■ 317

Development of Face Recognition on Raspberry Pi for Security Enhancement of Smart Home System

Teddy Surya Gunawan*[1], Muhammad Hamdan Hasan Gani[1],
Farah Diyana Abdul Rahman[1], Mira Kartiwi[2]
[1]Department of Electrical and Computer Engineering, International Islamic University Malaysia
[2]Department of Information Systems, International Islamic University Malaysia
Jalan Gombak, 53100 Kuala Lumpur, Malaysia
e-mail: tsgunawan@iium.edu.my

Abstract

Nowadays, there is a growing interest in the smart home system using Internet of Things. One of the important aspect in the smart home system is the security capability which can simply lock and unlock the door or the gate. In this paper, we proposed a face recognition security system using Raspberry Pi which can be connected to the smart home system. Eigenface was used the feature extraction, while Principal Component Analysis (PCA) was used as the classifier. The output of face recognition algorithm is then connected to the relay circuit, in which it will lock or unlock the magnetic lock placed at the door. Results showed the effectiveness of our proposed system, in which we obtain around 90% face recognition accuracy. We also proposed a hierarchical image processing approach to reduce the training or testing time while improving the recognition accuracy.

Keywords*: eigenface; Principal Component Analysis; face recognition; Smart Home System.*

1. Introduction

Nowadays, there is a growing interest in the smart home system using Internet of Things. According to the report in [1], 72% respondents said self-adjusting thermostat and 71% said that doors that can be locked from a remote location, were the most important features when it comes to the most desired smart home devices. Face recognition is one of the most used biometric identidication system beside fingerprint and iris [2]. A typical face recognition system is shown in Figure 1.
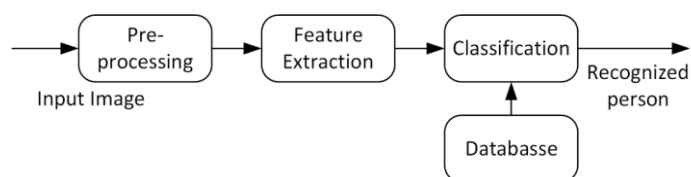

Figure 1. Typical Face Recognition System

There are many algorithms have been developed for face recognition algorithm, including appearance based [3], active appearance [4], support vector machines (SVM) [5], Bayesian model [6], deep learning neural network [7], and texture based [8]. In this research, we will focus on appearance based face recognition which includes direct correlation, eigenface, and fisherface. Unlike direct correlation algorithm that uses face image in their original image size, eigenface and fisherface algorithms reduce the image to the most discriminating factor and find similarity between images in a reduced dimension image size [9]. The fisherface algorithm uses inner class information for face classification and it can use multiple faces of a person to establish in-class variation to maximize class separation. In contrast, the eigenface algorithm uses one image per person applying the variation in one image to the overall recognition process. The eigenface algorithm is susceptible to the variation in illumination or facial expression. However, the computational requirement is lesser compared to the fisherface

algorithm. Hence, the eigenface algorithm will be implemented on Raspberry Pi as in [10] which has limited computational capability.

Raspberry Pi runs on Linux, a very light open source operating system that is available to anyone. The developers of this device seemed to think that it can be further reduced down hence they removed the complicated GUI (Graphical User Interface) and the unnecessary drivers and created the Raspbian OS. It is a minimalist operating system that also uses a cursor and not the command prompt as some servers do [11]. In this paper, the face recognition security system implemented on Raspberry Pi will be connected to the smart home control system proposed in [12]. The next sections will present the design, implementation, and experimental results.

2. Design of Face Recognition System using Raspberry Pi
In this section, the hardware design, software design, and algorithm design will be presented in more details.

2.1. Hardware Design
Figure 2 shows the proposed block diagram of face recognition system using Raspberry Pi. The Raspberry Pi is connected to the camera module which are shown in Figure 3. Raspberry Pi has the same processing capability of a single core processor with built-in graphics which can support up to 1080p video standard using its High Definition Media Input (HDMI) port. The ultrasonic sensor is used to detect the human presence. The relay circuit is connected to Raspberry Pi and provides an interface to the high voltage magnetic lock. The status could be shown using LED indicators and/or further send SMS using GMS module, or email using internet connection, or as part of smart home system as proposed in [12]. Finally, the overall power required is less than 1 A, so that a compact power supply, i.e. smartphone charger, could be used as the power supply.
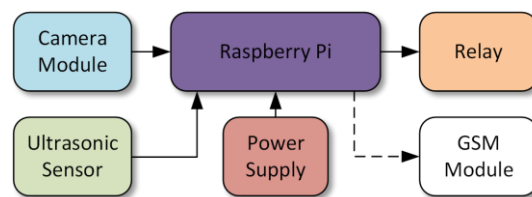

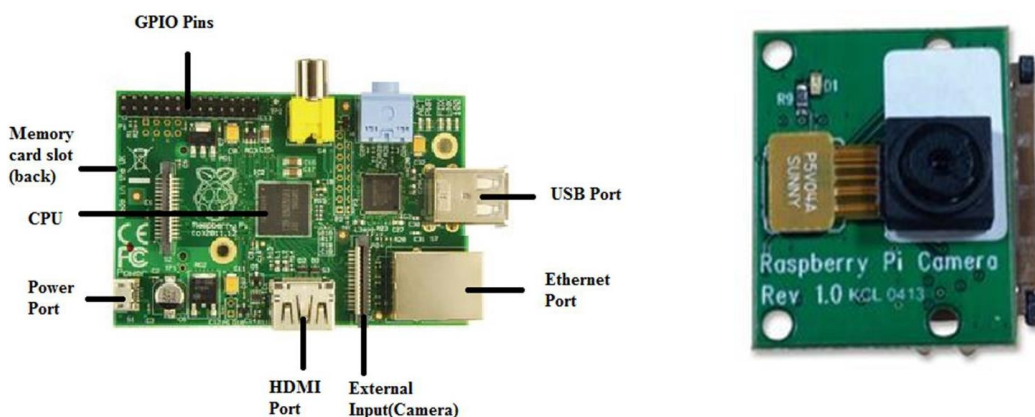Figure 2. Block Diagram of Proposed Hardware

Figure 3. Raspberry Pi and Camera Module

In this research, the external input port was attached with a camera made for Raspberry Pi called the Rpi Camera Board. It features a 5MP camera with a 1080p resolution and is capable of delivering high data rates of pictures and videos. Although other cameras also work together with the Raspberry PI board, some developers argued that using the RPi camera board provides faster processing compared to cameras that are attached using the USB port as described in [13]. This port shares the power with the other input output modules such as a keyboard and mouse, so an additional device requires an USB splitter or additional powered hubs to connect them.

2.2. Software Design

Figure 4 shows the flowchart of the proposed face recognition system. First, it reads ultrasonic sensor to detect the human presence, which could be set to check every second (configurable). If the human presence is detected, then the camera will capture the face image. The face detection routine will localize and segment the face region only. The face image is then fed in to the face recognition routine. If the recognized face is detected, the system will unlock the door by turning of the magnetic lock. After 30 seconds (configurable), the system will lock again the door by turning on the magnetic lock. The system will then start from the beginning.
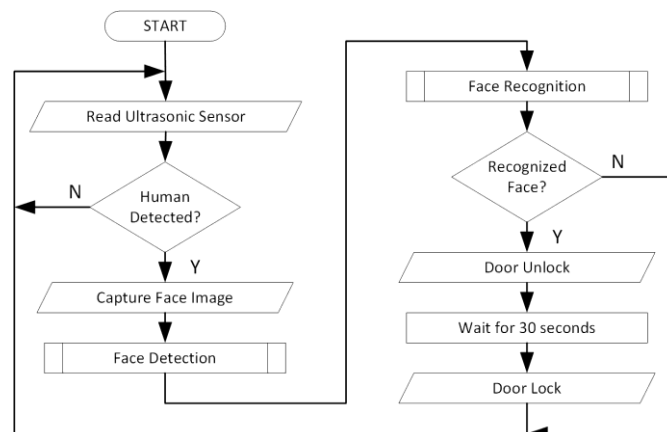

Figure 4. Flowchart of Face Recognition Security System

2.3. Algorithm Design

Figure 5 illustrates the proposed face recognition algorithm. The face detection algorithm will detect and segment the face from the overall image. Then, it does the necessary aligning and further cropping and conversion from color to grayscale. The eigenfaces feature vector will be extracted from this process. After that, the classification algorithm will use principal component analysis (PCA) to compare the input feature vector with the database, in which it will decide whether the input face image is similar with the registered face image. If it is recognized, then the system will turn on the magnetic lock to unlock the door.

The input face image occupies $p \times q$ dimensional vector space. For example, an image with 128×128 pixels has 16384-dimensional image space. The PCA selects a set of possibly correlated feature vectors into a smaller set of uncorrelated variables. In other words, PCA algorithm tries to reduce the dimensionality of the feature vectors while maintaining its unique characteristics.

Let $X = \{x_1, x_2, \cdots, x_N\}$ be a random vector with observations $x_i \in \mathbb{R}^d$. The mean μ, covariance matrix S, eigenvalues λ_i abd eigenvectors v_i of S could be calculated as shown in Eq. (1) to (3) as follows:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{1}$$

$$S = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)(x_i - \mu)^T \qquad (2)$$

$$Sv_i = \lambda_i v_i, i = 1,2,\cdots,N \qquad (3)$$

After that, we order the eigenvectors descending by their eigenvalue. The k principal components are the eigenvectors corresponding to the k largest eigenvalues. The k principal components of the observed vector x are given by:

$$y = W^T(x - \mu) \qquad (4)$$

where $W = (v_1, v_2, \cdots, v_k)$. The reconstruction from the PCA basis is then calculated as

$$x = Wy + \mu \qquad (5)$$

The Eigenfaces method then performs face recognition by projecting all training samples into the PCA subspace, projecting the query image into the PCA subspace, and finding the nearest neighbor between the projected training images and the projected query image.
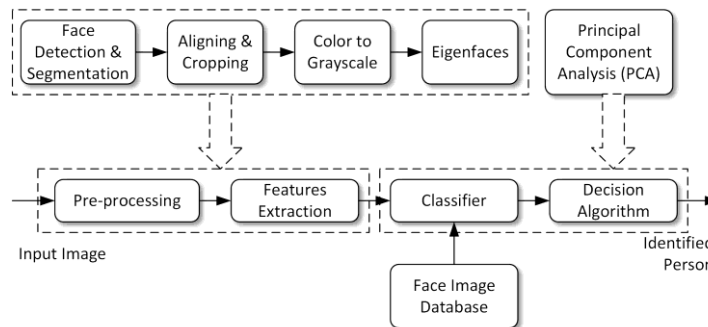

Figure 5. Face Recognition Algorithm

3. Implementation

Figure 6 shows the complete prototype of face recognition security system. The box contains the Raspberry Pi, ultrasonic sensor, reset button, LEDs and the power input and output. The prototype will be placed beside a door and connected to a magnetic lock as shown in Figure 7 which is turned off/unlock when the authorized user accesses the system. If an unauthorized personnel tries to access the door will stay locked and the user image will be stored in the memory. The prototype box will be placed around 1.6m from the ground and magnetic lock will be placed on top of the door. The power supply will be taken from a wall plug and these are the only input and output from the box.

On the software part, Raspbian OS is used as the operating systems for Raspberry Pi. Next the Python and OpenCV library was installed for the algorithm implementation. To train the faces into the library, we use the "train.py" algorithm in the OpenCV library. The training data should be loaded into the script. These images will be captured using the code "capture-positives.py". This code will continuously capture images into the training data folder. The training data also requires a negative training images that are built into the Raspberry pi library. Sets of 10 images for each person is trained at a time and "train.py" script is executed. Sample of the training images is shown in Figure 8.

The training data given will produce an output named "training.xml" file which contains the positive data processed into it. This process can also be done using a full fledged computer to shorten the training time. An average of 10 minutes were required using a smaller picture to process using the Raspberry Pi. Finally, ports are initialized using the terminal in root mode to access the GPIO pins. The initial set up was done using a servo but this set up can be replaced to magnetic lock or solenoid lock by simply changing the Pulse Width Modulation (PWM) pin to

any other GPIO pins and initialize the pin in the script. To run the code, the terminal window on the Raspberry Pi is opened and the python code "box.py" is executed. When the red light is turned on the system is ready to execute face recognition.
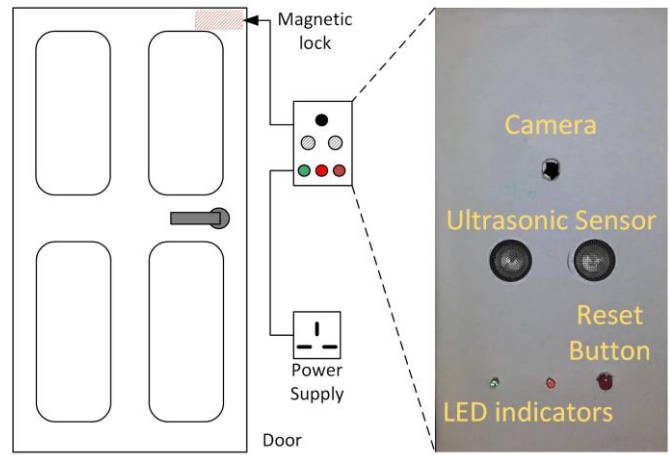

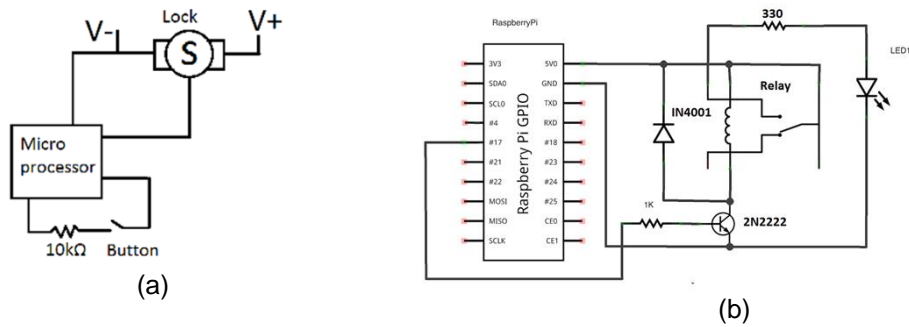Figure 6. Complete Prototype of Face Recognition Security System using Raspberry Pi

(a)

(b)

Figure 7. Circuit diagram of (a) Magnetic Lock, (b) Relay

positive_001
503 kB
651 × 792

positive_005
481 kB
637 × 774

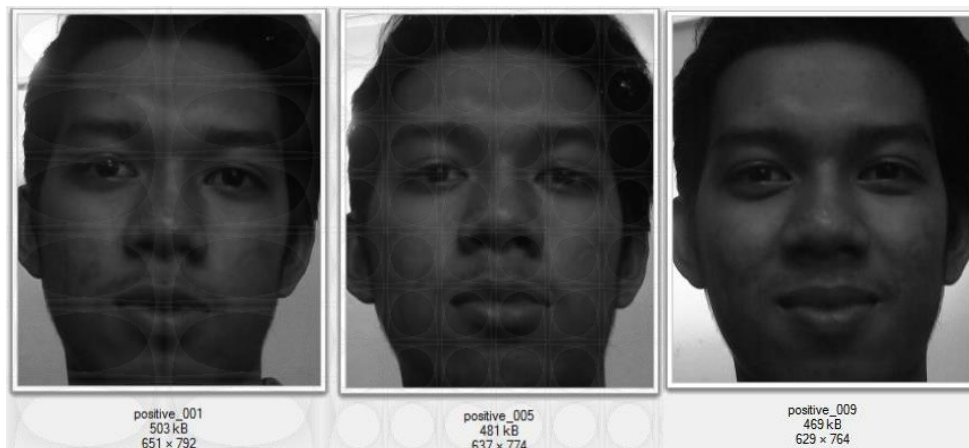positive_009
469 kB
629 × 764

Figure 8. Samples of Training Images for Face Recognition

4. Results and Discussion

In this section, the output of the training stage, experiment on face detection, and experiment of face recognition will be elaborated. The performance evaluation includes recognition rate, reliability, memory management, and power management.

4.1. Output of the Training Stage

Figure 9 shows the result of the training process, in which the positive eigenfaces were produced for each recognized face. The feature extraction will collect the data on the sample image using PCA and all the information is sent to another program for face recognition. The face recognition algorithm will compare this information with previously collected data from the library compiled during image training process and outputs the result if the person is recognized or not.

Figure 9. Positive Eigenfaces of the Training Data

4.2. Face Detection Experiment

For identifying the minimum time it takes to run an image of multiple quality and size the face detect algorithm was used as a baseline algorithm. Initial results by varying the size of image captured is shown in Figure 10. The problem with the Raspberry Pi is that it has limited processing power. Although the image captured by the camera has very high resolution but the processing time required for an image is approximately 30 seconds per frame for an image of resolution 1920x1080 which is a HD quality picture. However if a smaller image size is used the processing time is also reduced. By using a picture with dimension 640x480, the image seems to process much faster at below 10 seconds and a smaller frame size of 320x240 only takes about 200 milliseconds (ms).
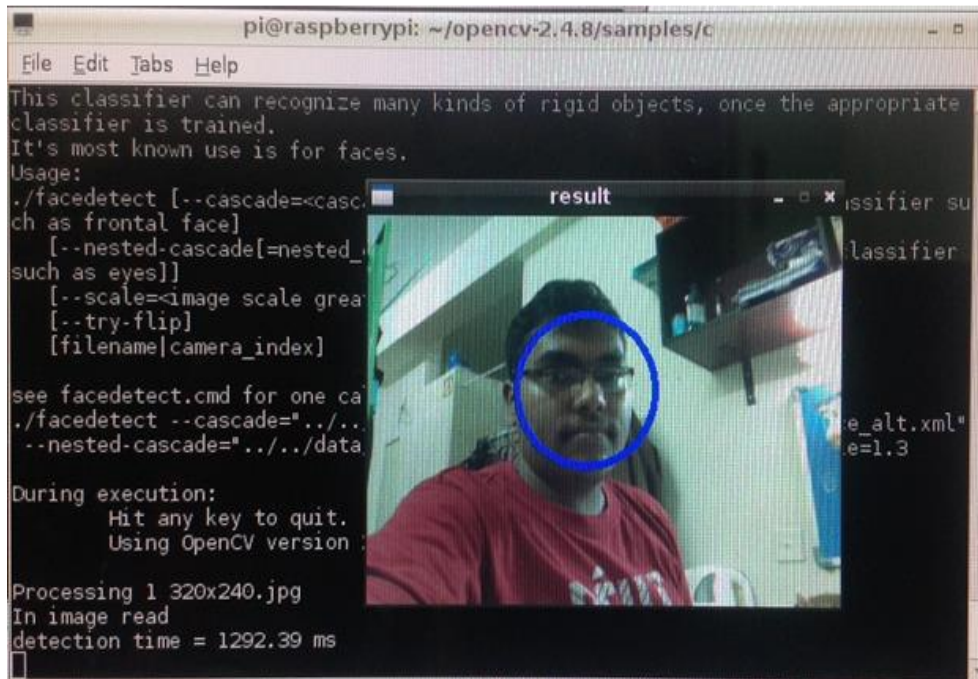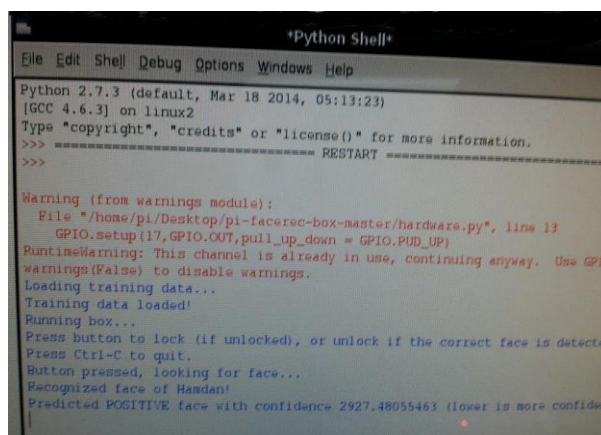

Figure 10. Face Detection Experiment

Table 1. Image Resolution versus Processing Time

Image Resolution	Processing Time (ms)
1920x1080	109945
1280x960	25081
640x480	5695
320x240	1451

Another limitation of the facial recognition is the physical background of the image being processed. If an image is taken at low light conditions the algorithm fails to identify the target's parameters and fails to read the required data. To avoid this, the images are taken from a good light source room and with a bright background. The face recognition will use more resources if more image of people are given to it to process. Multiple samples in the same image will also result in higher time for processing. For example if this program is used for locking house doors then only 1 person can unlock it at a time for more accuracy. However if the implementation are only used to generate for one person that it will be just fine. Table 1 shows the average time taken for images of varying resolution.

4.3 Face Recognition Experiment

When the "box.py" algorithm is ran, it will load the previous trained data. When a positive image is identified, the green LED will light up signaling that the face is recognized and the door can be opened. Figure 11 shows the output from the python program. If an image is captured but the face could not be identified then the red LED will keep lighting up signaling that the face is not detected. To capture another image the reset button is pressed again. If a negative image is detected, the red LED light will be turned on and the user is denied permission as the lock will be kept close.

Figure 11. Output Sample of Recognized Face

The performance evaluation includes the recognition time, memory management, accuracy and power management. To measure the recognition time required, the system is evaluated ten times and the average was taken. It was found that the average recognition time was 15 seconds. It is worth mentioning that the time measured did not consider the background process, such as checking internet connection, system update, and other services

The system is tested using same person and different people including the authorized person and unauthorized person and the results show that the system recognizes the users 9 times out of 10. Therefore, it has an efficiency of 90% with a 10% tolerance. However, this only happened when the user changed some of the facial expressions.

The experiment was done using a brightly lit room with enough space between the user and the Pi camera. The best distance the user should stand to get the best accuracy is about 0.5m from the camera. Taking the picture from a closer distance will not get a good recognition as the whole face should be able to be seen inside the picture. When the captured image resolution is low, the accuracy seemed to drop because the image captures has not enough data to be processed. By using a hierarchical approach to this problem is reduced. The user image is captured in high resolution then shrunk to a smaller size before comparing to the training image. This produced a good recognition rate as well.

Using a training image of high resolution can also be used to get better accuracy but the Raspberry Pi's limited processing power causes the system to hang if the image above 720p is used for training. Therefore a smaller image is the best way to get the results. One of the limitation of the system is that the user must appear exactly as the training image captured by them. For example, if the intended user does not wears a spectacle when taking sample image then he cannot wear them during authorization. Sometimes, it is also sensitive when the user smiles wider than usual due to the size of mouth changing differently than the training image.

Memory management is also a very important aspect in the program execution. The program was tested using multiple input setting and number of input was varied to get the best recognition time. By training a set of 20 positive image with 200 negative image the "training.xml" data file was about 150MB and varying the negative image to 100 with the 10 positive image gives about 50MB training data. When the accuracy of these were tested, no changes in accuracy was detected but the data for training 3 authorized people cannot be ran at

the same time as Raspberry Pi does not have the computational power. The larger the memory the training data the longer the program takes to execute. So it is an important aspect where an authorized person does not want to wait long for the system to unlock the door.

The Raspberry Pi has a good power management system but the internal power supply is enough to the board itself but it is inefficient to provide power to external sources. Unlike the GPIO pins in the Arduino, the raspberry Pi has limited power to supply to the these pins making developers rely on different alternatives such as combining two boards together.

In this project the GPIO pins are the only source used to power up all the hardware to keep its simplicity and it is was insufficient to provide power to all the components. A low power relay was used to replace a normal relay but this can only be used for small voltage load. This is one of the limitation in the current Raspberry Pi board.

5. Conclusion

This paper has presented a face recognition security system using Raspberry Pi. Python and OpenCV was used to implement the feature extraction and classifier, in which we used eigenface and PCA. The prototype design for real world implementation has been elaborated, in which the output of face recognition algorithm will lock or unlock the magnetic lock placed at the door using relay circuit. We have discussed the limited processing capability of Rasperry Pi which affect the image resolution to be captured, processing time, as well as memory and power management. The recognition rate was found to be around 90% when tested with three persons. This proposed system could be connected using Internet to the smart home system for the added security capability. Further research includes optimization of hierarchical image processing, use different features extraction and classifer, or use parallel Raspberry Pi clusters to speedup the computation.

Acknowledgement

References

[1] i. networks. 2015 State of the Smart Home Report. https://www.icontrol.com/blog/2015-state-of-the-smart-home-report/], Retrieved on: 4 June 2017.

[2] J Galbally, S.Marcel, J Fierrez. Image quality assessment for fake biometric detection: Application to iris, fingerprint, and face recognition. *IEEE transactions on image processing*. 2014; 23: 710-724.

[3] MA Turk, AP Pentland. *Face recognition using eigenfaces.* in Computer Vision and Pattern Recognition. Proceedings CVPR'91. IEEE Computer Society Conference on. 1991: 586-591.

[4] GJ Edwards, TF Cootes, CJ Taylor. *Face recognition using active appearance models*. in European conference on computer vision. 1998: 581-595.

[5] G Guo, SZ Li, K Chan*. Face recognition by support vector machines*. in Automatic Face and Gesture Recognition. Proceedings of Fourth IEEE International Conference on. 2000: 196-201.

[6] B Moghaddam, T Jebara, A Pentland. Bayesian face recognition. *Pattern Recognition.* 2000; 33: 1771-1782.

[7] OM Parkhi, A Vedaldi, A Zisserman. Deep Face Recognition*. in BMVC*. 2015: 6.

[8] T Ahonen, A Hadid, M Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE transactions on pattern analysis and machine intelligence*. 2006; 28: 2037-2041.

[9] PN Belhumeur, JP Hespanha, DJ Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*. 1997; 19: 711-720.

[10] M Sajjad, M Nasir, K Muhammad, S Khan, Z Jan, AK Sangaiah, M Elhoseny, SW Baik. Raspberry Pi assisted face recognition framework for enhanced law-enforcement services in smart cities*. Future Generation Computer Systems.* 2017.

[11] S McManus, M Cook. Raspberry Pi for dummies. John Wiley & Sons. 2017.

[12] TS Gunawan, IRH Yaldi, M Kartiwi, N Ismail, NF Za'bah, H Mansor, AN Nordin. Prototype Design of Smart Home System using Internet of Things*. Indonesian Journal of Electrical Engineering and Computer Science.* 2017; 7: 107-115.

[13] E Upton, G Halfacree. Meet the Raspberry Pi. John Wiley & Sons. 2012.