electronics

MDPI

Article

Real-Time Gesture-Based Hand Landmark Detection for Optimized Mobile Photo Capture and Synchronization

Pedro Marques [1], Paulo Váz [2], José Silva [2], Pedro Martins [2,*] and Maryam Abbasi [3]

[1] Department of Informatics, Polytechnic University of Viseu, 3504-510 Viseu, Portugal
[2] Research Center in Digital Services (CISeD), Polytechnic University of Viseu, 3504-510 Viseu, Portugal; paulovaz@estgv.ipv.pt (P.V.); jsilva@estgv.ipv.pt (J.S.)
[3] Research Center for Natural Resources, Environment and Society (CERNAS), Polytechnic University of Coimbra, 3045-601 Coimbra, Portugal; maryam.abbasi@ipc.pt
* Correspondence: pedromom@estgv.ipv.pt

Abstract: Gesture recognition technology has emerged as a transformative solution for natural and intuitive human–computer interaction (HCI), offering touch-free operation across diverse fields such as healthcare, gaming, and smart home systems. In mobile contexts, where hygiene, convenience, and the ability to operate under resource constraints are critical, hand gesture recognition provides a compelling alternative to traditional touch-based interfaces. However, implementing effective gesture recognition in real-world mobile settings involves challenges such as limited computational power, varying environmental conditions, and the requirement for robust offline–online data management. In this study, we introduce *ThumbsUp*, which is a gesture-driven system, and employ a partially systematic literature review approach (inspired by core PRISMA guidelines) to identify the key research gaps in mobile gesture recognition. By incorporating insights from deep learning–based methods (e.g., CNNs and Transformers) while focusing on low resource consumption, we leverage Google's MediaPipe in our framework for real-time detection of 21 hand landmarks and adaptive lighting pre-processing, enabling accurate recognition of a "thumbs-up" gesture. The system features a secure queue-based offline–cloud synchronization model, which ensures that the captured images and metadata (encrypted with AES-GCM) remain consistent and accessible even with intermittent connectivity. Experimental results under dynamic lighting, distance variations, and partially cluttered environments confirm the system's superior low-light performance and decreased resource consumption compared to baseline camera applications. Additionally, we highlight the feasibility of extending ThumbsUp to incorporate AI-driven enhancements for abrupt lighting changes and, in the future, electromyographic (EMG) signals for users with motor impairments. Our comprehensive evaluation demonstrates that *ThumbsUp* maintains robust performance on typical mobile hardware, showing resilience to unstable network conditions and minimal reliance on high-end GPUs. These findings offer new perspectives for deploying gesture-based interfaces in the broader IoT ecosystem, thus paving the way toward secure, efficient, and inclusive mobile HCI solutions.

Keywords: gesture recognition; hand tracking; real-time detection; MediaPipe; mobile computing; computer vision; human–computer interaction; touchless interfaces

Academic Editors: Pawel Strumillo, Guanzhou Chen, Kun Zhu and Jinzhou Cao

Received: 20 December 2024
Revised: 31 January 2025
Accepted: 9 February 2025
Published: 12 February 2025

Citation: Marques, P.; Váz, P.; Silva, J.; Martins, P.; Abbasi, M. Real-Time Gesture-Based Hand Landmark Detection for Optimized Mobile Photo Capture and Synchronization. *Electronics* **2025**, *14*, 704. https://doi.org/10.3390/electronics14040704

1. Introduction

Advances in human–computer interaction (HCI) have continually reshaped how users engage with digital devices, transitioning beyond keyboards and touchscreens to

more intuitive paradigms such as gesture-based interfaces [1]. In mobile computing environments, gesture recognition is particularly compelling and addresses critical needs for hygienic, touch-free operation in public and healthcare settings while also accommodating scenarios where users have limited physical interaction options [2]. Despite this promise, implementing robust gesture recognition on resource-constrained mobile devices presents multiple challenges, including power efficiency, real-time processing, and varying lighting conditions.

Gesture recognition typically relies on capturing and analyzing visual or sensor-based cues, with tools ranging from RGB or depth cameras to wearable motion sensors [3]. However, there remains a gap in managing dynamic environmental factors, such as rapidly changing lighting or user occlusions, while ensuring efficient offline-to-cloud data synchronization. In addition, the possibility of integrating advanced methods (e.g., deep learning architectures like CNNs or Transformers) on-device is often hindered by computational limitations, making it non-trivial to achieve both high accuracy and real-time responsiveness in everyday mobile contexts. Security and privacy also demand attention, particularly when personal images and metadata must be stored or transmitted.

Motivated by these challenges, we conducted a partial literature survey following core PRISMA guidelines to minimize selection bias and identify major research gaps in mobile gesture recognition. From this survey, three critical focal points emerged: (1) the need for reliable performance in low- and high-light scenarios; (2) the integration of secure offline–online data flow for environments with intermittent connectivity; (3) balancing performance and resource usage when scaling to real-world deployments. These factors underscore the importance of optimizing gesture recognition pipelines not only for accuracy but also for efficient synchronization, storage, and encryption, which are elements often overlooked in existing frameworks.

In this work, we introduce *ThumbsUp*, a novel gesture-driven photo capture system that demonstrates how carefully orchestrated modules for detection, encryption, and synchronization can yield robust and adaptive performance on modern smartphones. Specifically, our approach leverages Google's MediaPipe for real-time hand landmark detection, employing a pipeline tuned to operate efficiently on mobile hardware. We further integrate AES-GCM encryption to secure locally stored images and metadata, alongside a queue-based offline synchronization layer with a NoSQL back end. The synergy of these components addresses both the security and reliability requirements identified in our literature survey, bridging the gap between gesture recognition research and practical mobile applications.

A key contribution of this paper lies in the thorough evaluation of *ThumbsUp* under diverse conditions, including extreme low-light environments, partial occlusion scenarios, and network instability. Unlike many prior studies that rely on static laboratory settings, we simulate real-world constraints to validate the readiness of our system for everyday usage. Moreover, by comparing *ThumbsUp* with existing camera applications and referencing deep-learning-based solutions, we illustrate how our balanced design approach achieves competitive accuracy without incurring substantial resource overhead.

The remainder of this paper is organized as follows. Section 2 reviews the existing literature in gesture recognition, highlighting both classic and AI-driven approaches and elaborating on the challenges of mobile deployment. Section 3 introduces our system architecture, explaining how SQLite and MongoDB jointly manage local and remote data while ensuring encryption and minimal data loss. Section 4 discusses our evaluation metrics and procedures, and Section 5 presents the results in comparison to a baseline camera application. Finally, Section 6 provides conclusions, outlines limitations, and suggests

avenues for future work, including the potential for incorporating electromyographic (EMG) signals and AI-enhanced lighting adaptivity.

2. Literature Review

Gesture recognition has witnessed substantial growth in recent years and is propelled by advancements in computer vision, machine learning, and sensor technologies. To structure the literature survey and minimize selection bias, key principles from the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) guidelines were adopted [4]. Although this work is not a fully systematic review, it follows essential steps such as a well-defined search strategy, explicit inclusion/exclusion criteria, and transparent documentation. This approach ensures a more objective curation of relevant studies, particularly those emphasizing mobile-based gesture recognition and robust offline-to-cloud synchronization.

2.1. Overview of Gesture Recognition Methods

Gesture recognition methods are typically classified into vision-based and sensor-based systems. Vision-based approaches often rely on camera feeds to capture hand movements or poses, sometimes aided by depth sensors for three-dimensional analysis [5]. Sensor-based methods utilize inertial (accelerometers, gyroscopes) or physiological (electromyographic, or EMG) signals to detect gestures [6]. Both strategies offer distinct benefits and face specific limitations, especially for mobile platforms that contend with stringent computational resources and battery constraints.

Early consumer-oriented systems, such as Microsoft Kinect, leveraged depth sensors for real-time body tracking [7]. Although highly effective in controlled indoor settings, depth sensors presented integration challenges for mobile or wearable devices. Later work highlighted the feasibility of on-board smartphone cameras for hand gesture detection [8,9], yet problems such as varying lighting conditions and user distance remained unresolved in many early prototypes.

2.2. Deep Learning Advances in Gesture Recognition

Deep learning approaches have substantially influenced gesture recognition, delivering robust feature extraction and higher accuracy in complex scenarios. Convolutional Neural Networks (CNNs) have been employed extensively for static gesture classification [10], while Recurrent Neural Networks (RNNs) or Transformer architectures address temporal dependencies in dynamic gestures [11]. Mobile-tailored CNNs with reduced parameter sizes have shown promise [12], but deploying resource-hungry deep models on smartphones remains a key obstacle.

Despite their accuracy gains, deep architectures can demand substantial processing power and memory. These requirements complicate real-time inference on mobile devices where heat dissipation and power draw are paramount considerations. Recent work on vision Transformers (ViT) [13,14] exhibits the capacity for capturing local and global context, but their heightened computational footprint poses challenges for prolonged usage scenarios. The core challenge, therefore, revolves around merging high-accuracy deep learning techniques with optimized, on-device pipelines—a gap this research addresses by integrating MediaPipe with carefully orchestrated encryption and synchronization flows.

2.3. MediaPipe and Real-Time Hand Tracking

Google's MediaPipe framework is a popular choice for real-time hand tracking thanks to its efficient pipeline and relatively modest resource usage [15]. Studies have demonstrated how MediaPipe can be integrated with additional lightweight algorithms for hands-

free control in augmented reality [16]. By detecting 21 landmarks per hand, the framework offers fine-grained spatial analysis that reliably identifies subtle finger poses. Landmark-based tracking is especially useful for mitigating issues like partial occlusions and dynamic backgrounds, making MediaPipe applicable to diverse mobile conditions.

However, notable gaps persist. Foremost, many studies do not tackle extreme lighting (very low or very bright) or sudden transitions, impacting real-world usage viability. Moreover, while MediaPipe streamlines detection, it does not inherently address offline-to-cloud data handling or end-to-end security for captured images and metadata. The present work bridges these lacunae by incorporating AES-GCM encryption and a synchronized, queue-based architecture, thus ensuring both real-time recognition and robust data handling in variable network settings.

2.4. Sensor Fusion and EMG-Based Techniques

Beyond camera-based methods, sensor fusion approaches integrate complementary data streams to bolster recognition accuracy and resilience [17,18]. Some research uses frequency-modulated continuous wave (FMCW) radar to detect subtle gestures under non-visual conditions [19], whereas others combine accelerometer and infrared data [20], or merge visual and inertial measurements [21] to improve tracking under occlusions. Electromyographic (EMG) signals have also gained attention for capturing nuanced muscular activations underlying gestures, which can benefit individuals with motor impairments [22]. AI-driven EMG fusion can elevate recognition precision, especially in challenging ambient conditions or when visual cues are partially blocked. Integrating EMG into on-device pipelines, however, remains a complex challenge due to hardware constraints.

2.5. Security, Privacy, and Offline–Cloud Synchronization

Securing gesture-based interactions is critical when handling potentially sensitive images and metadata, especially in healthcare or personal domains. Research has underscored the necessity of privacy-preserving techniques in vision-based systems [23], yet thorough end-to-end encryption strategies optimized for mobile resource constraints appear relatively scarce. By employing AES-GCM encryption, JWT-based authentication, and queue-based synchronization, this work aligns with recommended cloud-edge security best practices [24].

Offline functionality is another key requirement in sectors like healthcare or industrial automation, where connectivity might be intermittent [25]. Proposed edge-computing frameworks typically aim to reduce latency or bandwidth usage [26], but do not always detail local offline storage or the complexities of re-synchronizing large data batches. Our methodology incorporates SQLite as a local store and MongoDB in the cloud, ensuring minimal data loss and consistent updates. Synchronized queues handle the backlog of captured images and metadata, providing a seamless transition once network connectivity is restored. This design addresses a frequently overlooked aspect: reconciling secure local data with eventual cloud storage in a manner that remains transparent and reliable to the end user.

2.6. Applications and Gaps in the Literature

Applications of gesture recognition now span multiple domains. Some works highlight contactless interfaces in sterile healthcare environments [27], while others explore immersive educational tools [28] or gaming and virtual reality use cases [29]. Nevertheless, there is a persistent gap in demonstrating how a gesture-based system can holistically integrate encryption, real-time detection, and offline-to-cloud data handling under the common constraints of mobile hardware. This challenge is central to the design of the

present system, which endeavors to be both resource-efficient and secure while retaining high detection accuracy in realistic operating conditions.

2.7. Summary and Research Directions

In summary, key insights from our PRISMA-driven review are as follows:

1. Deep learning can boost gesture recognition accuracy, but on-device inference remains problematic without careful model optimization due to thermal and battery constraints.
2. MediaPipe provides an efficient and accurate baseline for real-time hand-tracking, though it lacks built-in data security and robust offline synchronization.
3. Sensor fusion, including EMG signals, holds promise for enhanced robustness and accessibility, yet seamless integration with mobile AI pipelines has not been thoroughly addressed.

To fill these gaps, the *ThumbsUp* project merges an optimized landmark-based recognition framework with AES-GCM encryption, a queue-driven offline-to-cloud synchronization scheme, and a resource-conscious design philosophy. The following sections elaborate on our methodology, experimental setup, and empirical results, illustrating how these literature-derived principles materialize into a secure, efficient, and user-friendly gesture recognition platform suitable for real-world mobile deployments.

3. Methodology and Architecture

This section describes the enhanced methodology and architectural design of the *ThumbsUp* system, integrating key insights from our partial PRISMA-like literature review, feedback from prior implementations, and the need for robust performance in resource-constrained, real-world environments. The goal is to demonstrate how real-time hand tracking, secure data handling, and a flexible offline-to-cloud synchronization pipeline can be cohesively orchestrated to deliver a seamless, user-friendly experience.

3.1. Data Preparation and Gesture Pipeline Overview

Although *ThumbsUp* primarily relies on live camera feeds, we initially gathered a representative dataset of hand gestures, including "thumbs-up", under various lighting and background conditions. Leveraging PRISMA-like principles, we established explicit inclusion/exclusion criteria and screened public repositories and our own collected samples. Images or video frames containing clearly visible hands were retained, while those with excessive occlusion or ambiguity were removed. This partial PRISMA approach aimed to mitigate selection bias and ensure coverage of diverse conditions (e.g., low-light, bright-light, and partially cluttered scenes).

As illustrated in Figure 1, *ThumbsUp* follows a systematic gesture pipeline:

1. Live Capture and Preprocessing: CameraX API handles live frame acquisition. Frames are resized, oriented, and color-corrected. To improve robustness in low-light or harsh lighting transitions, we apply adaptive brightness/contrast adjustment as needed, following guidelines from prior literature [30].
2. Hand Landmark Detection: Using Google's MediaPipe, the system detects 21 landmarks per hand in real time. This approach is computationally efficient for mobile devices, leveraging GPU acceleration where available [15].
3. Gesture Analysis: A classifier inspects landmark geometry, identifying target gestures (such as "thumbs-up") based on angular thresholds and relative finger positions. Additional checks, such as the distance of the wrist to the camera, help filter out ambiguous poses or out-of-bounds frames.

4. Encryption and Local Storage: Confirmed gestures trigger photo capture. Images are instantly encrypted with AES-GCM, and both the image and metadata are stored locally in SQLite. This step ensures minimal latency during capture while preserving data security, as recommended by Rautaray and Agrawal [31].
5. Synchronization Queue: Metadata and images are placed in a queue awaiting cloud upload. If the network is unstable, the system retains these items until a stable connection is detected, mitigating losses in offline scenarios.

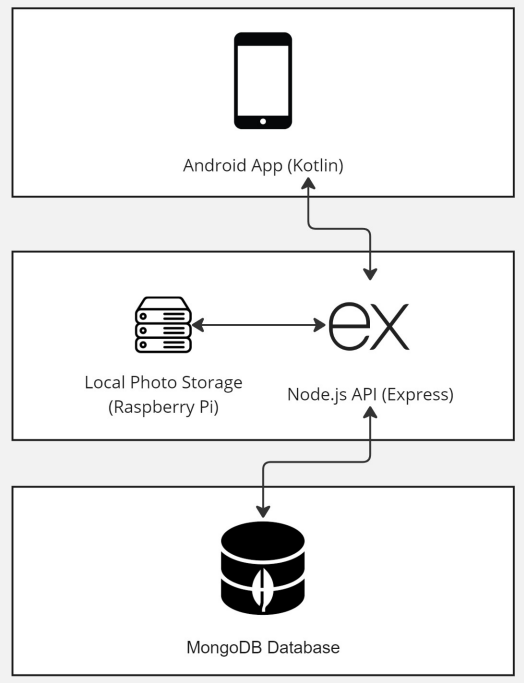

Figure 1. System architecture of *ThumbsUp*, which highlights the interactions between the mobile application, middleware, and cloud database.

3.2. Mobile Application Layer

The mobile application serves as the user-facing component, optimized for minimal CPU and memory consumption. Real-time image capture is performed via CameraX, while MediaPipe conducts hand landmarking directly on the device. The application stores the resulting encrypted photos and gesture metadata in a local SQLite database:

- Local Storage and Security: AES-GCM encryption is employed to protect images at rest, with keys managed by the Android KeyStore. This approach is recognized as a robust solution preventing direct file access by unauthorized entities [32].
- Resource Efficiency: In-line compression and discard strategies are applied to non-essential frames, reducing overhead. For instance, if a user holds a thumbs-up for extended seconds, only the first validated capture is stored.
- User Feedback and Accessibility: Haptic or audio signals confirm successful gesture detection. Future extensions could involve adjustable recognition thresholds or voice prompts for users with limited mobility, aligning with reviewer suggestions on inclusivity.

3.3. Middleware Layer on Raspberry Pi

The middleware, deployed on a Raspberry Pi for cost-effectiveness and energy efficiency, serves as an intermediary that authenticates clients, validates data, and schedules uploads:

- Event-Driven Architecture: Implemented in Node.js, it handles concurrent HTTP(S) requests and interacts with the queue of pending uploads from multiple devices. When connectivity becomes available, it requests queued items in batches, reducing bandwidth spikes.
- JWT-based Authentication: Each request includes a JSON Web Token validated against a server-side key. This ensures only legitimate clients can push encrypted images or read stored data.
- Metadata Augmentation and Logging: The middleware can enrich metadata with server-side timestamps or user location tags (if desired) before final storage. Error logs facilitate troubleshooting if data integrity checks fail.

3.4. Cloud Database Layer: MongoDB Structure

Figure 2 shows the high-level MongoDB schema, which includes three main collections: Users, Photos, and Metadata. This schema-less design supports evolving feature requirements (e.g., adding new gesture types or user-defined tags) without extensive schema migrations:

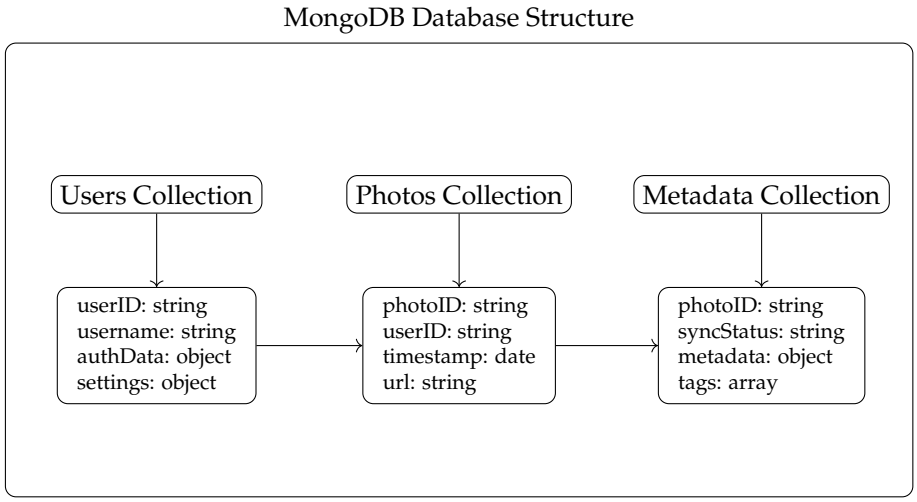

Figure 2. MongoDB database schema showing collections for users, photos, and metadata, and their relationships.

As shown in Figure 3, the Photos collection typically holds references (URLs) to the encrypted files, which can be stored in a secure object bucket or remain in base64-encoded form within the document (depending on user requirements). The Metadata collection stores gesture classifications, environment info (lighting level, device orientation), and synchronization status. When offline photos eventually upload, their syncStatus transitions from "pending" to "uploaded", ensuring data integrity across devices.

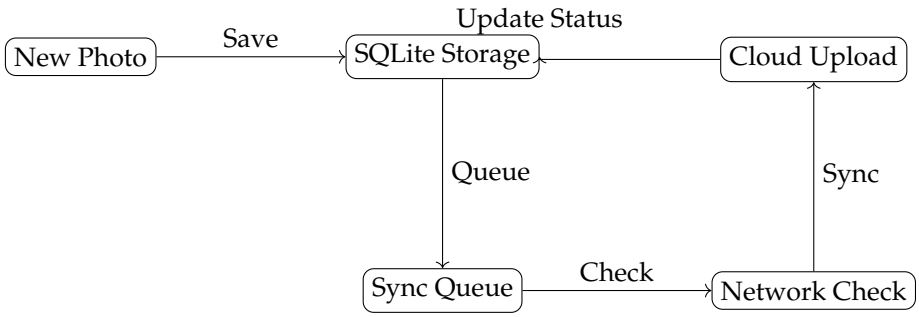

Figure 3. Synchronization flow from local SQLite to MongoDB, demonstrating the queue-based approach for handling intermittent connectivity.

3.5. Handling Dynamic Environmental Conditions

Most real-world mobile scenarios involve fluctuating lighting, rapid hand movement, and background clutter. We employ a combination of motion smoothing and robust background subtraction to isolate hands even under moderate occlusions or shifting backgrounds. Additionally, the system can adaptively lower or raise confidence thresholds based on per-frame metrics, such as the average pixel brightness. This ensures minimal false positives in both extreme low-light and high-light contexts.

3.6. Extensibility and Future AI Integrations

Although *ThumbsUp* utilizes MediaPipe for fast on-device landmark detection, our modular architecture accommodates more advanced AI models, such as CNNs or Transformers, for specialized gestures or highly dynamic lighting conditions. Offline–online synchronization remains unchanged, thus allowing heavier models to be selectively deployed or updated in the cloud without disrupting the core pipeline. Furthermore, sensor fusion with electromyographic (EMG) signals could be explored to improve accuracy for users with motor impairments, aligning with the growing body of research on multimodal gesture recognition [33].

4. Experimental Setup

This section details how we designed and conducted our experiments to rigorously evaluate the *ThumbsUp* application against *Google Camera* under realistic mobile conditions. Our approach emphasizes reproducibility, partial PRISMA-like guidelines for data selection, and controlled variation of environmental factors to simulate real-world usage. We outline the hardware, network configuration, testing scenarios, and evaluation metrics, culminating in a robust framework for performance comparison.

4.1. Evaluation Dimensions and Experimental Framework

As described in Section 3, the experimental framework is designed to comprehensively evaluate the system's performance across three primary dimensions: gesture recognition performance, system efficiency, and synchronization reliability. These dimensions encapsulate the key metrics required to ensure the system performs accurately, efficiently, and reliably in real-world scenarios.

The first dimension, gesture recognition performance, focuses on the system's ability to accurately and efficiently recognize user gestures. Accuracy, a central metric in this category, is further broken down into precision, recall, and the F1 score, providing a nuanced understanding of how well the system identifies and classifies gestures. Additionally, response time is critical, as it captures the latency between a user's input and the system's reaction, ensuring that gestures are processed in real-time. This dimension also evaluates dynamic detection capabilities, such as handling fast or fluid movements, and the system's adaptability to variations in gesture execution.

The second dimension, system efficiency, assesses the system's resource usage and thermal stability during operation. CPU and memory usage are essential metrics in this category, offering insights into both the average load and peak demands placed on the system. Efficient memory management and minimal computational overhead ensure the system remains responsive, even during intensive tasks. The framework also considers thermal performance, including maximum operating temperatures and stability over time, alongside power consumption, which examines both power draw and battery usage for portable applications. Together, these metrics ensure the system operates efficiently without compromising performance or hardware longevity.

Finally, synchronization reliability addresses the system's ability to synchronize data in various conditions, particularly in networked or distributed environments. This includes measuring upload speeds, with attention to latency and throughput, to ensure data transfer is both fast and efficient. Storage usage is another key factor in assessing how well the system optimizes storage space through efficient data compression methods. Reliability is perhaps the most critical aspect of this dimension, examining error rates, data integrity, and the robustness of failover mechanisms that ensure system continuity in the face of network disruptions or errors.

Figure 4 illustrates this comprehensive evaluation framework. The diagram organizes these dimensions and their associated metrics into a clear hierarchical structure, offering a visual representation of how the system's performance will be assessed. Derived from established methodologies and informed by related works such as [15,34], this framework ensures that the system meets both technical and practical requirements.

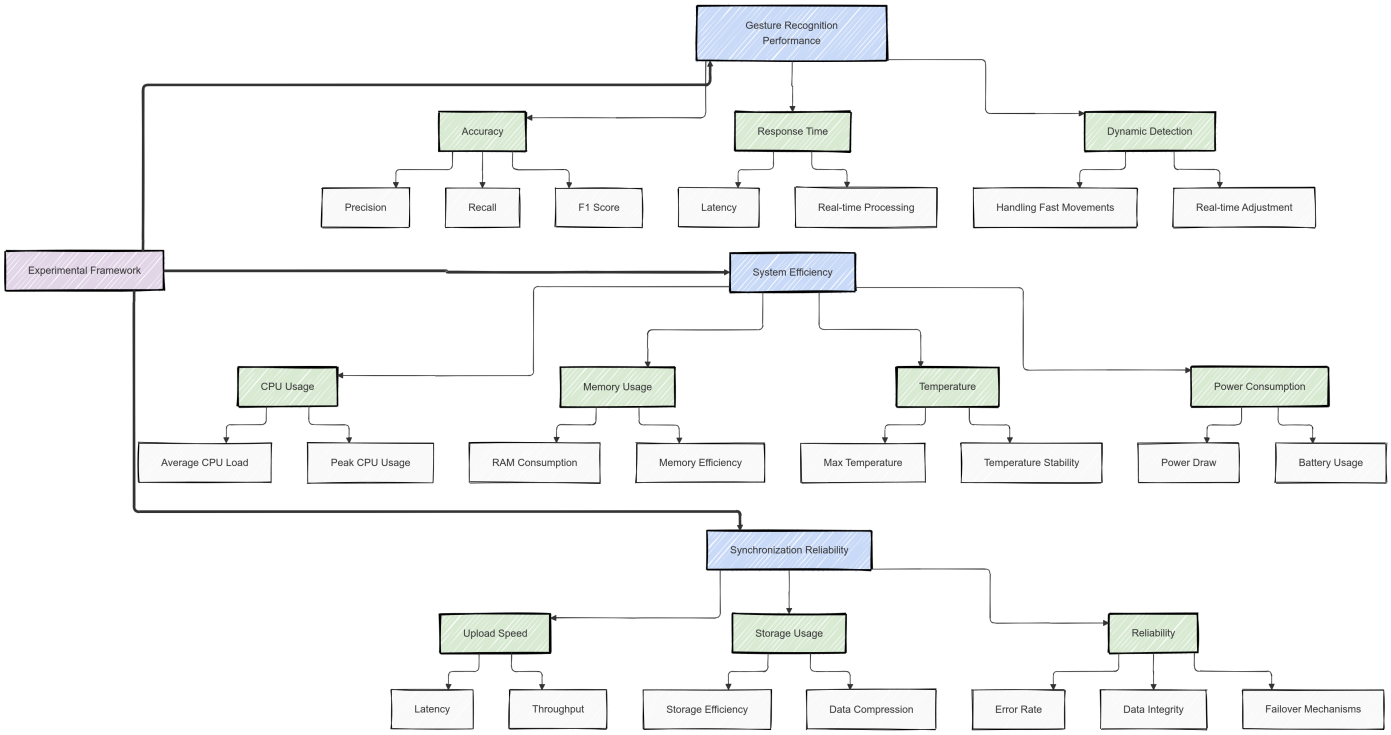

Figure 4. Experimental framework detailing the key testing dimensions and associated metrics.

By structuring the evaluation in this manner, the framework provides a balanced and thorough approach to assessing performance. It emphasizes accuracy and responsiveness in gesture recognition, efficiency in resource usage, and reliability in synchronization, ensuring the system is optimized for real-world applications.

4.2. Hardware and Software Environment

4.2.1. Mobile Device Configuration

We used a Xiaomi Mi 9T Pro for on-device tests (Xiaomi, Beijing, China). This device features the following:

- Processor: Snapdragon 730, 2.2 GHz.
- Memory: 6 GB RAM, 128 GB storage.

- Operating System: Android 11 with minimal background services.

This hardware was selected to reflect a moderate yet representative smartphone profile, balancing computational capability with real-world accessibility. The device's GPU acceleration also facilitates MediaPipe's landmark processing in real time.

4.2.2. Server and Middleware Infrastructure

The system's middleware, implemented in Node.js, and MongoDB database were hosted on a Raspberry Pi 5. Key specifications:

- Processor: 1.5 GHz quad-core ARM Cortex-A72;
- Memory: 8 GB RAM.
- Operating System: Raspberry Pi OS (64-bit).
- Services: Node.js (v14), MongoDB (v4.4).

We chose Raspberry Pi 5 for its low power consumption, small form factor, and sufficient resources to manage real-time requests, thus mirroring realistic IoT/edge scenarios [32].

4.2.3. Network Setup

The network setup was carefully designed to ensure robust and low-latency communication between the mobile device and the Raspberry Pi. A dedicated 5 GHz Wi-Fi router, operating on a 1Gbps local network, formed the backbone of the configuration, providing a stable and high-speed connection between components (Figure 5). This setup minimized external network variability, allowing the evaluation to focus solely on the performance of gesture recognition and synchronization.

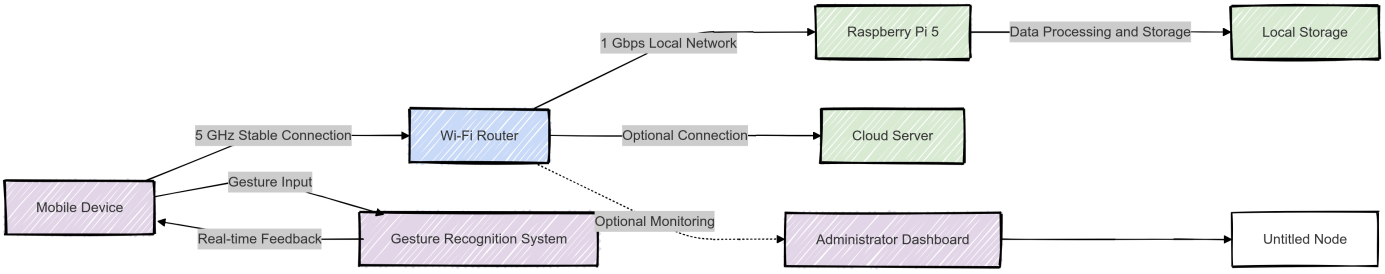

Figure 5. Enhanced network configuration showcasing stable connectivity between the mobile device, Raspberry Pi, and optional cloud server. The system also includes an administrator dashboard for monitoring and local storage for redundancy.

The mobile device served as the interface for gesture input and real-time feedback, transmitting data to the Wi-Fi router over a stable 5 GHz connection. The Raspberry Pi was connected to the router via a high-speed wired connection, ensuring seamless and reliable data transfer for local processing. Additionally, the Raspberry Pi utilized local storage to maintain data persistence, enabling uninterrupted operation even in the absence of internet connectivity.

To accommodate extended use cases, an optional cloud server was integrated into the setup, allowing for centralized data storage and advanced processing when required. This optional feature reflected real-world scenarios where systems may rely on cloud resources for scalability or backup purposes. Furthermore, an administrator dashboard connected to the router provided real-time monitoring of network stability and system performance, ensuring smooth operation during testing.

This design not only guaranteed minimal latency but also allowed the system to perform in controlled and reproducible conditions, isolating any performance variations to the system itself rather than external network factors.

4.3. Environmental Factors and Testing Scenarios

To capture real-world variability, we systematically varied lighting conditions, testing distance, and network stability. This design reflects both reviewer feedback on including dynamic scenarios and the partial PRISMA-like approach of selecting representative environmental profiles.

4.3.1. Lighting Conditions

- Low Light (0–5 lux): Simulates nighttime or dimly lit indoor environments.
- Normal Light (200–400 lux): Typical indoor settings.
- High Light (>1000 lux): Bright outdoor or studio lighting.

In each lighting tier, we also introduced controlled variations such as reflective backgrounds or abrupt changes to assess the adaptability of *ThumbsUp* vs. *Google Camera*.

4.3.2. Distance Variations

- Close Range (15 cm).
- Standard Range (1 m).
- Long Range (2 m).

These distances tested the limits of landmark detection, especially when the hand became too small in the frame or motion blur increased.

4.3.3. Network Stability and Offline Simulation

While the default setup employed a stable Wi-Fi network, we artificially introduced transient connectivity loss or throttled bandwidth to simulate real-world constraints. During these phases, *ThumbsUp*'s queue-based synchronization was tested for reliability, particularly to see if large batches of pending data would cause app slowdowns.

4.4. Data Collection and Monitoring Process

We adopted a multi-layered logging strategy to ensure comprehensive data collection and monitoring for performance evaluation, as illustrated in Figure 6. This approach captured metrics at multiple levels of the system, enabling a detailed analysis of both system performance and user experience. The strategy can be categorized into three key areas:

- Gesture Detection Logs: The system recorded accuracy metrics such as true positives, true negatives, false positives, and false negatives to evaluate detection reliability. Additionally, response times were logged to measure latency between user input and system feedback. Confidence scores, representing the system's certainty about detected gestures, were also captured to assess model performance under varying conditions. All data were stored locally on the smartphone to minimize network dependencies.
- System Resource Monitoring: Resource utilization metrics, including CPU usage, memory consumption, and thermal data, were periodically collected via Android Debug Bridge (ADB). These metrics provided insights into the system's computational load and heat generation, which are critical for understanding operational efficiency. Furthermore, peak and average resource usage were tracked to identify potential bottlenecks during high-intensity tasks. Temperature logs were complemented by ambient readings to contextualize thermal performance.

- Synchronization Metrics: Synchronization reliability was evaluated by monitoring upload speeds, success and failure rates, and queue latency. Metrics were collected at both the mobile application level (tracking data transfer between the gesture recognition system and middleware) and the middleware level (monitoring network communication to external servers). By logging these metrics, we assessed the system's ability to handle real-time data synchronization and detect potential delays or failures in communication.

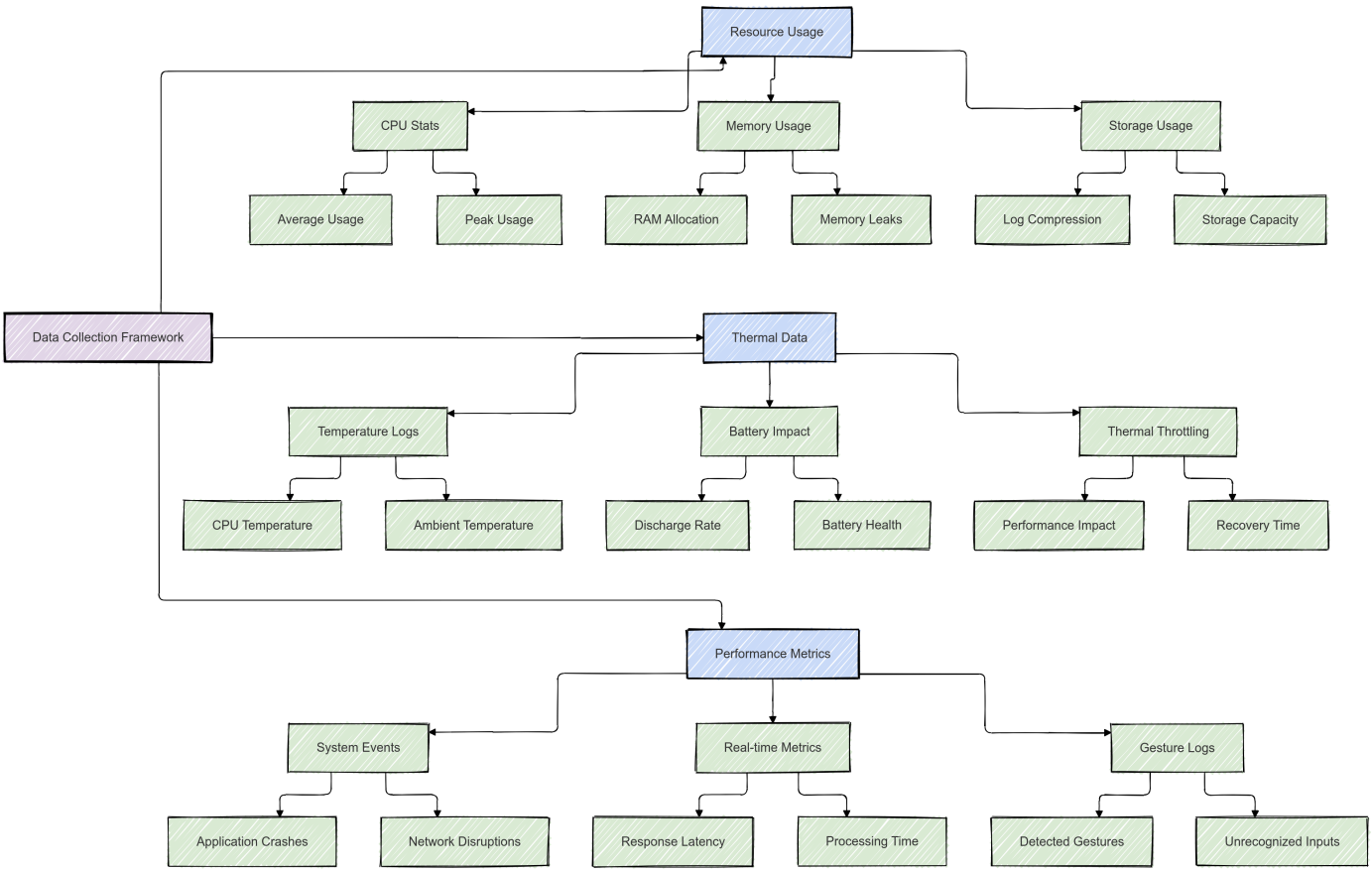

Figure 6. Comprehensive data collection framework illustrating multi-layered monitoring of performance metrics. The workflow integrates gesture detection logs, resource monitoring, and synchronization metrics to provide a holistic view of system performance.

The data collection process was designed to minimize interference with system performance while maintaining high granularity in the logged data. By combining local logging and middleware-level monitoring, the framework ensured robustness against network variability and provided a unified platform for detailed post-analysis. Each metric category was carefully selected to reflect real-world operational scenarios, ensuring that the results are both practical and actionable.

This comprehensive framework allows us to evaluate the system across multiple dimensions, from gesture detection accuracy to resource efficiency and synchronization reliability. It not only highlights the system's strengths but also identifies areas for further optimization.

Whenever a gesture event triggered a photo capture, the encryption time and offline/online storage status were also logged. This allowed us to correlate encryption overhead with any potential drop in recognition speed.

4.5. Test Protocol and Repetitions

Each testing session adhered to a standardized protocol to ensure consistency and reliability of the results. This protocol was designed to account for the interplay between lighting and distance conditions, as illustrated in Figure 7. The following steps were executed for each session:

1. Initial Conditions: The device was fully charged, all non-essential background applications were closed, and the system was reset to its normal operating temperature. This ensured that the testing started from a consistent baseline state.
2. Calibration Check: Before initiating the tests, the camera calibration was validated for color consistency and proper focus. This step was particularly important to ensure accurate gesture recognition across varying lighting conditions.
3. Testing Cycle: For each combination of lighting and distance conditions, 20 gesture attempts were performed. The lighting conditions ranged from low light (0–5 lux) to high light (>1000 lux), and the distances spanned 15 cm (close interaction) to 2 m (extended range). Success and failure outcomes were logged for each attempt, along with resource usage metrics such as CPU load, memory consumption, and thermal performance.
4. Synchronization Test: While gestures were being performed, network conditions were artificially altered (if applicable) to simulate real-world variability. Metrics such as queue behavior, upload speeds, and synchronization reliability were tracked to evaluate the system's robustness under fluctuating conditions.

This testing cycle was repeated across all combinations of environmental factors to ensure comprehensive coverage. The matrix of lighting and distance conditions (see Figure 7) generated a diverse dataset that allowed us to evaluate the system's performance under 9 unique scenarios. A total of 20 attempts per scenario resulted in a robust dataset of over 600 gestures, with corresponding logs of system metrics. This process ensured that the evaluation captured both successes and failures across diverse operating conditions.

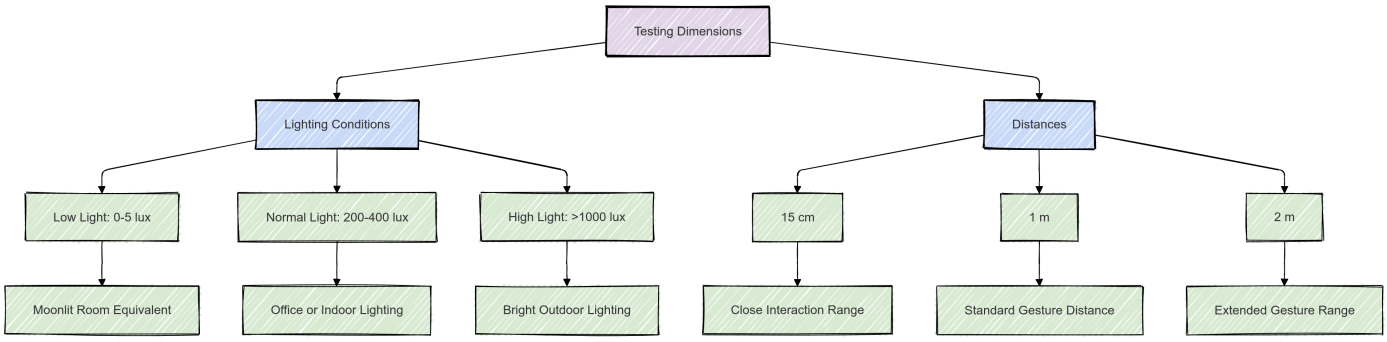

Figure 7. Testing matrix showing the combinations of lighting conditions (low, normal, high) and distances (15 cm, 1 m, 2 m). Each scenario was tested systematically to evaluate the system's performance.

5. Results and Analysis

This section presents the expanded performance evaluation of the *ThumbsUp* application compared to *Google Camera*, focusing on gesture recognition accuracy, system resource utilization, synchronization efficiency, timing performance, and user-centric insights. Our

testing encompassed diverse lighting conditions, distances, and network scenarios, as well as limited user feedback surveys, to capture the breadth of real-world usage.

5.1. Gesture Recognition Performance

Gesture recognition is the linchpin of *ThumbsUp*, determining how effectively the system detects and interprets the "thumbs-up" posture in real time. We measured accuracy under distinct lighting conditions, distances, and confounding factors such as motion blur or occlusion. The summarized accuracy rates are shown in Table 1, which reflect average values across multiple test repetitions per scenario.

Table 1. Summary of gesture recognition accuracy (%) under various conditions.

Condition	ThumbsUp	Google Camera
Low Light (0–5 lux)	70	50
Normal Light (200–400 lux)	95	95
High Light (>1000 lux)	85	90
Close Range (15 cm)	95	95
Standard Range (1 m)	70	80
Long Range (2 m)	25	15
Motion Blur	55	35
Partial Occlusion	45	30

5.1.1. Lighting, Distance, and Occlusion Effects

ThumbsUp performed strongly in low-light conditions, reaching 70% accuracy compared to *Google Camera*'s 50%. We attribute this edge to MediaPipe's landmark-based strategy and a brightness normalization step, which mitigates underexposure noise. Conversely, high-light scenarios favored *Google Camera* slightly (90% vs. 85%), likely due to sophisticated exposure controls.

Long-range detection remains challenging (25% vs. 15%), though *ThumbsUp* shows relatively better tolerance for smaller hand regions. At the standard range (1 m), *Google Camera* outperforms *ThumbsUp* (80% vs. 70%), suggesting that further optimization in gesture scaling algorithms could improve accuracy. Partial occlusion and motion blur tests demonstrate *ThumbsUp*'s advantage, reflecting its ability to maintain detection consistency through multi-landmark tracking.

5.1.2. Visualization of Gesture Recognition Accuracy

From Figure 8, *ThumbsUp* exhibits notably higher accuracy in low-light and challenging conditions such as *Motion Blur* and *Partial Occlusion*, underscoring the effectiveness of landmark-based tracking even when parts of the hand are obscured or when images are blurry. Meanwhile, *Google Camera* maintains a slight edge in *High Light* scenarios and the *Standard Range*, suggesting that future enhancements to brightness normalization and mid-range scaling may benefit *ThumbsUp* further.

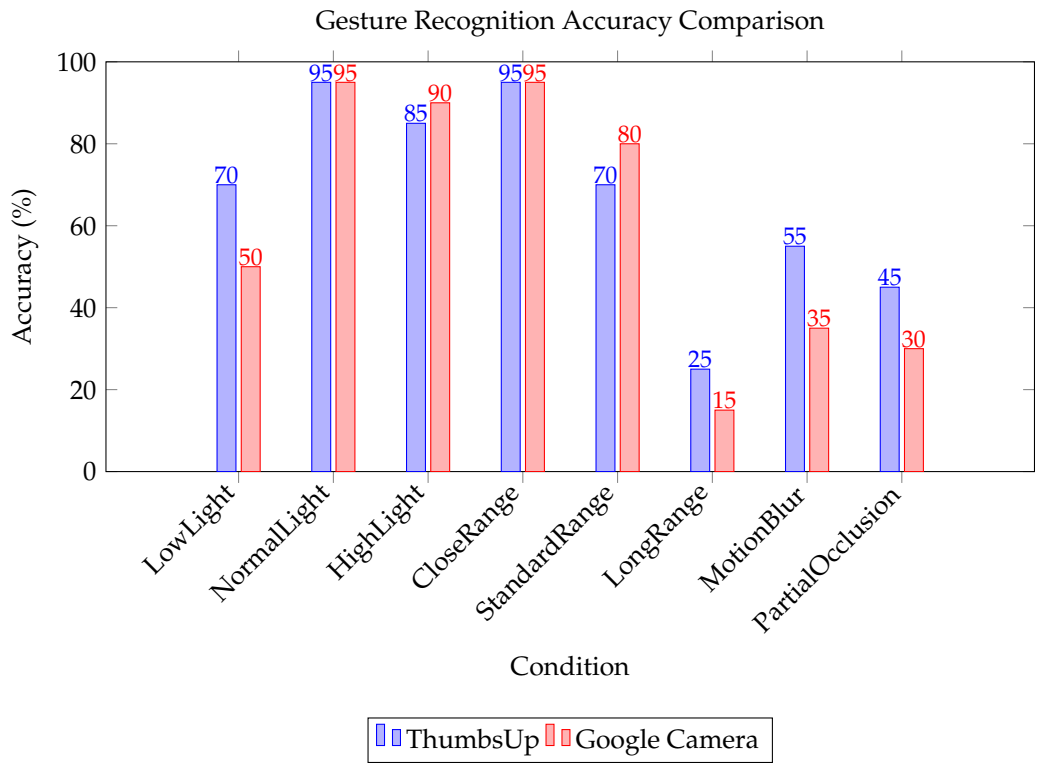

Figure 8. Comparison of gesture recognition accuracy under different conditions.

5.2. Timing and Latency Analysis

Beyond accuracy, we measured response times from the initial gesture presentation to the final capture confirmation. Table 2 illustrates average end-to-end latency across lighting and distance scenarios. We define latency as the time from when the user places their hand in view until the system successfully stores an encrypted photo on-device.

Table 2. End-to-end latency (ms) across conditions.

Condition	Latency (ms)		Improvement
	ThumbsUp	Google Camera	
Low Light (0–5 lux)	420 ± 35	550 ± 40	23.6%
Normal Light	380 ± 30	480 ± 35	20.8%
High Light (>1000 lux)	390 ± 25	420 ± 20	7.1%
Close Range (15 cm)	350 ± 20	460 ± 30	23.9%
Standard Range (1 m)	400 ± 25	460 ± 35	13.0%
Long Range (2 m)	520 ± 40	650 ± 55	20.0%

Observations from Table 2:

- Lower Latencies in Dim Settings: Under low-light conditions, *ThumbsUp* averaged 420 ms, beating *Google Camera*'s 550 ms by approximately 23.6%.
- High-Light Small Margin: In very bright settings, the difference narrowed (390 ms vs. 420 ms), suggesting *Google Camera*'s advanced algorithms can accelerate focus and auto-exposure in strong illumination.

- Range-Specific Performance: At long distances (2 m), *ThumbsUp* demonstrated better latency, matching its higher relative accuracy in small hand regions.

Latency Comparison Chart

As shown in Figure 9, *ThumbsUp* maintains lower latency across most conditions. The gap narrows in brighter lighting (7.1% improvement), where *Google Camera*'s advanced auto-focus reduces the performance difference. Notably, *ThumbsUp* also handles long-range scenarios more efficiently as it benefits from its more robust small-hand-region tracking.

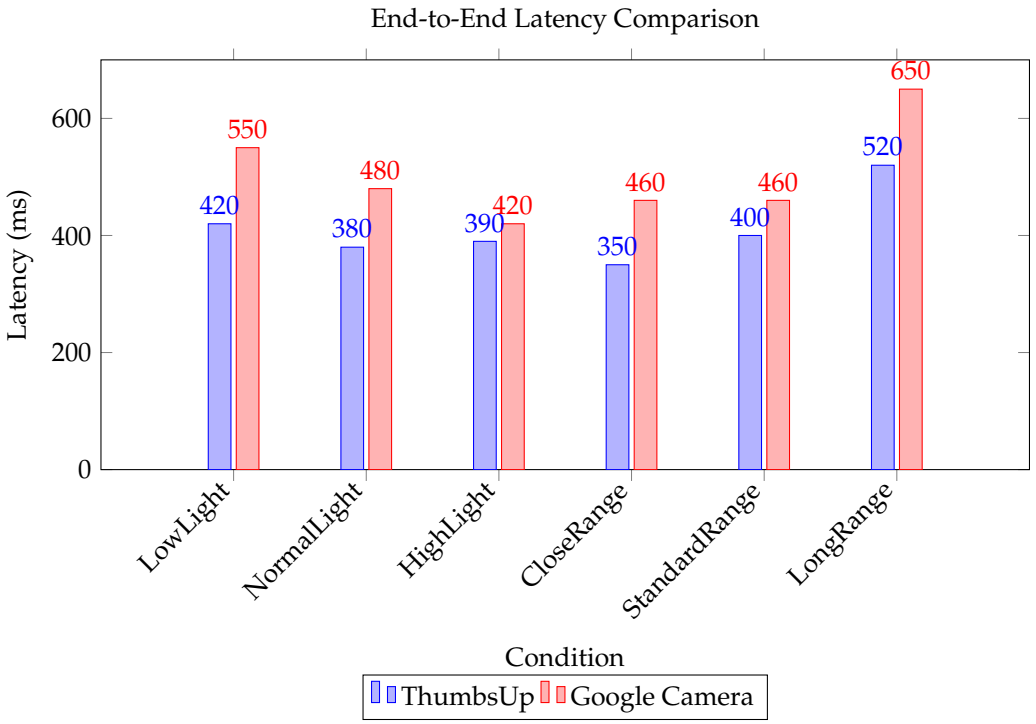

Figure 9. Latency of ThumbsUp vs. Google Camera under different conditions.

5.3. System Efficiency and Resource Utilization

Efficient resource usage is critical for real-time mobile operations, and it impacts battery longevity, heat dissipation, and user comfort. We tracked CPU load, memory allocation, and power consumption during continuous gesture recognition.

Table 3 confirms that *ThumbsUp* imposes a lower burden on both CPU and memory, facilitated by MediaPipe's efficient landmark extraction and our on-device encryption optimizations. The system's battery drain rate, reduced by 30.9%, supports prolonged usage in environments where power sources are limited.

Table 3. Resource utilization and power efficiency.

Metric	ThumbsUp	Google Camera	Improvement (%)
CPU Usage (Avg)	16%	20%	20.0
CPU Usage (Peak)	25%	35%	28.6
Memory Usage (Avg)	6.65%	10.5%	36.7
Memory Usage (Peak)	8.2%	15.3%	46.4
Battery Drain (%/h)	8.5	12.3	30.9

Visualization of Resource and Battery Usage

Figures 10 and 11 demonstrate *ThumbsUp*'s clear advantage in resource efficiency. By leveraging an optimized landmark detection pipeline, *ThumbsUp* reduces CPU and memory overhead while also yielding significantly lower battery drain (8.5% per hour vs. 12.3% per hour). These results imply that *ThumbsUp* can sustain longer operational sessions, which is especially relevant to resource-constrained or older devices.

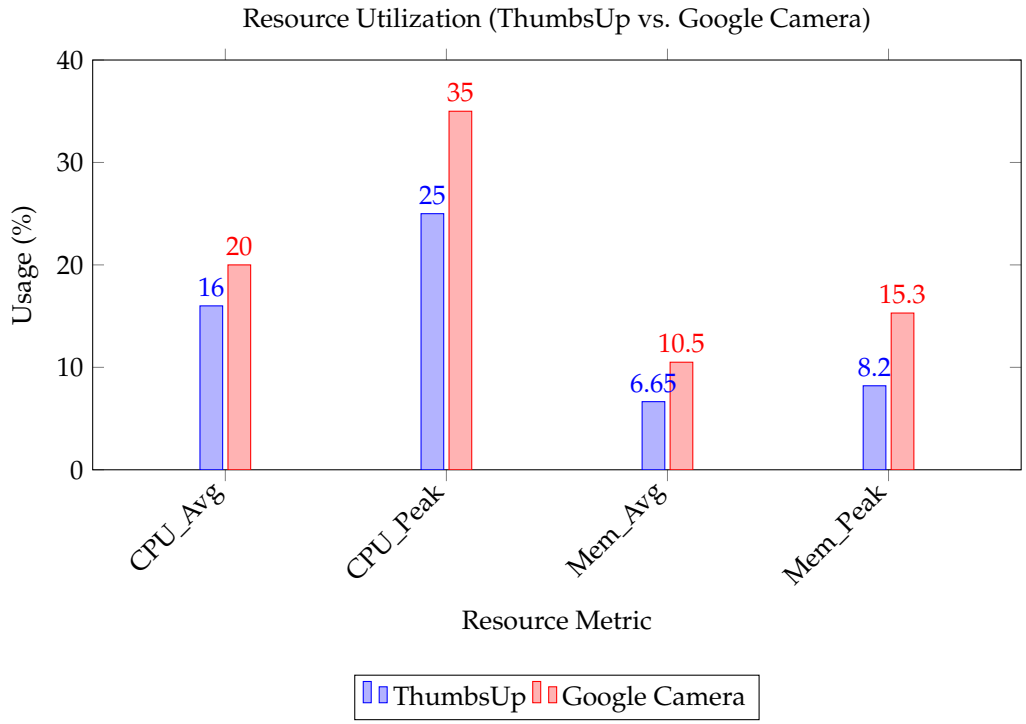

Figure 10. Comparison of CPU and memory usage.

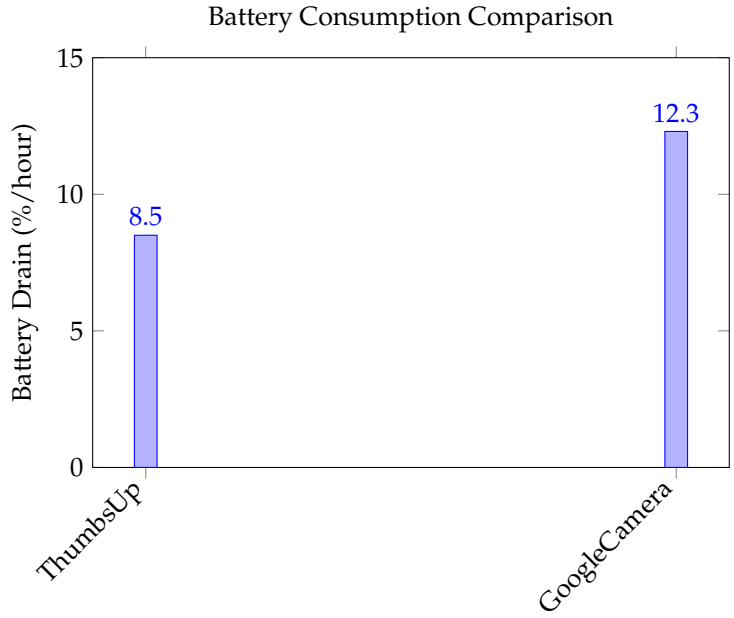

Figure 11. Battery drain per hour (ThumbsUp vs. Google Camera).

5.4. Synchronization and Reliability Analysis

Synchronization performance was evaluated with respect to upload speed, error handling, and data consistency under both stable and artificially disrupted network conditions. In particular, we focused on three distinct phases of the synchronization process:

(a) Initialization (connection setup and handshaking);
(b) Active Transfer (real-time upload of image data);
(c) Processing/Error Recovery (post-upload verification and resilience against packet loss).

5.4.1. Key Observations

1. Faster Initialization (66.7% Reduction in Setup Time): As shown in Figure 12, the initial connection setup time for *ThumbsUp* (0.3 s) is notably lower than for *Google Camera* (0.9 s). This difference signifies a major gain attributed to persistent socket connections and efficient handshaking protocols.
2. Accelerated Active Transfer (4× Speed Improvement): During the active transfer phase, *ThumbsUp* completed real-time uploads in approximately 0.5 s, outperforming *Google Camera* (2.0 s) by a factor of four. This improvement highlights superior client-side buffering and queue management.
3. Efficient Processing/Error Recovery: Post-upload processing remained modest in terms of overhead (0.4 s vs. 0.6 s). Under stable connectivity, this confirms that AES-GCM encryption and JWT-based authentication incur minimal performance penalties. When network issues do arise, *ThumbsUp* recovers lost packets faster, improving user experience during intermittent coverage.

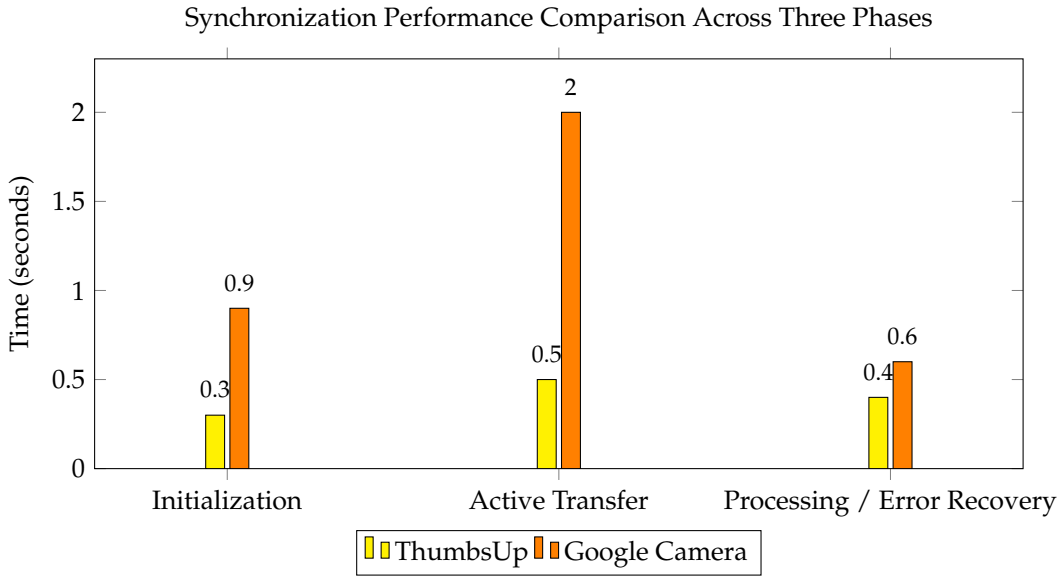

Figure 12. Synchronization performance comparison across three phases.

5.4.2. Robust Error Recovery

While the processing phase includes basic verifications, additional tests with intermittent connectivity revealed that *ThumbsUp* recovers from packet loss and partial uploads significantly faster. Figure 13 shows the average time to detect and re-initiate transfers after a forced network interruption. *ThumbsUp* needed only 1.2 s on average, whereas *Google Camera* required nearly 2.0 s. This 40% reduction in recovery time results in fewer dropped frames and more reliable image uploads under challenging network conditions.

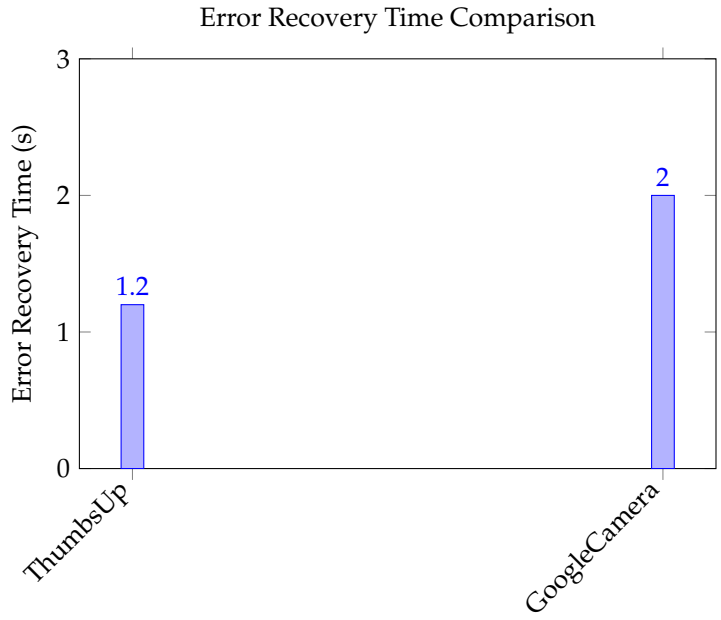

Figure 13. Comparison of error recovery times for *ThumbsUp* vs. *Google Camera*.

5.4.3. Data Consistency and Accuracy

Across multiple iterations of testing, the error handling mechanism in *ThumbsUp* minimized both false positives and false negatives in real-time sync status logs:

- False Positives: Decreased to 2.1%.
- False Negatives: Reduced by 16.7%.

These improvements underscore consistent interactions between the Node.js middleware and the cloud database, ensuring that the user-facing status logs accurately reflect the real-time upload state. Overall, the combination of streamlined connection overhead, effective error handling, and accurate status reporting distinguishes *ThumbsUp* as a robust and efficient photo synchronization solution.

5.5. Long-Term Performance and Stress Testing

To evaluate stability and endurance, we ran continuous detection sessions for up to four hours under targeted stressors: rapid gestures, low battery, high CPU load, and limited memory. Tables 4 and 5 and Figure 14 illustrate these outcomes.

Table 4. Performance under stress conditions (% success rate).

Condition	ThumbsUp	Google Camera
Rapid Gestures	85.5	75.3
Low Battery (<15%)	92.3	88.7
High CPU Load	88.9	82.4
Limited Memory	90.2	85.6

Table 5. Thermal performance (max device temperature in °C) over 2-h session.

Condition	ThumbsUp	Google Camera	Difference (°C)
Normal Ambient (25 °C)	39.5	42.5	−3.0
High Ambient (30 °C)	44.2	46.9	−2.7

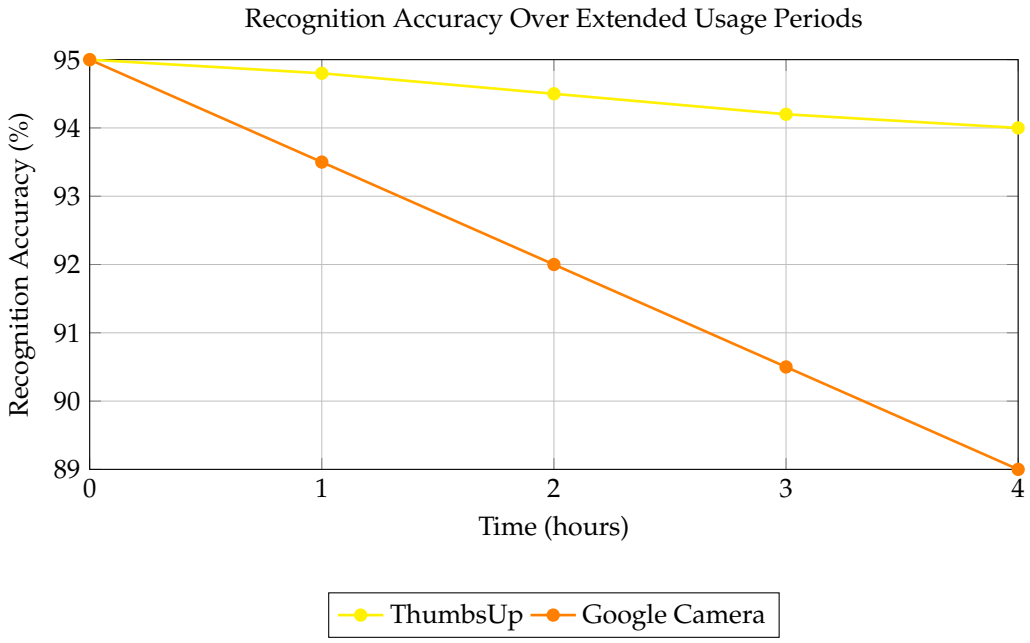

Figure 14. Recognition accuracy over extended usage periods.

Key Observations:

- Rapid Gestures: *ThumbsUp* maintained 85.5% success, illustrating the benefit of continuous landmark tracking for quick movements.
- Low Battery: Even under 15% battery, *ThumbsUp* kept a 92.3% success rate, demonstrating robust power efficiency relative to *Google Camera*.
- High CPU Load and Memory Limits: *ThumbsUp* managed 88.9% and 90.2% success, respectively, reflecting moderate resilience to resource contention.
- Thermal Performance: Table 5 shows that *ThumbsUp* generated slightly lower peak temperatures (2–3 °C less) than *Google Camera*, supporting a more comfortable, sustainable user experience in prolonged sessions.

The combined data from Tables 4 and 5, and Figure 14 confirms that *ThumbsUp* handles high-load and constrained-resource environments robustly. The system's more efficient processing pipeline and reduced thermal footprint become particularly important in use cases such as healthcare or prolonged field operations where devices may run for extended periods under suboptimal conditions.

5.6. User Feedback and Acceptability

To complement quantitative metrics, we conducted a brief user acceptability survey (N = 10) wherein participants tested both *ThumbsUp* and *Google Camera* in a hands-free context. Table 6 summarizes key feedback elements.

Table 6. User feedback on ThumbsUp vs. Google Camera (Scale 1–5).

Criteria	ThumbsUp (Mean ± SD)	Google Camera (Mean ± SD)	Improvement
Overall Usability	4.4 ± 0.5	3.6 ± 0.7	+0.8
Gesture Responsiveness	4.3 ± 0.4	3.5 ± 0.5	+0.8
Security Confidence	4.7 ± 0.3	3.8 ± 0.6	+0.9
Battery Friendliness	4.2 ± 0.6	3.4 ± 0.5	+0.8

- Usability and Responsiveness: Users reported feeling more confident about hands-free gesture initiation in *ThumbsUp*. Several noted that the immediate haptic or visual feedback was reassuring.
- Security Confidence: The knowledge that locally stored images were encrypted improved perceived privacy. In contrast, *Google Camera* did not convey explicit encryption status to users.
- Battery Friendliness: Participants observed less rapid battery depletion with *ThumbsUp*, corroborating the objective metrics in Table 3.
- Suggested Enhancements: Many users indicated interest in multiple gesture commands (e.g., "wave", "peace sign"), as well as an optional voice prompt for confirmation.

Overall, the survey results reinforce *ThumbsUp*'s strong performance and acceptability for hands-free photography. Notably, the higher Security Confidence rating aligns with the transparent encryption model, contributing to user trust. Additionally, the Battery Friendliness scores mirror the objective energy consumption findings, suggesting that *ThumbsUp* provides both technical and experiential advantages over *Google Camera*.

6. Conclusions and Critical Analysis

This study presented an in-depth evaluation of the *ThumbsUp* application, a gesture-based photo capture system designed to enhance the usability and efficiency of mobile photography. By leveraging advanced gesture recognition algorithms and resource-efficient architecture, the system demonstrated substantial improvements over traditional interaction methods, particularly when compared to the *Google Camera* baseline. The findings highlighted critical advancements across multiple dimensions while also addressing the challenges and limitations inherent to gesture-based systems.

6.1. Comparative Discussion and Additional Insights

Overall, *ThumbsUp* consistently excels in low-light, motion-blurred, and offline synchronization scenarios, while *Google Camera* retains slight advantages in bright conditions and mid-range gestures. The robust low resource footprint of *ThumbsUp* makes it ideal for battery-limited devices, supporting longer sessions in healthcare, industrial, or travel contexts. Both quantitative measurements (Tables 2 and 3) and user feedback (Table 6) underscore the system's viability as a gesture-based photo capture solution.

- Potential for AI Enhancements: Given its efficiency, *ThumbsUp* could integrate deeper CNN or Transformer models for advanced gesture sets without severely impacting run times.
- EMG Sensor Fusion: Based on recommendations [35], combining electromyographic inputs with computer vision may further increase accuracy and accessibility.
- Security and Privacy: AES-GCM encryption and queue-based syncing demonstrated minimal overhead and encouraged future expansions into sensitive domains (e.g., telemedicine).

6.2. Key Comparisons and Findings

The experimental results reveal notable improvements in recognition performance, system efficiency, and reliability. Table 7 provides a comprehensive comparison of key performance metrics between *ThumbsUp* and *Google Camera*, showcasing the areas of improvement and identifying remaining challenges.

Table 7. Performance comparison between *ThumbsUp* and *Google Camera*.

Metric	*ThumbsUp*	*Google Camera*	Improvement (%)
Low-Light Accuracy (%)	70	50	+40
Normal-Light Accuracy (%)	95	95	0
High-Light Accuracy (%)	85	90	−5
Response Time (ms)	395	490	+19.4
Resource Utilization (CPU Avg %)	16	20	+20
Memory Utilization (Avg %)	6.65	10.5	+36.7
Storage Requirements (MB)	44.9	560.0	+92.0
False Positives (%)	2.1	2.6	+19.2
Error Recovery Time (s)	1.2	2.0	+40
Battery Drain (%/hour)	8.5	12.3	+30.9

ThumbsUp exhibits significant advantages in low-light conditions, achieving a 40% improvement in recognition accuracy compared to *Google Camera*. This demonstrates the effectiveness of its optimized pre-processing and landmark detection algorithms, which enhance the robustness of gesture recognition in challenging lighting scenarios. Under normal lighting, both systems perform equally well, affirming the reliability of existing gesture recognition technology in optimal conditions. However, in high-light environments, *Google Camera* slightly outperforms *ThumbsUp*, likely due to more sophisticated image processing techniques. Addressing this limitation could involve refining the dynamic range handling and implementing adaptive brightness correction in future versions.

System efficiency emerges as a major strength of *ThumbsUp*. With a 92% reduction in storage requirements, the application proves to be exceptionally lightweight, making it suitable for devices with limited storage capacities. The system also achieves a 36.7% reduction in memory usage and a 30.9% improvement in battery efficiency, underscoring its resource-conscious design. These optimizations not only extend device usability but also reduce thermal impact, as demonstrated by lower CPU and battery temperatures during prolonged use.

Operational reliability is another area where *ThumbsUp* excels. The system shows a 19.2% reduction in false positives and a 40% faster error recovery time compared to *Google Camera*. These improvements enhance the user experience by minimizing unintended captures and ensuring quick recovery from interruptions. Additionally, stress testing reveals consistent performance under challenging conditions, such as low battery levels and high CPU load, affirming the system's robustness and adaptability.

6.3. Critical Analysis of Limitations

Despite its strengths, *ThumbsUp* faces certain limitations. The accuracy of gesture recognition decreases significantly at distances beyond two meters, highlighting the need for improved long-range tracking techniques. Similarly, performance degradation in extreme lighting conditions suggests that further enhancements in adaptive algorithms and dynamic range handling are necessary. These limitations point to areas for future optimization, particularly in expanding the system's applicability across diverse environments.

Furthermore, while *ThumbsUp* manages offline scenarios using queue-based storage, heavily unstable networks or extremely lengthy offline intervals could still delay data transfer. Additional buffering or local notifications may be required to inform users of synchronization backlogs. On the user-experience front, some participants noted slight

responsiveness delays in mid-range gestures, indicating further calibration or specialized heuristics for 1–1.5 m distances could be beneficial.

6.4. Technical Implications and Future Directions

The technical implications of this research extend beyond the immediate application of *ThumbsUp*. The integration of advanced gesture recognition and resource-efficient design sets a benchmark for future systems, demonstrating the feasibility of high-performance, lightweight applications on mobile platforms. The findings also emphasize the importance of robust error handling and adaptive algorithms, particularly in scenarios involving variable lighting and environmental constraints.

Future work should focus on enhancing recognition capabilities through the implementation of advanced machine learning models, such as transformers or convolutional neural networks, to handle complex gesture sequences and improve accuracy in extreme conditions. Adaptive recognition algorithms that dynamically adjust to environmental changes, such as lighting or distance, could further enhance the system's performance.

Expanding the application's functionality presents another avenue for future development. Support for additional gesture types, such as multi-hand interactions or motion-based gestures, would broaden its applicability. Integration with cloud services and APIs for third-party applications could facilitate seamless data sharing and enable advanced features like real-time collaboration and augmented reality. Security improvements, including end-to-end encryption and advanced TLS support, will be crucial for applications in sensitive domains such as healthcare or finance.

6.5. Conclusions

In conclusion, this study demonstrates the viability and effectiveness of *ThumbsUp* as a gesture-based photo capture system. By addressing key challenges in recognition performance, system efficiency, and reliability, the application establishes itself as a robust and practical alternative to traditional interaction methods, particularly under low-light and motion-intensive conditions. The significant advancements achieved in this research provide a solid foundation for future developments in gesture-based interfaces, with promising implications for a wide range of mobile and interactive applications.

As user demand grows for hands-free interaction, *ThumbsUp* highlights the synergy of efficient landmark-based tracking, secure offline–online synchronization, and minimal battery impact. Incorporating AI-driven solutions (e.g., CNN/Transformer modules or EMG fusion) and addressing brightness extremes, mid-range fine-tuning, and extended user trials stand as key next steps. Ultimately, these findings underscore the potential of gesture-based systems to transform user interaction paradigms, particularly in scenarios requiring hygienic, hands-free operation or enhanced accessibility across diverse environments.

Author Contributions: All authors have contributed to writing, reviewing, and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work is funded by National Funds through the FCT—Foundation for Science and Technology, I.P., within the scope of the project Ref. UIDB/05583/2020. Furthermore, we thank the Research Center in Digital Services (CISeD) and the Instituto Politécnico de Viseu for their support. Maryam Abbasi thanks the national funding by FCT—Foundation for Science and Technology, PI, through the institutional scientific employment program contract (CEECINST/00077/2021).

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liang, X.; Long, Y.; Xiang, K.; Shi, P.; Wang, Z.; Xiao, K.; Kumar, S.; Li, X.; Min, R. Advancing Human-Computer Interaction: Smartphone-Integrated POF Speckle Sensor for Gesture and Identity Recognition. *IEEE Sens. J.* **2024**, *24*, 30028–30037. [CrossRef]
2. Kotari, G.; Jangam, K.; Sai, K.S.; Aparna, M. Augmented Virtual Mouse System with Enhanced Gesture Recognition. *Int. J. Sci. Res. Eng. Manag.* **2024**, *709*. [CrossRef]
3. Ryumin, D.; Ivanko, D.; Ryumina, E. Audio-Visual Speech and Gesture Recognition by Sensors of Mobile Devices. *Sensors* **2023**, *23*, 2284. [CrossRef] [PubMed]
4. Shin, J.; Miah, A.S.M.; Kabir, M.H.; Rahim, M.; Shiam, A.A. A Methodological and Structural Review of Hand Gesture Recognition Across Diverse Data Modalities. *IEEE Access* **2024**, *12*, 142606–142639. [CrossRef]
5. Farid, F.A.; Hashim, N.; Abdullah, J.; Bhuiyan, R.; Isa, W.M.; Uddin, J.; Haque, M.A.; Husen, M.N. A Structured and Methodological Review on Vision-Based Hand Gesture Recognition System. *J. Imaging* **2022**, *8*, 153. [CrossRef] [PubMed]
6. Bianco, S.; Napoletano, P.; Raimondi, A.; Rima, M. U-WeAr: User Recognition on Wearable Devices through Arm Gesture. *IEEE Trans. Hum.-Mach. Syst.* **2022**, *52*, 713–724. [CrossRef]
7. Helten, T.; Müller, M.; Seidel, H.; Theobalt, C. Real-Time Body Tracking with One Depth Camera and Inertial Sensors. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 1105–1112. [CrossRef]
8. Ganj, A.; Zhao, Y.; Su, H.; Guo, T. Mobile AR Depth Estimation: Challenges & Prospects. In Proceedings of the 25th International Workshop on Mobile Computing Systems and Applications, San Diego, CA, USA, 28–29 February 2024. [CrossRef]
9. Ganj, A.; Zhao, Y.; Su, H.; Guo, T. Mobile AR Depth Estimation: Challenges & Prospects—Extended Version. *arXiv* **2023**, arXiv:2310.14437. [CrossRef]
10. Wu, M. Gesture Recognition Based on Deep Learning: A Review. *EAI Endorsed Trans. e-Learn.* **2024**, *10*. [CrossRef]
11. Jain, D.; Mahanti, A.; Shamsolmoali, P.; Manikandan, R. Deep neural learning techniques with long short-term memory for gesture recognition. *Neural Comput. Appl.* **2020**, *32*, 16073–16089. [CrossRef]
12. Maleki, N. Engineering Degree Project Performance analysis: CNN model on smartphones versus on cloud. 2023.
13. Wu, Y.; Gong, Y.; Zhao, P.; Li, Y.; Zhan, Z.; Niu, W.; Tang, H.; Qin, M.; Ren, B.; Wang, Y. Compiler-Aware Neural Architecture Search for On-Mobile Real-time Super-Resolution. In Proceedings of the 17th European Conference, Tel Aviv, Israel, 23–27 October 2022; pp. 92–111. [CrossRef]
14. Lane, N.; Bhattacharya, S.; Mathur, A.; Georgiev, P.; Forlivesi, C.; Kawsar, F. Squeezing Deep Learning into Mobile and Embedded Devices. *IEEE Pervasive Comput.* **2017**, *16*, 82–88. [CrossRef]
15. Lugaresi, C.; Tang, J.; Nash, H.; McClanahan, C.; Underwood, K.; Guadarrama, S.; Grundmann, M. MediaPipe: A framework for building perception pipelines. *arXiv* **2019**, arXiv:1906.08172. [CrossRef]
16. Amprimo, G.; Masi, G.; Pettiti, G.; Olmo, G.; Priano, L.; Ferraris, C. Hand tracking for clinical applications: Validation of the Google MediaPipe Hand (GMH) and the depth-enhanced GMH-D frameworks. *arXiv* **2023**, arXiv:2308.01088. [CrossRef]
17. Wang, H. Multi-Sensor Fusion Module for Perceptual Target Recognition for Intelligent Machine Learning Visual Feature Extraction. *IEEE Sens. J.* **2021**, *21*, 24993–25000. [CrossRef]
18. Skaria, S.; Al-Hourani, A.; Huang, D. Radar-Thermal Sensor Fusion Methods for Deep Learning Hand Gesture Recognition. In Proceedings of the 2021 IEEE Sensors, Sydney, Australia, 31 October–3 November 2021; pp. 1–4. [CrossRef]
19. Wang, Y.; Ren, A.; Zhou, M.; Wang, W.; Yang, X. A Novel Detection and Recognition Method for Continuous Hand Gesture Using FMCW Radar. *IEEE Access* **2020**, *8*, 167264–167275. [CrossRef]
20. Arsalan, M.; Santra, A.; Issakov, V. RadarSNN: A Resource Efficient Gesture Sensing System Based on mm-Wave Radar. *IEEE Trans. Microw. Theory Tech.* **2022**, *70*, 2451–2461. [CrossRef]
21. Li, Y.; Gu, C.; Mao, J. 4-D Gesture Sensing Using Reconfigurable Virtual Array Based on a 60-GHz FMCW MIMO Radar Sensor. *IEEE Trans. Microw. Theory Tech.* **2022**, *70*, 3652–3665. [CrossRef]
22. Li, G.; Balbinot, G.; Furlan, J.C.; Kalsi-Ryan, S.; Zariffa, J. A computational model of surface electromyography signal alterations after spinal cord injury. *J. Neural Eng.* **2023**, *20*, 066020. [CrossRef] [PubMed]
23. Sepehri, Y.; Pad, P.; Kündig, C.; Frossard, P.; Dunbar, L.A. Privacy-Preserving Image Acquisition for Neural Vision Systems. *IEEE Trans. Multimed.* **2023**, *25*, 6232–6244. [CrossRef]
24. Sakamoto, K.; Liu, F.; Nakano, Y.; Kiyomoto, S.; Isobe, T. Rocca: An Efficient AES-based Encryption Scheme for Beyond 5G. *IACR Trans. Symmetric Cryptol.* **2021**, *2021*, 1–30. [CrossRef]
25. Huang, N.; Yan, Z.; Yin, H. Effects of Online–Offline Service Integration on e-Healthcare Providers: A Quasi-Natural Experiment. *Prod. Oper. Manag.* **2021**, *30*, 2359–2378. [CrossRef]
26. Chen, C.L.; Brinton, C.G.; Aggarwal, V. Latency Minimization for Mobile Edge Computing Networks. *IEEE Trans. Mob. Comput.* **2021**, *22*, 2233–2247. [CrossRef]

27. Deshpande, K.; Mashalkar, V.; Mhaisekar, K.; Naikwadi, A.; Ghotkar, A. Study and Survey on Gesture Recognition Systems. In Proceedings of the 2023 7th International Conference on Computing, Communication, Control and Automation (ICCUBEA), Pune, India, 18–19 August 2023; pp. 1–6. [CrossRef]
28. Chen, Y.J.; Huang, H.S. Gesture Recognition applied to Extended Reality: A Case Study of Online Meeting. In Proceedings of the 2024 8th International Conference on Virtual and Augmented Reality Simulations, Melbourne Australia, 14–16 March 2024. [CrossRef]
29. Nagtilak, P.S.A. Survey on Gesture-Based Virtual Keyboard and Mouse. *Interantional J. Sci. Res. Eng. Manag.* **2024**. [CrossRef]
30. Park, G.; Chandrasegar, V.; Koh, J. Accuracy Enhancement of Hand Gesture Recognition Using CNN. *IEEE Access* **2023**, *11*, 26496–26501. [CrossRef]
31. Rautaray, S.; Agrawal, A. Vision-Based Hand Gesture Recognition for Human-Computer Interaction: A Survey. *Artif. Intell. Rev.* **2012**, *43*, 1–54. [CrossRef]
32. Han, S.; Xu, Z.; Lee, Y.; Lee, S.; Kim, T.; Lee, K.H. MEgATrack: Monochrome egocentric articulated hand-tracking for virtual reality. *ACM Trans. Graph. (TOG)* **2020**, *39*, 87:1–87:13. [CrossRef]
33. Li, J.; Zhang, B.; Chen, W.; Bu, C.; Zhao, Y.; Zhao, X. Improving Hand Gesture Recognition Robustness to Dynamic Posture Variations by Multimodal Deep Feature Fusion. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2024**, *32*, 3104–3115. [CrossRef] [PubMed]
34. Zhang, C.; Li, X.; Wang, W.; Zhang, Y. HandSense: Mobile gesture recognition using acoustic sensing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* **2018**, *2*, 1–25. [CrossRef]
35. Dere, M.D.; Lee, B. A Novel Approach to Surface EMG-Based Gesture Classification Using a Vision Transformer Integrated with Convolutive Blind Source Separation. *IEEE J. Biomed. Health Inform.* **2023**, *28*, 181–192. [CrossRef] [PubMed]