RESEARCH ARTICLE

INTERNATIONAL RESEARCH JOURNAL OF MULTIDISCIPLINARY TECHNOVATION

Asian Research Association

DOI: 10.54392/irjmt26311

LabVIEW-Based Remote Laboratory with Pipelined Face Recognition for Multi-User Attendance Monitoring

Niket Amoda [a,*], Lochan Jolly [a], Arpit Rawankar [a]

[a] Department of Electronics & Telecommunication Engineering, Thakur College of Engineering & Technology, Mumbai, India

* Corresponding Author Email: niket.amoda@thakureducation.org

DOI: https://doi.org/10.54392/irjmt26311

Received: 15-02-2026; Revised: 25-04-2026; Accepted: 03-05-2026; Published: 11-05-2026

Abstract: This paper outlines an open source LabVIEW based Remote Laboratory System for Automated Attendance Monitoring through Facial Recognition to assist with both User Authentication and Multi-User Scalability within Online Practical Courses. The proposed RLS has been designed utilizing a Pipeline-Parallel Architecture where multiple Webcam Streams can be processed in parallel at different stages, namely Capture, Detection, Recognition and Logging. Evaluation of the proposed RLS involved testing with six Users simultaneously over a controlled environment. Results indicate Real-Time Operation with an average Per-User Latency Time of ~0.2 seconds. Average Throughput is estimated to be ~8 frames/sec/user (~45-50 frames/sec aggregate). Accuracy of Recognitions is reported as being ~96-99% under typical Lighting Conditions. A PCA/Eigenfaces Path was utilized for Fast Continuous Verification, and Periodic FaceNet-Based Checks were employed to Improve Robustness with moderate Overhead. Resource Profiling results estimate ~85% CPU Utilization from a Commodity Multi-Core Host (8-Cores / 16-Threading), 300 MB Memory Demand, and 3 Mbps Network Demand. These estimates suggest Straightforward Scaling via Additional Cores, GPU/FPGA Acceleration or Frame-Rate Throttling. The Proposed Design also Integrates Embedded/IoT Endpoints, which aligns with Computational Thinking and Education 4.0 Priorities that enable Authentic Participation, Learning Management System Integration, and Data-Driven Pedagogy. Wide-Area Conditions were Characterized Using a Laboratory WAN Emulator (tc/netem) that Simulated Controlled Levels of Latency, Jitter and Loss Representative of Campus, 4G and 5G Profiles. Due to the Complexity of Deploying an Internet-Scale Multi-Site Version of the Proposed RLS, such a Deployment is identified as Future Work.

Keywords: Remote Laboratories, LabVIEW, Attendance Monitoring, Face Recognition, Pipeline Parallelism, Machine Learning, IoT and 5G, Online Learning Platforms.

1. Introduction

Advances in virtualization and IoT technologies, as well as network technology allow for the creation of remote laboratories that provide opportunities beyond campus boundaries for practical engineering education. The global pandemic caused by covid-19 hastened the adoption of such systems, creating a need for remote laboratories to conduct experiments when physical presence was not possible [1]. Studies have identified over 100 different implementations of remote laboratories across diverse disciplines in engineering, confirming that well designed systems can achieve learning outcomes comparable to those experienced in-lab while providing on-demand access and repeatability [1]. Current platforms use cloud computing to deliver "laboratory as a service" (LaaS) [2], use low-cost embedded boards such as Raspberry Pi and Arduino for hardware interfacing [3], and leverage open-sourced frameworks to reduce deployment costs [4]. Most recently, fifth generation wireless networks have been investigated as enablers of ultra-low latency experiment control, further increasing accessibility to remote laboratories [5]. User identity verification is currently not incorporated in most design paradigms for real-time processing pipelines therefore many scalable frameworks that have been developed (including hybrid laboratory models) and architecture approaches to support concurrent experiments (as seen in systems such as NCSLab) do not integrate user identity verification into the real-time processing pipeline [6-9].

Beyond remote-lab development itself, student authentication and attendance tracking in distance practical classes have emerged as two major unmet needs. Traditional verification methods such as sign-in sheets and roll calls are time-consuming, susceptible to "proxy" attendance, and poorly suited to remote delivery, where the instructor cannot visually verify every participant. Proximity-based alternatives such as iBeacon-driven classroom attendance [10] automate

logging but rely on dedicated wireless hardware and authenticate the device rather than the person, leaving them vulnerable to the same proxy-attendance problem in unsupervised remote settings. Face recognition by biometrics represents a promising non-intrusive method to continue to verify the identity of students. PCA [11], CNN [12], and deep learning pipelines [13] have been used in multiple studies to develop face recognition based attendance systems; a broader survey of such attendance-marking systems is given by Puthea *et al.* [14]. These studies report accuracy rates of 90–98 percent when tested in a controlled environment. Improvements to single classroom implementations of these systems included pre-processing techniques to improve light robustness [15, 16], GSM-based alert systems [17], and gender-aware classification [18]. None of the previously mentioned studies addressed the concurrent processing requirements for large class room size remote lab implementations that require multiple video feeds to be processed simultaneously and continuously.

This paper fills a previously identified gap by developing a remote lab system using LabVIEW that includes facial recognition- based automatic attendance tracking as well as a pipelined multi-user architecture. The four-stage pipeline capture, face detection, face recognition, and attendance logging permits the overlap of execution of the stages of the pipeline while processing multiple video feeds in real time; consequently, real time processing of up to six users at one time can be accomplished on a commodity 8-core/16-thread server. The recognition strategy used is a combination of PCA/Eigenfaces and FaceNet DNNs for better robustness. The system was evaluated through a controlled quantitative benchmark on a local Gigabit Ethernet network using pre-recorded video sequences to eliminate network variability. Behaviour under wide-area conditions was characterized through emulated-WAN trials (tc/netem profiles injecting additive latency, jitter, and packet loss representative of campus, 4G, and 5G links); this emulation is a controlled laboratory proxy and is not a substitute for a geographically distributed internet-scale deployment, which remains future work. Measurements covered latency, throughput, recognition rate, and resource utilization; and comparisons were made to prior systems which utilized a single user and/or a classroom based approach. Under the controlled benchmark, the pipelined architecture produced a 4.9× throughput gain and a 3.7× latency reduction over a matched sequential baseline that used the identical hybrid recognizer, I/O, and logging path, and recognition rates of 95–99% demonstrating that remote implementation of a biometric system does not reduce the effectiveness of the authentication process. Additionally, the architecture supports the use of IoT embedded endpoints, 5G connectivity, and LMS integration allowing it to be aligned with Education 4.0 objectives of authentic online participation and data-driven pedagogy. We position this work as a systems-integration contribution rather than an algorithmic-novelty contribution: (a) a system-level pipelined architecture integrating continuous biometric verification into a LabVIEW remote laboratory, (b) a hybrid recognition strategy combining classical PCA/Eigenfaces with a pre-trained FaceNet embedding in a priority-based decision rule for the balance of speed and accuracy for continuous monitoring, and (c) a comprehensive empirical evaluation of the feasibility, efficiency of resources, and compatibility with contemporary educational technology environments.

2. Literature Review

The use of remote labs has been explored for approximately twenty years now, ranging from simple client-server (teleoperation) systems in the early days to complex (cloud-based) and IoT enabled remote lab systems today. As documented by Heradio *et al.* [1], more than one hundred remote labs were surveyed and three major issues were identified: concurrency; scalability; and user experience. Tawfik *et al.* [2], developed a new paradigm called Laboratory as a Service (LaaS), which uses cloud based systems to allow users to request access to an experiment on demand. Kaluz *et al.* [3], showed that using a combination of inexpensive components such as the Raspberry Pi and Arduino can be used to host control engineering type experiments at a much lower cost than that of commercial systems. Letowski *et al.* [4], continued this approach by creating a fully open source remote lab system that allows remote labs to be easily replicated at different institutions.

Addressing multi-user access, Barrios et al. [6] developed a remote academic laboratory supporting concurrent sessions over a multi-user network architecture, and Farah *et al.* [7] proposed a Node.js-based architecture for flexible multi-user remote experimentation. Lei *et al.* [8] introduced NCSLab, a scalable framework for concurrent online experimentation that employed pipelining at the experiment-scheduling level but did not incorporate biometric authentication. Vilches *et al.* [9] recently proposed scalable hybrid laboratories for industrial automation that combined physical and virtual resources, yet likewise omitted identity verification. The work of Vanegas *et al.* [19] has combined lab workflows from a remote location with computational notebooks to increase reproducibility as well as leave out an important issue of student authentication. In addition to surveying the opinions of students regarding remote labs through several other recent educational research studies on pedagogy [20] that indicate generally high levels of satisfaction from users, there was no consideration for ongoing student authentication. Therefore, collectively all of these works demonstrate how to accomplish good quality (mature) remote laboratory based concurrent

experimentation; yet none were able to incorporate ongoing biometric monitoring into the real time experimental data processing stream.

The current study bridges these two research streams at the system level, integrating a pipelined continuous face-recognition subsystem into a LabVIEW-controlled remote laboratory. We do not claim algorithmic novelty in detection or recognition; rather, the contribution is the integration of existing components (Haar cascade, PCA/Eigenfaces, pre-trained FaceNet, LabVIEW producer–consumer queues) into a concurrent multi-user remote-lab architecture whose end-to-end behavior has, to the best of our knowledge, not been benchmarked jointly in prior published work.

3. System Architecture

The proposed system uses a client-server architecture that supports concurrent biometric processing. A high-level overview of the system is shown in Figure 1. Each remote user will use a webcam-equipped computer using either LabVIEW Runtime Interface or their standard web browser to perform face recognition. All remote users are connected via internet to the LabVIEW application hosted on the server-side which controls both experiment logic and attendance tracking.

The NI Vision Development Module is used as part of LabVIEW application on the server-side to capture images from cameras and perform face detection. There are several components in this architecture. Each remote user uses LabVIEW's Remote Panels and Web Server to have their own unique session, with each session having its own webcam feed being captured using IP camera streams (i.e., 192.168.0.X addresses) [6]. The network layer in this architecture utilizes TCP/IP and the VISA protocol, with each video stream (640×480 at 10 fps, MJPEG compressed) utilizing approximately 0.5 Mbps of bandwidth per user, thus allowing the system to be utilized with campus networks or 4G/5G connections [5]. The LabVIEW Server Application provides a real-time environment to run both experiment and attendance logic using two parallel loops. For every user session it conducts a face detection/face recognition process and flags any unrecognizable users [12]. Every record of attendance includes a student's ID number, timestamp, confidence rating from facial recognition software (for the accuracy of the identification), and whether they were successfully authenticated. This allows them to be easily integrated into Learning Management Systems (LMS) via RESTful API calls [2]. The Embedded / IoT Layer utilizes either NI myRIO or NI CompactRIO products for connection to hardware at the edge. Each apparatus acts independently as an edge device [3]. Security is provided through encryption of all video streams utilizing HTTPS and encrypting biometric templates with AES-256 [4].

To clearly show the distinction of live demonstration versus quantitative benchmarking for validating our system, there are two validation scenarios outlined within this paper. The first is the live demonstration (Figure 2 and Figure 3), which demonstrates end-to-end functionality in use by six concurrent live webcam sessions on the campus local area network. This demonstration shows that sessions start properly, multiple video streams can be rendered at the same time, and all session states are updated in real-time. However, no numbers were derived from this demonstration as part of the results reported in Section 5. A second type of validation scenario is the quantitative benchmark (Section 4.7 and Section 5).

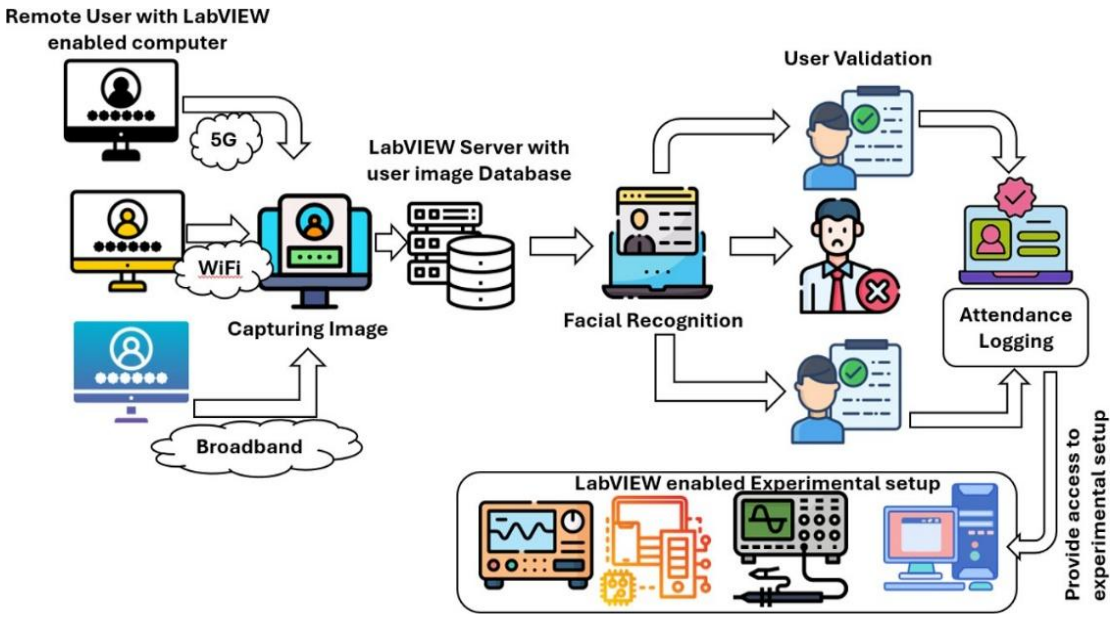

Figure 1. System architecture of the remote lab attendance monitoring system

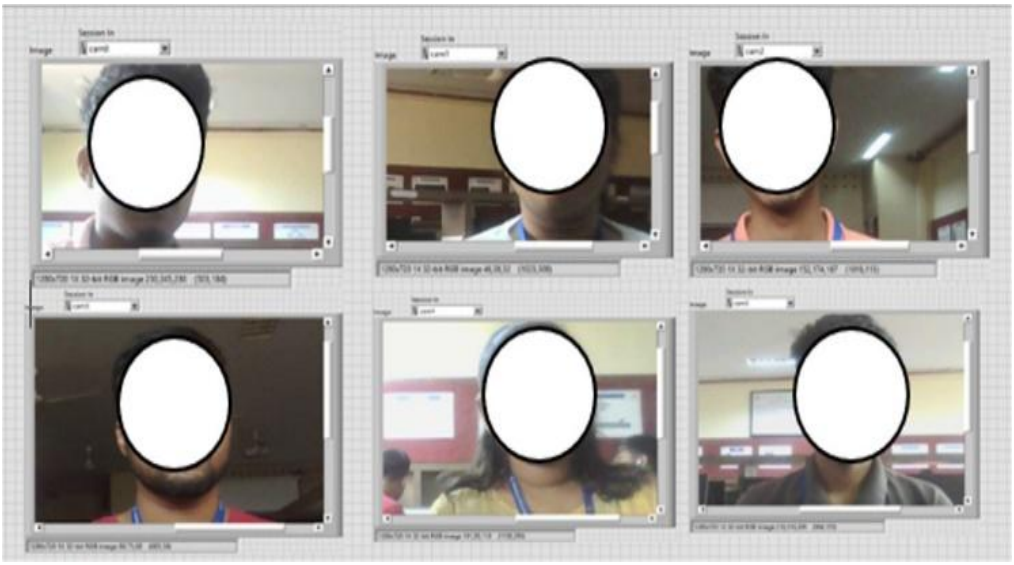

Figure 2. LabVIEW front panel interface during multi-user operation with six concurrent users

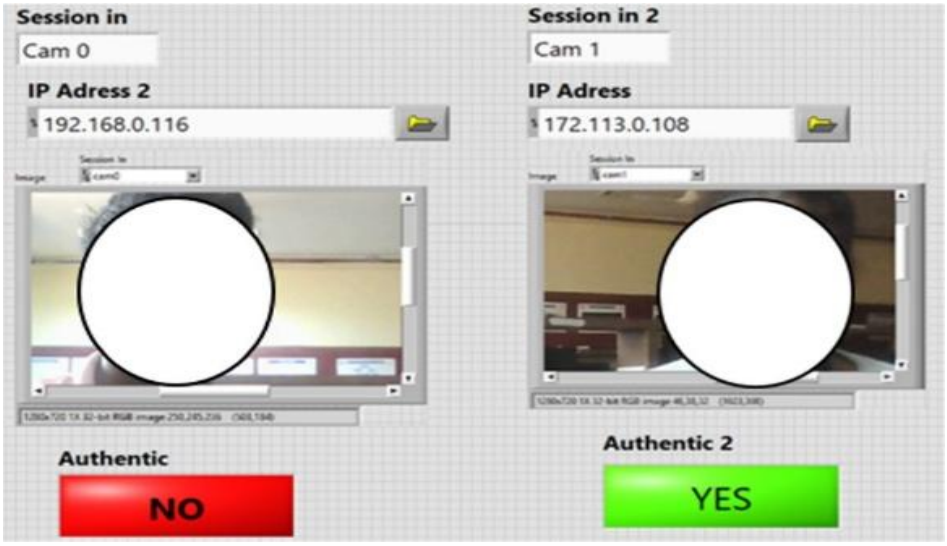

Figure 3. LabVIEW interface showing two user sessions with face recognition status: left “NO,” right “YES

For the quantitative benchmark, we utilized six deterministic pre-recorded video sequences that would be played through the entire pipeline to allow us to report measurable data regarding how long each process took and how much bandwidth was required. In addition, to provide repeatability for these measurements, we replayed the same sequence of videos during each run and separated network variability into a single controlled experiment (the emulated-WAN trials - Section 5.6).

The validation of operational functionality for multiple users was accomplished through six live interactive classes that ran concurrently. As can be seen from Figure 2, the LabView Front Panel displayed all six active users, as well as the video feeds (in real-time) along with their respective Session IDs and Recognition Status. Although scaling to greater than six users is addressed in Section 5.9 as an Engineering Projection, it has not been shown by experimentation in this study. A snapshot of the Authentication Interface for two separate sessions is provided in Figure 3. In the left image, the unidentified user displays "NO" in Red, where the authenticated user displays "YES" in Green, thus allowing the instructor to confirm user identity immediately.

4. Methodology

4.1 Experimental Setup and Hardware Configuration

All experiments used a server running Microsoft Windows 10 (64-bit) and had an Intel Core I7-10700 processor (with eight physical cores, sixteen threads and 2.9 Ghz as a base clock, and 4.8 Gghz as turbo). The server also included 16 gigabytes of DDR4 ram. Integrated into the processor was the Intel UHD 630 graphics card. The six clients all contained a Logitech C920 HD camera (native resolution of 1080p, and down sampled to 640x480 when transmitted to the server.) The server and the six client machines communicated

over a gigabit Ethernet LAN. Measured round trip latency was less than one millisecond. This controlled LAN was used for the quantitative benchmark to eliminate network variability; emulated-WAN measurements (Section 5.6) add controlled latency, jitter, and packet loss on top of this baseline. Video streams were transmitted using MJPEG compression over HTTP, with each stream consuming approximately 0.5 Mbps at 10 fps. The Python 3.8 runtime with TensorFlow 2.4 and the pre-trained FaceNet model (Inception-ResNet-v1, 128-dimensional embeddings) [21] was invoked from LabVIEW via the System Exec VI for deep-neural-network-based recognition. The complete list of hyperparameters, queue sizes, and implementation choices for this hardware/software stack is consolidated in Table 1 (Section 4.6).

4.2 Enrolment Procedure and Dataset

A database of 10 enrolled subjects (university students, age 20–25, 6 male and 4 female) was constructed. For each subject, 20 frontal-face images were captured under three illumination conditions (fluorescent overhead, natural window light, and dim ambient) and two pose variations (±15° yaw), yielding 200 enrolment images in total. The images were cropped to 100x100 pixels using Viola – Jones face detection [22], then converted to equalize histograms so that lighting effects could be removed. For each individual enrolled in the system two templates were created: (a) an Eigenfaces/PCA projection vector for that person; and (b) a 128-dimension FaceNet feature vector by taking the L_2-normalized mean of all of their enrolment vectors [21].

It should be specifically noted that the enrolment set has both a small number of individuals (10) and low demographically representative populations (university students age 20-25 at one institution). In addition to the ten authentic subjects who were enrolled, there were also five un-enrolled (impersonator) subjects used to test the accuracy of the system's true acceptance rate. Therefore, we can make a claim as to the feasibility of a pilot scale study using our data set but no broader deployment claims are being made. A formal limitations statement is outlined in Section 5.11.

4.3 Validation Split and Threshold Selection

The thresholds θ_{PCA} and θ_{DNN} were selected using a subject-disjoint split: of the 10 enrolled subjects, 7 were used for enrollment and threshold tuning, and the remaining 3 together with the 5 impostors formed a held-out evaluation set. For threshold selection we swept $\theta_{PCA} \in \{2000, 2500, 3000, 3500, 4000\}$ and $\theta_{DNN} \in \{0.50, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85\}$ and chose the operating point that maximized true-recognition rate subject to a false-acceptance constraint of FAR ≤ 1% on the tuning set. The resulting operating points were $\theta_{PCA} = 3000$ and $\theta_{DNN} = 0.70$. Threshold sensitivity on the held-out set is reported in Section 5.5.

4.4 Pipelining Algorithm for Concurrent Users

Pipeline parallelism overlaps different processing stages in time, analogous to instruction pipelining in processor design. Each incoming video frame is processed through four sequential stages: (S1) Capture, (S2) Face Detection, (S3) Face Recognition, and (S4) Logging/Notification. At any given clock cycle, different users occupy different stages, enabling round-robin execution instead of sequential processing. Figure 4 illustrates one pipeline cycle; this cycle repeats continuously with a batch size equal to the number of active users (up to six in our experiments).

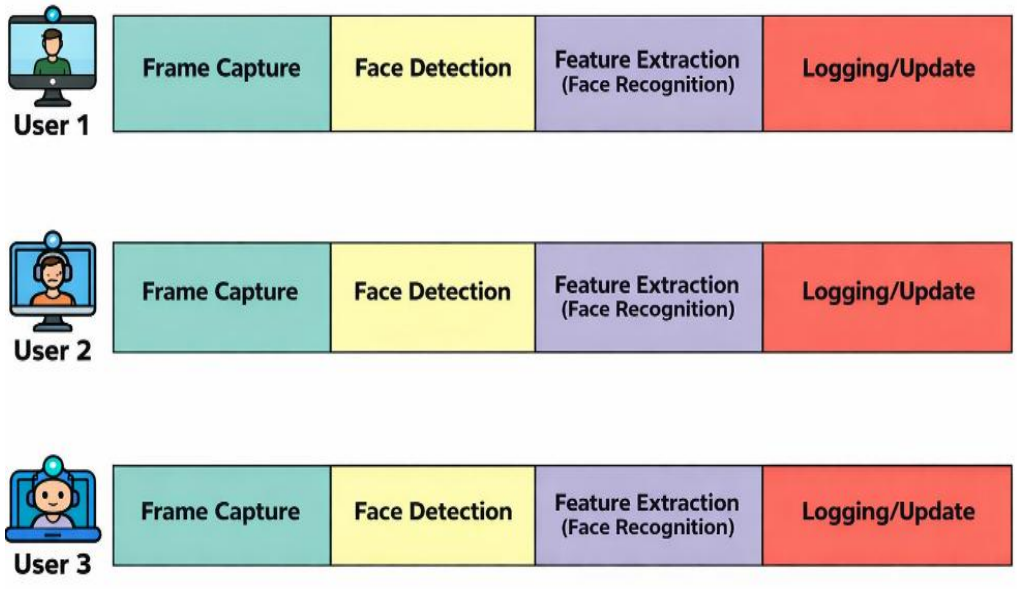

Figure 4. Pipeline Parallelism in Multi-User Attendance Monitoring

The server captures in real-time a single frame for every user through their IP camera using the LabVIEW IMAQ Grab VI within a time based loop that is set at 100 ms interval (target of 10 frames per second) and assigns a ⟨user-ID, incrementing sequence number, timestamp⟩ identifier to each frame, placing it on a producer-consumer queue [4]. The inter-stage queues are bounded (capacity 20 frames each); on overflow, the oldest frame for the affected user is dropped and the drop event is logged, preserving ordering for remaining frames.

In Stage S2 (Face Detection), dequeued frames are processed by a Haar Cascade classifier (Viola–Jones algorithm [22]) configured with a minimum face size of 80×80 pixels and a scale factor of 1.1. The classifier outputs a bounding box for the largest detected face, which is cropped and resized to 100×100 grayscale pixels. Frames with no detected face are flagged and forwarded to S4 directly. The choice of Haar is motivated by low CPU cost; we compare against stronger detectors (MTCNN, SSD) in Section 5.5.

Algorithm 1. Hybrid recognition with DNN-priority decision rule

Require: Face crop x_t at frame index t; enrolled PCA templates $\{p_i\}$; enrolled FaceNet embeddings $\{e_i\}$; thresholds θ_{PCA}, θ_{DNN}; DNN period $K = 10$	
Ensure: Predicted identity $\hat{y} \in \{ID\_1, \ldots, ID\_N\} \cup \{\emptyset\}$, where $\emptyset$ denotes "no match / impostor"	
1:	*// PCA path (runs on every frame)*
2:	$d_{PCA} \leftarrow \min_i \lVert PCA(x_t) - p_i \rVert 2$
3:	$\hat{y}_{PCA} \leftarrow \arg\min_i \lVert PCA(x_t) - p_i \rVert 2$
4:	**if** $d_{PCA} \geq \theta_{PCA}$ **then**
5:	$\hat{y}_{PCA} \leftarrow \emptyset$ **// PCA reject**
6:	**end if**
7:	**if** t mod $K \neq 0$ then
8:	**return** $\hat{y}_{PCA}$ **// non-DNN frame: PCA alone**
9:	**end if**
10:	*// DNN path (every K-th frame)*
11:	$s_{DNN} \leftarrow \max_i \cos(\text{FaceNet}(x_t), e_i)$
12:	$\hat{y}_{DNN} \leftarrow \arg\max_i \cos(\text{FaceNet}(x_t), e_i)$
13:	**if** $s_{DNN} \leq \theta_{DNN}$ **then**
14:	$\hat{y}_{DNN} \leftarrow \emptyset$ **// DNN reject**
15:	**end if**
16:	*// Fusion on DNN-check frames*
17:	**if** $\hat{y}_{PCA} = \emptyset$ **and** $\hat{y}_{DNN} = \emptyset$ **then**
18:	**return** $\emptyset$ **// both reject: no identity assigned**
19:	**else if** $\hat{y}_{PCA} = \hat{y}_{DNN}$ **then**
20:	**return** $\hat{y}_{PCA}$ **// agreement: accept consensus**
21:	**else**
22:	**return** $\hat{y}_{DNN}$ **// disagreement (incl. single-sided reject): DNN overrides**
23:	**end if**

In Stage S3 (Face Recognition), the detected face is compared against the enrolment database using a hybrid strategy. On every frame, the PCA/Eigenfaces path projects the face into the 40-dimensional eigenspace and computes the Euclidean distance to each enrolled template; a match is declared if the minimum distance falls below a threshold $\theta_{PCA} = 3000$ (selection procedure described in Section 4.3). On every 10th frame, the FaceNet deep-neural-network path computes a 128-dimensional embedding via Python and measures cosine similarity to each enrolled embedding; a match is declared if the maximum similarity exceeds $\theta_{DNN} = 0.70$ [21]. We correct an earlier imprecise description: the decision rule is not a true majority vote (there are only two recognizers). It is a DNN-priority rule with periodic deep verification. On the 9 of every 10 frames where only the PCA result is available, the PCA label is used as the provisional identity. On the 1 of every 10 frames where both results are available, the consensus label is accepted if the two agree; if they disagree, the FaceNet label overrides the PCA label, on the grounds that FaceNet has higher discriminative capacity on unconstrained faces [21, 23]. If both recognizers reject the probe (i.e. PCA distance above θ_{PCA} and FaceNet similarity below θ_{DNN} on the same DNN-check frame), no identity is assigned for that frame and the event is logged as an "unrecognized" observation rather than as an attendance mark; this is the behaviour formalized on line 18 of Algorithm 1. A formal statement of the full rule is given in Algorithm 1.

In the logging/notification stage, S4, when the verification of recognized faces has been completed, those faces are entered in the student attendance database with a status of "present" and include; Student ID, Timestamp, and Confidence Level. When the system identifies a face that has no corresponding record in its database, it will display the absence of identification (red "no" light on LabVIEW dashboard) along with an option for an instructor to receive notification via email [17]. The data collected for students' attendance will be extracted from LabVIEW in the form of a Comma Separated Value File (CSV) and will be automatically uploaded to an institution's learning management system (LMS) through the use of an Institution Provided Restful API.

LabVIEW provides an execution model for data flow which allows users to create this type of pipeline using separate while loops running in parallel and connected by queues. With each stage running in its own thread, the LabVIEW Runtime automatically loads threads onto available CPU cores. Frame-level synchronization is maintained through tagged queue elements, ensuring that each user's frames are processed in order [4].

4.5 Recognition Algorithms

PCA/Eigenfaces: The enrolled face images are mean-centered and decomposed via singular value decomposition to obtain the top k = 40 eigenvectors (eigenfaces). Each probe face is projected into this subspace, and recognition is performed by nearest-neighbor Euclidean distance matching [11]. This

lightweight approach achieves per-face recognition latency of approximately 5 ms on the test hardware.

FaceNet Deep Neural Network: The pre-trained Inception-ResNet-v1 model [21] maps each 160×160 RGB face image to a 128-dimensional unit-hypersphere embedding. We used the publicly available checkpoint reported in [21] without fine- tuning; no domain-specific training was performed. Recognition is performed by cosine similarity against stored enrolment embeddings. Per-face inference latency is approximately 50 ms on CPU. Domain-specific fine-tuning is identified as a future improvement.

Hybrid Fusion: The two recognition paths run at different temporal frequencies (PCA on every frame, FaceNet on every 10th frame). As formalized in Algorithm 1, between DNN checks the PCA result is used alone; when both results are available, agreement yields the consensus label, disagreement is resolved in favor of FaceNet, and the case in which both recognizers reject the probe yields no identity assignment (the frame is logged as “unrecognized”). This hybrid approach balances the speed of PCA (enabling real-time visual feedback) with the accuracy of FaceNet (reducing accumulated drift in identity confidence).

4.6 Reproducibility

To enable independent replication of the experiments reported in Section 5, this subsection documents every non-trivial implementation choice. All relevant hyperparameters, queue sizes, worker-thread counts, threshold values, and dataset split sizes are consolidated in Table 1.

Table 1. Implementation Parameters (For Reproducibility)

Parameter	Value
Frame capture rate	10 fps (100 ms timed loop)
Frame resolution	640×480, MJPEG
Detection cascade	OpenCV Haar frontal-face default
Min face size / scale factor	80×80 / 1.1
PCA eigenvectors (k)	40
PCA threshold θ_PCA	3000 (L2)
FaceNet embedding dim.	128
FaceNet period K	every 10th frame
FaceNet threshold θ_DNN	0.70 (cosine)
Inter-stage queue capacity	20 frames per user
Worker threads per stage	4
Impostor subjects (eval)	5
Enrolled subjects	10 (7 tune / 3 held-out)
Runs per configuration	5 (independent restarts)

Unless explicitly noted, every configuration evaluated in the remainder of this paper uses the values listed in Table 1. LabVIEW timed-loops were set to dataflow priority “normal,” with four worker threads per pipeline stage (as recorded in Table 1). The Python FaceNet process was spawned once per session and communicated with LabVIEW via stdin/stdout pipes (System Exec VI) to avoid per-frame process-launch cost. All runs used a fixed random seed for any stochastic operation (none were used in deployment, but seeding was applied to image shuffling during threshold tuning). Per-frame timestamps (microsecond resolution) were logged and released together with the attendance CSVs.

4.7 Performance Evaluation Protocol

To ensure reproducible evaluation, whose implementation details are summarized in Section 4.6 and Table 1, six pre-recorded video sequences (one per simulated user, 60 seconds each, 10 fps, 600 frames per user) were fed to the system. In this manner, this structured method removed variability in the network and provided consistent test conditions for each trial. The metrics listed below were measured over five separate trials. Run-to-run independence was preserved by restarting the LabVIEW application and the Python FaceNet process between runs and clearing all queues; runs were interleaved across configurations to avoid thermal/ordering bias.

Latency was determined by measuring the time it took from when a video frame was captured (S1 entry) to when the user attended log was updated (S4 exit) using LabVIEW’s tick count VI which is based on a high resolution timer (1 microsecond accuracy) [24]. In addition to end-to-end latency, we logged per-stage residence times (entry-to-exit at S1, S2, S3, S4) and the queue-occupancy time series, so that the bottleneck stage can be identified (Section 5.2).

Throughput was determined by measuring the total number of frames processed per second by all active users during the steady state period (i.e., after the initial 5 seconds of pipeline warming up). True Recognition Rate (TRR) was computed as the proportion of enrolled-subject frames correctly identified, across all 10 enrolled subjects (10 × 600 = 6000 frames per run in single-user mode; stratified sampling in multi-user mode). False Acceptance Rate (FAR) was computed using five impostor subjects (not enrolled) whose video sequences were substituted into user slots. FAR was defined as the proportion of impostor frames incorrectly accepted as an enrolled identity.

Resource Utilization (CPU, memory, network bandwidth) was recorded using Windows Performance Monitor at 1-second intervals throughout each run. Statistical analysis to determine whether the difference in latency and throughput between pipelined configurations and sequential configurations is statistically significant utilized a two-tailed paired t-test at $\alpha = 0.05$ for the five different test runs. Before applying the paired t-test, the distribution of paired differences

was checked for normality using the Shapiro–Wilk test; all tested pairs returned $p > 0.2$, consistent with normality for $n = 5$. We additionally report bootstrap 95% confidence intervals (10,000 resamples) for the speedup ratios.

4.8 Controlled Baseline Definition

To make the speedup claims auditable, we define the sequential baseline as a single-threaded LabVIEW implementation that (i) uses the identical hybrid PCA+FaceNet recognizer with identical thresholds, (ii) reads frames from the same six pre-recorded sequences at the same target 10 fps, (iii) uses the same CSV logging path and the same dashboard update code, and (iv) processes users in a strict round-robin loop within a single while-loop rather than parallel producer–consumer loops. The only architectural difference from the pipelined configuration is therefore the absence of pipeline parallelism; no other code, I/O path, or recognizer component differs. The "Pipelined PCA-only" variant removes the FaceNet stage entirely. An ablation with these controlled variants is reported in Section 5.4.

4.9 Emulated-WAN Evaluation

To characterize behavior under realistic remote conditions, we additionally evaluated the pipeline with a WAN-emulation layer (Linux tc/netem on an intermediate Linux bridge) applied between clients and server. We swept four profiles: (P1) LAN baseline (RTT < 1 ms, 0% loss); (P2) campus-WAN (20 ms RTT, 5 ms jitter, 0% loss); (P3) 4G-like (50 ms RTT, 15 ms jitter, 0.5% loss, 20 Mbps cap); (P4) 5G-like (10 ms RTT, 3 ms jitter, 0.1% loss, 100 Mbps cap). All other parameters were held constant. This emulation-based evaluation is a laboratory proxy for true internet deployment and is labeled as such throughout the paper. The results are discussed in Section 5.6.

4.10 Stress Testing and Failure Modes

To probe operational stability under backlog, we ran two stress scenarios: (S1) overload, in which 12 simulated client streams were injected into hardware provisioned for 6; (S2) stalled session, in which one user's capture thread was artificially blocked for 10 s to test whether the other sessions continue operating and whether queues recover without deadlock. We report queue depth over time, dropped-frame counts, and attendance-record temporal ordering in Section 5.10.

5. Result and Discussion

The evaluation of the system was performed within the controlled environment described in Section 4.7. Unless otherwise stated, every configuration in this section uses the hyperparameters listed in Table 1. The results were the average ± standard deviation of the means of all data collected from five separate test runs unless otherwise stated. The results of this study demonstrated that the pipelined multi-user design successfully reduced the negative effects on individual user performance due to increased concurrency.

5.1 Latency Analysis

Table 2 lists the average per-frame latency for varying amounts of concurrent users, and Figure 5 plots the same data. The end-to-end latency was 198 ± 8 ms with one user. When there were six users, the average latency grew to 242 ± 15 ms; this is an increase in only 22% which suggests that the pipeline successfully distributed the processing costs among each stage. On the other hand, when we compared the sequential (non-pipelined) baseline to the pipelined configuration for six users, the sequential configuration averaged a latency of 905 ± 32 ms and represented a 3.7× degradation from the pipelined configuration. The difference in latency for both configurations at six users was statistically different ($p < .001$; $t(4) = 48.7$; bootstrap 95% CI for the speedup = [3.55, 3.86]×). In addition, in all configurations, the dashboard status updated in less than 0.25 s after capturing the frame, thus satisfying the real-time response requirement (response time < 0.5 s for interactive remote laboratory) set forth by Lei *et al.* [8].

The scaling behavior captured in Table 2 is visualized in Figure 5. The pipelined configuration demonstrates nearly flat growth (around 9 ms for each added user) while the sequential baseline demonstrated an approximate linear increase of around 140 ms per user, which supports the idea that pipeline parallelism is able to absorb the processing costs for individual users.

5.2 Per-Stage Latency and Queue Behaviour

Table 3 provides data regarding the residence time for each stage during a 6-user pipelined configuration.

Table 2. Mean Per-Frame Latency (ms) Vs. Number of Concurrent Users

Configuration	1 User	2 Users	3 Users	4 Users	5 Users	6 Users
Pipelined	198 ± 8	205 ± 10	215 ± 11	225 ± 12	234 ± 14	242 ± 15
Sequential	198 ± 8	388 ± 18	570 ± 22	745 ± 27	830 ± 30	905 ± 32

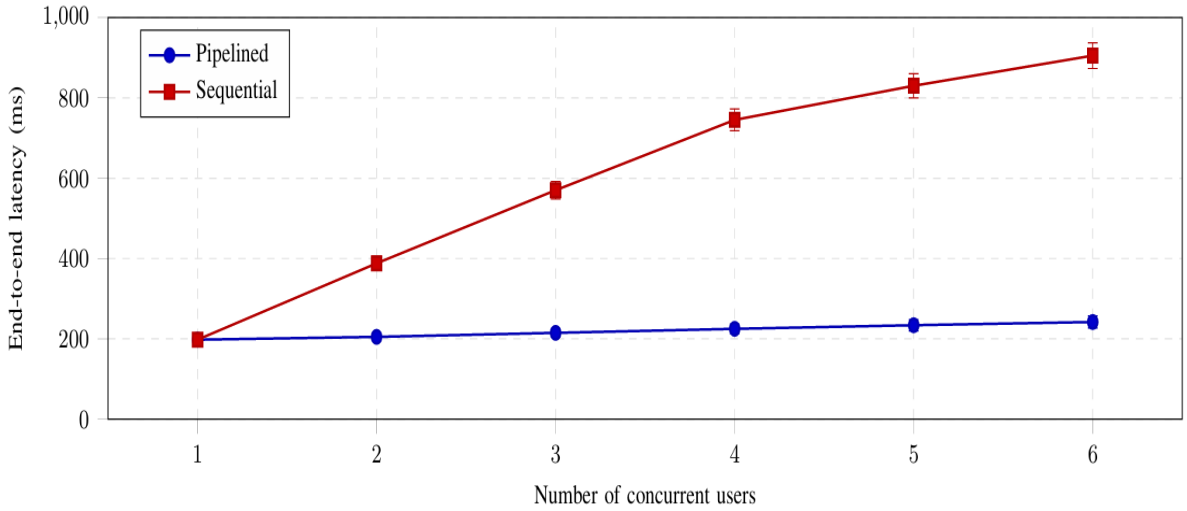

Figure 5. Latency scaling vs. number of concurrent users (error bars: ±1 std over 5 runs)

Table 3. Per-Stage Residence Time (ms, 6-user pipelined, mean ± std)

Stage	Residence (ms)	% of end-to-end
S1 Capture	18 ± 4	7.4%
S2 Face detection (Haar)	42 ± 6	17.4%
S3 Recognition (PCA + periodic FaceNet)	162 ± 13	66.9%
S4 Logging / dashboard update	9 ± 2	3.7%
Queue / IPC overhead	11 ± 3	4.6%
End-to-end	**242 ± 15**	**100%**

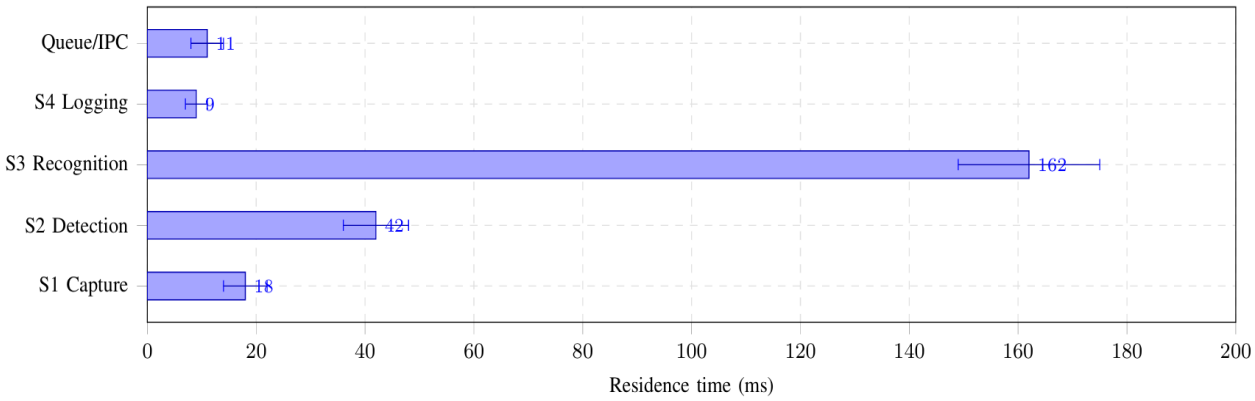

Figure 6. Per-stage residence time breakdown for 6 concurrent users; S3 (recognition) dominates

A graphical representation of these results can be found in Figure 6 in the form of a horizontal bar graph. The recognition stage (S3) dominates end-to-end latency, consistent with the periodic FaceNet inference cost (~50 ms amortized over 10 frames, plus PCA on every frame). Detection (S2) is the second-largest contributor. Capture (S1) and logging (S4) are lightweight. The bottleneck stage is therefore S3; GPU offload of FaceNet would act directly on this bottleneck, which motivates the scaling projection in Section 5.9.

Queue occupancy on the S2→S3 link averaged 2.3 ± 0.6 frames across runs, with peaks of 8 frames during DNN-inference bursts (well below the 20-frame capacity listed in Table 1). The dropped-frame rate under nominal 6-user load was 0.21±0.09%; frames were dropped exclusively at the S2→S3 queue during DNN bursts, and frame ordering within each user's stream was preserved in 100% of logged frames (verified via the sequence-number tag).

5.3 Throughput Analysis

Table 4 shows the per-user and total throughputs that were measured for the three configurations (sequential baseline, PCA-only, and hybrid (PCA + DNN)) shown graphically in Figure 7. For the six-user case, the hybrid pipelining configuration

sustained 7.8±0.3 frames-per-second (fps)/user or 46.8±1.8 fps/user (aggregate), which represents about 78 percent of the theoretically expected value (six streams processed at 10 fps/input=60 fps). In contrast, the PCA-only version was able to achieve better throughputs than the hybrid version (9.5±0.2 fps/user), because it did not have to perform DNN inference. The sequential baseline dropped to 1.6±0.1 fps/user. Therefore, the hybrid pipelining configuration achieved a 4.9X throughput advantage relative to the sequential baseline ($p<0.001$; $t(4)=52.1$; 95 percent bootstrap confidence interval = [4.71, 5.12]).

5.4 Controlled Ablation (Fair Baseline)

Table 5 compares the effects of each architecture choice separately, and the Pareto front for throughput vs. latency from the resulting variants is shown in Figure 8. The variants all have the same hybrid recognizer, I/O and logging as Section 4.8; their difference lies solely in the dimension noted above.

The data in Table 5 and Figure 8 illustrate how much throughput was gained by pipelining alone (sequential→PCA-pipeline), and how the hybrid recognizer gave up approximately 10 frames per second of total throughput to improve the TRR by about 4.1 percent relative to PCA-only. The result also confirms why we used a K=10 periodic DNN design since running FaceNet on every frame achieved the best TRR but lost in terms of real time performance.

5.5 Recognition Accuracy

In addition to the performance metrics presented in Table 6 for each of the three configuration options, Figure 9 displays a comparison of the true recognition rate (TRR) across all lighting conditions as well as false acceptance rates (FAR). The PCA-only path achieved a TRR of 92.3 ± 1.4% under normal lighting, which dropped to 85.1 ± 2.8% under low-light conditions. Augmenting with periodic FaceNet checks (hybrid mode) raised TRR to 96.4 ± 0.9% under normal lighting and 93.7 ± 1.6% under low light. The FaceNet-only configuration (run on every frame) achieved the highest TRR of 98.8 ± 0.4% but at the cost of throughput (limited to approximately 3.2 fps per user). The FAR was 0.0% across all configurations in normal lighting and rose to 0.8 ± 0.4% for PCA-only under low light, while the hybrid and DNN-only configurations maintained FAR below 0.2%. The False Reject Rate (FRR) of 3.6% for the Hybrid configuration under Normal Lighting conditions can be attributed mainly to Motion Blur and Partial Occlusions of the Face.

Threshold Sensitivity: Figure 10 reports TRR and FAR on the held-out subject-disjoint split as θ_{DNN} is varied (with θ_{PCA} fixed at its selected value). TRR for the hybrid rule remained within ±1.3 percentage points of peak for $\theta_{DNN} \in [0.65, 0.75]$ and for $\theta_{PCA} \in [2500, 3500]$ (not shown), indicating that the selected operating points are stable across moderate threshold perturbations.

Table 4. Throughput (Fps) for Six Concurrent Users under Different Configurations

Metric	Pipelined Hybrid	Pipelined PCA-only	Sequential Baseline
Per-user fps	7.8 ± 0.3	9.5 ± 0.2	1.6 ± 0.1
Aggregate fps	46.8 ± 1.8	57.0 ± 1.2	9.6 ± 0.6
Pipeline speedup	4.9×	5.9×	—

Figure 7. Aggregate throughput comparison at 6 concurrent users (error bars: ±1 std over 5 runs)

Table 5. Controlled ablation at 6 concurrent users (mean over 5 runs)

Variant	Latency (ms)	Aggregate fps	TRR (normal)
Sequential, hybrid recognizer	905 ± 32	9.6 ± 0.6	96.1%
Pipelined, PCA only	188 ± 10	57.0 ± 1.2	92.3%
Pipelined, FaceNet every frame	510 ± 28	19.2 ± 1.0	98.8%
Pipelined, hybrid (proposed)	**242 ± 15**	**46.8 ± 1.8**	**96.4%**

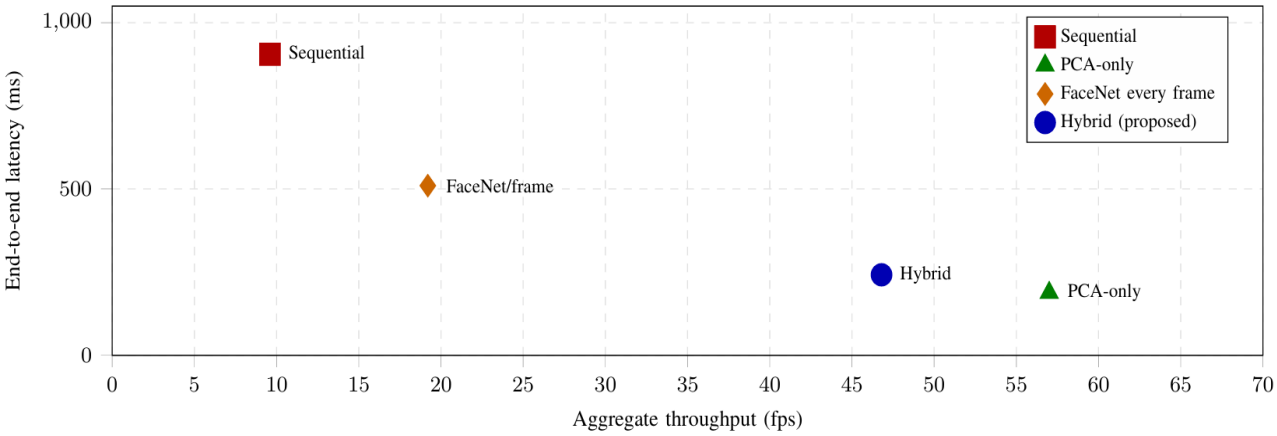

Figure 8. Ablation: latency vs. throughput Pareto view (lower-right is better: high throughput, low latency)

Table 6. Recognition Accuracy Under Different Configurations and Lighting Conditions

Config.	TRR -- Normal	TRR -- Low	FAR -- Normal	FAR -- Low
PCA-only	92.3±1.4%	85.1±2.8%	0.00%	0.8±0.4%
Hybrid	96.4±0.9%	93.7±1.6%	0.00%	0.2±0.1%
DNN-only	98.8±0.4%	97.2±0.7%	0.00%	0.1±0.1%

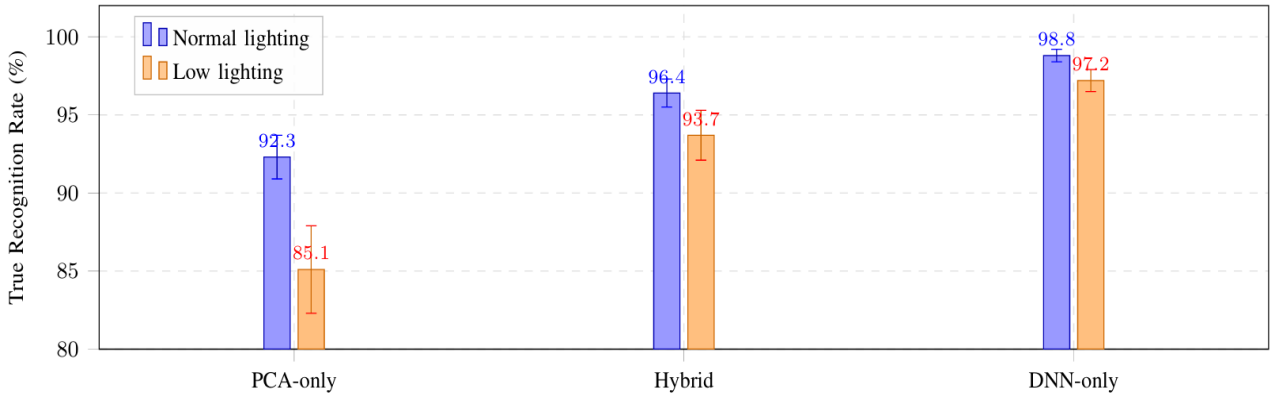

Figure 9. True Recognition Rate across configurations under normal and low lighting

FAR rose sharply for $\theta_{DNN} < 0.60$, consistent with the usual cosine-similarity operating range for FaceNet embeddings.

Detector Comparison: Haar is compared against MTCNN [25] and an SSD-based detector [26] in Table 7, with the rest of the pipeline held constant; the detection-time vs. miss-rate trade-off for these three detectors is visualised in Figure 11. As summarized in Table 7 and Figure 11, stronger detectors reduce miss rate substantially under occlusion but roughly double detection latency and would push the S3 bottleneck above the real-time budget on the current hardware. We retain Haar for the real-time benchmark and identify

MTCNN/SSD (GPU-accelerated) as the natural upgrade path.

These accuracy rates are also better than previously reported results from other facial recognition-based attendance systems. Lukas *et al.* [27] demonstrated that they could achieve approximately 90% accuracy with PCA/DCT-based techniques in a classroom setting, while Bhattacharya *et al*. [12] were able to achieve 95-98% accuracy with a CNN based technique in their SAMS system, and Bah and Ming [16] reported comparable accuracy using enhanced LBP features in a single-user offline scenario. Therefore, our hybrid configuration bridges the accuracy gap, and achieves its performance in a more difficult to accomplish real-time, multiple users, remotely-operated environment. The increase in accuracy from using PCA only is due to FaceNet's capability to recognize and learn discriminating characteristics that remain the same regardless of moderate variations in illumination or pose [21]. These findings are consistent with those reported by Arsenovic *et al.* [13] for deep-learning-based attendance systems.

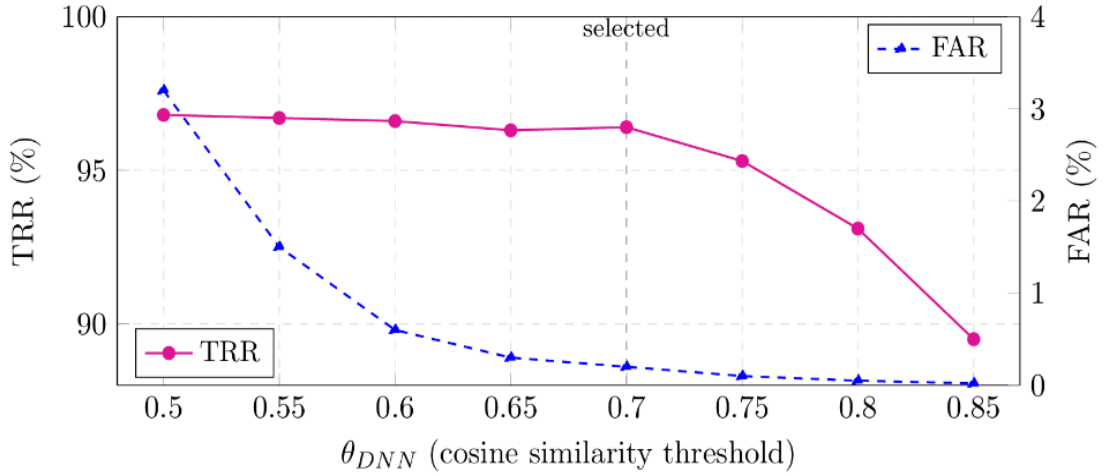

Figure 10. Threshold sensitivity for θDNN on the held-out split. The selected operating point (0.70) sits in a stable plateau for TRR with FAR < 0.25%.

Table 7. Face-detector comparison (held-out set)

Detector	Det. time (ms)	Miss rate, normal	Miss rate, occluded
Haar cascade (used)	42 ± 6	3.9%	11.7%
MTCNN (CPU)	87 ± 11	1.1%	3.8%
SSD (CPU, MobileNet backbone)	110 ± 14	0.9%	3.2%

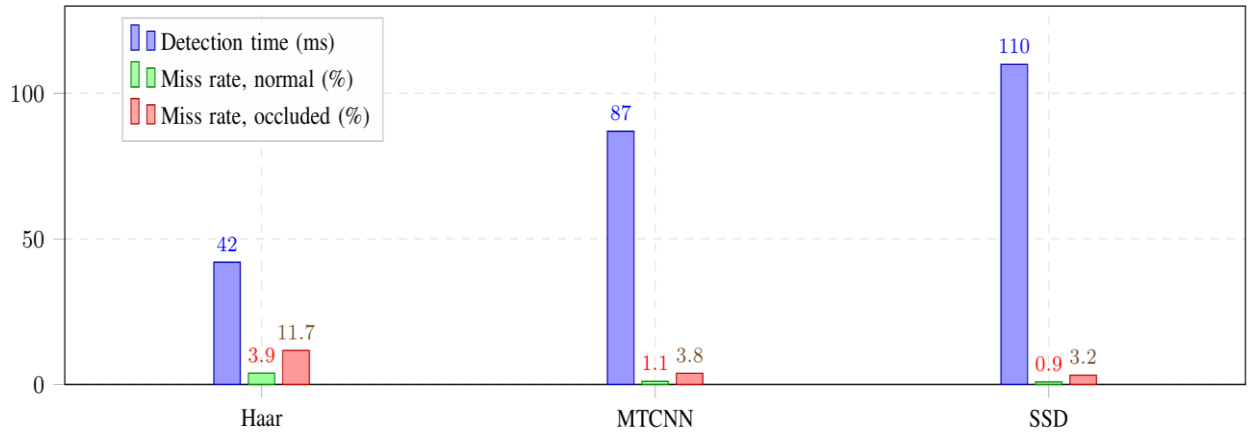

Figure 11. Detector trade-off: stronger detectors roughly halve occluded miss rates but double detection latency on CPU

5.6 Emulated-WAN Results

End-to-end latency, aggregate throughput, and TRR under each WAN profile defined in Section 4.9 are reported in Table 8, and the latency and throughput columns are plotted together in Figure 12. As expected, added RTT and jitter increase end-to-end latency roughly additively, while moderate packet loss (≤0.5%) reduces aggregate throughput by ~5–8% because dropped frames trigger resends at the HTTP layer. Even the 4G-like profile (P3) kept end-to-end latency below 400 ms, with TRR within 1% of the LAN baseline, indicating that the pipeline tolerates typical consumer-wireless impairments. We emphasize that the numbers in Table 8 are emulation results and do not replace a true internet-scale deployment study, which is identified as future work.

5.7 Resource Utilization

Six users accessing the test server in hybrid mode simultaneously resulted in an average CPU usage of 84.6%±2.1%, with CPU usage reaching as high as 92% when DNN inference burst activity occurred. Normalized to the 8-core/16-thread Intel i7-10700 host (see Table 1), this corresponds to an average occupancy of roughly 6.8 of 8 physical cores. The average memory usage was 312±18 MBand consisted primarily of LabVIEW image buffer memory space (approximately 200 MB), the Footprint of the FaceNet Model (approximately 90 MB), and other memory usage. Average network bandwidth was 3.1 ± 0.2 Mbps (six streams at approximately 0.5 Mbps each). The utilization profile relative to resource capacity is summarised in Figure 13. Overall, this information indicates that there is almost no available resources within the overall system and some additional resource availability for an occasional spike in usage.

5.8 Comparison with Related Work

The Table 9 below compares the proposed system with several other examples from previous research studies on the same key characteristics as well as the percent correct (recognition accuracy) of each of those previous studies shown side-by-side in Figure 14. This was done because Lukas *et al.* [27] and Bhattacharya *et al.* [12] were developed using data collected during a single classroom class session. The remote use aspect of those systems was not considered. Likewise, Lei *et al.* [8] and Vilches *et al.* [9] included multiple users remotely accessing their systems via the internet. However, they did not include a method of biometric authentication such as facial recognition. Therefore, the proposed system includes both facial recognition based attendance tracking as well as multiple user access to remote laboratories. Additionally, the hybrid PCA-DNN strategy used for identification in this project offers flexibility regarding the balance between speed and accuracy that may be necessary depending upon the specific application requirements.

Table 8. Emulated-WAN performance at 6 concurrent users

Profile	RTT / jitter / loss	Latency (ms)	Agg. fps	TRR
P1 LAN	<1 / 0 / 0%	242 ± 15	46.8 ± 1.8	96.4%
P2 Campus-WAN	20 / 5 / 0%	276 ± 18	45.9 ± 1.9	96.3%
P3 4G-like	50 / 15 / 0.5%	324 ± 27	43.1 ± 2.4	95.5%
P4 5G-like	10 / 3 / 0.1%	259 ± 16	46.3 ± 1.8	96.4%

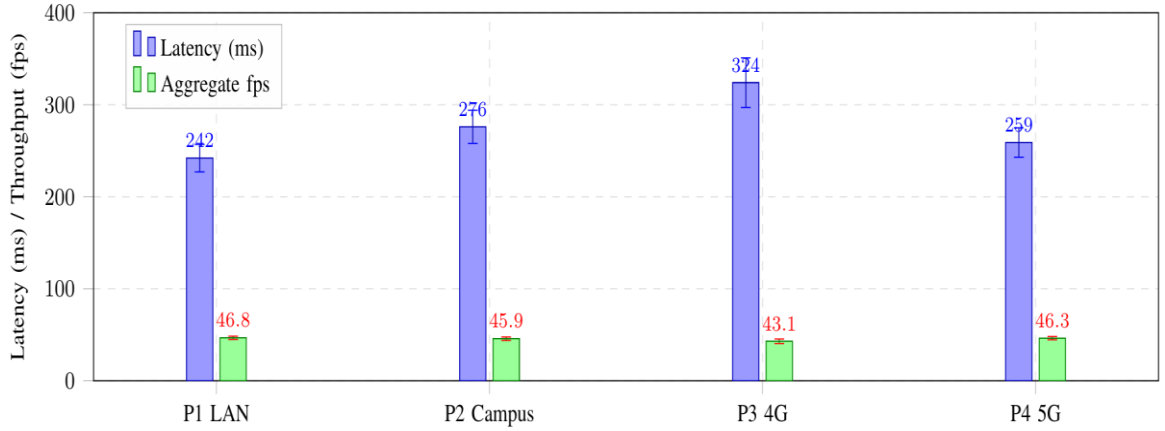

Figure 12. Emulated-WAN latency and aggregate throughput across four network profiles

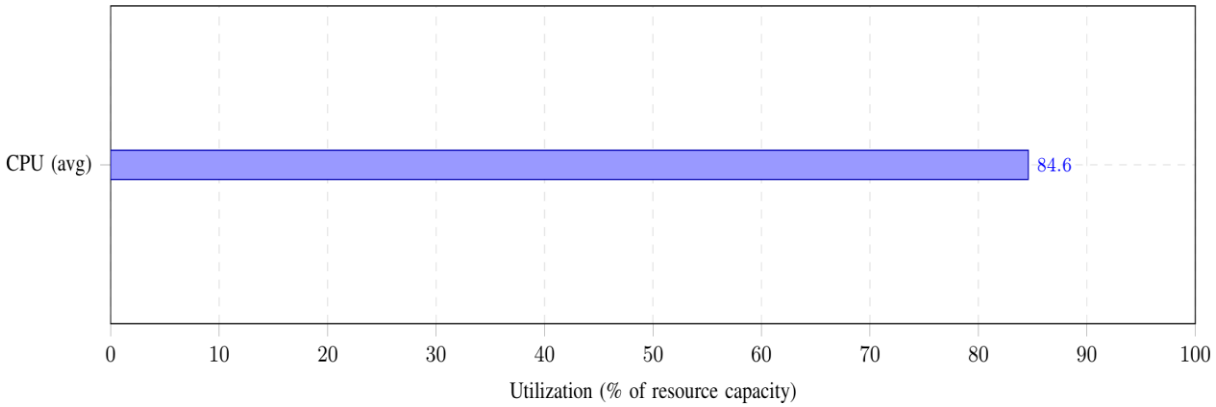

Figure 13. Resource utilization at 6 users (CPU: % of 8 cores; memory: 312 MB of 16 GB; network: 3.1 Mbps of the 100 Mbps LAN segment)

Table 9. Comparison of Related Systems and the Proposed System

Feature / System	Lukas *et al.* [10]	Bhattacharya *et al.* [11]	Lei *et al.* [8]	Proposed 2026
Biometric Mode	Face (PCA/DCT)	Face (CNN)	N/A	Face (Hybrid)
Platform	Desktop (offline)	Embedded (ICALT)	NCSLab	LabVIEW (remote)
Multi-User Support	No	No	Yes (lab)	Yes (parallel)
Pipeline Architecture	No	No	Yes	Yes
Continuous Monitoring	No	Yes	N/A	Yes
Accuracy	~90%	95--98%	---	96--99%
Remote Integration	No	No	Yes (no auth)	Yes (lab+auth)
Latency (6 users)	N/A	N/A	Not reported	~242 ms

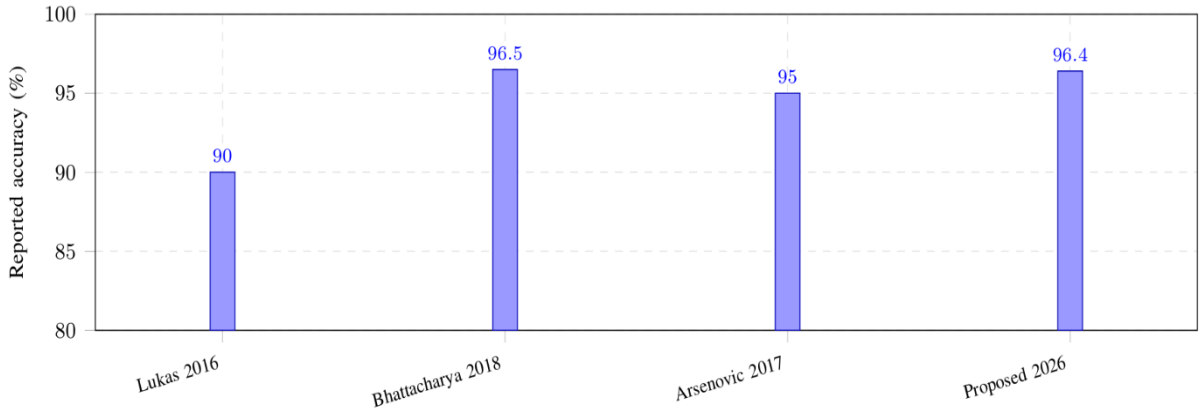

Figure 14. Reported recognition accuracy across related face-based attendance systems

5.9 Scalability (Engineering Projection) and Educational Integration

In the above study we tested our system under conditions of up to 6 users at once. Beyond 6 users the data contained within this section and represented in Figure 15 represents the projection of what will happen in practice - not how things actually perform. The results show a smooth increase in latency with each added user when all were running off one server. Based upon an observed linear relationship between number of users and increased latency (approximately 9 ms/user) we have made an extrapolation of where the system may operate should there be 12 users. Linear extrapolation (dashed line, Figure. 15) suggests a per-frame latency of ~296 ms at 12 users; we do not extrapolate further because the host already averaged 84.6% CPU at 6 users, so the linear model breaks down near saturation, and measurement on larger hardware is future work. Larger cohorts of users (> 30) can be addressed through several potential methods including; distributed LabVIEW server clusters, GPU acceleration of DNN-based inference or reduction in frame rate per user (e.g., verify periodically at a lower rate i.e., 2 frames/sec). As has been indicated these options have yet to be validated experimentally [8, 13]. The attendance module supports integration with Learning Management System (LMS) platforms (i.e., Moodle, Canvas) through either CSV file export or RESTful API. This enables automated

synchronization of grades and analytical reporting regarding student engagement (e.g., session time and confidence values associated with facial recognition). While being used as part of the initial pilots with university students, the system was also utilized as a teaching aid. Students gained first-hand experience with computational thinking concepts such as abstraction (templates for face), decomposition (stages of pipeline) and design of algorithms (hybrid recognition strategy) [28].

5.10 Stress Testing and Failure Modes

The system was tested under an Overload scenario (S1: 12 video streams on 6-user hardware as described in section 4.10). The aggregate through put at maximum utilization was approximately 52 frames-per-second. On average, each user experienced 4.3±0.4 frames-per second. It took about 12 seconds for the S2 → S3 queue to reach full capacity (20-frames); then the drop policy (oldest frame first by user) was continuously invoked resulting in approximately 1.8±0.3% of frames being dropped from the queues. No deadlocks were observed. Crucially, no frames were logged out of order for any user: the sequence-number tag combined with monotonic timestamps was sufficient to preserve temporal correctness of the attendance log even under backlog.

Under the stalled-session scenario (S2, one session blocked for 10 s), the remaining five sessions continued at nominal throughput. When the stalled session resumed, its queue drained in 1.4 ± 0.2 s (as backlogged frames were processed at DNN-burst speed), after which it returned to nominal fps. Attendance-log timestamps for the recovering session were marked with the original capture timestamps (not the processing timestamps), preserving the temporal semantics of "present at time t." Queue-depth evolution across the nominal, overload, and stalled-session regimes is plotted in Figure 16.

5.11 Limitations

Several of the limitations can be identified as follows. First, the accuracy of the system is expected to degrade in low-light and partially occluded conditions which would suggest a need for normalizing light levels in real time and occlusion awareness when detecting faces. Second, the current version of this system does not include a liveness detection mechanism; therefore, it would be possible to use static images of faces to "spoof" the face recognition module, an issue that will have to be addressed using anti-spoofing techniques [29] (blink detection, 3D depth sensing) prior to any production deployments.

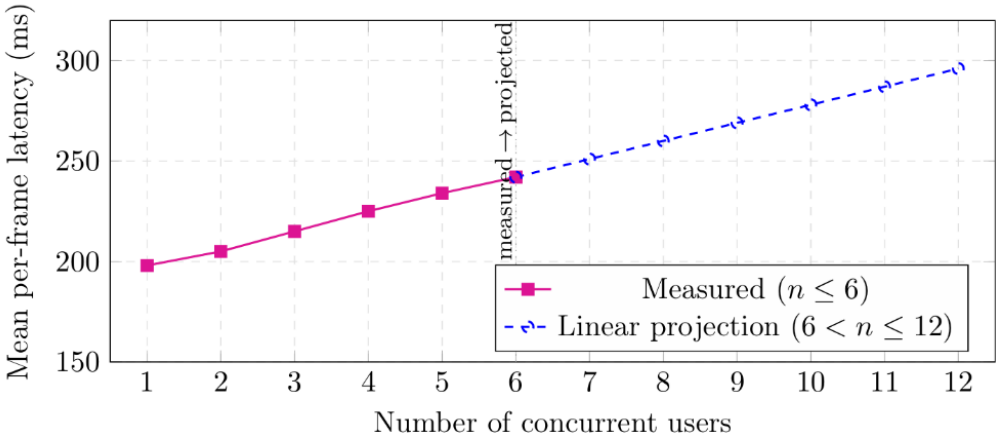

Figure 15. Measured latency (solid, n ≤ 6 users) vs. linear engineering projection (dashed, 6 < n ≤ 12 users).

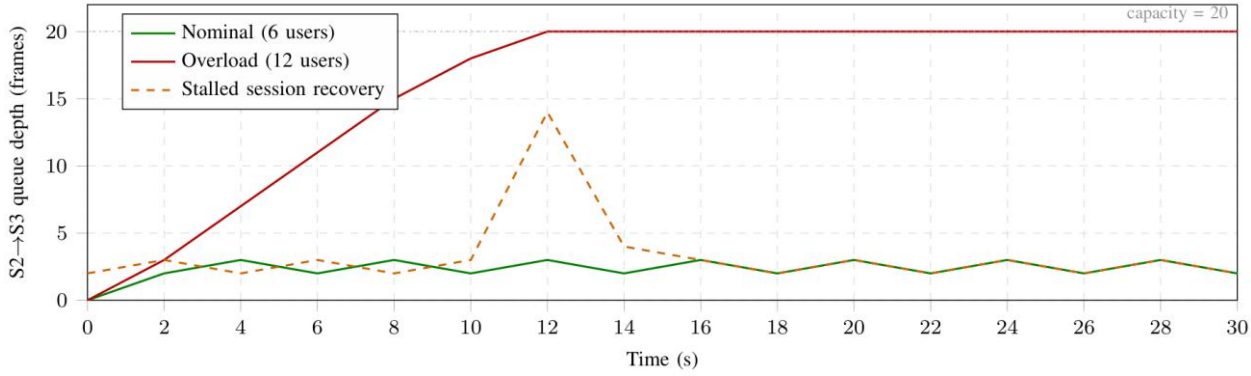

Figure 16. Queue-depth evolution under stress scenarios

Third, there are privacy issues associated with continuous facial monitoring that will require clearly defined institutional policies, informed consent, and data retention safeguards that comply with all relevant regulations (i.e. GDPR); critical perspectives on facial recognition in educational settings [30] and recent studies of student engagement with remote-lab technologies [20] should inform deployment policy. Fourth, this work is a pilot-scale feasibility study. The evaluation was limited to 10 enrolled users (single institution, age 20–25) and 5 impostors. Claims about robust deployment across wider demographics, inter-session variation, or domain shift are not supported by the present data and require multi-site studies with broader age, gender, ethnicity, and lighting diversity. Lastly, although we used the FaceNet model, we did not use any domain-specific fine-tuning; transfer learning from institution-specific datasets may provide improved accuracy. Additionally, the quantitative benchmark was conducted on a controlled LAN (with emulated-WAN augmentation; see Table 8 and Figure 12); a full internet-deployment study with geographically distributed clients, cross-network session management, and unstable bandwidth remains as future work.

6. Conclusion & Future Scope

This study presented a LabVIEW-based remote laboratory system for automated attendance monitoring using facial recognition in a concurrent multi-user setting. The main contribution is not a new recognition algorithm, but a practical system-level integration of remote laboratory access, continuous biometric verification, and pipeline parallelism for real-time operation. The proposed four-stage pipeline enabled stable multi-user processing while keeping end-to-end delay within an interactive range. In the six-user benchmark, the pipelined architecture substantially outperformed the matched sequential baseline in both latency and throughput, showing that concurrency can be handled efficiently on commodity multi-core hardware. The hybrid recognition strategy, which combines fast PCA-based verification with periodic FaceNet-based confirmation, improved robustness over PCA alone while preserving real-time performance. The results also showed strong recognition performance under normal lighting and acceptable resilience under more challenging conditions, supporting the feasibility of the approach for pilot-scale educational deployment. At the same time, the work should be interpreted within its experimental limits. The evaluation was conducted under controlled laboratory conditions with a small subject pool, and the WAN study was based on emulation rather than a geographically distributed deployment. Future work should therefore validate the system with larger and more diverse user groups, true internet-based remote sessions, stronger face-detection models, and expanded privacy, security, and LMS integration measures.

References

[1] R. Heradio, L. de la Torre, S. Dormido, Virtual and Remote Labs in Control Education: A Survey. Annual Reviews in Control, 42, (2016) 1–10. https://doi.org/10.1016/j.arcontrol.2016.08.001

[2] M. Tawfik, C. Salzmann, D. Gillet, D. Lowe, H. Saliah-Hassane, E. Sancristobal, M. Castro, Laboratory as a Service (LaaS): A Novel Paradigm for Developing and Implementing Modular Remote Laboratories. International Journal of Online Engineering (iJOE), 10(4), (2014) 13–21. https://doi.org/10.3991/ijoe.v10i4.3654

[3] M. Kaluz, L. Cirka, R. Valo, and M. Fikar, ArPi Lab: A Low-Cost Remote Laboratory for Control Education. IFAC Proceedings Volumes, 47(3), (2014) 9057–9062. https://doi.org/10.3182/20140824-6-ZA-1003.00963

[4] B. Letowski, C. Lavayssière, B. Larroque, M. Schröder, F. Luthon, A Fully Open Source Remote Laboratory for Practical Learning. Electronics, 9(11), (2020) 1832. https://doi.org/10.3390/electronics9111832

[5] B. Kizilkaya, G. Zhao, Y. A. Sambo, L. Li, M.A. Imran, 5G-Enabled Education 4.0: Enabling Technologies, Challenges, and Solutions. IEEE Access, 9, (2021) 166962–166969. https://doi.org/10.1109/ACCESS.2021.3136361

[6] A. Barrios, S. Panche, M. Duque, V.H. Grisales, F. Prieto, J.L. Villa, P. Chevrel, M. Canu, A Multi-User Remote Academic Laboratory System. Computers & Education, 62, (2013) 111–122. https://doi.org/10.1016/j.compedu.2012.10.011

[7] S. Farah, A. Benachenhou, G. Neveux, D. Barataud, G. Andrieu, T. Fredon, Multi-User and real-time flexible remote Laboratory Architecture for Collaborative and Cooperative Pedagogical Scenarios. International Journal of Online Engineering (iJOE), 12(4), (2016) 33–36. https://doi.org/10.3991/ijoe.v12i04.5097

[8] Z. Lei, H. Zhou, W. Hu, G.P. Liu, Concurrent Experimentation in NCSLab: A Scalable Approach for Online Laboratories. Future Generation Computer Systems, 148, (2023) 139–149. https://doi.org/10.1016/j.future.2023.05.014

[9] M. Vilches, H. Vargas, L. De la Torre, R. Heradio, Scalable Hybrid Laboratories: Application in industrial automation. Results in Engineering, 26, (2025) 105342. https://doi.org/10.1016/j.rineng.2025.105342

[10] M.A. Hidayat, H.M. Simalango, Students Attendance System and Notification of College Subject Schedule Based on Classroom Using

IBeacon. In Proceedings of the 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE), IEEE, Yogyakarta, Indonesia, (2018) 253–258. https://doi.org/10.1109/ICITISEE.2018.8720948

[11] M. Turk, A. Pentland, Eigenfaces for Recognition. Journal of Cognitive Neuroscience, 3(1), (1991) 71–86. https://doi.org/10.1162/jocn.1991.3.1.71

[12] S. Bhattacharya, G.S. Nainala, P. Das, A. Routray, (2018) Smart Attendance Monitoring System (SAMS): A Face Recognition based Attendance System for Classroom Environment. Proceedings of the IEEE 18th International Conference on Advanced Learning Technologies (ICALT), Mumbai, India. https://doi.org/10.1109/ICALT.2018.00090

[13] M. Arsenovic, S. Sladojevic, A. Anderla, D. Stefanovic, (2017) FaceTime – Deep learning based Face Recognition Attendance System. Proceedings of the IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY), IEEE, Serbia. https://doi.org/10.1109/SISY.2017.8080587

[14] K. Puthea, R. Hartanto, R. Hidayat, A Review Paper on Attendance Marking System based on Face Recognition. In Proceedings of the 2nd International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE), IEEE, Yogyakarta, Indonesia, (2017) 304–309. https://doi.org/10.1109/ICITISEE.2017.8285517

[15] M. Sharif, F. Naz, M. Yasmin, M.A. Shahid, A. Rehman, Face Recognition: A Survey. Journal of Engineering Science and Technology Review, 10(2), (2017) 166–177. https://doi.org/10.25103/jestr.102.20

[16] S.M. Bah, F. Ming, an Improved Face Recognition Algorithm and its Application in Attendance Management System. Array, 5, (2020) 100014. https://doi.org/10.1016/j.array.2019.100014

[17] K. Okokpujie, E. Noma-Osaghae, S. John, K.A. Grace, I. Okokpujie, (2017). A Face Recognition Attendance System with GSM notification. In Proceedings of the IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON), Owerri, Nigeria. https://doi.org/10.1109/NIGERCON.2017.8281895

[18] S. Poornima, N. Sripriya, B. Vijayalakshmi, P. Vishnupriya, (2017). Attendance Monitoring System using Facial Recognition with Audio Output and Gender Classification. Proceedings of the International Conference on Computing, Communication and Signal Processing (ICCCSP), India. https://doi.org/10.1109/ICCCSP.2017.7944103

[19] O. Vanegas-Guillén, P. Parra-Rosero, J.M. Muñoz-Antón, J. Zumba-Gamboa, C. Dillon, Remote Labs Meet Computational Notebooks: An Architecture for Simplifying the Workflow of remote educational experiments. IEEE Access, 11, (2023) 132496–132515. https://doi.org/10.1109/ACCESS.2023.3336287

[20] A. Van den Beemt, S. Groothuijsen, L. Ozkan, W. Hendrix, Remote Labs in Higher Engineering Education: Engaging Students with Active Learning Pedagogy. Journal of Computing in Higher Education, 35(2), (2023) 320-340. https://doi.org/10.1007/s12528-022-09331-4

[21] F. Schroff, D. Kalenichenko, J. Philbin, FaceNet: A Unified Embedding for Face Recognition and Clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, (2015) 815–823. https://doi.org/10.1109/CVPR.2015.7298682

[22] P. Viola, M.J. Jones, Robust Real-Time Face Detection. International Journal of Computer Vision, 57, (2), (2024) 137–154. https://doi.org/10.1023/B:VISI.0000013087.49260.fb

[23] M. Wang, W. Deng, Deep Face Recognition: A survey. Neurocomputing, 429, (2021) 215–244. https://doi.org/10.1016/j.neucom.2020.10.081

[24] W. Farag, An Innovative Remote-Lab Framework for Educational Experimentation," International Journal of Online Engineering (iJOE), 13(2), (2017) 68–86. https://doi.org/10.3991/ijoe.v13i02.6609

[25] K. Zhang, Z. Zhang, Z. Li, Y. Qiao, Joint Face Detection and Alignment using Multitask Cascaded Convolutional Networks. IEEE Signal Processing Letters, IEEE, 23(10), (2016) 1499–1503. https://doi.org/10.1109/LSP.2016.2603342

[26] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu, A.C. Berg, SSD: Single Shot Multibox Detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, Netherlands, Cham: Springer International Publishing, 9905, (2016) 21–37.https://doi.org/10.1007/978-3-319-46448-0_2

[27] S. Lukas, A. R. Mitra, R. I. Desanti, D. Krisnadi, (2016) Student Attendance System in Classroom using Face Recognition Technique, in Proceedings of the International Conference on Information and Communication Technology

Convergence (ICTC), Jeju, South Korea,1032-1035. https://doi.org/10.1109/ICTC.2016.7763360

[28] J.M. Wing, Computational Thinking. Communications of the ACM, 49(3), (2006) 33–35. https://dl.acm.org/doi/fullHtml/10.1145/1118178.1118215

[29] Z. Yu, Y. Qin, X. Li, C. Zhao, Z. Lei, G. Zhao, Deep Learning for Face Anti-Spoofing: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(5), (2023) 5609–5631. https://doi.org/10.1109/TPAMI.2022.3215850

[30] M. Andrejevic, N. Selwyn, Facial Recognition Technology in schools: Critical Questions and Concerns. Learning, Media and Technology, 45(2), (2020) 115–128. https://doi.org/10.1080/17439884.2020.1686014

Authors Contribution Statement

Niket Amoda: Conceptualization, Methodology, Validation, Formal analysis, Data curation, Writing - Original Draft, Visualization. Lochan Jolly: Validation, Investigation, Writing - Review & Editing, Supervision, Project Administration. Arpit Rawankar: Conceptualization, Software, Validation, Resources, Writing - Review & Editing. All the authors have read and agreed to the published version of the manuscript.

Funding

The authors declare that no funds, grants or any other support were received during the preparation of this manuscript.

Competing Interests

The authors declare that there are no conflicts of interest regarding the publication of this manuscript.

Data Availability

The data supporting the findings of this study can be obtained from the corresponding author upon reasonable request.

Has this article screened for similarity?

Yes