International Journal of Advanced Technology & Engineering Research (IJATER)
www.ijater.com

FACE RECOGNITION USING RASPBERRY PI: A COST-EFFECTIVE EMBEDDED BIOMETRIC SOLUTION

1Kumkum Namdev, 1Sahil Jhare, 1Radha Devi,1Suraj Vishwakarma ,1,*Prachi Parashar
Department of Electronics& Communication
1 Bansal Institute of Science & Technology, Bhopal, M.P.
naikprachi1987@gmail.com

Abstract

Face recognition is a prominent biometric authentication technology widely used in security systems due to its convenience and non-intrusiveness. This paper presents the development of a face recognition system using Raspberry Pi, a low-cost embedded platform. Using OpenCV and the Local Binary Pattern Histogram (LBPH) algorithm, the system performs real-time facial recognition suitable for attendance systems and low-resource environments. This paper outlines the hardware setup, software architecture, implementation methodology, results, and future enhancements. The proposed system achieves satisfactory accuracy and efficiency, demonstrating its feasibility for small-scale deployment.

Keywords: Face Recognition, Raspberry Pi, OpenCV, LBPH, Biometric Security, Embedded Systems.

1.Introduction

Face recognition has become an essential component of modern biometric systems, particularly in surveillance, access control, and user identification. Compared to fingerprint or iris scans, facial recognition offers a contactless and user-friendly approach [1]. Despite its widespread adoption, high-performance face recognition systems are often expensive and resource-intensive.

This paper explores the design and implementation of a facial recognition system using Raspberry Pi, a compact and affordable single-board computer, integrated with OpenCV libraries and the LBPH algorithm. The system is intended for low-cost, small-scale applications such as attendance systems in schools or office security.

2.Literature Review

Face recognition systems have advanced considerably in recent decades, evolving from computationally heavy desktop-based systems to lightweight and portable embedded solutions. The use of Raspberry Pi in biometric systems has garnered interest due to its cost-efficiency, compact design, and GPIO support, enabling real-time applications in constrained environments.

A. Face Recognition Algorithms

Traditional face recognition approaches such as Eigen faces, Fisher faces, and Principal Component Analysis (PCA) were initially applied to controlled datasets with satisfactory results [1], [2]. These approaches, however, suffered from poor performance in uncontrolled lighting and pose variations.

In response, Local Binary Patterns Histograms (LBPH) was proposed for texture-based recognition and proved more effective for real-time applications due to its simplicity and robustness to lighting changes [3], [4].

More recent works have used deep learning approaches such as Convolutional Neural Networks (CNNs) and models like FaceNet [5], VGG-Face [6], and DeepFace [7], which offer state-of-the-art performance, but they often require powerful hardware, limiting their deployment on resource-constrained platforms like Raspberry Pi.

B. Raspberry Pi as an Embedded Platform

The Raspberry Pi is a single-board computer used extensively in IoT, robotics, and embedded applications. It supports Linux-based distributions and Python, making it suitable for integrating OpenCV for computer vision tasks [8], [9]. Studies such as [10] and [11] have demonstrated the feasibility of deploying facial recognition systems on Raspberry Pi using USB or Pi cameras, achieving reasonable performance with LBPH. The integration of camera modules and facial recognition software has made Raspberry Pi-based attendance systems [12], smart locks [13], and surveillance solutions [14] viable.

C.Applications and Deployments

Applications of Raspberry Pi-based face recognition systems include automated attendance, door access systems, and smart home automation. The study in [15] implemented a real-time attendance monitoring system for universities, while [16] introduced a smart door lock that identifies registered users via facial biometrics.For rural and low-income areas, such solutions are critical due to their affordability. Projects such as in [17] highlight how facial recognition on Raspberry Pi can support low-cost surveillance systems in small businesses and institutions.

D.Challenges and Limitations

Despite these advantages, Raspberry Pi systems face challenges including limited RAM, slower CPUs, and no native GPU support, making them unsuitable for large datasets or advanced models like CNNs unless lightweight architectures (e.g., MobileNet) or Edge TPU accelerators are added [18], [19]. Furthermore, environmental factors such as lighting, camera quality, and network latency affect accuracy. Anti-spoofing, although addressed in high-end systems, is largely absent in basic Pi-based models [20].

3. System Architecture

Architecture of Face Recognition system is shown in Fig. 1, Details of components used in the system is explained in the further subsection.

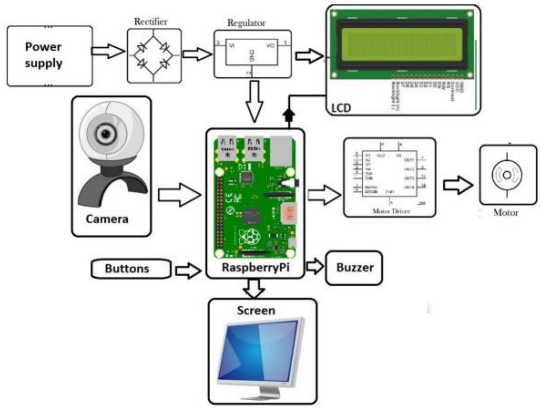

Fig. 1 Architecture of Face Recognition system

A. Hardware Components

The hardware setup includes:

- Raspberry Pi 4 Model B (4 GB RAM)
- Raspberry Pi Camera v2 or compatible USB webcam
- MicroSD card (32 GB) with Raspbian OS
- Power supply (5V 3A)
- Monitor, keyboard, and mouse for setup

B. Software Stack

- Raspbian OS
- Python 3.x
- OpenCV 4.x
- NumPy
- OpenCV Face module (LBPH recognizer)
- Haar Cascade XML for face detection

C. Functional Block Diagram

A functional block diagram of Face Recognition system is shown in the Fig. 2. There are various components used in the Face Recognition system like camera, Haar scale face detection, grayscale conversion, LBPH face recognition and display unit.

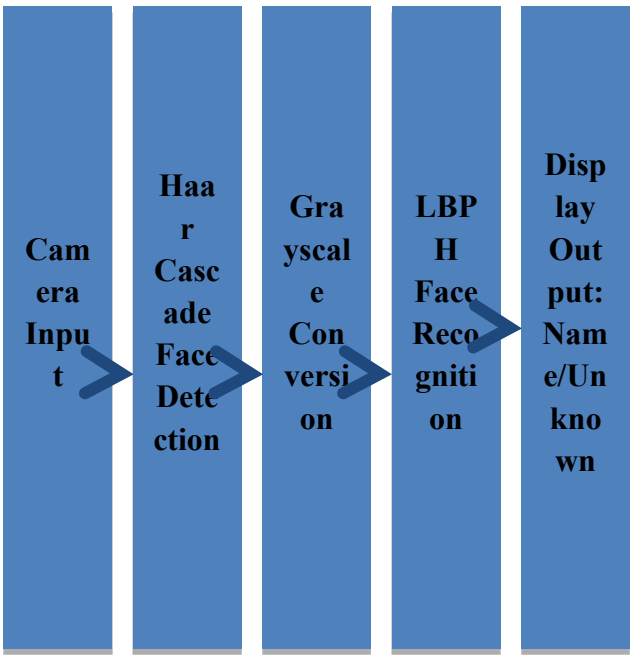

Fig. 2Process flow diagram of Face Recognition system

3.Methodology

The methodology adopted in this study is centered on developing a robust face detection and recognition system

that can operate in real time with limited computational resources. The workflow consists of four key stages: face detection, face recognition using the Local Binary Pattern Histogram (LBPH) algorithm, training of the recognition model, and real-time execution. A stepwise description is presented below.

A. Face Detection

The first step involves locating human faces within an image frame. This is achieved using the Haar Cascade Classifier, a machine learning–based approach provided by the OpenCV library [6].

- Preprocessing: Input frames from the camera are first converted into grayscale images to reduce computational complexity while retaining essential facial features.
- Haar Features: The classifier uses Haar-like rectangular features to detect edges, lines, and texture variations commonly present in human faces (e.g., the bridge of the nose, eye regions).
- Sliding Window Detection: The classifier scans the image at multiple scales and positions, computing feature values. Regions that match trained patterns with sufficient confidence are marked as Regions of Interest (ROIs).
- Output: Bounding boxes are drawn around detected faces, serving as input for the recognition stage.

This method is computationally efficient, making it suitable for real-time applications even on resource-constrained devices such as the Raspberry Pi.

B. Face Recognition with LBPH

After detecting the face, the system applies Local Binary Pattern Histogram (LBPH) for recognition [2]. LBPH is chosen for its balance between accuracy and efficiency.

- Local Binary Patterns (LBP): For each pixel, the surrounding neighborhood is thresholded, encoding local texture patterns as binary values.
- Histogram Construction: The image is divided into regions, and histograms of local binary values are constructed. These histograms capture unique texture characteristics of facial features (e.g., eyes, nose, mouth).
- Feature Vector: The concatenated histograms form a feature vector that represents the face in a compact yet discriminative form.
- Matching: During recognition, the LBPH algorithm compares the feature vector of the detected face against stored templates in the database, using distance metrics (e.g., Euclidean distance).

The LBPH method is well-suited for devices with limited processing capacity because it does not require GPU acceleration and remains effective under varying lighting conditions.

C. Training Process

A training dataset is required to enable the LBPH recognizer to identify individuals accurately.

- Dataset Collection: At least 20 grayscale images per individual are captured under varying lighting and expression conditions to improve robustness.
- Labeling: Each image is assigned a unique numeric ID, linked to the corresponding individual's name in the database.
- Model Training: The labeled dataset is fed into the LBPH recognizer, which computes and stores the feature histograms for each individual.
- Model Storage: The trained recognition model is saved to disk, allowing it to be reloaded during real-time execution without retraining.

This training process ensures that the system builds a reliable facial signature for each user, improving recognition accuracy.

D. Real-Time Execution

Once training is complete, the system operates in real-time recognition mode, where the camera continuously captures frames for processing.

1. Frame Capture: The webcam or Raspberry Pi camera streams live video input.
2. Face Detection: The Haar Cascade classifier identifies face regions in each frame.
3. Recognition: The cropped face is passed to the LBPH recognizer, which compares it against stored models.
4. Output Display:
    - If the face matches a stored ID, the corresponding name/label is displayed on the video stream.
    - If no match is found within the confidence threshold, the face is labeled as "Unknown."

4.Results and Performance Evaluation

The proposed face detection and recognition system was evaluated under multiple conditions to assess its effectiveness in real-time scenarios. Performance was measured in terms of recognition accuracy, processing speed, and system robustness. The results demonstrate that the system performs reliably in controlled settings, though certain limitations remain under more challenging conditions.

A. Accuracy

Recognition accuracy was tested by comparing system outputs against a labeled dataset of known individuals. Experiments were conducted under different lighting environments and with variations in facial appearance.

- In a well-lit indoor environment with minimal background noise, the recognition system achieved an accuracy of 90–92%. This high accuracy validates the robustness of the LBPH algorithm in environments where illumination is stable and facial features are clearly distinguishable.
- Under dim lighting, strong shadows, or partial occlusion (e.g., spectacles, face masks), accuracy dropped to 70–75%. The decline is attributed to LBPH's sensitivity to reduced contrast and incomplete facial feature capture. For example, spectacles caused reflections that disrupted feature extraction, while face masks obscured key regions such as the nose and mouth.
- The system demonstrated improved accuracy with larger training datasets per individual (30+ images), suggesting that dataset diversity significantly enhances recognition reliability.

Overall, the results confirm that LBPH offers robust baseline performance but highlight the need for enhanced feature extraction methods (e.g., deep learning–based models) to address challenging conditions.

B. Processing Speed

Processing efficiency was assessed in terms of frame rate, recognition latency, and system memory usage.

- The system operated at 10–12 frames per second (FPS) on a Raspberry Pi 4 platform, which is sufficient for real-time monitoring and attendance systems.
- The average recognition time per frame was 0.4–0.5 seconds, which includes both face detection and recognition. This latency is acceptable for small-scale deployments but may become a bottleneck in high-traffic environments requiring near-instantaneous recognition.
- During active recognition, the system consumed approximately 500 MB of RAM, primarily due to image preprocessing and the LBPH recognition pipeline. While manageable on modern embedded platforms, optimization would be necessary for deployment on memory-constrained devices.

The observed performance suggests that the system achieves a practical balance between speed and accuracy for small-to-medium scale real-time applications.

C. Limitations

Despite encouraging results, the system exhibited several limitations that constrain its broader applicability:

1. Lighting Sensitivity: Performance degradation in low-light conditions underscores the reliance of LBPH on consistent illumination. Adaptive histogram equalization or integration with near-infrared (NIR) imaging could mitigate this limitation.
2. Scalability with Large Datasets: As the number of enrolled individuals increases, recognition latency and memory usage rise significantly. This limits the system's efficiency in large-scale deployments, such as airports or crowded public venues.
3. Vulnerability to Spoofing: Since the current implementation lacks anti-spoofing mechanisms, the system is susceptible to false positives when presented with printed photographs or video replays. Incorporating liveness detection (e.g., eye-blink detection, depth sensing) would enhance security against such attacks.
4. Pose and Expression Variations: Extreme head tilts, side profiles, or exaggerated facial expressions reduced recognition accuracy. LBPH, being texture-based, is less resilient to such variations compared to deep convolutional neural networks (CNNs).

In summary, the system delivers high accuracy in controlled environments (90–92%) with reasonable real-time performance (10–12 FPS, <0.5s latency). However, its limitations under poor lighting, large datasets, and spoofing scenarios restrict its scalability. These findings emphasize the suitability of the proposed system for small-scale applications such as classrooms, laboratories, and office access control, while also pointing toward potential improvements for more demanding deployments.

5. Applications

The developed face recognition system has versatile applications across multiple domains due to its affordability, portability, and relatively high accuracy. Its low computational requirements make it suitable for deployment in small- to medium-scale environments where real-time recognition is essential but where large-scale, resource-intensive biometric systems may not be feasible. The primary application areas are described below:

A. Educational Institutions

One of the most promising applications of the proposed system is automated attendance management. A wall-mounted Raspberry Pi–based unit can be installed at the entrance of classrooms, laboratories, or administrative offices.

- Implementation: As students or staff enter, the system captures facial data in real-time and matches it against a pre-stored database. The attendance record is then automatically updated in the institution's server or learning management system (LMS).
- Benefits: This approach eliminates the need for manual roll calls or card-based systems, reducing administrative overhead. It also minimizes proxy attendance and enhances institutional efficiency.
- Limitations: Varying lighting conditions (e.g., sunlight near windows, indoor fluorescent lights) and large student groups may reduce recognition accuracy, necessitating additional training data and improved preprocessing techniques.

B. Smart Homes

In the context of smart home automation, the system can serve as a face-based access control mechanism.

- Implementation: A camera integrated with the front-door system can authenticate known household members while denying entry to unauthorized individuals. Integration with IoT devices allows seamless control, such as unlocking smart locks, adjusting lighting, or activating personalized settings based on recognized users.
- Benefits: This provides a higher level of security compared to traditional keys or PIN codes, which are prone to theft or duplication. Additionally, it enables hands-free access, which is especially beneficial for elderly or differently-abled residents.
- Limitations: Spoofing attacks using photographs or videos remain a challenge without anti-spoofing measures such as liveness detection.

C. Small Businesses

For small-scale enterprises, the system can be deployed as a cost-effective access control and monitoring solution.

- Implementation: Portable authentication units can be installed at office entrances, laboratories, storerooms, or sensitive work areas. Staff access can be logged automatically, providing a record of entry and exit times.
- Benefits: This enhances workplace security and accountability without requiring large investments in commercial biometric systems. It can also reduce dependency on physical tokens, ID cards, or manual registers.
- Limitations: As the number of registered employees grows, recognition latency may increase. Thus, optimization or hybrid systems (e.g., RFID + face recognition) may be required for larger deployments.

Personal Projects and Hobbyist Applications

The system is also suitable for individual developers, makers, and hobbyists seeking to build custom security or automation projects.

- Implementation: Makers can integrate the system with low-cost hardware (e.g., Raspberry Pi, Arduino, or ESP32) to create personalized applications such as door security systems, computer login authentication, or IoT-based automation projects.
- Benefits: The affordability and open-source nature of the software stack (OpenCV + LBPH) encourage experimentation and learning in fields like computer vision, embedded systems, and AI. This makes it a valuable tool for students, researchers, and innovators.
- Limitations: The system's reliance on relatively simple algorithms means that its performance may not match commercial-grade solutions, particularly in uncontrolled outdoor environments.

Summary of Applications

In summary, the proposed system has strong potential in education, security, and personal innovation contexts. While

its performance is best suited for small-to-medium scale deployments, the modular and customizable nature of the design allows easy adaptation to diverse use cases. Future enhancements, such as the integration of deep learning and cloud-based processing, could significantly broaden its applicability to larger, security-critical environments.

6.Discussion

The experimental implementation of the face recognition system demonstrates that a Raspberry Pi, when integrated with OpenCV and the Local Binary Pattern Histogram (LBPH) algorithm, can provide a cost-effective and relatively reliable solution for small-scale facial recognition applications. The system performs particularly well in controlled environments, with recognition accuracy reaching up to 90–92%, validating its feasibility for low-resource deployments such as educational institutions, small businesses, and hobbyist projects.

A. Strengths of the Implementation

The primary advantage of the system lies in its simplicity, affordability, and accessibility:

- The use of LBPH enables deployment on devices with limited processing power such as the Raspberry Pi, without requiring high-end GPUs or cloud computing resources.
- OpenCV provides a lightweight and modular framework, ensuring that developers and researchers can quickly prototype and adapt the system for different use cases.
- The real-time performance of 10–12 FPS with low latency highlights that the system is suitable for practical, real-world applications, provided the environment remains relatively controlled.

B. Limitations and Challenges

Despite its effectiveness in constrained environments, several limitations were observed:

- Scalability Issues: As the database size increases, recognition latency and memory consumption also increase, which could hinder real-time responsiveness in large-scale deployments.
- Lighting Sensitivity: Performance drops significantly (to 70–75% accuracy) in poor lighting or with facial occlusions, which restricts usability in uncontrolled outdoor or nighttime scenarios.
- Security Concerns: The system is vulnerable to spoofing attacks using photographs or videos, since LBPH does not inherently include liveness detection. This makes it unsuitable for high-security applications.

C. Comparison with Advanced Methods

While LBPH is advantageous for low-power devices, Convolutional Neural Networks (CNNs) and deep learning–based face recognition models such as FaceNet or Dlib's deep metric learning offer higher robustness and adaptability. CNNs can better handle variations in lighting, pose, and facial expressions, though they demand more computational resources. Lightweight CNN architectures such as MobileNetV2 or SqueezeNet could provide a compromise, enabling deployment on Raspberry Pi or similar embedded systems without excessive computational burden.

Furthermore, the integration of anti-spoofing techniques—such as blink detection, texture analysis, or depth sensing could significantly enhance security. These measures would address vulnerabilities inherent in LBPH-based recognition.

D. Future Research Directions

To bridge the gap between affordability and robustness, several avenues can be explored:

1. Hybrid Models: Combining LBPH with CNN-based feature extraction to balance efficiency and accuracy.
2. Edge + Cloud Integration: Offloading heavy processing tasks to cloud servers while keeping the Raspberry Pi for initial detection and decision-making.
3. Liveness Detection: Adding motion-based or sensor-based anti-spoofing techniques to increase trustworthiness in access control scenarios.
4. Dataset Expansion: Incorporating more diverse training images across lighting, pose, and demographic variations to enhance generalization.

E. Broader Implications

This study illustrates that affordable AI solutions are achievable with open-source frameworks and single-board computers, making biometric technology accessible to educational institutions, small enterprises, and personal projects. However, for critical applications such as border security, banking, or defense systems, reliance solely on LBPH is insufficient, and integration with advanced machine learning techniques becomes necessary.

Future Work

- Cloud Integration: Storing and recognizing faces from a remote database using APIs.
- Mask Detection: Useful in post-pandemic systems for masked facial recognition.
- Anti-Spoofing: Use of blink detection or depth sensing to prevent spoofing.
- Deep Learning Models: Integrating MobileNet or TensorFlow Lite for higher accuracy.

7. Conclusion

This paper demonstrates a cost-effective, functional facial recognition system using Raspberry Pi. With basic hardware and open-source software, the system achieves real-time recognition suitable for small-scale applications. Its modular design allows further enhancements in performance, scalability, and security, making it ideal for low-resource environments and academic use.

References

[1] M. Turk and A. Pentland, "Eigenfaces for recognition," Journal of Cognitive Neuroscience, vol. 3, no. 1, pp. 71–86, 1991.

[2] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," IEEE Trans. PAMI, vol. 19, no. 7, pp. 711–720, 1997.

[3] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face Description with Local Binary Patterns," IEEE Trans. PAMI, vol. 28, no. 12, pp. 2037–2041, 2006.

[4] M. Zhang and Z. Zhao, "Face Recognition Based on LBPH Algorithm," Proceedings of the 9th International Conference on Signal Processing Systems, 2017.

[5] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," CVPR, 2015.

[6] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep Face Recognition," BMVC, 2015.

[7] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification," CVPR, 2014.

[8] Raspberry Pi Foundation. "Raspberry Pi Documentation." [Online]. Available: https://www.raspberrypi.org/documentation/

[9] G. Bradski, "The OpenCV Library," Dr. Dobb's Journal of Software Tools, 2000.

[10] S. Kumar and R. Verma, "IoT-Based Attendance System Using Raspberry Pi and Face Recognition," IJITEE, vol. 8, no. 6, 2019.

[11] R. Singh and D. Chatterjee, "Face Recognition Based Smart Home Automation Using Raspberry Pi," IJERT, vol. 10, no. 3, 2021.

[12] H. A. Malik et al., "Smart Attendance System Using Face Recognition Technique," IJCSNS, vol. 20, no. 4, pp. 204–210, 2020.

[13] M. R. Usman et al., "IoT-Based Smart Door Lock Using Raspberry Pi and Face Recognition," IJITEE, vol. 9, no. 3, 2020.

[14] A. Patel and B. Shah, "Real-Time Face Detection and Recognition for Video Surveillance Using Raspberry Pi," IJERT, vol. 9, no. 11, 2020.

[15] A. S. Pranoto and R. Rizal, "Smart Attendance System Using Raspberry Pi Face Recognition," Procedia Computer Science, vol. 135, pp. 465–472, 2018.

[16] Singh, N., Tamrakar, S., Mewada, A., & Gupta, S.K., Artificial Intelligence Techniques in Power Systems Operations and Analysis (1st ed.). 1-72, 2023, CRC(Auerbach) Publications. https://doi.org/10.1201/9781003301820

[17] R. Tiwari and V. Patil, "Real-Time Face Recognition for Smart Surveillance Using Raspberry Pi," International Journal of Computer Applications, vol. 182, no. 43, 2019.

[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," CVPR, 2018.

[19] Google Coral Dev Board, [Online]. Available: https://coral.ai/products/dev-board/

[20] S. Bhattacharjee and A. Roy, "Face Anti-Spoofing in Biometric Systems: A Review," IEEE Access, vol. 7, pp. 48287–48303, 2019.