Stereoscopic Liveness Detection and Face Authentication for Secure Smart Locks

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 user is recognized by the camera, thereby automating the unlocking process. However, such systems remain susceptible to spoofing attacks, including the use of photos or videos, which compromise security.

This paper presents a smart locking system that integrates face recognition with liveness detection to provide secure authentication and mitigate spoofing attacks. The proposed architecture employs a stereo vision framework to estimate facial depth via epipolar geometry, effectively distinguishing between actual faces and presentation attacks. Upon successful liveness verification, the face recognition module, authenticates the user. Once the identity is confirmed, the locking mechanism is disengaged. A web interface facilitates remote user management and real-time monitoring. The experimental evaluations demonstrate that the proposed solution achieves high accuracy and an ultra-low latency, validating its feasibility for real-time, robust, and secure access control.

Keywords

computer vision, liveness detection, smart access, artificial intelligence, iot.

1 Introduction

Smart surveillance systems [14] play a crucial role in home security allowing authorized access. However, traditional methods such as PIN codes, RFID cards, and fingerprints are increasingly vulnerable to forgery, and tampering [20]. 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 [3]. Furthermore, modern solutions utilize complex machine learning models that improve accuracy but require high computational resources, limiting their use on lightweight platforms like the Raspberry Pi. These advanced systems also rely on cloud infrastructure for database storage [13], which adds to the overall financial cost of the system.

To overcome these challenges, we propose liveness using epipolar geometry [8] and facial landmarks (eyes and nose) to find depth, a lightweight yet effective measure to thwart 2-D spoofing (Fig.1). High-speed face detection and recognition are achieved using YuNet [19] and SFace [22], specifically optimized for resource-constrained environments [15]. An Arduino controls a solenoid lock, while a web-based dashboard provides live monitoring and remote admin approval [1]. The proposed pipeline achieves a balance between security, accuracy, and performance.

2 Related Work

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

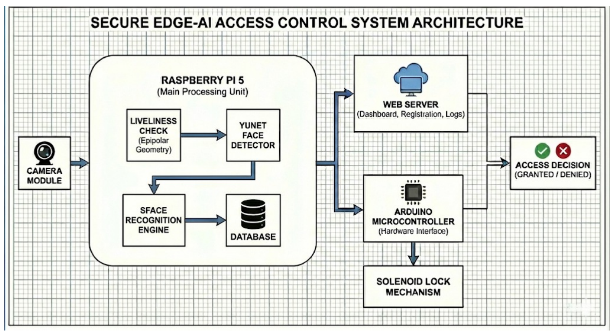


Figure 1: System Block Diagram. We show an overview of the proposed smart lock with liveness detection, face recognition, and hardware.

2.1 Non-Biometric Smart Locks

Vongchumyen et al. [17] developed a Web-based solution, while Aswini et al. [2] introduced an RFID-OTP system. While these solve the lost-key problem, they fail to offer convenience. Whether using a phone 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 by Maheshwari and Nalini [13] offer accuracy but suffer from latency and internet dependency. Alternatively, local offline systems like Elechi et al. [7] often lack the robustness to distinguish real faces from photos. To counter this, Dasare [5] introduced active liveness (hand gestures), which unfortunately adds user friction. A robust system must be both passive and local.

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 [21] standard for camera calibration to ensure geometric accuracy and Hartley and Sturm's [9] triangulation framework to reconstruct facial landmarks. Crucially, we utilize Hirschmüller's [11] 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. [18] offer high accuracy, they are computationally prohibitive for microcontrollers.

3 Technical Prerequisites

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

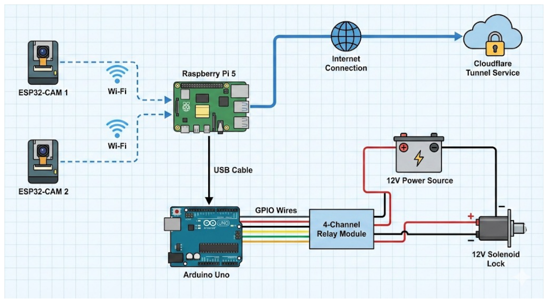


Figure 2: Integrated System Architecture. System components with data flow from stereo capture to lock actuation.

3.1 Stereo Calibration & Epipolar Geometry

To reconstruct 3D information, we model the camera using the Pinhole Model [8]. First, we estimate the **Intrinsic Matrix** (K) using a flexible calibration technique [21]. This accounts for the focal length (f_x, f_y) and optical center (c_x, c_y) of cameras (Eq.1):

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{1}$$

For stereo consistency, we utilize the Epipolar Constraint and the triangulation framework described in [9]. Given a point projected onto the left ($\mathbf{p}_L$) and right ($\mathbf{p}_R$) image planes, their relationship is governed by the Fundamental Matrix ($\mathbf{F}$), which encapsulates the epipolar geometry of the stereo pair (Eq.2):

$$\mathbf{p}_R^T \cdot \mathbf{F} \cdot \mathbf{p}_L = 0. \tag{2}$$

3.2 Disparity & Depth Estimation

To compute depth, we first calculate pixel disparity (d) using the **Semi-Global Block Matching (SGBM)** algorithm [10]. Unlike simple pixel matching, SGBM enforces a global smoothness (WLS) constraint to reduce noise on textureless surfaces like skin. The depth Z is inversely proportional to disparity [9]:

$$Z = \frac{f \cdot B}{d}. \tag{3}$$

Eq.3, where Z is the distance (depth) of the object from the camera, f is the rectified focal length (derived from f_x), B is the baseline distance between cameras, d is the Disparity and ($x_L - x_R$) horizontal pixel difference between the left and right image points. This relationship allows the system to distinguish between a real face (varying Z) and a spoof photograph (constant Z).

3.3 Deep Metric Learning and Loss Function

To enable robust identity verification, the system map the facial images into a 128-dimensional Euclidean space. It utilizes a Convolutional Neural Network (CNN), denoted as $f_\theta(\cdot)$, to extract a feature vector. For any input face x, the network outputs a normalized embedding $\mathbf{v}$:

$$\mathbf{v} = \frac{f_\theta(x)}{||f_\theta(x)||_2}, \quad \text{where } \mathbf{v} \in \mathbb{R}^{128}. \tag{4}$$

This normalization projects all facial features onto a hypersphere of fixed radius s, ensuring that magnitude does not affect the matching process.

To ensure these embeddings are discriminative, the network is trained using the SFace loss function [22]. SFace optimizes the embedding space by enforcing explicit geometric constraints via a combined loss function $\mathcal{L}_S$:

$$\mathcal{L}_S = \underbrace{\log\left(1 + e^{s(\theta_{y_i} - \beta)}\right)}_{\text{Intra-class Compactness}} + \lambda \underbrace{\log\left(1 + e^{s(\alpha - \theta_{neg})}\right)}_{\text{Inter-class Separability}}. \tag{5}$$

Eq.5, the first term forces features of the same identity to stay within an angular boundary β, while the second term pushes different identities apart by a margin α. λ is a hyperparameter that balances the two terms. This clustering during training enables the use of simple distance metrics for verification.

Cosine Similarity is a metric used to measure how similar two vectors are irrespective of their size. Mathematically, it measures the cosine of the angle θ between two vectors projected in a multi-dimensional space.

$$\text{Similarity}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{||\mathbf{A}||\,||\mathbf{B}||} = \cos(\theta). \tag{6}$$

The resulting score ranges from -1 (completely different) to 1 (exactly the same). In the context of face recognition, this metric is preferred over Euclidean distance because it focuses on the orientation of the feature vectors rather than their magnitude.

4 Methodology

4.1 System Overview

The system workflow begins with stereo vision for depth perception and anti-spoofing, followed by face recognition and a control interface for hardware actuation. The system captures synchronized video streams from two cameras, applies epipolar rectification and depth estimation to verify facial liveness, and subsequently performs identity recognition using learned facial embeddings. After successful authentication, an Arduino-controlled solenoid lock is actuated to grant physical access as in Fig.2.

4.2 System Design and Implementation

Epipolar Geometry. The stereo vision module follows the calibrated stereo camera model described in Eq.1. Intrinsic and extrinsic parameters are obtained offline using a standard chessboard calibration procedure [21]. Using these parameters, epipolar rectification is applied to align corresponding scan lines, as in Eq. 2. Disparity is then computed along horizontal epipolar lines using the Semi-Global Block Matching (SGBM) algorithm, producing a dense disparity map (Fig.4b).

Depth Estimation. The depth (Z) of any pixel is inversely proportional to its disparity (d) (Eq.3). This allows reliable estimation of relative facial depth using the calibrated stereo configuration.

Facial Landmark Localization. Facial landmarks are detected using the YuNet [19], which identifies both bounding boxes and precise coordinates for key facial features. These landmarks are detected in the image space and mapped to corresponding coordinates in the disparity map. This mapping enables direct association between facial landmarks and their estimated depth values.

Liveness Detection. A human face is a 3D object where the nose is closer to the camera than the eyes. Conversely, a 2D spoof (photo) is planar, meaning the depths of the nose and eyes are approximately equal. Depth values are extracted at the landmark positions corresponding to the left eye, right eye, and nose tip:

$$\Delta Z = \frac{Z_{\text{left\_eye}} + Z_{\text{right\_eye}}}{2} - Z_{\text{nose}}, \tag{7}$$

where $Z_{\text{left\_eye}}$, $Z_{\text{right\_eye}}$, and Z_{nose} denote the estimated depths at the corresponding facial landmark. The liveness condition is satisfied if the resulting geometric protrusion exceeds a predefined threshold. The threshold $[\Delta Z_{min}, \Delta Z_{max}]$ defined on the empirical separation between planar spoofs ($\approx$ 0m) and actaul facial depth profiles ($>$ 0.020m), as in Appendix A. This range effectively rejects both 2D presentation attacks and depth estimation outliers. Liveness is validated across 3 consecutive frames to filter transient noise.

Face Recognition. The face recognition follows a 3 stage pipeline: detection, alignment, and feature extraction. The input image I is first processed by YuNet [19], an efficiency-optimized convolutional neural network. For every detected face, it outputs a bounding box $B = (x, y, w, h)$ and 5 landmarks (nose, eyes and mouth corners).

We apply Affine Transformation [6] to map the detected facial landmarks to a standard template. This accounts for rotation (θ), isotropic scaling (s), and translation (t_x, t_y) via the matrix M, Eq.8:

$$M = \begin{bmatrix} s\cos\theta & s\sin\theta & t_x \\ -s\sin\theta & s\cos\theta & t_y \end{bmatrix}. \tag{8}$$

The optimal matrix parameters are estimated using the Least Squares method, minimizing the error between the detected source points (L_{src}) and the target template (L_{dst}):

$$\min_{M} \sum_{j=1}^{5} \|L_{dst}^{(j)} - M \cdot \tilde{L}_{src}^{(j)}\|^2. \tag{9}$$

Finally, the aligned image $I_{aligned}$ (112 × 112 pixels) is generated by inverse transformation:

$$I_{aligned}(x', y') = I(M^{-1}[x', y', 1]^T). \tag{10}$$

The aligned image is passed to SFace [22], which maps the high-dimensional pixel data into a compact 128-dimensional embedding vector $\mathbf{v} \in \mathbb{R}^{128}$. These embeddings are normalized to lie on a hypersphere, to find semantic similarity.

Cosine similarity (Eq.6) is used to compare the extracted embedding with stored embeddings in the database. Access is granted if the similarity exceeds a predefined threshold (0.40 for SFace), ensuring that only authorized users are recognized.

Hardware Actuation. The system governs transitions between liveness verification, recognition, and access states. Upon successful authentication, a signal is transmitted via serial communication to an Arduino, which actuates a solenoid lock. A cooldown interval of 5 seconds is enforced to prevent repeated triggering.

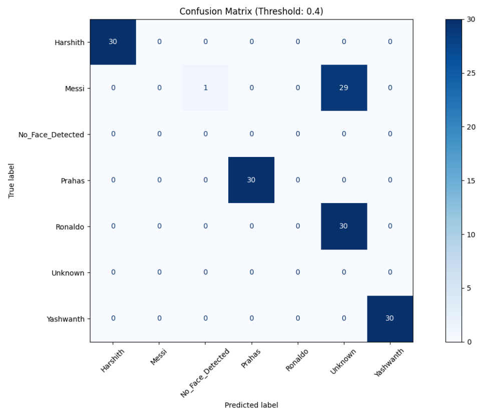


Figure 3: Confusion Matrix for YuNet + SFace. Demonstrating perfect accuracy and low error.

5 Experiments and Evaluation

5.1 Dataset

We collect a dataset using a laptop web camera, including 200 high-quality images from three participants who perform various head movements and facial expressions during 60-second videos. For security, images are temporarily stored on a Raspberry Pi. Our evaluation uses a test dataset of 150 images. The model creates 128-dimensional vector embeddings for each training image and averages them to form a mean embedding for each identity, stored locally on the Raspberry Pi for real-time recognition and evaluation.

5.2 Baselines

Our examination includes traditional approaches such as the Haar Cascade Classifier [16] and the Histogram of Oriented Gradients in conjunction with support vector machine (SVM) [4]. Additionally, we have evaluated deep learning-based models, including the Dlib (OpenCV) [12] and the Insightface (Arcface) [6].

5.3 Evaluation Metrics

To evaluate the performance of our system, we utilize standard classification metrics derived from the confusion matrix (Fig. 3): True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). We measure **Accuracy**, representing the overall correctness of the model. Furthermore, we calculate **Precision**, defined as the ratio of correctly predicted positive observations to the total predicted positives, and **Recall** (True Positive Rate), which measures the ratio of correctly predicted positive observations to the actual class. Finally, we report the **F1-Score**, the harmonic mean of Precision and Recall that balances the two metrics.

5.4 Quantitative Analysis

The Haar Cascade model demonstrated a very low Accuracy (0.08) and a high False Negative rate (FN 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 authorized users (Recall

Table 1: Model Evaluation

Models	Recall (TP)	FP	TN	FN	Precision	F1-Score	Accuracy
Haar Cascade	0.13	0.70	0.30	0.87	0.16	0.14	0.08
HOG + SVM	0.14	0.13	0.87	0.86	0.52	0.22	0.52
Dlib (OpenCV)	0.83	0.55	0.45	0.00	0.60	0.70	0.60
Insightface*	0.90	0.00	1.00	0.10	1.00	0.95	0.94
YuNet + SFace (Ours)	**1.00**	**0.00**	**1.00**	**0.00**	**1.00**	**1.00**	**1.00**

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

Table 2: Ablation Study

Configuration	Accuracy	FP	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.

0.14), likely due to the inflexibility of HOG descriptors when faced with the varied poses and facial expressions.

Among the deep learning approaches, Dlib (OpenCV) achieved improved recognition performance (Recall 0.83), but it was accompanied by a significant security concern: a high False Positive rate (FP 0.55). This indicates a lack of discriminative ability to differentiate between users and impostors. Although InsightFace demonstrated high Accuracy (0.94) and perfect Precision (1.00), it was not selected due to its computationally intensive architecture, exceeding the capabilities of the Raspberry Pi 5 for edge deployment.

Ultimately, YuNet + SFace provided the optimal solution. It achieved perfect metrics (1.00 Accuracy, 1.00 Precision, 0.00 FP), demonstrating the ideal balance of computational efficiency and high-precision security required for the final system (Table 1).

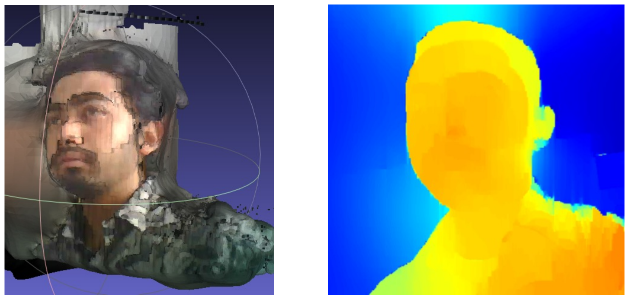
5.5 Ablation Study

We conducted an ablation study to isolate the impact of the detector (YuNet) and the recognizer (SFace) independently (Table 2). We tested four configurations: YuNet + LBPH, Haar Cascade + SFace, Retinaface + Arcface, and the proposed YuNet + SFace. We first assessed the YuNet + LBPH. Despite the integration of a robust detector, system accuracy decreased to 0.45, accompanied by a high FP (0.40). These results indicates, Local Binary Pattern 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.

The Retinaface + Arcface configuration matched our model's perfect accuracy (1.0) and FP (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 FP) 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.



(a) 3D Reconstruction. Facial depth reconstructed as a mesh.

(b) Disparity. Disparity map showing depth variations.

Figure 4: Liveness cues. 3D reconstruction and disparity visualization.

5.6 Qualitative Analysis

3D Reconstruction. Fig.4a shows the rebuilt 3D point cloud mesh overlaid with measured facial depths. The close match between the mesh and the person's facial shape confirms that the depth measurements are accurate. The reconstruction clearly shows the shape and depth of the face. This accuracy makes sure that the depth difference (ΔZ) (Eq.7) comes from real features of the face.

Disparity Mapping. Fig.4b shows the disparity map by the Semi-Global Block Matching (SGBM) algorithm. Looking at the image, you can clearly see the difference between the person in front and the background. The heatmap uses warmer colors to show areas that are closer, and cooler colors for areas that are farther away. The depth across the face is not the same; there is a clear change from the tip of the nose out to the ears and neck.

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 thank the Technical University of Applied Sciences Würzburg-Schweinfurt (THWS) for providing the 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.

A Depth Analysis and Liveness Thresholds

Table 3: Expected Depth Differences Between Eyes and Nose

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

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

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.

References

[1] Arduino Expert. 2025. IoT Smart Door Lock Using ESP32, Solenoid Lock & Blynk App. Online: https://arduinoexpert.com/iot-smart-door-lock-using-esp32-and-blynk/.

[2] D Aswini et al. 2021. Smart locking system using RFID and OTP. In *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Vol. 1. IEEE, 1876–1879.

[3] S. Bhattacharjee. 2018. Spoofing Deep Face Recognition with Custom Silicone Masks. Conference paper (ICB 2018) – online reference when available.

[4] N. Dalal and B. Triggs. 2005. Histograms of Oriented Gradients for Human Detection. CVPR 2005.

[5] P Dasare et al. 2021. Smart Door Lock System with Face Recognition and Active Liveness Detection. In *2021 International Conference on Communication information and Computing Technology (ICCICT)*. IEEE, 1–6.

[6] J. Deng et al. 2019. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. CVPR 2019.

[7] H Elechi et al. 2022. Face recognition-based smart door lock system using Raspberry Pi. *International Journal of Engineering and Manufacturing (IJEM)* 12, 1 (2022), 23–34.

[8] Richard Hartley and Andrew Zisserman. 2003. *Multiple view geometry in computer vision*. Cambridge University Press.

[9] Richard I. Hartley and Peter Sturm. 1997. Triangulation. *Computer Vision and Image Understanding* 68, 2 (1997), 146–157. doi:10.1006/cviu.1997.0547

[10] Heiko Hirschmller. 2008. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence* 30, 2 (2008), 328–341.

[11] Heiko Hirschmuller. 2005. Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, 807–814.

[12] D. E. King. 2009. Dlib-ml: A Machine Learning Toolkit. Journal of Machine Learning Research (software/library).

[13] S Maheshwari and N Nalini. 2017. Face recognition using IoT based smart door lock system. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*. IEEE, 3346–3350.

[14] Mordor Intelligence. 2026. Smart Home Video Surveillance Market Size & Growth Analysis Report. Online: https://www.mordorintelligence.com/industry-reports/smart-home-video-surveillance-market.

[15] S. Serengil. 2024. DeepFace: A Lightweight Face Recognition and Facial Attribute Analysis Framework. GitHub repository: https://github.com/serengil/deepface.

[16] P. Viola and M. Jones. 2001. Rapid object detection using a boosted cascade of simple features. CVPR 2001 (classic paper).

[17] C Vongchumyen, P Thirachit, and K Khongritti. 2017. Door lock system via web application. In *2017 International Electrical Engineering Congress (iEECON)*. IEEE, 1–4.

[18] Y Wen et al. 2025. Foundation Stereo: A Large-Scale Stereo Matching Model. *arXiv preprint arXiv:2501.xxxxx* (2025).

[19] X. Wu and OpenCV contributors. 2023. YuNet: A tiny millisecond-level face detector. Online: https://github.com/opencv/opencv/blob/master/samples/dnn/face_detection_yunet.md.

[20] A. Zainuddin. 2024. Vulnerability Analysis of Smart Lock Using NIST SP 800-115 Method. Online: https://www.researchgate.net/publication/396254787.

[21] Zhengyou Zhang. 2000. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 11 (2000), 1330–1334. doi:10.1109/34.888718

[22] H. Zhong. 2021. SFace: a lightweight face recognition model. Preprint / conference reference when available.