face Recognition Grand Challenge (FRGC). High-resolution face images, 3-D face scans, and iris images were used in the tests. The results indicated that the new algorithms are 10 times more accurate than the face recognition algorithms of 2002 and 100 times more accurate than those of 1995. Some of the algorithms were able to outperform human participants in recognizing faces and could uniquely identify identical twins.

Since 1993, the error rate of automatic face-recognition systems has decreased by a factor of 272. The reduction applies to systems that match people with face images captured in studio or mugshot environments. In Moore's law terms, the error rate decreased by one-half every two years. The table.1 shows the algorithms developed back then.

Table1: Existing technique for Face recognition

Year	Authors	Method
1973	Kanade	First automated system
1987	Sirovich & Kirby	Principal Component Analysis
1991	Turk & Pentland	Eigenface
1996	Etemad & Chellapa	Fisherface
2001	Viola & Jones	AdaBoost + Haar Cascade
2007	Naruniec & Skarbek	Gabor Jets

II. METHODOLOGY

Firstly, the raspberry pi is installed with noobs operating system and python is installed in it. Our first target is to get face recognition, for that we write code in python, we use eigen faces concept for face recognition and use camera for capturing images. And for voice recognition we use alexa speech to text API which we install in raspberry pi. we use microphone, speaker and camera module.

III. FACE RECOGNITION

Face recognition has always been a very challenging task for the researchers. On the one hand, its applications may be very useful for personal verification and recognition. On the other hand, it has always been very difficult to implement due to all different situations that a human face can be found. Nevertheless, the approaches of the last decades have been determining for face recognition development.

Due to the difficulty of the face recognition task, the number of techniques is large and diverse. In addition, the applications involve a huge number of situations. Although we can find many other identification and verification techniques, the main motivation for face recognition is because it is considered a passive, non-intrusive system to verify and identify people. There are many other types of identification such as password, PIN (personal identification number) or token systems. Moreover, it is nowadays very instilled the usage of fingerprints and iris as a physiological identification system. They are very useful when we need an active identification system; the fact that a person has to expose their body to some device makes people feel being scanned and identified.

It is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. There are multiple methods in which facial recognition systems work, but in general, they work by comparing selected facial features from given image with faces within a database. While initially a form of computer application, it has seen wider uses in recent times on mobile platforms and in other forms of technology, such as robotics. It is typically used in security systems and can be compared to other biometrics such as fingerprint or eye iris recognition systems. Recently, it has also become popular as a commercial identification and marketing tool.

3.1 Eigen face method of face recognition

In mathematical terms, we wish to find the principal components of the distribution of faces, or the eigen vectors of the covariance matrix of the set of face images treating an image as a point in a very high dimensional space. The eigenvectors are ordered, each one accounting for a different amount of the variation among the face images. These eigen vectors can be thought of as a set of features that together characterize the variation between face images. Each image location contributes more or less to each eigenvectors so that

we can display the eigen vectors as a sort of ghostly face which we call an eigen face. Some of the faces we studied are illustrated in figure below and the corresponding eigenfaces are shown in second figure. Each eigenface deviates from uniform gray where some facial feature differs among the set of training features; they are a sort of map of the variations between faces.

Each individual face can be represented exactly in terms of a linear combination of the Eigen faces. Each face can also be approximated using only the best eigenfaces –those that have the largest eigen values and which therefore account for the most variance within the set of face images. The best M Eigen faces span an M-dimensional subspace –“face space“ of all possible images. Perhaps an efficient way to learn and recognize faces would be to build up the characteristic features by experience over time and recognize particular faces by comparing the feature weights needed to reconstruct them with the weights associated with known individuals. Each individual, therefore would be characterized by the small set of features or eigen picture weights needed to describe and reconstruct them –an extremely compact representation when compared with the images themselves.

This approach to face recognition involves the following initialization operations:

1. Acquire an initial set of face images
2. Calculate the eigenfaces from the training set, keeping only the M images that correspond to the highest eigen values. These M images define the face space. As new faces are experienced, the eigenfaces can be updated or recalculated.
3. Calculate the corresponding distribution in M-dimensional weight space for each known individual by projecting f it is face their face images onto the face space.

These operations can also be performed from time to time whenever there is free excess computational capacity.

Having initialized the system, the following steps are then used to recognize new face images

1. Calculate a set of weights based on the input image and the M eigenfaces by projecting the input image onto each of the eigenfaces.
2. Determine if the image is a face at all by checking to see if the image is sufficiently close to face space
3. If it is a face, classify the weight pattern as either a known or unknown face.
4. (Optional) Update the Eigen faces and/or weight patterns.
5. (Optional) If the same unknown face is seen several times, calculate its characteristic weight pattern and incorporate into the known faces.

3.2 Using Eigenfaces to Classify a Face Image

The eigenface Images calculated from the eigenvectors of L span a basis set with which to describe face images. Sirovich and Kirby (1991) evaluated a limited version of this framework on an ensemble of = 115 images of Caucasian males, digitized in a controlled manner, and found that about 40 eigenfaces were sufficient for a very good description of the set of face images. With 3r -i0 eigenfaces. KIAS pixel-by-pixel errors in representing cropped versions of face Images were about 2%. Since the eigenfaces seem adequate for describing face images under very controlled conditions, we decided to investigate their usefulness as a tool for fat identification.



Fig.1: Seven of the eigen faces calculated from input images

In practice, a smaller is sufficient for identification, since accurate reconstruction of the image is not a requirement. In this framework, identification becomes a pattern recognition task The eigenfaces span M dimensional subspace of the original, N^2 image space. Them significant eigenvectors of the L matrix are chosen as those with the largest associated eigen values. In many of our test cases fused on M = 16 face images M' = 7 eigenfaces were used.

A new face image (Γ) is transformed into its eigenface components transformed into "face space" by a simple for k = 1, This describes a set of point to point image multiplications and .summations. Fig.1 shows an image and its projection into the seven-dimensional face space. The weights form a vector Ω = (ω1, ω2,...ωn) describes the contribution of each eigenface in representing the input fact image. Treating the eigenfaces as a basis set for face images. The vector may then be used in a standard pattern recognition algorithm to find which of a number of predefined face classes. If any. best de. scribes the face. The simplest method for determining which face class provides the best description of an input face image is to find the face class k that minimizes the Euclidian distance.

$$\epsilon_k^2 = \|(\Omega - \Omega_k)\|^2 \tag{1}$$

where Ω is a vector describing the kth face cars. The face classes Ωt, are calculated by avenging the results of the eigenface representation over a small number of face images (as few as one) of each individual. A face is classified as belonging to class k when the minimum es is below come chosen threshold. Otherwise the face is classified as "unknown: and optionally used to create a new fate class. Because creating the vector of weights is equivalent to protecting the original face image onto the low dimensional face space, many images (meta of them looking nothing like a face) will protea onto a given pattern vector. This is not a problem for the system, however. since the distance e between the Image and the face space b simply the squared distance between the mean- adjusted input image, its protection onto face space.

Thus, there are four possibilities for an input image and its pattern vector:

- (1) Near face space and near a face class
- (2) Near face space but not near a known face CUSS.
- (3) Distant from face space and near a face class, and
- (4) Distant from face space and not near a known face class.

In the first case, an individual is recognized and identified. In the second case, an unknown individual is present. The last two cases indicate that the image is not a face image. Case three typically shows up as a false positive in most recognition systems. In our framework. However, the false recognition may be detected because of the significant distance between the image and the subspace of expected face images.

IV. VOICE RECOGNITION

A Voice Command System essentially means a system that processes voice as an input, decodes or understands the meaning of that input processes it and generates an appropriate voice output. Any voice command system need three basic components which are speech to text converter, query processor and a text to speech converter. Voice has been a very integral part of communication nowadays. Since, it is faster to process sound and voices than to process written text, hence voice command systems are omnipresent in computer devices. There have been some very good innovations in the field of speech recognition. Some of the latest innovations have been due to the improvements and high usage of big data and deep learning in this field. These innovations have attributed to the technology industry using deep learning methods in making and using some of the speech recognition systems. Using big data for speech systems, Alexa was able to reduce word error rate by 6% to 10% relative, for the system that had the word error rate of 17% to 52%.

Speech Recognition is the ability of machine for instance a computer to understand words and sentences spoken in any language. These words or sentences are then converted to a

format that could be understood by the machine. Speech recognition is basically implemented using vocabulary systems. A speech recognition system may be a Small Vocabulary-many user system or a Large Vocabulary- small user system.

1.1 System Architecture

The block diagram of voice recognition system is shown in fig2

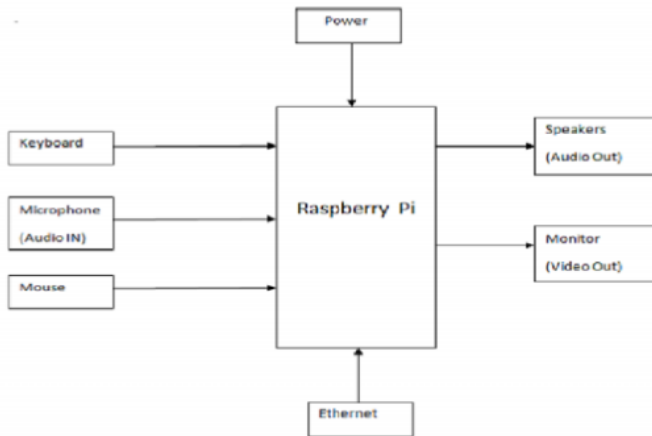


Fig.2: Block diagram of voice system

Mouse also acts an interface between the system and the developer and does not have a direct interaction with the end user. RaspberryPi is the heart of the voice command system as it is involved in every step of processing data to connecting components together. The Raspbian OS is mounted onto the SD card which is then loaded in the card slot to provide a functioning operating system. Ethernet is being used to provide internet connection to the voice command system. Since the system relies on online text to speech conversion, online query processing and online speech to text conversion hence we need a constant connection to achieve all this. Monitor provides the developer an additional way to look at the code and make any edits if any. It is not required for any sort of communication with the end user. Once the query put forward by the user has been processed, the text output of that query is converted to speech using the online text to speech converter. Now this speech which is the audio output is sent to the user using the speakers which are running on audio out.

1.2 Flow of events in voice command system

First, when the user starts the system, he uses a microphone to send in the input. Basically, what it does is that it takes sound input from the user and it is fed to the computer to process it further. Then, that sound input if fed to the speech to text converter, which converts audio input to text output which is recognizable by the computer and can also be processed by it. Then that text is parsed and searched for keywords. Our voice command system is built around the system of keywords where it searches the text for key words

to match. And once key words are matched then it gives the relevant output. This output is in the form of text. This is then converted to speech output using a text to speech converter which involves using an optical character recognition system.

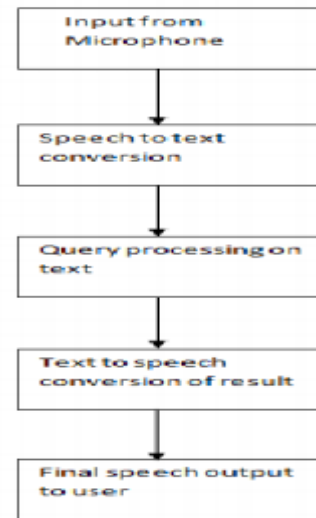


Fig.3: Flow of voice command system

OCR categorizes and identifies the text and then the text to speech engine converts it to the audio output. This output is transmitted via the speakers which are connected to the audio jack of the raspberry pi.

V. RESULT

The Voice Command System works on the idea and the logic it was designed with. Our system uses the keyword “listen” to take a command. Each of the commands given to it is matched with the names of the modules written in the program code. If the name of the command matches with any set of keywords, then those set of actions are performed by the Voice Command System. The modules of Find my iPhone, Wikipedia and Movies are based upon API calling. We have used open source text to speech and speech to text converters which provide us the features of customizability. If the system is unable to match any of the said commands with the provided keywords for each command, then the system apologizes for not able to perform the said task. All in all, the system works on the expected lines with all the features that were initially proposed. Additionally, the system also provides enough promise for the future as it is highly customizable and new modules can be added any time without disturbing the working of current modules.

1.3 Process of voice system

Step One: Register for an Amazon Developer Account Before you do anything, you’ll need to register for a free Amazon Developer Account, then create a profile for your DIY Echo. This is pretty straightforward: > Log into your Amazon Developer Account.

1. Click on the Alexa Tab.
2. Click Register a Product Type > Device.

3. Name your device type and display name (We chose “Raspberry Pi” for both).
4. Click Next.
5. On the Security Profile screen, click “Create new profile.”
6. Under the General tab, next to “Security Profile Name” name your profile. Do the same for the description. Click Next.
7. Make a note of the Product ID, Client ID, and Client Secret that the site generates for you.
8. Click the Web Settings tab, then click the Edit button next to the profile dropdown. > Next to Allowed Origins, click, “Add Another” and type in: https://localhost:3000.
9. Next to Allowed Return URLs, click “Add Another” and type in: https://localh
10. The Device Details tab is next. It doesn’t matter much what you enter here. Pick a category, write a description, pick an expected timeline, and enter a 0 on the form next to how many devices you plan on using this on. Click Next.
11. Finally, you can choose to add in Amazon Music here. This does not work on the Pi powered device, so leave it checked as “No.” Click Save.

Now you have an Amazon Developer Account and It’s time to head over to the Raspberry Pi and get Alexa working.

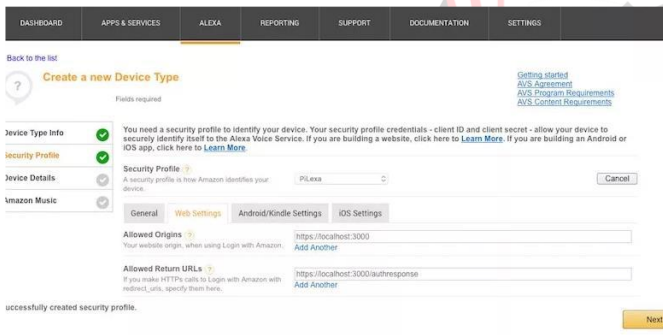


Fig.4: Alexa developing page

Step Two: Clone and Install Alexa

1. Open the Terminal application on the Raspberry Pi and type: cd Desktop and press Enter. > Type in git clone https://github.com/alexa/alexa-avs-sample-app.git and press Enter.
2. Once that’s complete, type in: cd~/Desktop/alexa-avs-sample-app and press Enter.
3. Type in nano automated_install.sh and press Enter.
4. This pulls up your text editor. Here, you’ll need to enter your ProductID, ClientID, and ClientSecret that you notes in the step above. Use the arrow keys to navigate to each entry. Enter each detail after the = sign as noted in the image above. When you’re done, tap CTRL+X to save and exit.

5. You’re now back at the command line. It’s time to run the install script. Type in cd ~/Desktop/alexa-avs-sample-app and press Enter.
6. Type in .automated_install.sh and press Enter.
7. When prompted, press Y for the different questions, and answer as you see fit for the rest. This will configure your Pi and install some extra software. This can take up to 30 minutes.

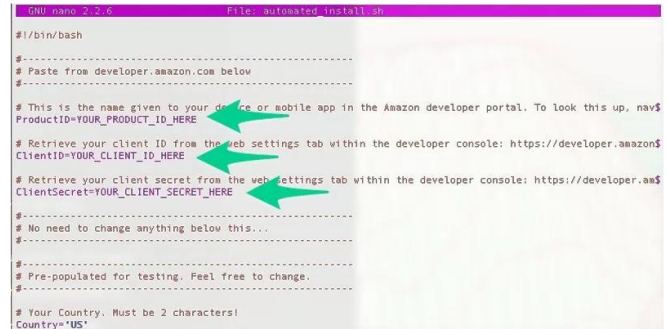


Fig.5:Command line code

Step Three: Run the Alexa Web Service

Next, Run three sets of commands at once in three different Terminal windows. Create a new Terminal window for each of the following steps. Don’t close any windows! We need to do steps three (this one,) four, and five every time you reboot your Raspberry Pi.

The first one you’ll start is the Alexa Web Service:

1. Type in cd ~/Desktop/alexa-avs-sample-app/samples and press Enter.
2. Type in cd companion Service && npm start and press Enter.

This starts the companion service and opens up a port to communicate. Leave this window open.

Step Four: Run the Sample App and Confirm Your Account

Open up a second Terminal window (File > New Window). This next step runs a Java app and launches a web browser that registers your Pi-powered Echo with the Alexa web service.

1. In your new Terminal window, type in cd ~/Desktop/alexa-avs-sample-app/samples and press Enter.
2. Type in cd javaclient && mvn exec:exec and press Enter.
3. A window will pop up asking you to authenticate your device. Click Yes. This opens up a browser window. A second pop-up will appear in the Java app asking you to click Ok. Do not click this yet.
4. Log into your Amazon account in the browser.
5. You’ll see an authentication screen for your device. Click Okay. Your browser will now display “device tokens ready.”

6. You can now Click the Ok pop-up in the Java app.

Now, Raspberry Pi has the necessary tokens to communicate with Amazon's server. Leave this Terminal window open.

Step Five: Start Your Wake Word Engine

1. Type in `cd ~/Desktop/alexa-avs-sample-app/samples` and press Enter.

2. Type in `cd wake WorAgent/src && ./wakeWordAgent -e kitt_ai`

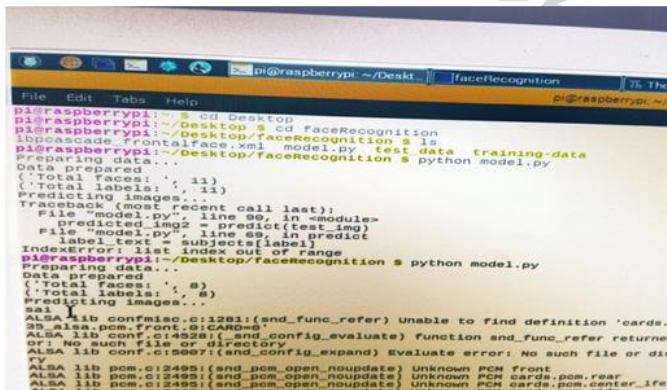
3. This by default uses "Alexa" as wakeword. To change the wakeword into any other follow the following steps. > Visit <https://snowboy.kitt.ai/> and register/login.

4. Go through the process to create your own hotword. We used "Tom" as a Wakeword.

5. Download the created "Tom.pmdl" file and rename it to "alexa.umdl".

6. Stop wake word detection on your device and replace the file "`~/Desktop/alexa-avs-sample-app/samples/wakeWordAgent/kitt_ai/snowboy/resources/alexa.umdl`" with renamed file.

Run the wakeword command and now it will respond to your new wakeword (Tom).



```
pi@raspberrypi: ~/Desktop/faceRecognition
pi@raspberrypi: ~$ cd Desktop
pi@raspberrypi: ~/Desktop$ cd faceRecognition
pi@raspberrypi: ~/Desktop/faceRecognition$ ls
cascade_frontend.xml model.py test_data training-data
pi@raspberrypi: ~/Desktop/faceRecognition$ python model.py
Data prepared...
({'Total faces': 11})
({'Total labels': 11})
Predicting faces...
Traceback (most recent call last):
  File "model.py", line 99, in <module>
    predicted_img = predict(test_img)
  File "model.py", line 99, in predict
    label_text = subjects[label]
IndexError: list index out of range
pi@raspberrypi: ~/Desktop/faceRecognition$ python model.py
Data prepared...
({'Total faces': 8})
({'Total labels': 8})
Predicting images...
sa1
ALSA lib confdefs.c:1281:(snd_func_refer) Unable to find definition 'cards...
ALSA lib pcm_front.c:1489:(snd_pcm_open_noupdate) Unknown PCM front
ALSA lib pcm.c:2495:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.front
ALSA lib pcm.c:2495:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2495:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.cen...
```

Fig.6: Output when unknown image is given

The fig.6 shows the output when the face is recognized. It is seen that "sa1" is printed on the screen when that particular image is given. There is a folder named trained data in which 10 images of same person are given, and in test data one photo is given which is tested to find whether it is the same photo as in training data folder.

VI. CONCLUSIONS

Various components like Raspberry Pi 3, Speakers (earphones), Microphone and camera modules are used. In this proposed model first we implemented face recognition using the concept of Eigen faces. Next Voice Controlled Personal Assistant is implemented by taking use of ALEXA API. Voice controlled Personal Assistant (VPA) activates when a user defined Wakeword is given as input. We used Wakeword "TOM". When it hears the appropriate Wakeword, it gives user a time of 5 seconds for giving any

query to PI. After 5 seconds it processes the query and gives the output with good accuracy. Advanced voice technology will soon be ubiquitous, as natural and intelligent user interface technology integrates seamlessly into daily life. Voice will be a primary interface for the connected home, providing a natural means to communicate with alarm systems, lights, kitchen appliances, sound systems and more, as users go about their day-to-day lives.

REFERENCES

1. Rukhasar Jamadar, Snehal Patil, Sayama Solkar, "Voice Control Music System Using Raspberry Pi", JETIR April 2019, Volume 6, Issue 4
2. Lalitha S, Ashwini V, Madhusudhan.K.N, Sachin B S, "Person Authentication Using Face And Voice Modalities", International Journal of Advances in Science, Engineering and Technology(IJASEAT), pp. 71-79, Volume-1, Issue-2, 2013
3. Ishwar S. Jadhav, V. T. Gaikwad, Gajanan U. Patil, "Human Identification using Face and Voice Recognition", International Journal of Computer Science and Information Technologies, Vol. 2 (3), 2011, 1248-1252
4. Smith and Pentland, "Eigen based facial recognition" Dimitri PISSARENKO December 1, 2002
5. Mathew Turk and Alex Pentland, "Eigen faces for face recognition" vision and modelling group, Journal of cognitive neuroscience, Volume 3, issue 1, 1991
6. George Bebis, "Face Recognition Using Principal Components Analysis", Journal of Cognitive Neuroscience, 3(1), pp. 71-86, 1991.