Smart Security System Using Face Recognition on Raspberry Pi

Fahim Faisal
Dept. of Computer Science and Engineering
Daffodil International University
Dhaka, Bangladesh.
embeddedfahim@gmail.com

Syed Akhter Hossain
Dept. of Computer Science and Engineering
Daffodil International University
Dhaka, Bangladesh.
aktarhossain@daffodilvarsity.edu.bd

***Abstract*—Privacy and Security are two of the most important universal rights. To ensure security in our daily life through technology, a lot of research is going on. Among them facial recognition is a popular and well-established technology. In this technology, faces are detected and identified out of images and with the help of Internet of Things (IoT), it becomes even more useful and precise. Using face recognition and IoT, we aim to create a smart door, which secures the gateway on the basis of who we are. In our proof of the concept of such a smart security system, we have used Viola-Jones method to detect faces and Eigenfaces method to recognize people. A Raspberry Pi has been used as the microprocessor for ensuring low-cost and small size of the system. The door will open automatically for the known person upon receiving command from the processor. On the other hand, photo of the unknown will be uploaded to a webserver while an email is sent to the owner of the place containing a link to the image. From the website of the system, the owner can then allow/block entry and also trigger an alarm from anywhere in the world.**

Keywords—Viola-Jones face detection method, Microprocessor, PCA, Eigenvector, Covariance matrix, Euclidean distance, Eigenface, Microcontroller

I. INTRODUCTION

Human beings can be precisely identified by their unique facial characteristics. Nowadays, automatic person identification in access control has become popular by using biometric data instead of using cards, passwords or patterns. Most of the biometric data have to be collected by using special hardware such as fingerprint scanners, palm print scanners, DNA analyzers etc. But face recognition, in this particular area, has some advantages. It's hassle-free in a sense that one doesn't need to touch anything in order to be identified or recognized. As a result, in the field of biometrics, face recognition technology is one of the fastest growing fields. This recent interest in face recognition can be attributed to the increase of commercial interest and the development of feasible technologies to support the development of face recognition. Major areas of the commercial use of face recognition include biometrics, law enforcement and surveillance, human-computer interaction, multimedia management (for example, automatic tagging of a particular individual within a collection of digital photographs), smart cards, passport check, criminal investigations, access control etc. However, face detection can be a bit challenging at times owing to some unstable characteristics of a human face. For example, glasses and beard will affect the detecting accuracy. Moreover, different kinds and angles of lighting will generate uneven brightness on the face, which will have influence on the detection process. To overcome these problems, the system used Viola-Jones method [1] for face detection and Eigenfaces method [2] for face recognition. Eigenfaces are generated using a mathematical process called Principal Component Analysis (PCA). If a face is recognized, it is known and if not, it is unknown. Since PCA reduces the dimensions of facial images without losing important features, facial images for many people can be stored in the database without losing significant efficiency. Therefore, face recognition using PCA is more useful for home/office security systems than other face recognition techniques [3].

II. RELATED WORKS

Face recognition in an automatic manner started to gain popularity from the 1960s. 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) [4]. Several modern systems including the implementation of automatic face detection and recognition have been developed since then.

Paul Viola and Michael J. Jones, in their paper, discussed about a new face detection framework that is capable of processing images extremely rapidly while achieving high detection rates [1]. Ole Helvig Jensen's paper documents all relevant aspects of the implementation of the Viola-Jones face detection algorithm [5]. M. A. Turk and A. P. Pentland. introduced a new technique for image representation in order to recognize faces in an unsupervised manner [2]. Daniel Georgescu introduced a system which is capable of performing real-time face detection, face recognition and sending an e-mail notification to interested institutions [6]. Liton Chandra Paul and Abdulla Al Sumam, in their paper, addressed the building of a face recognition system by using Principal Component Analysis (PCA) [7]. Shervin Emami and Valentin Petruț Suciu, in their paper, have introduced a face recognition system using OpenCV and Microsoft's .NET framework [8]. Hteik Htar Lwin, Aung Soe Khaing and Hla Myo Tun, in their paper, discussed about an automatic door lock system using OpenCV on MATLAB [3]. I.Yugashini, S. Vidhyasri and K. Gayathri Devi, in their paper, discussed about a face recognition and detection process implemented by modifying principal component analysis (PCA) approach to fast based principal component analysis (FBPCA) approach [9]. Prathamesh Timse *et al.*, in their paper, demonstrated the implementation of Viola-Jones and Eigenfaces method in C# in the development of a face recognition system. The system can upload photos of

978-1-7281-2741-5/19/$31.00 ©2019 IEEE

unauthorized people to a remote webserver [10]. Ayushi Gupta *et al.*, used Viola-Jones method and Principal Component Analysis to develop a face recognition system [11].

III. METHODOLOGY

Face recognition can be divided into a few simple stages and those stages can be further divided into more sophisticated stages. At first, images are captured with the help of a camera and then the images are taken as inputs. Faces are differentiated from the images and only the important features of a face are kept in the database which reduces space complexity and in turn the overall computational complexity. After that, the machine is trained with these features for further evaluation. A simple face recognition procedure is shown in the flowchart in Fig. 1.

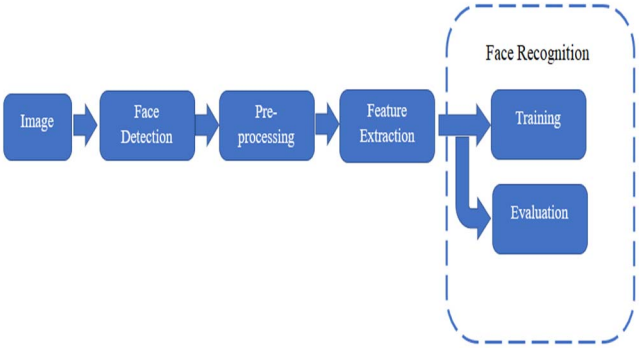

Fig. 1: A Simple Face Recognition Flowchart.

A. Viola-Jones Face Detection Method and AdaBoost Algorithm

This method can be divided into three main steps. The first step of the Viola-Jones face detection algorithm is to turn the input image into a new image representation called an integral image that allows a very fast feature evaluation. The used features are comparable to Haar-basis functions. The Viola-Jones method analyzes a 24 * 24 sub-window using features consisting of two or more rectangles. Each feature results in a single value which is obtained by subtracting the sum of the white rectangle(s) from the sum of the black rectangle(s) [5]. The different types of features are shown in Fig. 2.

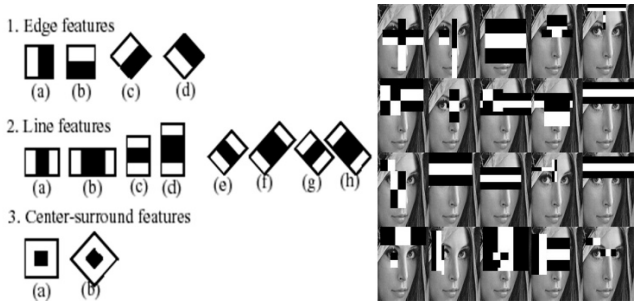

Fig. 2: Different types of features.

For a fast processing of these features, we used a new image representation method, called the integral image representation. This is accomplished by making each pixel equal to the entire sum of all pixels above and to the left of the concerned pixel [6]. It is calculated by the following equation,

$$g(x, y) = \sum_{x' \le x,\, y' \ge y} i(x', y') \quad \ldots\ldots\ldots \quad [1]$$

Where g(x, y) is the integral image and i(x , y) is the original image. The integral image can be computed in one pass over the original image by using the following pair of equations:

$$s(x, y) = s(x, y - 1) + i(x, y) \quad \ldots\ldots\ldots\ldots \quad [2]$$
$$g(x, y) = g(x - 1, y) + s(x, y) \quad \ldots\ldots\ldots\ldots \quad [3]$$

Where s(x, y) is the cumulative row sum, s(x, −1) = 0, and g(−1, y) = 0. The second step is constructing a classifier in order to select a small number of important features using the AdaBoost learning algorithm. AdaBoost is a machine learning boosting algorithm, which is capable of constructing a strong classifier by using a weighted combination of weak classifiers [5]. A weak classifier is calculated by the following equation,

$$h(x, f, p, \theta) = \begin{cases} 1, if\ p(f(x)) > p(\theta) \\ 0, \qquad otherwise \end{cases}$$

where x is a 24 * 24 pixels sub-window of an image, f is the applied feature, p indicates the direction of the inequality, and θ is a threshold that decides whether x should be classified as a positive (face) or a negative (non-face). The final strong classifier is obtained after applying the AdaBoost algorithm described in equation 1. In the third step, the cascaded classifier is used to determine whether a given sub-window classifier is definitely not a face or maybe a face. The cascaded classifier is composed of stages where each consists of a strong classifier. The concept is illustrated with two stages in figure 3.

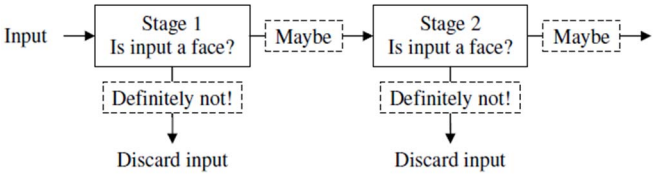

Fig. 3: Cascaded Classifier.

B. Principal Component Analysis and Eigenfaces Method

To extract the relevant features of facial images, Principal Component Analysis (PCA) method is used. As mentioned earlier, face recognition based on PCA is generally referred to as the use of Eigenfaces. Eigenfaces are principal components of the distribution of faces, or equivalently, the Eigen vectors of the covariance matrix of the set of the training images, where an image with N by N pixels is considered as a point in N^2 dimensional space. The PCA algorithm is shown in the following steps:

Step 1: The image matrix I of size (N × N) pixels is converted to the image vector Γ of size (P × 1) where P = (N × N).

Training set,

$$\theta = [\theta_1 \theta_2 \ldots \theta_M]$$

Step 2: Average face image is calculated by,

$$\Psi = \frac{1}{M} \sum_{i=1}^{M} \theta_i,$$ where M is the total number of images.

Each face differs from the average by,

$$\Phi_i = \theta_i - \Psi, \text{ where } \Phi_i \text{ is the i-th difference.}$$

Difference matrix,

$$A = [\Phi_1\ \Phi_2\ \dots\ \Phi_M]$$

Step 3: A covariance matrix C, is constructed as,

$$C = AA^T, \text{ where size of C is } (P \times P).$$

This covariance matrix is huge in dimension, which increases computational complexity. In order to work with this covariance matrix, we need to reduce its dimensions. Such a covariance matrix L with reduced dimensionality is,

$$L = A^TA, \text{ where size of L is } (M \times M).$$

For obtaining the Eigenvectors of the original covariance matrix, it can be calculated by the following equations:

$$A^TAX_i = \lambda_iX_i$$

By multiplying both sides of the above equation with A,

$$AA^TAXi = A\lambda_iX_iAA^T(AX_i) = \lambda_i(AX_i)$$

AX_i are the Eigenvectors of the covariance matrix which is denoted by U_i and eigenvalues λ_i are the same for the two covariance matrices.

Step 4: A face image can be projected into this face space by,

$$\Omega_k = U_k^T\Phi_i$$

Step 5: Test image vector = θ_t, Mean subtracted image vector,

$$\Phi_t = \theta_t - \Psi$$

The test image is projected onto the face space to obtain a vector,

$$\Omega = U_k^T\Phi_t$$

Step 6: The last step is finding the minimum distance between the test image and the training images. The face with minimum Euclidian distance shows similarity to the test image. The distance of test image Ω to each training image is called the Euclidean distance and is defined by,

$$\boldsymbol{\varepsilon}_k^2 = \| \Omega - \Omega_k \|^2$$

By choosing a threshold value α, that is the maximum acceptable value for known images and comparing it with the minimum $\boldsymbol{\varepsilon}_k$, test image can be recognized as known or unknown face.

If $\boldsymbol{\varepsilon}_{k(min)} \geq \alpha$, the test image is recognized as an unknown face.

If $\boldsymbol{\varepsilon}_{k(min)} < \alpha$, the test image is a known face.

C. Internet of Things and Alarm System

Whenever the system detects an unknown person, the image of the unknown is uploaded to a remote webserver, an email is sent to the owner/client and at the same time the microprocessor commands the microcontroller to trigger the buzzer [12], which acts as the alarm system. The alarm beeps with 0.25 second delay between two consequent beeps. This beeping continues for 10 seconds and then it stops automatically.

The web address of the system is sss.embeddedfahim.com, which is hosted on the aforementioned remote server. The uploaded image is retrieved and displayed on the homepage. The email that is sent to the owner/client, also contains a link to the image. Upon visiting the link, the owner is taken to the login page. From there the homepage can be accessed after typing in the required credentials, i.e. username and password. The credentials can be collected from the system administrator.

Additionally, the website contains two buttons which can be used to open and close the door remotely, from anywhere around the world. These buttons each hold a value and the value is passed on to a text file placed inside the server. The microprocessor has an infinite loop running inside it, which continuously checks the value inside this text file. As soon as the buttons are pressed the value stored inside the text file is updated and the microprocessor detects the updated value and commands the microcontroller to open or close the door accordingly. The microcontroller then generates appropriate signals to rotate the servos which are attached to the door.

```
import requests
import hardware

door = hardware.Door()

while 1:
        response = requests.get('https://sss.embeddedfahim.com/test.php')
        if response.text == '1':
                door.unlock()
                while 1:
                        response = requests.get('https://sss.embeddedfahim.com/test.php')
                        if response.text == '0':
                                door.lock()
                                break
```

Fig. 4: Partial Python Code (For Web Control).

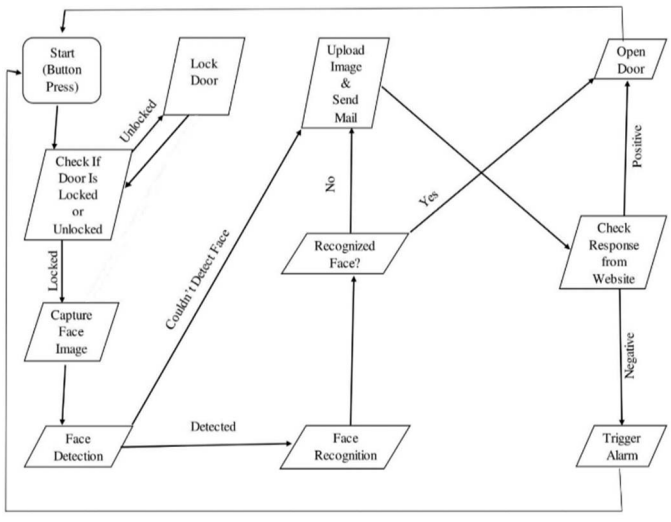

Fig. 5: System Algorithm Flowchart.

IV. Hardware Components

The main controller of the system is the single board computer *Raspberry Pi 3* [13]. In other words, it's the brain of the system. An *Arduino Uno* [14] has been used to assist the *Pi* with handling sensors, servo motors and other electronic components. The *Arduino Uno* [14] has a logic level of 5V whereas the *Raspberry Pi* [13] has a logic level of 3.3V, which some modules and sensors don't accept. Moreover, the *Arduino Uno* [14] can generate powerful Pulse Width Modulation (PWM) signals which are required to control the servo motors [15], the *Pi* can also generate PWM signals but adjusting the duty cycle [16] with the *Pi* is slightly inefficient.

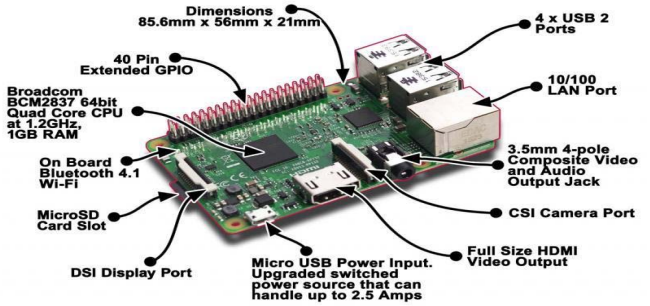

Fig. 6: Raspberry Pi 3 Model B.

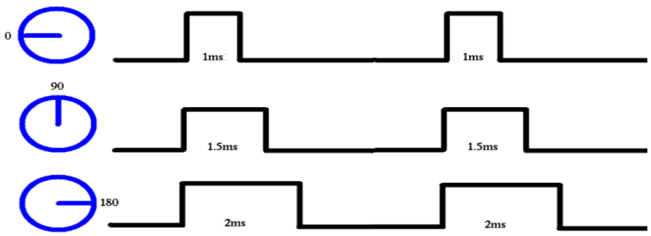

Fig. 7: Different PWM Signals for Servo Control.

A custom-built 5V Lithium-Ion rechargeable power supply has been used to power the system. Two 2000 mAh 18650 Li-Ion batteries [17] have been used in parallel in the power supply. The 3.7V of the batteries is further boosted in the Battery Management System (BMS) [18]. It has 5V output and can deliver 4A of current at any instance through two channels, 2A per channel. One channel provides power to the servo motors, buzzer and the 20x4 LCD [19]. The *Pi* needs steady voltage to operate thus it is given power through a separate channel of the power supply. The servo motors, buzzer and the LCD, all require 5V to perform properly. So, all of them are given 5V. The *Arduino Uno* [14] is connected to the *Raspberry Pi* [13] via an USB A-to-B cable, hence it receives 5V directly from the *Pi.* The BMS has its own voltage regulators on each channel so additional voltage regulation wasn't required.

Fig. 8: Raspberry Pi Camera Module (5 MP).

There is a button which works as the shutter for the camera. It is connected to the *Pi* through a pull-down resistor [20] and thus receives power directly from the *Pi*. The 5 MP camera module [21] is powered through the Camera Serial Interface (CSI) port on the *Raspberry Pi* [13]. Two 1-Watt LEDs [22] have been used for additional brightness while capturing images. These LEDs are driven by two BC547 N-P-N Bipolar Junction Transistors (BJT). In average, the total power consumption of the system is around 5W. The used power supply can handle the load flawlessly. The system can run on batteries for 4 hours straight in case of power failure. The Battery Management System (BMS) [18] automatically switches to live power from the power adapter when electricity comes on.

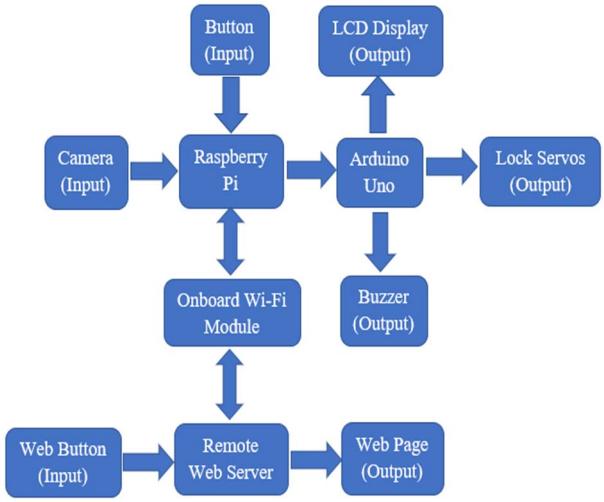

Fig. 9: Connection Diagram.

V. Software and Communication

The Viola-Jones face detection algorithm and the Eigenfaces algorithm, both are pre-included in Intel's OpenCV [23] library, which is offered in multiple programming languages, including Python [24]. The official programming language for the *Raspberry Pi* [13] is also Python [24]. As such, Python [24] was used to program the *Pi*.

At first, OpenCV [23] was compiled and installed on the *Pi*. After that it was imported as a module in the Python [24] code. There are separate code segments for training and evaluation. The input images were captured through the camera module by pressing the shutter button, which is connected directly to the *Pi.*

In the database folder, 50 different face images of 10 different people were used as the training images. While constructing the database, the captured images are cropped by the face detection module in order to obtain only the facial parts of captured images with different directions. For instance, 5 images of a person with different face directions are shown in figure 10. We trained the machine to treat these images as positive, i.e. the face in these images will be considered as the face of a known person.

Fig. 10: 5 Images of a Single Person (Positive).

We also trained the machine with 400 negative images, all obtained from the AT&T Laboratories Cambridge face database [25].

Fig. 11: 5 Images of a Single Person (Negative).

All training images are cropped and converted to 92x112 grayscale images by using "crop" and "cvtColor" built-in functions of OpenCV [23]. Mean centered (or subtracted) images are evaluated by subtracting average image from the original training images. The eigenvectors corresponding to the covariance matrix define the Eigenfaces, which look like ghostly faces. Since 50 positive training images are used, 50 positive Eigenfaces are generated, which are then superimposed on each other. Eigenfaces of the training images are shown in figure 13.

```
image = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
# Getting coordinates of single face in captured image..
result = face.detect_single(image)
if result is None:
        print 'Could not detect single face! Check the image capture.jpg' \
              ' to see what was captured and try again with only one face visible.'
        continue
x, y, w, h = result
# Cropping image as close as possible to desired face aspect ratio..
# Might be smaller if face is near the edge of the image..
crop = face.crop(image, x, y, w, h)
# Saving image to file..
filename = os.path.join(config.POSITIVE_DIR, POSITIVE_FILE_PREFIX + '%03d.pgm' % count)
cv2.imwrite(filename, crop)
print 'Found face and wrote training image', filename
count += 1
```

Fig. 12: Partial Python Code (Image Crop & Conversion).

Fig. 13: Positive Eigenface (leftmost), Negative Eigenface (middle) & Mean Eigenface (rightmost).

The *Arduino Uno* [14] was programmed in the *Arduino* language (a subset of C++) for ease of programming.

Server-side programming is done using PHP [26] and the website is written using HTML [27] and CSS [28]. Several PHP [26] scripts are placed inside the server which handle different things. For example, the text file containing the value from the web buttons is handled by a PHP [26] script which extracts the value from the text file and passes it on to the *Pi* when the *Pi* sends a request to the server. Owner/client credentials are stored inside a MySQL [29] database, which is also located inside the server. Photo of the unknown person is uploaded using the FTPLIB [30] module and email is sent using the SMPTPLIB [31] module of the Python language. A separate email account is associated with the system, which is created using Gmail.

```
<?php
        echo file_get_contents('test.txt');
?>
```

Fig. 14: Partial PHP Code.

FCC standards and regulations were followed regarding the communication between the prototype and the internet. A 2.4 GHz Wi-Fi Router [32] has been used to transfer images from the prototype to the server and control signals from the server to the prototype. The onboard Wi-Fi module of the *Raspberry Pi* [13] which connects to the Wi-Fi router [32], was used to transmit data to the server and receive control signals from the server. IEEE 802.11 b/g/n wireless standard [33] was utilized in all the network components.

VI. Observations and Experiments

Several experiments were conducted to ensure proper functionality of the system. This helps to accurately determine the future complications that may occur in actual field. There were two tasks that needed to be tested. First one is automatic door opening for a known person and the second one is the triggering of the alarm system, image upload and email notification for an unknown person. The prototype showed a descent success rate in all aspects when tested.

For testing purpose, we used different configurations of the system. Two of the most important variables in the configuration were the minimum-neighbors threshold and the detection threshold. There are also two other parameters for face recognition in OpenCV's [23] built-in haar classifier and face recognizer functions; scale increase rate and minimum detection scale.

The minimum-neighbors threshold is the value which sets the cutoff level for discarding or keeping rectangle groups as "face" or "non-face", based on how many raw detections are in the group. This parameter's value ranges from 0 to 10, as set by the "face.detect_single" built-in function of OpenCV [23]. We have set this value to 8, i.e. we only want an object to be marked as a face if it has 80% probability of being a "face". If we set minimum-neighbors to a value n, then the face detector will mark an object as a "face" in any image if and only if there is a group of n rectangles (hits) identifying it as a "face".

Detection threshold is the maximum value of the Euclidean distance between the database image and input image. It indicates the amount of difference between the test image and the mean image.

Minimum detection scale is the parameter in the call to "DetectHaarCascade" built-in function. It is the size of the smallest face to search for. We have set it to 25x25, which gives us the best results. A good rule of thumb is to use some fraction of your input image's width or height as the minimum scale, for example, $1/4^{th}$ of the image width. If one specifies a minimum scale other than the default, its aspect ratio (the ratio of width to height) must be the same as the defaults, i.e., the aspect ratio should be 1:1.

Scale increase rate is the parameter in the call to "DetectHaarCascade" function which specifies how quickly OpenCV [23] should increase the scale for face detections with each pass it makes over an image. Setting this higher makes the detector run faster (by running fewer passes), but if it's too high, the machine may jump too quickly between scales and miss faces. The default in OpenCV [23] is 1.1, in other words, scale increases by a factor of 1.1 (10%) after each pass. This parameter may have a value of 1.1, 1.2, 1.3 or 1.4. We have set it to 1.2, which means it will run a moderate number of passes, thus the face detection will be accurate as well as fast. The lower the value, the more "thoroughly" haar-detector will check the image for a "face", but naturally, will take more time.

As mentioned earlier, a database of 10 individuals was created, where 5 images of each person were stored. For simulation, we tested the face recognition system for 10 known people and 5 unknown people. The observations were carried out in artificial fluorescent lighting conditions. Subjects were asked to show their faces 10 times before the camera for each threshold value. Some of the results obtained were as follows:

TABLE I. KNOWN PERSON 1

Threshold	Correct recognition	False recognition	Not recognized
1350	9	0	1
1400	9	0	1
1450	8	0	2
1500	10	0	0
1550	10	0	0
1600	9	1	0
1650	7	3	0
1700	5	5	0

TABLE II. KNOWN PERSON 2

Threshold	Correct recognition	False recognition	Not recognized
1350	2	0	8
1400	4	0	6
1450	7	0	3
1500	9	0	1
1550	9	0	1
1600	9	1	0
1650	7	3	0
1700	5	5	0

TABLE III. KNOWN PERSON 3

Threshold	Correct recognition	False recognition	Not recognized
1350	4	0	6
1400	5	0	5
1450	6	0	4
1500	9	0	1
1550	8	0	2
1600	9	1	0
1650	7	3	0
1700	6	4	0

TABLE IV. UNKNOWN PERSON

Threshold	Correct recognition	False recognition
1350	10	0
1400	10	0
1450	10	0
1500	10	0
1550	10	0
1600	9	1
1650	7	3
1700	6	4

It can be observed from the tables above that, the most promising threshold value is 1500. So, we used the value 1500 in our configuration which gave us an average recognition accuracy of 95%.

Fig. 15: Top View of the Prototype.

Fig. 16: Door Opened for Known Person.

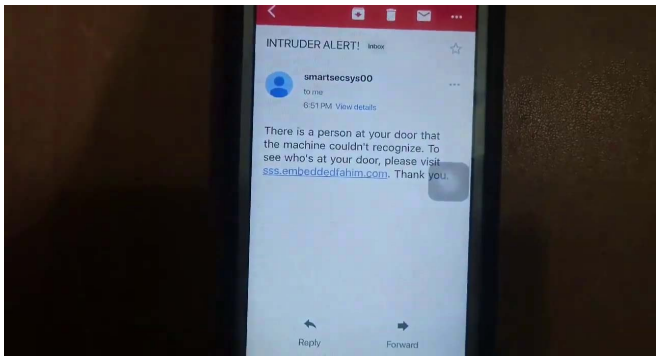

Fig. 17: Email Notification.

VII. Advantages

The main advantage of this system is that it's very small in size and lightweight, thus can be easily integrated into wooden/metal doors. It has a good recognition rate with 95% accuracy and the recognition time is less than 0.5 second. It's extremely power efficient requiring only 5 watts of power to operate. Low power consumption enables the system to run on batteries for hours at a stretch in case of power failure. Also, it costs only around 80 USD (including web hosting & domain cost for 1 year) to build this system with the current facilities. The low cost and long battery life increase usability and overall efficiency.

The aim of this system is to assist people in maintaining security in their homes, offices and other important places. The system can efficiently manage the entry of people at a particular place and with the help of IoT it can also materialize remote controlling of entrances. Also, it saves valuable time of people and reduces the hassles of the security guards and staff, increasing the degree of security.

VIII. Areas of Further Development

As this is only the first prototype of the system, there is a lot of room for improvements. For instance, there is no Graphical User Interface (GUI) for the system on the *Pi*. So, a user-friendly Graphical User Interface (GUI) will be developed using Python [24] in the future. Furthermore, the Eigenfaces algorithm may be replaced with a more efficient face recognition algorithm such as the Fisherface algorithm [34]. Addition of GSM/GPRS module would make the system self-reliant in terms of communication. Addition of a fingerprint sensor and Radio Frequency Identification (RFID) module would further increase the level of security and precision. Night vision camera module will increase the efficiency of recognition in dark environments.

IX. Conclusion

In this modern era, machine learning and IoT have become two of the most prominent fields which have made our lives easier, safer and efficient through variety of their applications. In almost every aspect of our daily life, we can see the benefits of these fields. However, our effort was to develop a helping hand for maintaining security at important places. We used Viola-Jones algorithm to detect faces and Eigenfaces algorithm to recognize people. Faces were extracted out of images and the machine was trained with some positive and negative images. We then successfully tested the system with different configurations and different known and unknown people. The test results were recorded and we achieved 95% accuracy in recognition in fluorescent lighting conditions. Lastly, areas of further development were discussed. As there is no end of perfection, with more resources and logistics support plus keen observation of its performance on different sectors, the scope of improvement of the system is still on table.

X. Acknowledgement

The authors of this paper would like to express profound gratitude to the Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh for their endorsement. Without their support, we could never pursue such a complicated project.

References

[1] Paul Viola, Michael J. Jones, "Robust Real-Time Face Detection", published on the International Journal of Computer Vision 57(2), 2004.

[2] M. A. Turk and A. P. Pentland, "Face Recognition Using Eigenfaces", published on the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 91, pages 586 - 591, 1991.

[3] Hteik Htar Lwin, Aung Soe Khaing, Hla Myo Tun, "Automatic Door Access System Using Face Recognition", published on the International Journal of Scientific & Technology Research (IJSTR), Volume 4, Issue 06, June 2015, pages 294 – 299, ISSN: 2277-8616.

[4] de Leeuw, Karl; Bergstra, January 2007. "The History of Information Security: A Comprehensive Handbook". Amsterdam: Elsevier. pp. 264–265. ISBN: 9780444516084.

[5] Ole Helvig Jensen, "Implementing the Viola-Jones Face Detection Algorithm", 2008.

[6] Daniel Georgescu, "A Real-Time Face Recognition System Using Eigenfaces", published on the Journal of Mobile, Embedded and Distributed Systems, vol. III, no. 4, 2011.

[7] Liton Chandra Paul, Abdulla Al Sumam, "Face Recognition Using Principal Component Analysis Method", published on the International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Volume 1, Issue 9, November 2012.

[8] Shervin Emami, Valentin Petruț Suciu, "Facial Recognition using OpenCV', published on the Journal of Mobile, Embedded and Distributed Systems, vol. IV, no. 1, 2012, ISSN 2067 – 4074.

[9] I.Yugashini, S. Vidhyasri, K. Gayathri Devi, Design and Implementation of Automated Door Accessing System with Face Recognition, International Journal of Science and Modern Engineering (IJISME), Volume-1, Issue-12, November 2013.

[10] Prathamesh Timse, Pranav Aggarwal, Prakhar Sinha and Neel Vora, "Face Recognition Based Door Lock System Using OpenCV and C# with Remote Access and Security Features", published on the International Journal of Engineering Research and Applications, Vol. 4, Issue 4 (Version 6), April 2014, pp.52-57, ISSN: 2248-9622.

[11] Ayushi Gupta, Ekta Sharma, Neha Sachan and Neha Tiwari, "Door Lock System through Face Recognition Using MATLAB", published on the International Journal of Scientific Research in Computer Science and Engineering, Vol-1, Issue-3, 30 June 2013.

[12] “Buzzer,’ Last accessed on February 1, 2019, at 12:00:00PM [Online]. Available: https://www.techshopbd.com/product-categories/buzzers/177/buzzer-techshop-bangladesh/
[13] “Raspberry Pi,” Last accessed on February 1, 2019, at 12:05:00PM [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/
[14] “Arduino Uno,” Last accessed on February 1, 2019, at 12:10:00PM [Online]. Available: https://www.arduino.cc/en/Guide/ArduinoUno
[15] “Servo Motor,” Last accessed on February 1, 2019, at 12:15:00PM [Online]. Available: https://www.techshopbd.com/product-categories/motors/1337/servo-motor-mg996-techshop-bangladesh
[16] Barrett, Steven Frank; Pack, Daniel J. (2006). “Microcontrollers Fundamentals for Engineers and Scientists”. Morgan and Claypool Publishers. pp. 51–64. ISBN 1-598-29058-4.
[17] “2000 mAh 18650 Li-Ion Battery,” Last accessed on February 1, 2019, at 12:20:00PM [Online]. Available: https://www.rcproductbd.com/product/li-po-18650-battery-3-7v-2000mah-capacity-tasted/
[18] “Battery Management System,” Last accessed on February 1, 2019, at 12:25:00PM [Online]. Available: https://www.allmartbd.com/5v-2a-power-bank-board.html
[19] “20x4 LCD,” Last accessed on February 1, 2019, at 12:30:00PM [Online]. Available: https://www.allmartbd.com/lcd2004-5v-blue-backlight.html
[20] “Pull-down resistor,” Last accessed on February 1, 2019, at 12:35:00PM [Online]. Available: https://grantwinney.com/using-pullup-and-pulldown-resistors-on-the-raspberry-pi/
[21] “Camera Module,” Last accessed on February 1, 2019, at 12:40:00PM [Online]. Available: http://bdspeedytech.com/index.php?route=product/product&product_id=1878
[22] “1-watt LED,” Last accessed on February 1, 2019, at 12:45:00PM [Online]. Available: https://components101.com/diodes/1-watt-led
[23] “OpenCV,” Last accessed on February 1, 2019, at 12:50:00PM [Online]. Available: https://docs.opencv.org/2.4.13.7/
[24] “Python,” Last accessed on February 1, 2019, at 12:55:00PM [Online]. Available: https://docs.python.org/2/index.html
[25] “AT&T Laboratories Cambridge face database,” Last accessed on February 1, 2019, at 01:00:00PM [Online]. Available: https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[26] “PHP,” Last accessed on February 1, 2019, at 1:05:00PM [Online]. Available: http://php.net/manual/en/
[27] “HTML,” Last accessed on February 1, 2019, at 1:10:00PM [Online]. Available: https://devdocs.io/html/
[28] “CSS,” Last accessed on February 1, 2019, at 1:15:00PM [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/CSS
[29] “MySQL,” Last accessed on February 1, 2019, at 1:20:00PM [Online]. Available: https://dev.mysql.com/doc/
[30] “FTPLIB,” Last accessed on February 1, 2019, at 1:25:00PM [Online]. Available: https://docs.python.org/2/library/ftplib.html
[31] “SMTPLIB,” Last accessed on February 1, 2019, at 1:30:00PM [Online]. Available: https://docs.python.org/2/library/smtplib.html
[32] “Wi-Fi Router,” Last accessed on February 1, 2019, at 1:35:00PM [Online]. Available: https://ryanscomputers.com/network/router/tp-link-tl-wr840n-v2-300mbps-router.html
[33] “IEEE 802.11 b/g/n Standard,” Last accessed on February 1, 2019, at 1:40:00PM [Online]. Available: https://www.webopedia.com/TERM/8/802_11.html
[34] “Fisherface Algorithm,” Last accessed on February 1, 2019, at 1:45:00PM [Online]. Available: http://www.scholarpedia.org/article/Fisherfaces