

VISION PRO SURVEILLANCE

Anonymous Author(s)

Abstract

Recent advances in artificial intelligence (AI) and Internet of Things (IoT) technologies have led to the replacement of traditional locking mechanisms with intelligent, computer vision-based access systems. These systems grant entry exclusively when a registered user is recognized by the camera, thereby automating the unlocking process. However, such systems remain susceptible to spoofing attacks, including the use of 2D photographs or video replays, which compromise security.

This paper presents a smart locking system that integrates face recognition with liveness detection to provide secure authentication and prevent spoofing. The system utilizes two ESP-32 camera modules for image acquisition and employs a Raspberry Pi 5 as the primary processing unit. The liveness detection module leverages epipolar geometry to estimate depth between key facial landmarks, preventing spoofing. Upon successful liveness verification, the face recognition module, featuring YuNet for face detection and SFace for recognition, authenticates the user. Once the identity is confirmed, the locking mechanism is disengaged. The lock is actuated through coordinated control of an Arduino, relay, and 12V DC power supply. In addition, an interactive web interface facilitates remote user registration, real-time monitoring, administrative privileges, including user approval and audit logs. The proposed solution offers a cost-effective, robust, and secure smart lock, demonstrating the seamless integration of advanced AI with embedded systems.

Keywords

Computer Vision, Liveliness Detection, Smart Access, Artificial Intelligence, IoT

1 Introduction

Smart surveillance systems [8] play a crucial role in home security by enabling convenient authorized access. However, traditional methods such as PIN codes, RFID cards, and fingerprints are increasingly vulnerable to forgery, cloning, and physical tampering [14]. While existing face recognition algorithms offer a more seamless approach, they remain susceptible to spoofing attacks using high-resolution photos or videos and may struggle in different lighting conditions. [2]. Furthermore, modern solutions utilize complex machine learning models that improve accuracy but require high computational resources, which limits their use on lightweight platforms like Raspberry Pi and often difficult to scale [9]. These advanced systems rely on cloud infrastructure for database storage and thus, adds to the financial cost of the entire system..

To overcome these challenges, the proposed system confirms liveness using epipolar geometry and facial landmarks (eyes and nose) to find depth—a lightweight yet effective measure to thwart 2-D spoofing [13]. High-speed face detection and recognition are achieved using YuNet and SFace, specifically optimized for resource-constrained environments [9, 11]. An Arduino to control a solenoid lock, while a web-based dashboard provides live monitoring and remote admin approval [1]. Overall, this pipeline achieves a balance between security, accuracy, and performance.

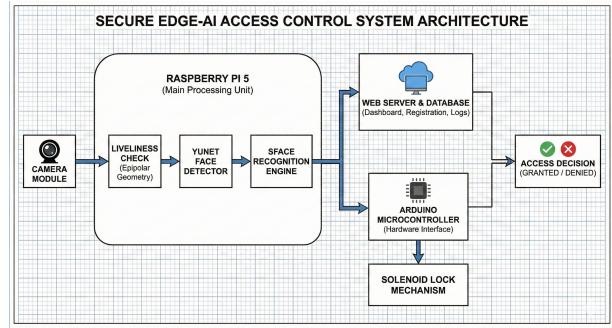


Figure 1: System Block Diagram: An overview of the proposed smart lock, including liveness detection, face recognition, and hardware.

2 Related Work

This section reviews existing systems and the computer vision algorithms that underpin our proposed solution.

2.1 Non-Biometric Smart Locks

Vongchumyen et al. (2017) proposed a Web-based system and Aswini D. et al. (2021) proposed RFID-OTP systems. While these solve the “lost key” problem, they fail to offer true convenience. Whether using a phone app or scanning a tag, the user is required to manually interact with a device, negating the “hands-free” benefit of home automation.

2.2 Monocular Biometric Systems

To improve convenience, facial recognition was adopted, but single-camera systems face a trade-off between security and speed. Cloud-based solutions (Maheshwari & Nalini, 2017) offer accuracy but suffer from latency and internet dependency. Alternatively, local offline systems (Elechi et al., 2022) often lack the robustness to distinguish real faces from photos. To counter this, Dasare (2021) introduced “active liveness” (hand gestures), which unfortunately adds user friction. A robust system must be both passive (no gestures) and local (offline).

2.3 Stereo Vision and The Research Gap

To achieve passive, secure liveness detection, our work leverages Stereo Vision for 3D depth extraction. Our methodology relies on Zhang’s (1998) standard for camera calibration to ensure geometric accuracy and Hartley & Sturm’s (1997) triangulation framework to reconstruct facial landmarks. Crucially, we utilize Hirschmüller’s (2005) Semi-Global Matching (SGM) algorithm. By applying epipolar geometry constraints, SGM produces dense disparity maps efficiently on embedded hardware. While recent “Foundation Models” like Wen et al. (2025) offer high accuracy, they are computationally prohibitive for microcontrollers.

117 3 Technical Prerequisites

118 This section outlines the mathematical foundations of stereo vision,
 119 calibration, and metric learning required for the proposed system.
 120

121 3.1 Stereo Calibration & Epipolar Geometry

122 To reconstruct 3D information, we model the camera using the
 123 Pinhole Model. First, we estimate the **Intrinsic Matrix** (K) using
 124 the flexible calibration technique [15]. This accounts for focal length
 125 (f_x, f_y) and optical center (c_x, c_y) from low-cost sensors:
 126

$$127 \quad K = \begin{bmatrix} f_x & 0 & c_x \\ 128 \quad 0 & f_y & c_y \\ 129 \quad 0 & 0 & 1 \end{bmatrix} \quad (1)$$

130 For stereo consistency, we utilize the **Epipolar Constraint** and
 131 triangulation framework [5]. Given a point projected onto the left
 132 (\mathbf{p}_L) and right (\mathbf{p}_R) image planes, their relationship is governed by
 133 the Fundamental Matrix (\mathbf{F}):
 134

$$135 \quad \mathbf{p}_R^T \cdot \mathbf{F} \cdot \mathbf{p}_L = 0 \quad (2)$$

137 3.2 Disparity & Depth Estimation

138 To compute depth, we first calculate pixel disparity (d) using the
 139 **Semi-Global Block Matching (SGBM)** algorithm [6]. Unlike simple
 140 pixel matching, SGBM enforces a global smoothness (WLS)
 141 constraint to reduce noise on textureless surfaces like skin. The
 142 depth Z is inversely proportional to disparity [5]:
 143

$$144 \quad Z = \frac{f \cdot B}{d} \quad (3)$$

145 Where Z is the distance (depth) of the object from the camera, f
 146 is the focal length, B is the baseline distance between cameras, and
 147 d is the Disparity ($x_L - x_R$). This relationship allows the system to
 148 distinguish between a real face (varying Z) and a spoof photograph
 149 (constant Z).
 150

152 3.3 Deep Metric Learning

153 Identity verification utilizes a neural network trained with **Triplet**
 154 **Loss** [10] to map facial features into a 128-dimensional embedding
 155 space. Verification is performed by calculating the **Euclidean**
 156 **Distance** between the live embedding (\mathbf{p}) and stored embedding
 157 (\mathbf{q}):
 158

$$159 \quad d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{128} (p_i - q_i)^2} \quad (4)$$

160 If this distance is below a learned threshold (e.g., $\tau < 0.6$), the
 161 identity is confirmed.
 162

165 4 Methodology

166 4.1 System Overview

167 The proposed system comprises a sequential processing pipeline
 168 designed for real-time secure access control. The workflow begins
 169 with stereo vision for depth perception and anti-spoofing, followed
 170 by deep learning-based face recognition and a control interface
 171 for hardware actuation. The system captures synchronized video
 172 streams from two cameras, applies epipolar rectification and depth
 173 estimation to verify facial liveness, and subsequently performs iden-
 174 tity recognition using learned facial embeddings. After successful
 175 authentication, an Arduino-controlled solenoid lock is actuated to
 176 grant physical access.
 177

178 4.2 System Design and Implementation

- 179 • **Epipolar Geometry:** The stereo vision module follows
 180 the calibrated stereo camera model described in Eq. 1. In-
 181trinsic and extrinsic parameters are obtained offline using
 182 a standard chessboard calibration procedure. Using these
 183 parameters, epipolar rectification is applied to align corre-
 184 sponding scan lines, as in Eq. 2. Disparity is then computed
 185 along horizontal epipolar lines using the Semi-Global Block
 186 Matching (SGBM) algorithm, producing a dense disparity
 187 map (Fig.2b) suitable for real-time processing.
 188
- 189 • **Depth Estimation:** The depth (Z) of any pixel is inversely
 190 proportional to its disparity (d). The relationship is defined
 191 in Eq.3 The formulation allows reliable estimation of rela-
 192 tive facial depth using the calibrated stereo configuration.
 193
- 194 • **Facial Landmark Localization:** Facial landmarks are de-
 195 tected using the YuNet model, which identifies both bound-
 196 ing boxes and precise coordinates for key facial features.
 197 These landmarks are detected in the rectified image space
 198 and mapped to the corresponding coordinates in the disper-
 199 sity map. This mapping enables direct association between
 200 facial landmarks and their estimated depth values.
 201
- 202 • **Liveness Detection:** A human face is a 3D object where
 203 the nose is closer to the camera than eyes. Conversely, a 2D
 204 spoof (photo) is planar, meaning the depths of the nose and
 205 eyes are approximately equal. Depth values are extracted at
 206 the landmark positions corresponding to the left eye, right
 207 eye, and nose tip.
 208

$$209 \quad \Delta Z = \frac{Z_{\text{left_eye}} + Z_{\text{right_eye}}}{2} - Z_{\text{nose}} \quad (5)$$

210 where $Z_{\text{left_eye}}$, $Z_{\text{right_eye}}$, and Z_{nose} denote the estimated
 211 depths at the corresponding facial landmark. The liveness
 212 condition is satisfied if the resulting geometric protrusion
 213 exceeds a predefined threshold. To ensure robustness against
 214 noise and transient estimation errors, liveness verification
 215 is performed over multiple consecutive frames.
 216

- 217 • **Face Recognition:** After successful liveness verification,
 218 the system employs a deep learning pipeline consisting of
 219 two models: YuNet for face detection and SFace for face
 220 recognition. To ensure robustness against pose variations,
 221 Affine transformation is applied using the facial landmarks
 222 provided by YuNet. This transformation aligns the detected
 223 face by correcting rotational offsets before it is passed to
 224 SFace. SFace converts the aligned facial image into a com-
 225 pact 128-dimensional embedding vector that represents the
 226 user's unique facial characteristics.
 227

228 Identity matching is performed using cosine similarity:
 229

$$230 \quad S = \frac{f \cdot g}{\|f\| \|g\|} \quad (6)$$

231 where f represents the extracted facial feature vector and g
 232 denotes a stored enrollment embedding. Access is granted
 233



Figure 2: Liveness (3D Reconstruction and Disparity) and Integrated System Architecture.

only if the similarity score exceeds a threshold, thereby identifying the user with high confidence.

- **Hardware Actuation:** The system effectively governs transitions between liveness verification, recognition, and access states. Upon successful authentication, a command signal is transmitted via serial communication to an Arduino, which actuates a solenoid lock. A cooldown interval is enforced to prevent repeated triggering.

5 Experiments and Evaluation

5.1 Dataset

The initial dataset was collected using a laptop web camera and consists of three subfolders, each corresponding to an individual. Each subfolder contains 200 high-quality images, extracted from a 60-second video in which participants performed various head movements and facial expressions. Images are temporarily stored locally on the Raspberry Pi until they are converted to vector embeddings, ensuring protection from cyber attacks. A Test Dataset consisting of 150 images was used for evaluation.

Each image is converted into a 128-dimensional vector. The model generates vectors for all training images and averages them to create a 'mean' embedding that represents that person. These embeddings are saved locally in an 'embeddings.pkl' file on the Raspberry Pi 5. The file is used for real-time recognition and model evaluation.

5.2 Baseline

In this section, we are conducting a comparative analysis of our model alongside several others. The evaluation framework relies on: **True Acceptance Rate (TAR):** system correctly verifies a legitimate user who is enrolled in the database, **False Acceptance Rate (FAR):** system incorrectly verifies an unauthorized impostor as a legitimate user **False Rejection Rate (FRR):** system incorrectly rejects a legitimate user, failing to recognize their enrolled identity, **True Rejection Rate (TRR):** system correctly rejects an unauthorized impostor, denying them access, and **overall Accuracy.** Our examination includes traditional approaches such as the Haar Cascade Classifier [12] and the Histogram of Oriented Gradients in conjunction with support vector machine (SVM) [3]. Additionally, we have evaluated deep learning-based models, including the Dlib (OpenCV) [7] and the Insightface (Arcface) [4].

5.3 Quantitative Analysis

The Haar Cascade model demonstrated a very low accuracy (0.08) and a high FRR (0.87). Due to its basic feature-based approach, which is highly susceptible to changes in lighting conditions and non-frontal facial orientations. Similarly, the HOG + SVM was unable to correctly identify most of authorized users (TAR 0.14), likely due to the inflexibility of HOG descriptors when faced with the varied poses and facial expressions present in the test set.

Among the deep learning approaches, Dlib (OpenCV) achieved improved recognition performance (TAR 0.83), but it was accompanied by a significant security concern: (FAR 0.55). This indicates lack of discriminative ability to differentiate between users and impostors. Although InsightFace demonstrated high accuracy (0.94), it was not selected due to hardware constraints. Its computationally intensive architecture exceeded the capabilities of the Raspberry Pi 5 and rendered it unsuitable for edge deployment.

Table 1: Model Evaluation

Models	TAR	FAR	FRR	TRR	Accuracy
Haar Cascade	0.13	0.70	0.87	0.3	0.08
HOG + SVM	0.14	0.13	0.86	0.87	0.52
Dlib (OpenCV)	0.83	0.55	0.0	0.45	0.60
Insightface*	0.90	0.0	0.10	1.0	0.94
YuNet + SFace (Ours)	1.0	0.0	0.0	1.0	1.0

* indicates that the training/testing was conducted on a PC instead of the Raspberry Pi 5, due to the computational intensity.

Ultimately, YuNet + SFace provided the optimal solution. It achieved perfect metrics (1.0 Accuracy, 0.0 FAR), demonstrating the ideal balance of computational efficiency and high-precision security required for the final system.

To further validate our proposed system, we conducted an ablation study to isolate the impact of the detector (YuNet) and the recognizer (SFace) independently. We tested four configurations: YuNet + LBPH, Haar Cascade + SFace, Retinaface + Arcface, and the proposed YuNet + SFace. We first assessed the texture-based recognizer by evaluating the YuNet + LBPH. Despite the integration of a robust detector, system accuracy decreased to 0.45, accompanied by a high FAR (0.40). These results indicate Local Binary Pattern

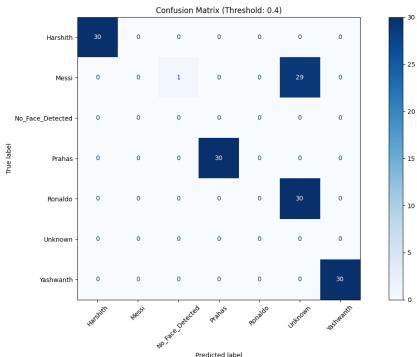


Figure 3: Confusion Matrix for YuNet + SFace: Demonstrating perfect accuracy and low error.

Table 2: Ablation Study

Configuration	Accuracy	FAR	Latency	Comments
YuNet + LBPH	0.85	0.80	18 ms	Good cropping cannot fix LBPH's texture limitations.
Haar Cascade + SFace	0.68	0.05	14 ms	Good recognizer limited by Haar's inability to find faces.
Retinaface + Arcface *	1.0	0.0	180 ms	Accurate but impractical for real-time use.
YuNet + SFace (ours)	1.0	0.0	15 ms	Optimal , perfect balance of speed, size, and security.

Histograms (LBPH) lack the necessary feature extraction depth for secure identification.

In contrast, we explored a less robust detector by pairing Haar Cascade with SFace. Although this setup achieved low latency (14 ms), the accuracy was constrained to 0.68. The system performance was hindered by the Haar Cascade detector's frequent failure to detect faces, resulting in a significant bottleneck. These findings confirm that traditional detectors are inadequate for providing reliably consistent input to the recognition pipeline.

The Retinaface + Arcface configuration matched our model's perfect accuracy (1.0) and FAR (0.0), it suffered from a massive latency penalty, taking 180 ms per frame. This is over 10 times slower than our proposed solution, making it impractical for real-time edge applications where rapid response is critical.

The YuNet + SFace configuration emerged as the optimal architecture. It achieved perfect accuracy (1.0) and security (0.0 FAR) and also maintained an ultra-low latency of 15 ms. This confirms that YuNet + SFace successfully bridges the gap between high-performance security and real-time efficiency, validating its selection as the core engine for our embedded system.

5.4 Qualitative Analysis

Table 3 shows expected depth differences for real human faces and planar surfaces. These theoretical values form the basis for liveness detection.

Table 3: Expected Depth Differences

Classification	Δz (m)
Average Adult Face	0.020–0.030
Average Child Face	0.015–0.020
Photo / Flat Surface	≈ 0.000

Table 4: Liveness Decision Parameters

Parameter	Value
Liveness Minimum (ΔZ_{\min})	0.015 m
Liveness Maximum (ΔZ_{\max})	0.080 m
Consensus Frames (N_c)	3

Table 4 lists the Liveness Decision parameters. The depth thresholds define valid liveness ranges, and the consensus frame count ensures stability across multiple frames.

Table 5: Depth Measurements and Liveness Results

Test Scenario	Z_{eyes} (m)	Z_{nose} (m)	ΔZ (m)	Result
Real Human Subject	0.430	0.393	0.037	PASS
Printed Photo	0.338	0.335	0.003	FAIL
Mobile Phone Screen	0.477	0.356	0.121	FAIL

Table 5 compares measured depth differences for real faces and spoof attacks. Human faces fall within the liveness range and pass the test, while printed photos and phone screens fail due to inconsistent depth variation. Mobile phone screens show higher depth differences than expected for a planar surface because reflections, and display curvature introduce erroneous stereo disparities, which the system interprets as depth variation, resulting in artificially inflated Z values. Despite this, the values remain outside the valid liveness range, so the spoof is correctly rejected.

6 Conclusion

This research presents an efficient real-time face recognition system for edge computing on the Raspberry Pi 5, featuring epipolar geometry-based liveness detection and a web management dashboard. The YuNet + SFace pipeline outperformed other models, demonstrating that high-precision biometric authentication is feasible on affordable embedded hardware.

7 Acknowledgment

We would like to express our sincere gratitude to the Technical University of Applied Sciences Würzburg-Schweinfurt (THWS) for providing the academic environment, infrastructure, and resources necessary to carry out this project. We also extend our heartfelt thanks to Prof. Dr. Andreas Lehrmann for his valuable guidance, continuous support, and insightful feedback throughout the development of this work. His expertise and encouragement played a crucial role for the successful implementation.

465 References

- 466 [1] Arduino Expert. 2025. IoT Smart Door Lock Using ESP32, Solenoid Lock & Blynk
467 App. Online: <https://arduinoexpert.com/iot-smart-door-lock-using-esp32-and-blynk/>.
468
- 469 [2] S. Bhattacharjee. 2018. Spoofing Deep Face Recognition with Custom Silicone
470 Masks. Conference paper (ICB 2018) – online reference when available.
471
- 472 [3] N. Dalal and B. Triggs. 2005. Histograms of Oriented Gradients for Human
473 Detection. CVPR 2005.
474
- 475 [4] J. Deng et al. 2019. ArcFace: Additive Angular Margin Loss for Deep Face
476 Recognition. CVPR 2019.
477
- 478 [5] Richard I. Hartley and Peter Sturm. 1997. Triangulation. *Computer Vision and
479 Image Understanding* 68, 2 (1997), 146–157. doi:10.1006/cviu.1997.0547
480
- 481 [6] Heiko Hirschmüller. 2008. Stereo processing by semiglobal matching and mutual
482 information. *IEEE Transactions on pattern analysis and machine intelligence* 30, 2
483 (2008), 328–341.
484
- 485 [7] D. E. King. 2009. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine
486 Learning Research* (software/library).
487
- 488 [8] Mordor Intelligence. 2026. Smart Home Video Surveillance Market Size & Growth
489 Analysis Report. Online: [https://www.mordorintelligence.com/industry-reports/
491 smart-home-video-surveillance-market](https://www.mordorintelligence.com/industry-reports/
490 smart-home-video-surveillance-market).
492
- 493 [9] OpenCV. 2023. YuNet: A tiny millisecond-level face detector. Online: https://huggingface.co/opencv/face_detection_yunet.
494
- 495 [10] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A
496 unified embedding for face recognition and clustering. In *2015 IEEE Conference
497 on Computer Vision and Pattern Recognition (CVPR)*. 815–823. doi:10.1109/CVPR.
498 2015.7298682
499
- 500 [11] S. Serengil. 2024. DeepFace: A Lightweight Face Recognition and Facial Attribute
501 Analysis Framework. GitHub repository: <https://github.com/serengil/deepface>.
502
- 503 [12] P. Viola and M. Jones. 2001. Rapid object detection using a boosted cascade of
504 simple features. CVPR 2001 (classic paper).
505
- 506 [13] L. Wang. 2016. Liveness Detection Using Texture and 3D Structure Analysis.
507 Online: <https://www.researchgate.net/publication/308383798>.
508
- 509 [14] A. Zainuddin. 2024. Vulnerability Analysis of Smart Lock Using NIST SP 800-115
510 Method. Online: <https://www.researchgate.net/publication/396254787>.
511
- 512 [15] Zhengyou Zhang. 2000. A flexible new technique for camera calibration. *IEEE
513 Transactions on Pattern Analysis and Machine Intelligence* 22, 11 (2000), 1330–1334.
514 doi:10.1109/34.888718
515
- 516
- 517
- 518
- 519
- 520
- 521
- 522
- 523
- 524
- 525
- 526
- 527
- 528
- 529
- 530
- 531
- 532
- 533
- 534
- 535
- 536
- 537
- 538
- 539
- 540
- 541
- 542
- 543
- 544
- 545
- 546
- 547
- 548
- 549
- 550
- 551
- 552
- 553
- 554
- 555
- 556
- 557
- 558
- 559
- 560
- 561
- 562
- 563
- 564
- 565
- 566
- 567
- 568
- 569
- 570
- 571
- 572
- 573
- 574
- 575
- 576
- 577
- 578
- 579
- 580