*Original Research Article*

# A cost-efficient real-time security surveillance system based on facial recognition using Raspberry Pi and OpenCV


## ABSTRACT

Home surveillance systems are still challenging, particularly for patrolling or tracking subjects through CCTV images despite recent developments. Therefore, it is crucial to instantly identify human faces based on captured facial images in protection and surveillance. Identification of people, intrusion detection, and follow up access control of objective sites are examples of applications of such systems. This paper represented a cost-efficient real-time facial recognition-based surveillance system for home and small offices using raspberry pi and computer vision. In the application, first, the system tracks the detected individuals' faces in the frame and only focuses on the image content in these facial regions. Then, a powerful algorithm for recognising detected faces is used using a pre-provided face database. For the implementation in this paper, the most common Haar Cascade and Local Binary Pattern Histogram (LBPH) algorithms are used for facial detection and recognition. The system works perfectly in normal lighting conditions with accepted accuracy.



Keywords: Face Detection, Face Recognition, Surveillance, Raspberry Pi, OpenCV, Home Security.




## 1. INTRODUCTION

Crime rates have risen considerably in recent years, especially in developing countries [1][2]. Numerous occurrences, such as theft, burglary, and unwanted invasions, occur unexpectedly. This necessitates the installation of a security system capable of preventing unwanted entrance to private residences or small companies [3]. Till date, many security systems such as security cameras, password-protected door locking systems, and the security guard are used to unauthorized access. However, none of these methods are well enough, like a password-protected door locking system is vulnerable to forgery, fraud, and memory loss [4]. Because the possibility exists that unauthorized individuals could gain access to the secure facility by obtaining the password in an illicit manner. Besides, hiring more security guards is expensive for private houses and small businesses. Moreover, an ordinary security camera does not have a threat detection or warning system. So low-cost real-time surveillance for private homes and small businesses is a significant challenge.

In recent years there has been a significant development in facial recognition. OpenCV (Open Source Computer Vision Library) development is a significant breakthrough since it can be easily used to implement commonly used facial recognition algorithms such as Eigenface [5], Fisherface [6], and LBPH [7]. Facial recognition-based surveillance and warning systems can be a solution to tight security [7].

Furthermore, to keep costs minimal, it needs to be implemented using the existing CCTV camera system and less hardware. So in this paper, we propose a surveillance and warning system based on Raspberry Pi and Computer Vision that is low-cost and can be used on existing camera systems and perform satisfyingly.

To demonstrate our proposed system, we made a demo using a Raspberry Pi 4 Model B with the existing CCTV camera system. A Raspberry Pi camera module is used to make the face database. And code is written in Python using the OpenCV library. Moreover, the haar cascade (explained in section 2.1) classification is used to detect the face region in the video, and the LBPH algorithm (explained in section 2.2) is used to carry out facial recognition.

## 2. RELATED WORK AND MOTIVATION

The method of identifying or verifying a person from a digital image or video frame is known as facial recognition. For psychology, earlier studies on facial recognition can be traced back to engineering literature in the 1950s and 1960s. Some of the earliest results include research on Darwin's facial expression feelings [8]. In the 1970s, following the initial work of Bledsoe, the baton was retrieved by Goldstein, Harmon, and Lesk [9]. To automate the identification, they expanded his work and accounted for 21 subjective facial symbols, including hair colour and lip thickness. In 1991, Turk and Pentland [5] carried out the work of Sirovich and Kirby [10] by learning how to recognize faces within an image, which led to the first automatic facial recognition scheme and paved the way for potential advances in facial recognition technology under technological and environmental restrictions.

In recent years, many works have been done based on Computer Vision and Raspberry Pi. G. Xiang, Z. Quiyu, W. Hui, and C. Yan introduced a facial recognition method based on a variation of Local Binary Pattern Histograms. The regression technique is used to obtain facial features with the least complexity [11][12]. S. Guennouni, A. Ahaitoufa, and A. Mansouri proposed two main facial recognition strategies Edge-Orientation Matching and Haar-like function selection combined cascade classifiers [13]. In the paper "Face Recognition in Poor-Quality Video: Evidence From Security Surveillance," A. Mike et al. created a foolproof security system approach that incorporates subsystems such as surveillance CCTV, video management, and wireless backbone. Within the designated area guarded by CCTV cameras, the system is capable of detecting an intruder [14]. And recently, Faisal, F, and Hossain, S A developed IoT based intelligent security system using face recognition to unlock the door [15].

As stated before, the motivation of our work is to develop a low-cost and low-power consumption system that works on a private home and small business using an already installed CCTV camera and to fulfil these two purposes, Raspberry Pi and OpenCV is used to design the system (figure 1). The Raspberry Pi is a low-cost gadget that uses a conventional mouse and keyboard to connect to a computer monitor or TV. It's a versatile and handy machine that supports programming languages such as Scratch and Python. We used Raspberry Pi 4 with 8Gb RAM and Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz for the implementation. For experiments carried out in the paper, we e used a Raspberry Pi camera with a resolution of 5 megapixels and 30 frames per second and a regular camera for video recording. And thanks to the python wrapper of OpenCV, Raspberry Pi can access the OpenCV library easily.

OpenCV is a massive open-source repository for computer vision, machine learning, and image analysis. It plays a crucial role in real-time computer vision tasks such as recognizing objects, faces, or even human handwriting from images. We will talk about face detection first.

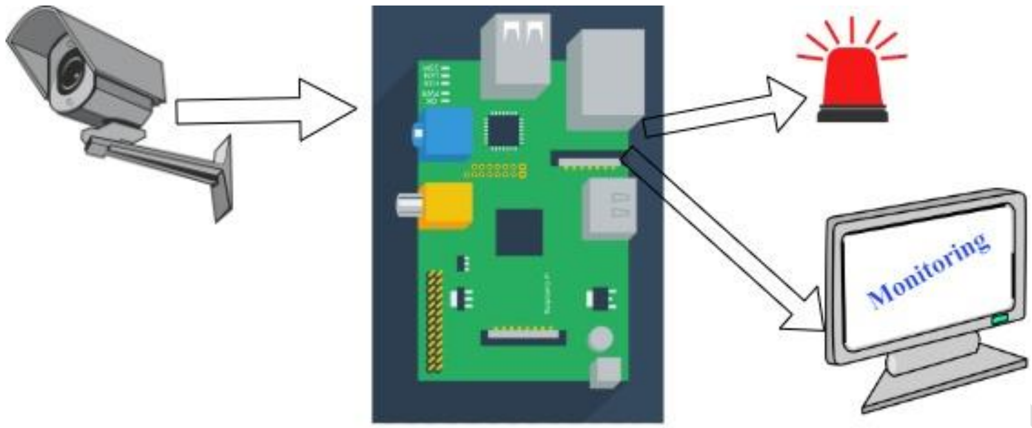



**Figure 1: Schematic diagram of the detailed idea of the operation (Image Source: CleanPNG and Pixabay)**

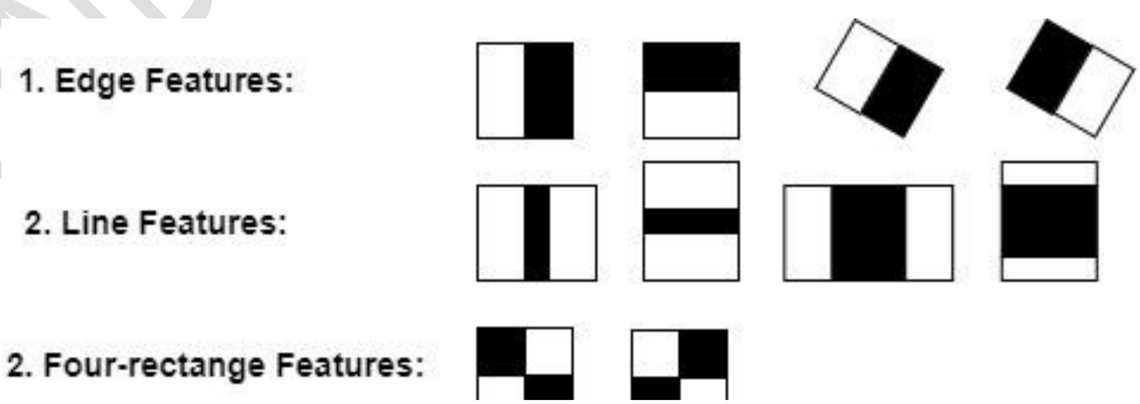
## 2.1 Face Detection

Haar Cascade is one of the most popular algorithms, and it is convenient to use in low computing power systems such as Raspberry Pi. However, when we deal with facial detection, the algorithm requires taking a lot of positive images (face images) and negative images (faceless images) and processing them in four stages to get the final classifier ranging of Haar Feature, Building Integral Images, Training in Adaboost, constructing Cascade Classifiers [16].

Firstly, extract the Haar features from the image. The Haar features shown in figure 2 are used in our system. Each function(what function?) is a single value calculated by subtracting the number of pixels covered by the white rectangle from the sum of pixel intensity values covered by the black rectangle [17].

Secondly, to construct an integral image (a data structure and algorithm to generate a sum of values from a rectangular subset of a grid). The goal of utilizing this integral image is to reduce computation consumption when obtaining pixel intensity summaries within a window. The idea is to convert the input images to a summarized area table, where the value is an add-up of intensities of all pixels on the upper and left side of any given point (x, y) of that table [18].

Thirdly, because some detected features are meaningless, a concept called Adaboost is used to detect the best features and train the classifiers with the chosen best features. This algorithm creates a "fortified" classification as a linear combination of simple, weighted "weak" classificatory. [19]

(a)


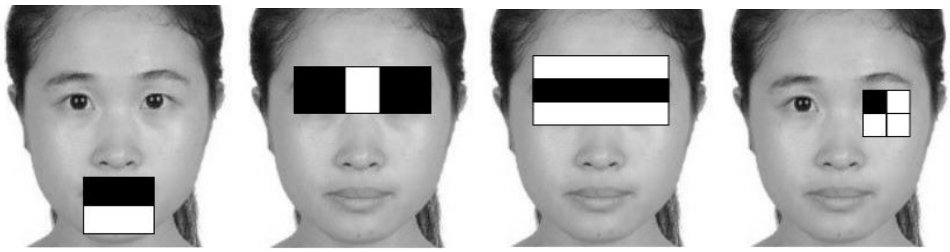
(b) 

**Figure 2: (a) Different Haar features. (b) Extracting Haar features from the face**

Fourthly, the cascade grouping consists of a series of steps, each containing a category of weak learners, which are basic classifiers known as stumps. Each classifier phase marks the region that is described as either positive or negative by the current position of the sliding window. Positive means an object has been found, and negative reveals that no objects have been found. The steps are designed as rapidly as possible to reject negative samples. The hypothesis is that there is no object of interest in the large majority of windows. On the other hand, positive things are rare and worth checking. [18]

### 2.2 Face Recognition

The world's leading and fast biometric technology is facial recognition. It uses the most accessible component of the human body, the face, in a non-intrusive way. [12] The algorithm for facial recognition analyzes the characteristics of a face from the input image. OpenCV[7] is widely used to implement algorithms for facial recognition. Some commonly used OpenCV algorithms are:

a. Eigenface: The principal analysis of the component is based on the classification of images to extract images from the same set of images. It is essential to match the eyes of each picture and image under the same conditions of lighting. [5].
b. Fisher face: This algorithm is based on linear discrimination analysis to detect patterns. It uses class labels and information about data points. Various lighting conditions have a minimal impact on its grading process [6].
c. LBPH: The picture is divided into pixel-size blocks of 3 x 3 in this algorithm. The luminosity intensity of the pixels surrounding the centre pixel is compared and a value of 1 or 0 for each surrounding pixel, depending on the difference. A result is several 8 bits that are converted into a decimal. The image's luminosity has no impact on the algorithm. Histograms are used to find and efficiently process frequencies of occurrences of values [20].

For the implementation, the LBPH algorithm has been used earlier. LBPH considers that the texture descriptor is helpful for the symbolization of the face since facial details can be divided into compositions of the micro-textures. In general, LBPH is conducted in three steps. The picture is first taken and internally converted to the grey image, next checked whether the pixels are mapped into a facial or a non-face picture based on haar characteristics and detect the face region in the picture. Then the Local Binary Pattern (LBP) operation, an easy and efficient texture user, will label a pixel image by threshing the pixels' neighbourhood and treating the output as a binary number. [21]. It also notes that detection efficiency is considerably enhanced when LBP is connected to the histograms of oriented gradient (HOG) descriptor [22].

A data vector is used to view the face images. It scans each block's pixel values and places them in 3*3 windows. Eight neighbours have the key pixel with the threshold value. Compared the current pixel value to neighbouring values, set it to 1 if it is greater than the centre; otherwise, it is 0. When compared, the binary values will be considered, and the corresponding decimal value will be changed in the central pixel by reading them in the clockwise direction. For example, the figure below shows a 3x3 intensity matrix where the middle value (100) is treated as the threshold value. Based on the threshold value (cells that have a higher value than threshold put as 1 and lower put as 0) a binary value matrix has been created. After converting the binary value that is saved in an array (clockwise) to decimal, we get a value for the middle cell, and that is 141 (figure 3).

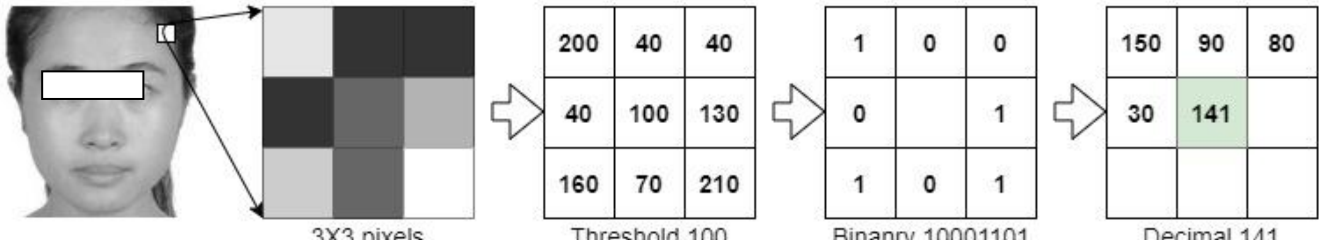



**Figure 3: Step-by-step demonstration of LBPH method**

## 3. IMPLEMENTATION

After studying and working on this project, it has been decided how to design and implement the system. The final implementation is as discussed conceptually before. It works in three steps. The first step is to create a dataset, the second step is to train the algorithm using that dataset, and the third step is based on the recognizer file the surveillance operation performed.

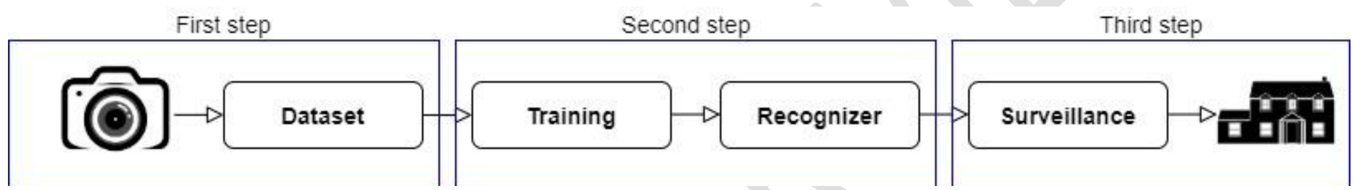



**Figure 4: The basic flowchart for how the system works**

Figure 4 represents the basic flowchart of the implementation. All the steps are discussed below -

### 3.1 Dataset

The most important part of this system is to create the data set. So the idea is to design the dataset creation in a more advanced way so that the system can handle captured photos instantly or import them from the available library. The workflow for data set creation is in figure 5.

From figure 5, it is clear that the system has two options, either capturing using the camera or importing images from local storage. Based on the user's decision, a new image will be captured/imported; then, the face detection algorithm will detect the faces and automatically crop only the face region of the frame and ready to store this face. For storage, there are different strategies for a new face or existing face. It will generate a face id for a new face and take the current input as the username, then store it in the dataset. For existing users, it will update the face image based on the existing user id and name. The implementation captures 60 images at once using the camera, and for updating based on the number of images present in the local storage, it stores 60 face images. Figure 6 shows examples of stored face data for a specific face ID in the dataset. When the dataset is ready or updated, the machine learning training algorithm can be called to train the model.

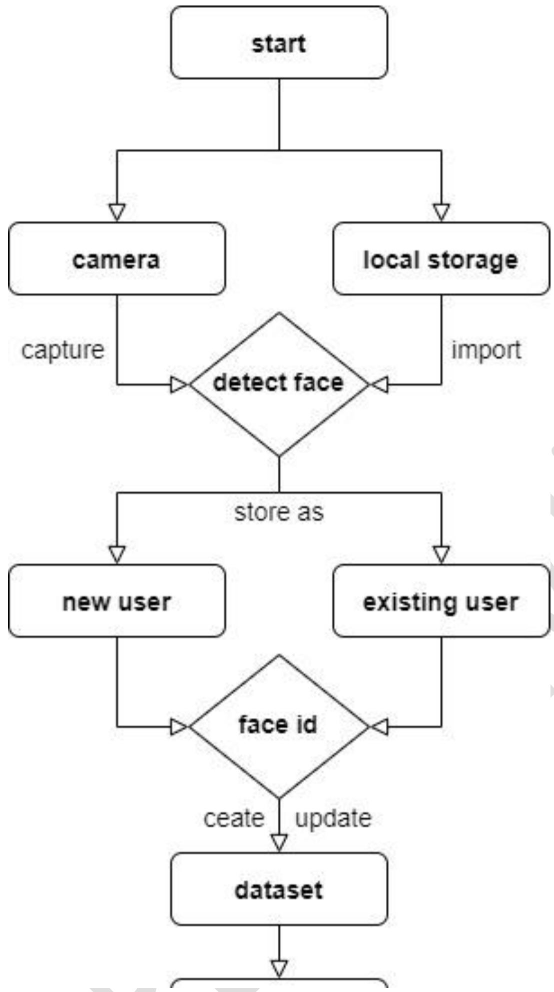



**Figure 5: Workflow for creating the data set**

**Figure 6: Sample of the dataset**

## 3.2 Training and Recognition

After finalizing the dataset, the next step is to train the algorithm to extract face features and store them for unseen futural images.

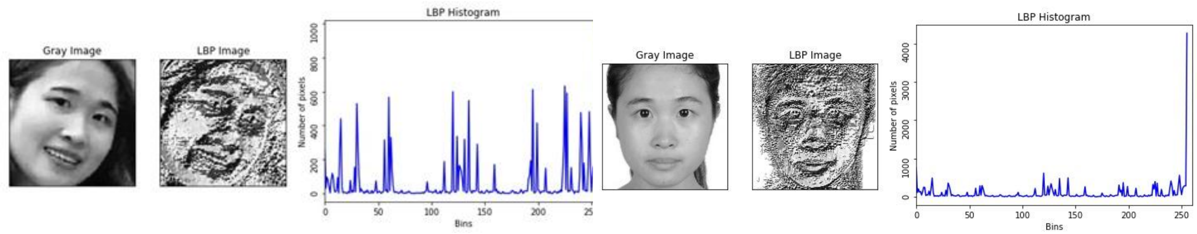



**Figure 7: LBP image and histogram for the training dataset**

Each face image is read one by one in the training process and add its label based on the face id. For training, first set a recognizer object by calling *LBPHFaceRecognizer_create()* function from the OpenCV library. The function extracted the LBP histogram (figure 7) information and saved the data trainer.yml file during training.

### 3.3 Operation

The goal of the system is to monitor the home or small business premises; if any unknown person enters the premises, then it will automatically notify the security person or house owner. The notification can be done in many ways like using an alarm (buzzer), lighting a led, sending an email, or a mobile text, and our designed system is capable of all three types of warning. Though the mobile SMS warning system is not implemented with the current system, it is still convenient to integrate this function by buying the SMS service from a push message service provider. Figure 8 shows the steps for the complete operations of the system. As shown in the flowchart, the camera will record video, and raspberry pi will analyze it constantly. Then the program will detect the face in the video, and it will call the recognizer function to check whether or not this face belongs to the database.

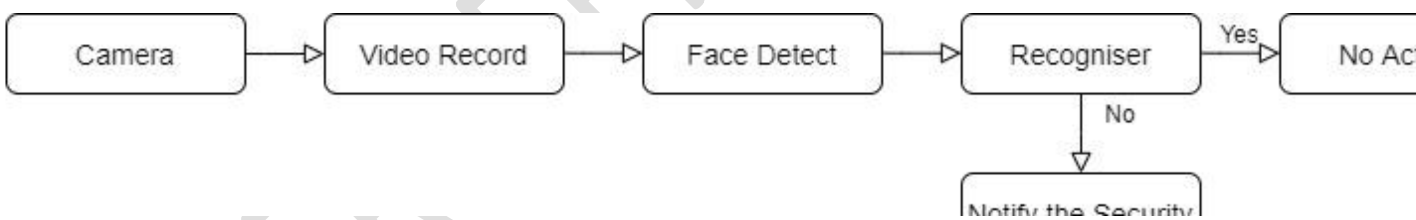



**Figure 8: The steps of how the system operates**

If a match of the detected face is found in the database, it will not take any action. In contrast, if it finds any unknown face, then the alarm notification will be automatically started; depending on the familiarity of the warning, it can, for example, light up a warning light and send an email to the owner or security person.

## 4. RESULT AND DISCUSSION

The Haar Cascade method in face detection is a feature-based methodology that is relatively resistant to place variations, as well as it supports the use of multiple cameras. The compact representation of face image and high-speed matching. However, it's found in our experiments that extreme differences in lighting conditions make it highly error-prone. These methods have trouble selecting the characteristics under bad lighting conditions, consequently making subjective decisions about which characteristics are relevant. The system has been tested with different faces and various lighting conditions. After checking the results, it is clear that the system detects quite accurately. Figure 9 represents some of the experiment results for real-time video footage.

Face recognition is much more complicated than face detection (in terms of greater error rate and involves more mathematical operations). Its performance depends primarily on the depth of the image, lighting conditions, facial poses and expressions, variations of faces in terms of poses, picture depths, and lighting conditions in recognized databases play a crucial role. For testing the implementation, some sample videos have been recorded under different lighting conditions and different facial poses, then checked the experiment results, either it worked well or not. Figure 10 represents screenshots of some of the experiments regarding face recognition and surveillance. Moreover, it can be clearly said that the system is recognizing the faces quite nicely though it is a low computation-powered system.

**Figure 9: Some screenshots of the face detection experiment**


**Figure 10: Some of the screenshots during the facial recognition experiment**

Also, it is performed very fast and quite accurately under artificial indoor lighting conditions in terms of face recognition. In terms of warning, in case of unknown people get found, the system can send warning signals in a couple of ways. An email notification is given in this paper (Figure 11). The system sends the administrator a warning email along with the timestamp and suspect image.


**Figure 11: Screenshot of the email notification in case of the unknown person detected**

Another goal was to replace the traditional high-cost computer with a low-cost raspberry pi. So it is always important to talk about the performance of Raspberry Pi in terms of training the algorithm and operation. Table.1 shows the data of the size of the dataset, total time consumption to train the model, and amount of memory consumed for saving the training data. For the experiment, the dataset was created with a total of five faces. For the first test, there were a total of 125 images (25 images for each face), and it has clearly shown the model has been trained fast; it took only 5.8 seconds and consumed only 18.1Mb memory. As we know, model prediction accuracy is dependent on a variety of factors, one of which is dataset size. As a result, we increased the dataset size incrementally while testing for accuracy and the computing power of the raspberry pi. Though raspberry pi has low computing power, it still trained the model relatively fast. For 500 images, it took around 29.57 seconds and consumed 72.5Mb memory from the storage.

**Table 1: Data chart of the number of photos, time consumption, and memory consumption**

| Number of photos in the dataset (5 faces) | Time is taken to train all the face data (seconds) | Size of the trainer (.yml) file (Mb) |
|---|---|---|
| 500 | 29.57 | 72.5 |
| 375 | 17.2 | 54.3 |
| 250 | 11.69 | 36.2 |
| 125 | 5.8 | 18.1 |

From figure 12(a) (the line chart of the total time taken to train against the number of photos), we can estimate that within 1 minute, it can train 1000 face images. The graph shows that up to 375 photos, the changes of time were regular, but for 375 to 500 photos, it took a little more time than previous. However, changes in memory consumption are pretty fixed, as shown in figure 12(b), the graph between trainer data against the number of images.

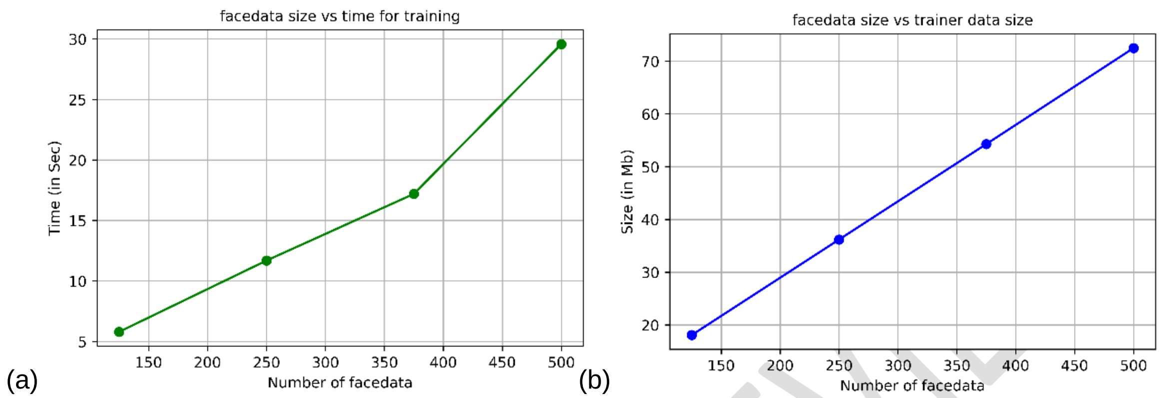



**Figure 12: (a) Represent the relation between time taken to train all the datasets (b) Represent the memory required to store trainer data to all the dataset**

So from the above result, it can be easily stated that raspberry pi can be confidently used in the home or small business surveillance application as a primary controller integrated with the existing security camera.

We have already talked about the implementation and results, but it is important to discuss whether it is an effective solution. Throughout the paper, we always underlined low-cost systems, and the device needs to keep running for 24 hours. Moreover, nowadays, most people own a personal laptop or desktop computer, so why not use the already available system rather than raspberry pi. Table 2 may clarify this confusion. We have compared the estimated power consumption of Raspberry Pi with traditional computers. It shows that a Raspberry Pi 3 consumes one-fourth (30.96kW) of the average power consumption of a laptop(126kW) and more than eight times less than a desktop computer(289.8kW) in a year. The newer Raspberry Pi 4(55.44kW) consumed little more power than Pi 3 but still very little than traditional computers. All the values calculated based on the power of datasheet available from the company and electricity price calculated based on the consumer electricity price in the USA.

Well, now we can say Raspberry Pi is very economical in terms of 24x7 surveillance. However, there are some other low-powered controlling boards available in the market. Why chose Raspberry Pi? Table 3 answers this question. First of all, Raspberry Pi is relatively cheap than other microcontrollers and easily available in the market.

**Table 2: Annual power consumption comparion between raspberry pi and traditional computers**

| Device | Load | Power Consumption (kW/h) | Price (USD) |
|---|---|---|---|
| Pi 3 B | full | 30.96 | 4.08 |
| Pi 4 B | full | 55.44 | 7.13 |
| Laptop | partial | 126 | 16.60 |
| Desktop Computer | Partial | 289.8 | 38.16 |

On the other hand, it always requires more computation power for computer vision applications, and Raspberry Pi has variants with up to 8Gb of RAM. Moreover, we did not consider another low-powered and low-cost board Arduino because Arduino is based on a microcontroller with less RAM and clock speed. So from our view, Raspberry Pi is a relatively decent choice for this application.

**Table 3: Comparision of price for respberry pi other similar comntrollers**

| Device | Price (USD) |
|---|---|
| Raspberry Pi 4 4/8 Gb RAM | $55/$75 |
| Odroid XU4 | $80 |
| Cubieboard4 CC-A80 | $130 |
| BeagleBoard X15 | $263 |

## 5. CONCLUSION

Face recognition systems will be increasingly used in the future for security purposes due to their superior performance compared to other security systems. An experimental study has been presented, which may be used for small private home surveillance and threat warnings. The result for the experimental system was found satisfactory. The system's software was written in Python and utilized facial detection and identification. Although this system's accuracy is more than 70%, it may be improved by incorporating new characteristics.

Additionally, due to the cameras' features, uncontrolled recording settings may result in ambient fluctuations such as illumination changes, changes in facial position, light shadowing, and blurring of the body or face. Following that, the quality of face captures made using video security cameras may influence the performance and efficacy of video-based facial recognition systems. This work can be extended with CNN-based facial recognition for better performance.

## 6. FUTURE WORK

In addition to our work, some recently developed methods can be used to improve performance and accuracy for security surveillance. For example, Lindner et al. has proposed a single-board computer-based facial identification and detection system in recent years. In addition to the Haar Cascade and LBPH algorithms, the authors used MTCNN [23] and FaceNet [24] for facial detection and recognition. According to their claims, they have a 97% success rate [25]. Therefore, when used for a human security service, it is appropriate. In addition to it, another work done by Gupta et al. where OpenCV is used to identify and extract the face, and a ResNeXt-inspired CNN [26] architecture called EmotionNet is used to classify the expressions of the subject [27]. So, emotion can be considered along with facial recognition regarding home security.

## REFERENCE:


[1] S. Ansari, A. Verma, and K. M. Dadkhah, “Crime Rates in India: A Trend Analysis,” *Int. Crim. Justice Rev.*, vol. 25, no. 4, pp. 318–336, Aug. 2015, doi: 10.1177/1057567715596047.

[2] J. van Dijk, P. Nieuwbeerta, and J. Joudo Larsen, “Global Crime Patterns: An Analysis of Survey Data from 166 Countries Around the World, 2006–2019,” *J. Quant. Criminol.*, 2021, doi: 10.1007/s10940-021-09501-0.

[3] E. L. Piza, B. C. Welsh, D. P. Farrington, and A. L. Thomas, “CCTV surveillance for crime prevention,” *Criminol. Public Policy*, vol. 18, no. 1, pp. 135–159, Feb. 2019, doi: https://doi.org/10.1111/1745-9133.12419.

[4] E. Jose, G. M., M. T. P. Haridas, and M. H. Supriya, “Face Recognition based Surveillance System Using FaceNet and MTCNN on Jetson TX2,” in *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, 2019, pp. 608–613, doi: 10.1109/ICACCS.2019.8728466.

[5] M. A. Turk and A. P. Pentland, “Face recognition using eigenfaces,” no. March 2019, pp. 586–591, 1991, doi: 10.5120/20740-3119.

[6] M. Anggo and La Arapu, “Face Recognition Using Fisherface Method,” *J. Phys. Conf. Ser.*, vol. 1028, no. 1, 2018, doi: 10.1088/1742-6596/1028/1/012119.

[7] S. Emami and V. P. Suciu, “Facial Recognition using OpenCV,” *J. Mobile, Embed. Distrib. Syst.*, vol. 4, no. 1, pp. 38–43, 2012, [Online]. Available: http://www.jmeds.eu/index.php/jmeds/article/view/57.

[8] P. Ekman, “Darwin’s contributions to our understanding of emotional expressions,” *Philos. Trans. R. Soc. Lond. B. Biol. Sci.*, vol. 364, no. 1535, pp. 3449–3451, Dec. 2009, doi: 10.1098/rstb.2009.0189.

[9] A. J. Goldstein, L. D. Harmon, and A. B. Lesk, “Identification of human faces,” *Proc. IEEE*, vol. 59, no. 5, pp. 748–760, 1971, doi: 10.1109/PROC.1971.8254.

[10] M. (1987). Sirovich, L., & Kirby, “kirbySirovich86.pdf,” *The journal of the Optical Society of America*, vol. 4. p. 519, 1987.

[11] G. Xiang, Z. Qiuyu, W. Hui, and C. Yan, “Face recognition based on LBPH and regression of local binary features,” in *2016 International Conference on Audio, Language and Image Processing (ICALIP)*, 2016, pp. 414–417.

[12] G. S. Nagpal, G. Singh, J. Singh, and N. Yadav, “Facial Detection and Recognition using OpenCV on Raspberry Pi Zero,” *Proc. - IEEE 2018 Int. Conf. Adv. Comput. Commun. Control Networking, ICACCCN 2018*, pp. 945–950, 2018, doi: 10.1109/ICACCCN.2018.8748389.

[13] S. Guennouni, A. Ahaitouf, and A. Mansouri, “Face detection: Comparing Haar-like combined with cascade classifiers and Edge Orientation Matching,” in *2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS)*, 2017, pp. 1–4.

[14] A. M. Burton, S. Wilson, M. Cowan, and V. Bruce, “Face recognition in poor-quality video: Evidence from security surveillance,” *Psychol. Sci.*, vol. 10, no. 3, pp. 243–248, 1999.

[15] F. Faisal and S. A. Hossain, “Smart Security System Using Face Recognition on Raspberry Pi,” in *2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 2019, pp. 1–8, doi: 10.1109/SKIMA47702.2019.8982466.

[16] A. Davison, “Face detection and tracking,” *Kill. Game Program. Java*, vol. 2, no. May, pp. 1–17, 2011, [Online]. Available: http://fivedots.coe.psu.ac.th/~ad/jg/nui07/.

[17] M. Khan, S. Chakraborty, R. Astya, and S. Khepra, “Face Detection and Recognition Using OpenCV,” *Proc. - 2019 Int. Conf. Comput. Commun. Intell. Syst. ICCCIS 2019*, vol. 2019-Janua, pp. 116–119, 2019, doi: 10.1109/ICCCIS48478.2019.8974493.

[18] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, 2001, vol. 1, pp. I–I.

[19] D. Rana, H. S. Sethi, S. P. Jena, and S. K. Pradhan, “Classroom Attendance System with IoT Notification,” no. 17051, pp. 17051–17057, 2020.

[20] P. Lavanya, G. L. Devi, and K. S. Rao, “LBPH-Based Face Recognition System for Attendance Management,” in *Microelectronics, Electromagnetics and Telecommunications*, Springer, 2020, pp. 55–61.

[21] X. Zhao and C. Wei, “A real-time face recognition system based on the improved LBPH algorithm,” in *2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP)*, 2017, pp.

72–76, doi: 10.1109/SIPROCESS.2017.8124508.

[22] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, 2005, vol. 1, pp. 886–893 vol. 1, doi: 10.1109/CVPR.2005.177.

[23] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint face detection and alignment using multitask cascaded convolutional networks,” *IEEE Signal Process. Lett.*, vol. 23, no. 10, pp. 1499–1503, 2016.

[24] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[25] T. Lindner, D. Wyrwał, M. Białek, and P. Nowak, “Face recognition system based on a single-board computer,” in *2020 International Conference Mechatronic Systems and Materials (MSM)*, 2020, pp. 1–6.

[26] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[27] V. Gupta, V. G. Panchal, V. Singh, D. Bansal, and P. Garg, “EmotionNet: ResNeXt Inspired CNN Architecture for Emotion Analysis on Raspberry Pi,” in *2021 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, 2021, pp. 262–267.