Face Detection System for Health Care Units Using Raspberry PI

Pronaya Bhattacharya
Assistant Professor, Department of CSE
Institute of Technology
Nirma University
Ahmedabad, India
pronoya.bhattacharya@nirmauni.ac.in

Anand Handa
Research Scholar
Dr. A.P.J Abdul Kalam Technical University
Lucknow, India
anandhanda1986@gmail.com

Mohd Zuhair
Assistant Professor, Department of CSE
Institute of Technology
Nirma University
Ahmedabad, India
mohd.zuhair@nirmauni.ac.in

2021 2nd International Conference on Intelligent Engineering and Management (ICIEM) | 978-1-6654-1450-0/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICIEM51511.2021.9445367

Abstract—**Computer vision has one of its primary applications as facial detection and recognition system, which can perform two essential functions of identifying and verifying an individual. An image capturing system with embedded computing as one of its significant parts extracts image information and requires no external processing system. To deliver results to other devices for the corresponding inputs, we use an interface device. The designed system is fast and efficient. It coordinates efficiently with the image recognition unit and the recognition algorithm. The system ensures a smooth flow of data stream between the two devices, namely, the Raspberry PI board and the camera. It also handles the data stream flow to occur smoothly between the camera and the Raspberry PI board. During this pandemic era, the application has a wide range of applicability for healthcare systems, helping in a contactless authentication of patients.**

Keywords—Image capturing, Remotely Operated Video Enhanced Receiver, Local Binary Pattern, Histogram of Oriented Gradients, Embedded Systems, Raspberry PI board.

I. INTRODUCTION

Modern healthcare ecosystems are sensor-driven that captures healthcare data from patients. However, before sending the data over wireless channels, identity authorization is a critical factor. It ensures that the devices, systems, and data are protected by malicious attackers. In traditional verification systems, passwords or secure keys were required to form identity authentication, but with advancements in technology, biometric-based authentication schemes are developed by researchers globally. In biometric-based ecosystems, specialized hardware and software units interact to form a secure identity verification system [1]. In a similar direction, face detection for biometric systems is studied heavily, where the detection technology matches a human face as digitized images or video frames against stored databases. The users are authenticated based on extracted facial templates captured, and being unique, they are matched and processed against stored templates [2],[3].

The face recognition systems are considered advantageous over other biometric systems. As face-recognition systems offer no physical contact with the hardware units for authentication, hence, the systems provide ease-of-access of users [4]. The systems are non-invasive as the random morphogenetic process creates the face. Moreover, the probability of having the same facial features for two different persons is highly unlikely, due to the uniqueness of facial features. Thus, a face-recognition system can allow high accuracy of healthcare user identification to access the health records, and verify updates on the healthcare records. The false recognition rate (FRR) is also low and is considered more stable as compared to other biometric identification. Face recognition has widely used in the security domain due to non-evasiveness, stability, uniqueness, and the slightest false recognition rate.

To design an efficient face recognition system, hardware and operating system (OS) units need to interface through computer vision libraries and dependencies [5]. Raspberry PI is considered a suitable choice for such designs due to the ease of interfacing with captured images, and high-swap space size, which allows process control (PC) block to schedule tasks in an efficient and useful manner [6]. PC interacts through different task units of the Raspberry PI board through a task control. For the design of the face recognition system, firstly the file system is expanded and *Open CV* dependencies are installed [7]. The PC then manages and executes the swap size before installing and linking the dependencies during execution. The Raspberry PI uses *NumPy*to interact with system processes through open ports and allows design portability to the recognition systems. However, the portability is limited due to its weight, high power consumption, and weight. One way to get rid of these limitations is to use the embedded systems [8].

In the proposed work, we design a real-time image capturing system based on a remotely operated video enhanced receiver (ROVER) design. ROVER first captures the images in real-time and then stores them into the database. Finally, it is compared with the other user's face. If it matches, then the access will be granted, and if it is not matched, then the key is denied. The ROVER embedded system supports a large variety of applications, including surveillance, motion analysis, and facial identification. Along with ROVER, we have developed an android application to control and interface with the ROVER [9], and the streaming can be easily viewed on multiple ports and display monitors.

The structure of the article is as follows. Section II depicts the system hardware design that consists of the assembly and interfacing of the Raspberry PI board. Section III depicts the proposed methodology that takes input images and computes the feature extraction, and subsequent image classification models. Based on extracted features, an android based application is designed that performs face recognition of patients in a healthcare ecosystem, and finally, section IV concludes the paper.

II. SYSTEM HARDWARE DESIGN

The section discusses the proposed hardware design of the framework. The following parts compose the system: an image capturing camera, Raspberry PI 3b, L298 Motor Driver, two 6 Volt direct current (DC) Motor, Jumper Wires, 4 AA battery, Robotic base, power bank, and USB cables. The hardware properties are now defined as follows.

A. Main Processing Chip

The primary signal processing chip used in our proposed system is a Broadcom 700MHz chip. The system CPU is a 32 bit ARM1176JZF-S RISC processor, which Advances reduced instruction set computer (RISC) Machines Ltd designs. This is also used to connect a camera and display unit because it a powerful peripheral device.

B. Memory Element

Secure digital (SD) card is used for booting and storage purposes because its accessing speed is high compared to the hard drives, and storage is also permanent. The board runs on a linux-based OS kernel. This Raspberry PI module has a Samsung class 10 micro-SD card. It uses the official Raspberry PI Raspbian Package, known as Jessie.

C. Raspberry PI BOARD Model

We now present the board design of the Raspberry PI model. Fig. 1 shows the details of the board. The board consists of the central and main module of the whole embedded system planned for image captivating and design processing. The main parts are as follows of the board are as follows major processing chip, memory, power supply, high definition multimedia interface (HDMI0 Out, ethernet port, universal serial bus (USB) ports, and significant global interfaces.

D. Interfaces Used

The ROVER consists of various interfaces because we have several interfaces on the board. The board consists of 2 USB ports, an HDMI output port for connecting monitors with HDMI input, or HDMI to Digital Visual Interface (DVI) lead for monitors having DVI input. If HDMI is not used, then the radio corporation of America (RCA) composite video lead can connect to the analog display. To update the Raspberry PI board and get a new version of the software, we have used an ethernet port. The Ethernet port is optional but helps in the networking module. The board also contains an audio lead for getting the audio output if we don't have HDMI support.

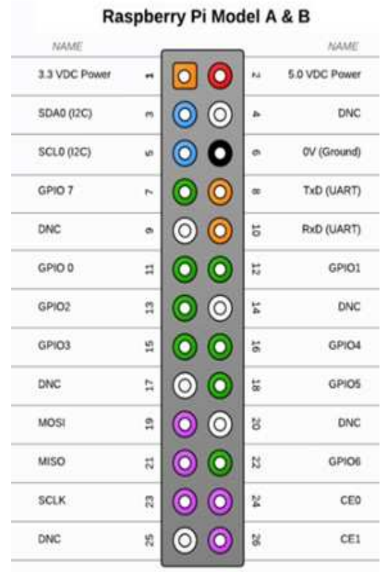

Fig. 1. Pin out Diagram and Raspberry PI Board

E. The Overall Assembly

Firstly, make the robotic base ready and connect the two motors to it. Connect the motors to the motor driver (L298N) and then click the motor driver with the 4 AA batteries. Connect the motor driver pins to the GPIO pins of the Raspberry PI using jumper wires. Lastly, connect the web camp utilizing the USB port in raspberry PI to power the Raspberry PI with the power bank. The webcam [10] [11] will stream the real-time images and video of the ROVER place on surveillance. The transmission will occur over an internet protocol (IP) address and a specific port number. Before the transmission, the configuration of the webcam is required But before that, we need to configure the webcam for Raspbian Jessie.

III. THE PROPOSED METHODOLOGY

In the proposed framework, we operate on the designed system in two different sessions [12]. In the first session, we capture the database, and in the next session, we create a database. Then, we capture the image and make a comparison of various images in the database to identify them. We use the Viola-Jones algorithm of face recognition for finding the accurate comparison or match [13]. Fig. 2 depicts the important components of the methodology.

A. Image Acquisition via Raspberry PI 3 ROVER

In this system, we have developed an android application that controls the ROVER motion. The installed camera collects the images, streams the video, and sends them to the central unit for pre-processing using MATLAB. Fig. 3 shows the detailed process. The interface developed for the application requires the IP address and the port number as input for the ROVER's working. The application consists of four directions motion control, i.e., front, back, left, and right. One can stop the ROVER by using the control *STOP*.

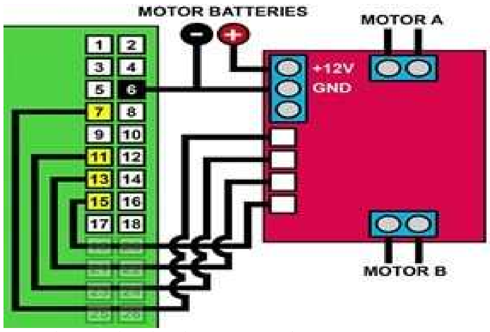

Fig. 2. Circuit Diagram Raspberry PI 3 ROVER

B. Face Detection Module

The face module detection of the project collects the images and creates a dataset [14] using the camera installed on the ROVER. Fig. 4 shows the detailed control steps of the face detection process.

1. *Image Acquisition:* The images are captured using a camera installed in the ROVER. The images captured are then transmitted using the Wi-Fi module to the processing unit [12].

2. *Face Detection:* We use the Viola-Jones algorithm [14] to detect the objects. Viola-Jones algorithm is more appropriate for frontal faces, rather than side faces. The image is first converted to a grey scale feature, to minimize the image data. The algorithm is based on the principle of location setup on the greyscale features, and the face detection is based on abounded box. The role of the box is to form outline detection of facial features, and the face is bounded through the box. Now, based on Haar features, the compounded data of various boxes are combined as single-box units. Then, the proposed algorithm is fed with sample face and non-face units, and we use the AdaBoost technique for feature selection, and then a classification-based training is conducted. The detailed explanation is now presented as follows.

   - *Haar-Feature selection:* In this, we have used five types of Haar features for face detection. Each feature results in a single value. The value is calculated by subtracting the sum of pixels in the white region from the pixels' sum in the dark area. The features are shown in Fig. 5.

   - *Creation of integral Image:* Next, we create an integral image, to reduce the overall computation time. For the same, we use four array references and then sum the pixel values within *D*. In rectangle *A*, the total of all the values at location one is equal to the sum of pixel values.

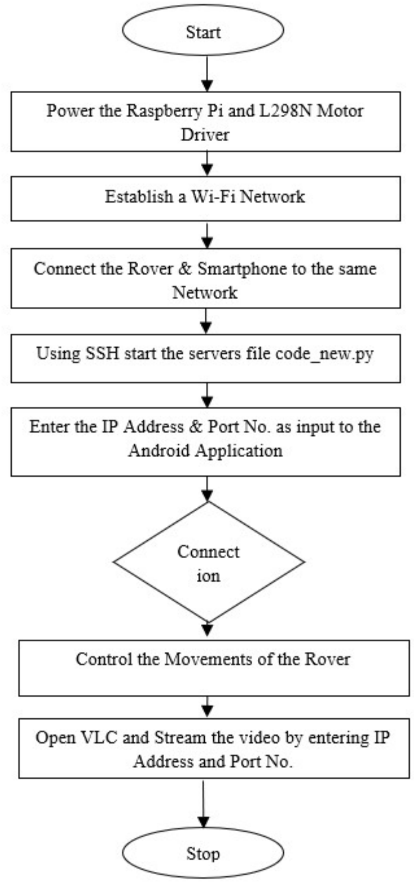

Fig. 3. Flow Diagram for Raspberry PI 3 ROVER Motion and Image Acquisition

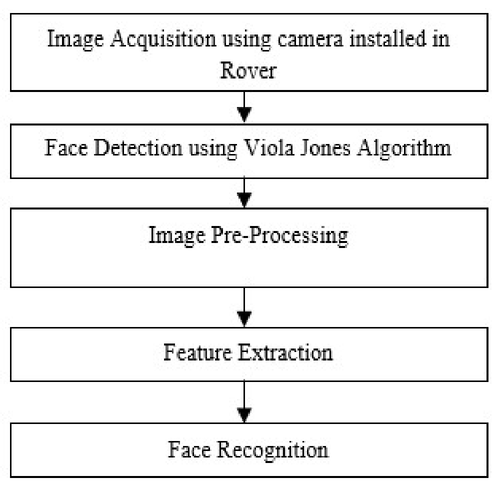

Fig. 4. Flow Diagram for Face Detection and Recognition

The value at location 2 is the sum of *A* and *B* values, for location 3 is equal to the sum of *A* and *C*, and at location 4 it is equal to the sum of *A*, *B*, *C*, and *D*.

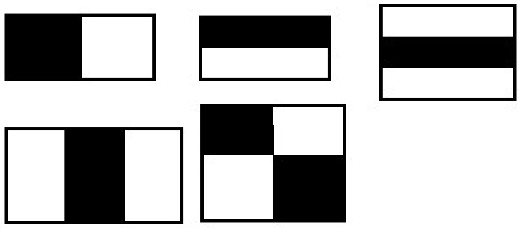

Fig. 5. Haar Features

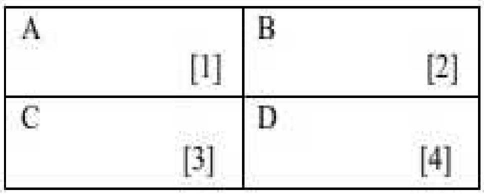

Fig. 6. A reference image

3. *Adaboost Training:* Adaboost is also known as the adaptive boost algorithm. It is particularly used for training a boosted classifier. It constructs a robust classifier from weak classifiers [15]. The function for the Adaboost classifier is presented as follows. In the method, we consider that weights $W = \{W_1, W_2, ..., W_n\}$ are assigned to n image samples $\{I_1, I_2, ..., I_n\}$. The initial weight is then computed as follows.

$$W_{I_j} = \frac{1}{n} (1)$$

Where I_j denotes the j^{th} training instance out of n training instances. A decision stump value D is selected based on weighted samples, and a binary decision $B\{-1,+1\}$] is formed based on I. The value depends on binary classification C_1, or C_2 of the images. The error $E(I)$, or mis-classification is computed as follows.

$$E(I) = \frac{C(I) - n}{n} (2)$$

Where $C(I)$ denotes the instances that are correctly classified with C_1 or C_2. Based on this, the modified error E_I0 for any j^{th} sample is computed as follows.

$$E_{I'} = \sum_{j \in I} W(j) \times E(I) / \sum_{W} (3)$$

Where W denotes the overall weighted sum, and the prediction error $E(I)$ denotes the error in j^{th} image during training. The modified weighted value D^0 is now computed as follows.

$$D^{`} = ln(1 - E(I^{`}))/E(I) \ (4)$$

Based on $D^{`}$, the Ada Boost ensembles weak models sequentially on the trained data, and the process iterates until no further improvement in learning rates are observed. The strong classifier is denoted as follows.

$$(x) = a_1f_1(x) + a_2f_2(x) + a_3f_3(x) + a_4f_4(x) + ..(5)$$

Where, $F(x)$ is termed as a strong classifier and a_i are the weighted values for any image x.

4. *Cascade Classifier:* It is the last stage of this algorithm. In this stage the algorithm identifies whether a sub window is to be classified as a face or a non-face. The initial stage classifier eliminates many negative examples and the subsequent layers eliminate additional negatives by using additional computation. The details of the cascade classifier is presented in Fig. 7:

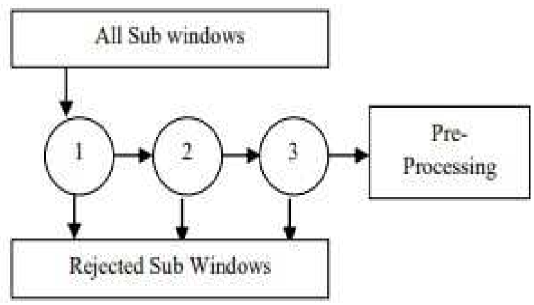

Fig. 7. Steps for Cascade Classifier

C. Pre-Processing

After pre-processing the images, the features would be extracted from the cropped detected faces. The contrast improvement and conversion of the grey-scale image into the black and white image and removing small objects from the binary image will be part of the pre-processing step.

D. Features Extraction

The features will be extracted from the image after it has been processed. Six features will be extracted in total, three each for the nose, mouth, and eyes, using two algorithms, local binary pattern (LBP) and histogram of oriented gradients (HOG) classifier would be applied as follows.

1. *Local Binary Pattern:* It's a type of feature descriptor that is used in face recognition. Each pixel value is converted into binary digits, which are either 0 or 1. The radial filter is used to select neighboring pixels $(P_0, P_1, P_2, ..., P_k)$, for any given central pixel value P_c in an image. The response at P_c is calculated as follows [16]

$$LBP = \sum_{n=0}^{n=i-1} s(P_n - P_c))2^n \quad (6)$$

Where $s(x)$ is 1 for $x \geq 0$ and otherwise 0.

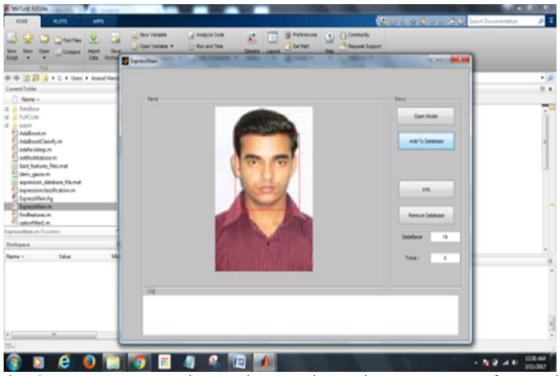

Fig. 8. Feature Extraction and Detection using MATLAB for Real Time Images

2. *Histogram of Orientation Gradients:* It is another feature descriptor used for face recognition. Firstly, it calculates the gradient values of a grey image which is obtained by calculating the *x* and *y* directional derivatives using convolution functions. The calculations are as follows [17]

$$I_x = I \cdot D_x \quad (7)$$
$$I_y = I \cdot D_y \quad (8)$$

Where D_x and D_y are kernels whose values are $[-1,0,1]$ and $[1,0,-1]^T$ respectively. The magnitude and the argument can be calculated as follows.

$$|I| = \sqrt{(I_x^2 + I_y^2)}$$
$$\theta = \tan^{-1}\left(\frac{I_y}{I_x}\right) \quad (9)$$

After this computation normalization is done.

Table 1. Experimental Results

Dataset	No. of Images	No. of Faces Identified	% Detection
JAFFE [15]	213	209	98.12 %
Faces94[16]	3044	3013	98.98 %
Subject	200	178	89.00 %

E. Face Recognition

The features extracted from the image will be compared with the stored features in the database with the help of a support vector machine (SVM) [18]. The basic functionality of SVM is to categorize every data object as point (x,y) is a n dimensional space (where n is the number of features we've identified), with each function's value being the value of a specific coordinate. Following that, we perform classification by determining the hyperplane that separates the two classes. Person observation coordinates are commonly used as support vectors. The ROVER identifies the human if the device recognizes the features. Table I displays the findings for different datasets, including the subject dataset provided by real-time image acquisition, and Fig. 8 shows attribute extraction and identification for real-time images.

IV. Conclusion

In this paper, we propose a framework of a face-recognition system for healthcare users that capture person images through the Raspberry PI 3 ROVER. The system can be easily operated from remote locations and can be controlled via the android application. It captures the live video and images and sends them back to the admin where the face recognition module is implemented using MATLAB.

References

[1] A. Handa, R. Agarwal, and N. Kohli, “Incremental approach for multimodal face expression recognition system using deep neural networks,” *International Journal of Computational Vision and Robotics*, vol. 11, no. 1, pp. 1–20, 2021.

[2] S. B. Patel, H. A. Kheruwala, M. Alazab, N. Patel, R. Damani, P. Bhattacharya, S. Tanwar, and N. Kumar, “Biouav: Blockchainenvisioned framework for digital identification to secure access in next-generation uavs,” in *Proceedings of the 2nd ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond*, ser. DroneCom ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 43–48. [Online]. Available:https://doi.org/10.1145/3414045.3415945

[3] A. Ladha, P. Bhattacharya, N. Chaubey, and U. Bodkhe, “Iigpts: Iotbased framework for intelligent green public transportation system,” in *Proceedings of First International Conference on Computing, Communications, and Cyber-Security (IC4S 2019)*, P. K. Singh, W. Pawłowski, S. Tanwar, N. Kumar, J. J. P. C. Rodrigues, and M. S. Obaidat, Eds. Singapore: Springer Singapore, 2020, pp. 183–195.

[4] A. Handa, R. Agarwal, and N. Kohli, “A survey of face recognition techniques and comparative study of various bi-modal and multi-modal techniques,” in *2016 11th International Conference on Industrial and Information Systems (ICIIS), Roorkee, India*, 2016, pp. 274–279.

[5] A. Handa, R. Agarwal, and N. Kohli, “A multimodel keyword spotting system based on lip movement and speech features,” *Multimedia Tools and Applications*, vol. 79, no. 27, pp. 20461–20481, 2020.

[6] M. Sajjad, M. Nasir, K. Muhammad, S. Khan, Z. Jan, A. K. Sangaiah, M. Elhoseny, and S. W. Baik, “Raspberry pi assisted face recognition framework for enhanced law-enforcement services in smart cities,” *Future Generation Computer Systems*, vol. 108, pp. 995–1007, 2020.

[7] M. Rashid, H. Singh, and V. GoyalI, “Analyzing functional magnetic resonance brain images with opencv2,” in *2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), Waknaghat, India*, 2020, pp. 451–454.

[8] M. T. Ravali and R. S. Komaragiri, “Image processing platform on raspberry pi for face recognition,” *Glob. J. Adv. Eng. Technol. ISSN*, vol. 3, pp. 2277–6370, 2014.

[9] S. Prasad, P. Mahalakshmi, A. J. C. Sunder, and R. Swathi, “Smart surveillance monitoring system using raspberry pi and pir sensor,” *Int. J. Comput. Sci. Inf. Technol*, vol. 5, no. 6, pp. 7107–7109, 2014.

[10] G. Senthilkumar, K. Gopalakrishnan, and V. S. Kumar, “Embedded image capturing system using raspberry pi system,” *International Journal of Emerging Trends & Technology in Computer Science*, vol. 3, no. 2, pp. 213–215, 2014.

[11] “Raspberry.” [Online]. Available: http://www.raspberrypi.org

[12] O. Deniz, G. Bueno, J. Salido, and F. De la Torre, "Face recognition´ using histograms of oriented gradients," *Pattern recognition letters*, vol. 32, no. 12, pp. 1598–1603, 2011.

[13] W. LU and M. YANG, "Face detection based on viola-jones algorithm applying composite features," in *2019 International Conference on Robots Intelligent System (ICRIS), Haikou, China*, 2019, pp. 82–85.

[14] Y.-Q. Wang, "An analysis of the viola-jones face detection algorithm," *Image Processing On Line*, vol. 4, pp. 128–148, 2014.

[15] A. Gupta, S. Sharma, S. Goyal, and M. Rashid, "Novel xgboost tuned machine learning model for software bug prediction," in *2020 International Conference on Intelligent Engineering and Management (ICIEM), London, UK*, 2020, pp. 376–380.

[16] C. Qin, Y. Hu, H. Yao, X. Duan, and L. Gao, "Perceptual image hashing based on weber local binary pattern and color angle representation," *IEEE Access*, vol. 7, pp. 45460–45471, 2019.

[17] E. Alionte and C. Lazar, "A practical implementation of face detection by using matlab cascade object detector," in *2015 19th international conference on system theory, control and computing (ICSTCC), CheileGradistei, Romania*. IEEE, 2015, pp. 785–790.

[18] A. N. Patil, R. B. Ranavare, D. V. Ballal, and A. Kotekar, "Raspberry pi based face recognition system for door unlocking," *International journal of innovative research in science and engineering vol*, no. 2.