

Smart Attendance System Using Face Recognition

Tarang Patel*, Krina Masharu**

*B Tech in Computer Science Engineering, Atmiya University, Rajkot, India

Email: tarangpatel133@gmail.com

** Faculty of Engineering and Technology (CE), Atmiya University, Rajkot, India

Email: krina.masharu@atmiyauni.ac.in

Abstract:

Automated attendance management is an essential administrative task in educational institutions and organizations, where accuracy, efficiency, and reliability are critical. Traditional methods, such as manual registers and RFID cards, often suffer from errors, proxy attendance, and time delays. The proposed Face Recognition Attendance System provides a contactless and automated solution using classical computer vision and machine learning techniques. Implemented in Python, the system uses OpenCV's Haar Cascade classifier for real-time face detection and the Local Binary Patterns Histograms (LBPH) algorithm for face recognition, which learns and matches facial patterns for identification. Tkinter offers an intuitive graphical interface for desktop interaction, while Flask manages backend functionalities, including attendance logging and report generation. The system operates in two modes—Registration, for capturing and storing face data, and Attendance, for real-time verification. Evaluation demonstrates robust performance under varying lighting conditions and facial orientations, highlighting its applicability as a secure, efficient, and scalable solution for modern attendance management.

Keywords: Face Recognition, Attendance System, Python, OpenCV, LBPH, Tkinter, Flask, Real-Time Detection, Automation, Proxy Prevention

I. INTRODUCTION

Automated attendance management is an essential administrative task in educational institutions and organizations, where accuracy, efficiency, and reliability are critical. Traditional methods, such as manual registers and RFID cards, often suffer from errors, proxy attendance, and time delays. The proposed Face Recognition Attendance System provides a contactless and automated solution using classical computer vision and machine learning techniques. Implemented in Python, the system uses OpenCV's Haar Cascade classifier for real-time face detection and the Local Binary Patterns Histograms (LBPH) algorithm for face recognition, which learns and matches facial patterns for identification. Tkinter offers an intuitive graphical interface for desktop interaction, while Flask manages backend functionalities, including

attendance logging and report generation. The system operates in two modes—Registration, for capturing and storing face data, and Attendance, for real-time verification. Evaluation demonstrates robust performance under varying lighting conditions and facial orientations, highlighting its applicability as a secure, efficient, and scalable solution for modern attendance management.

II. LITERATURE SURVEY

Recent developments in computer vision and classical machine learning methods have enabled the creation of automated attendance systems that rely on facial recognition. These systems aim to provide reliable, real-time, and contactless solutions for classrooms, offices, and other institutions, addressing the limitations of manual or card-based attendance systems. Researchers have implemented

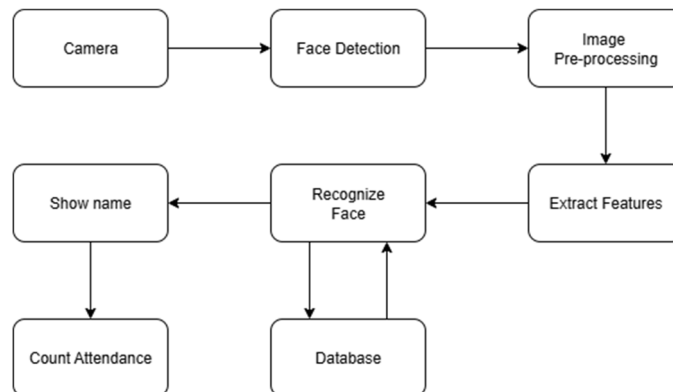
various approaches using Python and OpenCV to detect and recognize faces through webcam feeds, employing methods such as Haar Cascades for detection and LBPH for recognition.

Jireh Jam (2019) proposed a student attendance system using face detection and recognition algorithms, emphasizing frontal face detection and demonstrating basic real-time recognition in classroom settings. Visalakshi and Ashish (2019) developed a multi-face recognition system capable of detecting and identifying multiple individuals simultaneously, using OpenCV and local histogram-based recognition techniques, improving attendance marking efficiency in group scenarios.

Shraddha Shinde and Patil Priyanka (2020) introduced a webcam-based attendance system that continuously monitored faces during lectures. By employing a multi-frame matching strategy, their approach addressed intermittent recognition issues caused by occlusions or movement, ensuring reliable real-time attendance tracking. Rakshitha et al. (2021) implemented a Python-based system using classical methods for face recognition, emphasizing accuracy, scalability, and proxy prevention.

These studies demonstrate that classical ML methods such as LBPH provide a practical and efficient approach to attendance management. They are robust under various lighting conditions and facial orientations, while requiring limited computational resources compared to deep learning models. Furthermore, integrating GUI frameworks like Tkinter and backend systems like Flask enhances usability, reporting, and administrative efficiency. Collectively, these works highlight the applicability of Python-based, ML-driven attendance systems as scalable and reliable solutions for modern institutions.

METHODOLOGY



The block diagram represents the working flow of the Face Recognition Attendance System. The system operates by first capturing a live video feed from a camera, detecting faces in real-time.

1. Camera Setup

A camera (webcam or external camera) captures the live video stream.

The video feed is processed frame by frame to detect and analyse faces for further action.

2. Face Detection

The system utilizes face detection algorithms (e.g., Haar Cascade Classifier) through OpenCV to identify faces in each frame of the video feed.

The detected face region is isolated from the background for further processing.

3. Image Preprocessing

Preprocessing steps such as resizing, normalization, and colour space conversion (e.g., from BGR to RGB) are applied to ensure the facial data is consistent and standardized.

The processed face image is prepared for feature extraction, ensuring high accuracy in the next steps.

4. Feature Extraction

- The system uses specialized libraries like face recognition to extract unique facial features from the detected face.
- Facial landmarks such as the distance between the eyes, nose structure, and overall face geometry are translated into a 128-dimensional numerical encoding, which serves as the unique identifier for that face.

5. Face Recognition

- The extracted facial features (encoding) are compared against those stored in the face database.
- The system calculates the similarity between the input face and faces in the database. If the match exceeds a predefined threshold, the system identifies the person.

6. Database Management

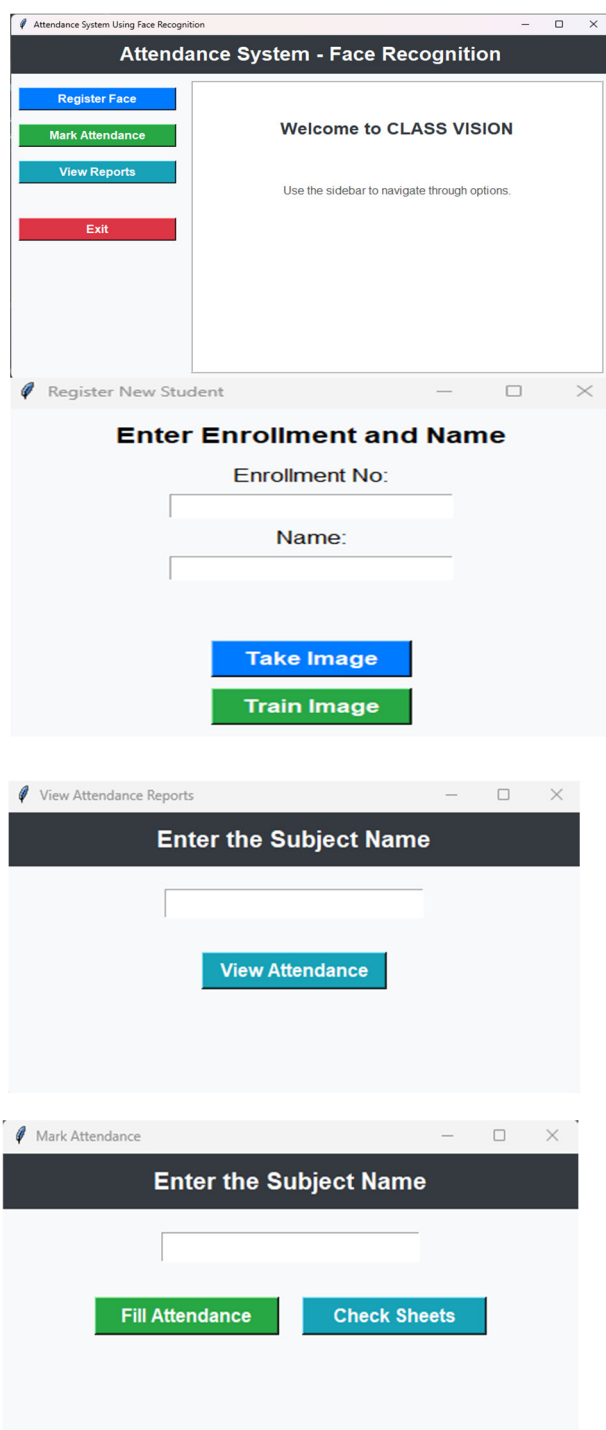
- The database stores facial encodings of registered users along with their identification details (name, ID, etc.).
- During attendance, the recognized face is cross-checked with the entries in the database, ensuring the person is a registered user.

7. Name Display and Attendance Marking

- Once the face is recognized, the system displays the name of the person on the user interface for confirmation.
- The system prevents duplicate attendance entries by cross-checking real-time recognition against previously logged data during the same session.

III. EXPERIMENTAL RESULTS AND ANALYSIS

- The proposed Face Recognition Attendance System was evaluated through experiments covering registration, training, and real-time attendance marking. Screenshots were captured at each stage to demonstrate the system's functionality.
- The home interface offers three options: "Register a New Student," "Take Attendance," and "View Attendance." Designed using Tkinter, the GUI allows easy navigation for administrators.
- During registration, administrators input the student's enrolment number and name, and multiple face images are captured using a webcam. Figures 4.2–4.4 illustrate this process. These images train the LBPH recognizer, which encodes facial patterns into histograms for the database, accounting for slight variations in expression and lighting.



IV. CONCLUSION

- In this project, an intelligent and automated Face Recognition Attendance System was successfully developed using computer vision and machine learning techniques. The system employs a real-time, efficient face detection and recognition

approach to accurately mark attendance without manual intervention or physical contact.

- Users can register their faces and record attendance through an interactive Tkinter-based desktop interface, while a Flask backend manages data storage and report generation. Real-time face detection, feature extraction, and recognition are performed seamlessly to ensure reliable and secure attendance marking.
- To enhance system performance and usability, a clean and user-friendly GUI was designed for easy operation. Additional features include prevention of duplicate entries during a session, integration with a database for record keeping, and automatic generation of CSV reports for future reference.
- The modular design ensures that the system can be extended in the future to include functionalities such as cloud storage, mobile application integration, and advanced liveness detection for improved security.

ACKNOWLEDGMENT

The heading of the Acknowledgment section and the References section must not be numbered.

Causal Productions wishes to acknowledge Michael Shell and other contributors for developing and maintaining the IEEE LaTeX style files which have been used in the preparation of this template. To see the list of contributors, please refer to the top of file IEEEtran.cls in the IEEE LaTeX distribution.

REFERENCES

- [1] **OpenCV**
G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000. [Online]. Available: <https://opencv.org/releases/>.
- [2] **pyttsx3**
R. Shevchuk, "Pyttsx3: Text-to-Speech conversion library in Python," 2018. [Online]. Available: <https://pypi.org/project/pyttsx3/>.
- [3] **Tkinter**
Python Software Foundation, "Tkinter: Python interface to Tcl/Tk," 2023. [Online]. Available: <https://docs.python.org/3/library/tkinter.html>.
- [4] **Python**
Python Software Foundation, "Python Language Reference, Version 3.15," 2025. [Online]. Available: <https://docs.python.org/3/>.
- [5] **Pandas**
W. McKinney, *Python for Data Analysis*, 2nd ed.,

O'Reilly Media, 2017. [Online]. Available:

<https://pandas.pydata.org/docs/>.

- [6] **NumPy**
C. Harris et al., "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, 2020. [Online]. Available: <https://numpy.org/doc/>.
- [7] **GitLab Repository**
T. Patel, "Face Recognition Attendance System Using Python," GitLab Repository, 2025. [Online]. Available: https://gitlab.com/TarangPatel0/attendance_system_using_facerecognition