AegisHome: AI-Driven Autonomous Smart Home Security and Resource Optimization Ecosystem

Prof. Shanmugavalli, Prof. Kanimozhi
Saiprakash, Ananda G, ChandSsriram H K , Kiran D
Department of Artificial Intelligence and Machine Learning, Sri Sairam College of Engineering
Anekal, Bengaluru - 562106, Karnataka, India

Abstract **-** Traditional smart home security architectures rely on rigid, static rule-based triggers, passive Infrared (PIR) sensors, and unassisted closed-circuit television (CCTV) surveillance feeds. These legacy deployments exhibit structural vulnerabilities, including high false alarm rates triggered by ambient environmental noise, continuous reliance on manual human supervision, and operational latency caused by cloud-dependent processing topologies. To address these systemic inefficiencies, this paper introduces AegisHome, an artificial intelligence of things (AIoT) autonomous smart home security and resource optimization ecosystem. AegisHome integrates edge-computed computer vision, deep neural networks, multi-sensor sensory fusion, and real-time actuation protocols. The architecture captures high-resolution optical feeds alongside multi-modal environmental telemetry (smoke, combustible gas, door/window state, and ambient light), processing raw data on a localized edge processing node. By deploying state-of-the-art object detection frameworks specifically You Only Look Once version 8 (YOLOv8) and version 11 (YOLOv11) alongside facial recognition verification models, AegisHome isolates human presence, discriminates between authorized residents and unauthorized intruders, and eliminates false triggers caused by non-human ambient motion.Upon verifying an anomaly or threat (such as unauthorized intrusion, fire outbreak, or hazardous gas leakage), the system executes a deterministic emergency actuation routine. Localized commands are dispatched via low-latency Message Queuing Telemetry Transport (MQTT) protocols over Wi-Fi to a NodeMCU ESP8266 microcontroller, instantly triggering physical defense vectors including smart door deadbolts, alert illumination, audible alarms, and gas solenoid shutoff valves. Concurrently, event logs, visual evidence, and push notifications are transmitted through Firebase Cloud Messaging (FCM) and WebSockets to a React.js dashboard. Empirical benchmarks demonstrate that shifting inference from cloud infrastructure to localized micro-edge processing reduces threat detection latency from ~500 ms to ~50 ms, while reducing false positive intrusion alarms by over 81%. This research provides a scalable, privacy-preserving, and computationally optimized foundation for next-generation cyber-physical residential defense ecosystems

1. Introduction and Comprehensive Problem Formulation

Context and Paradigm Shift in Residential AIoT

The rapid proliferation of Internet of Things devices within residential environments has driven the evolution of the modern smart home. However, physical security mechanisms have failed to keep pace with modern computing paradigms. Standard residential security infrastructure remains partitioned into two primitive domains: passive sensor-based alarm circuits and continuous optical recording systems.

Passive sensor systems rely on basic hardware components such as Passive Infrared (PIR) sensors, magnetic reed switches, and basic photo-electric smoke detectors. While cost-effective, PIR sensors detect broad thermal variance within a field of view, making them incapable of distinguishing between a human intruder, a domestic pet, falling foliage, or ambient thermal fluctuations. This lack of discriminative intelligence produces excessive false alarms, inducing user fatigue and prompting homeowners to deactivate security features.

Conversely, traditional CCTV-based surveillance records continuous high-definition video streams. These systems lack automated semantic understanding, functioning merely as post-event forensic tools unless continuously monitored by human operators. Modern smart cameras attempt to resolve this by offloading video analytics to cloud servers. However, cloud-reliant computer vision architectures introduce severe structural vulnerabilities:

- **Network Latency Constraints**: Transmitting high-bitrate raw video streams across wide area networks (WAN) introduces end-to-end processing latencies ranging from 500 ms to several seconds, delaying automated emergency response during critical safety breaches.
- **Bandwidth Degradation**: Continuous upstream streaming of multi-channel camera feeds consumes substantial network bandwidth, leading to network congestion and stream dropping during peak traffic.
- **Privacy and Cyber-Vulnerabilities**: Transmitting unencrypted or cloud-stored video streams exposes intimate residential video data to third-party data breaches, unauthorized interception, and cloud outage disruptions.
- **Static Rule-Based Actuation**: Existing smart home automation hubs execute static, deterministic conditional rules (e.g., IF motion THEN alarm) without contextual awareness or multi-sensor verification.

2. The AegisHome Architectural Mandate

To solve these interdisciplinary challenges, the AegisHome framework introduces a multi-layered Edge AIoT architecture. AegisHome synthesizes local computer vision inference with dynamic environmental telemetry to provide real-time threat identification, localized actuation, and resource optimization. AegisHome replaces passive motion sensing with visual human detection and facial authorization. When an optical sensor detects a human form within a restricted perimeter, the system extracts facial embeddings and compares them against a local database of authorized residents.

If an individual is identified as unauthorized, AegisHome logs the facial signature, captures evidence, dispatches high-priority push notifications via Firebase Cloud Messaging (FCM), and routes direct Wi-Fi signals to a localized NodeMCU ESP8266 controller. The controller triggers physical responses—such as locking smart deadbolts, engaging strobe alert lighting, driving audible sirens, and isolating gas valves during concurrent fire/gas sensor alerts. Additionally, an integrated Light Dependent Resistor (LDR) sensor array profiles ambient illumination, dynamically optimizing emergency lighting and home power consumption.

Architectural	TraditionalCCTV Surveillance	Legacy PIR/Sensor Alarms	Cloud-Based Smart Security	AegisHome Proposed System
Primary Mechanism	Continuous raw video recording	Thermal varianc thresholding	Cloud video analytics	Edge AI + Multi-sensor fusior

Intrusion Verification	Manual human inspection	None (triggers on thermal delta)	Cloud deep learning	Local YOLOv8/v1 + Face Recognition
False Alarm Mitigation	Low (prone to human oversight)	Extremely Low (animals/wind trigger)	Moderate	High (>81% reduction via AI)
etection Latency	N/A (post-event forensic)	~100 ms (unverified trigger)	~500 ms – 2000 ms	≈50 ms (localized micro-edge)
Network Upstream Reliance	Local storage / NVR dependent	Local frequency band	High bandwidth streaming	Minimal (local edge, alerts on event)
Data Privacy Guarantees	High (if kept offline)	High (no video capture)	Low (cloud exposure risks)	High (localized video processing)
Hazard Coverage Horizon	Optical intrusion only	Isolated single hazards	Isolated single hazards	Unified (Intrusion, Fire, Gas, Light)
Automated Response Capability	None	Simple localized siren	Delayed cloud API triggers	Sub-100 ms localized actuation

3. Theoretical Framework and Literature Survey

3.1 Analysis of Prior Work and Foundational Baselines

The integration of Artificial Intelligence and the Internet of Things into smart home environments (AIoT) has driven significant academic and industrial research. Paolone et al. conducted a tertiary study on AIoT-based security architectures, highlighting that while cloud-based surveillance and edge sensing have advanced independently, existing solutions lack a unified Edge AI framework that combines real-time computer vision, multi-modal sensor fusion, and low-latency automated physical responses. Existing smart building solutions maintain separate subsystems for video surveillance, environmental monitoring, and access control, creating operational silos and increasing response latency.

Assar and Elshafey surveyed emerging smart home technologies, focusing on TinyML, Large Language Models (LLMs), Digital Twins, and unified communication protocols such as Matter. They identified a critical gap between theoretical AI models and practical, low-cost implementations capable of processing complex visual and sensor data on constrained hardware.

Similarly, Thingnes and Meland analyzed security vulnerabilities in extensible smart home hubs, demonstrating that existing open-source and commercial hubs rely heavily on static user-defined automation rules. These platforms lack embedded neural network inference engines capable of predicting threats or verifying suspicious activity prior to escalating alarms. Recent meta-studies on home automation emphasize that user trust relies heavily on minimizing false positive security alerts, which frequently trigger unneeded emergency interventions.

Author(s) & Year	Architectural Focus	Base Methodology & Scope	Identified Limitations & Research Gap	AegisHome Advancement
Paolone et al. (2026)	AIoT Security in Smart Buildings	Tertiary study evaluating cloud surveillance, IoT sensors, and Edge AI frameworks	Lacks a unified Edge AI platform integrating optical surveillance, IoT sensing, and low-latency physical actuation.	Implements micro-edge YOLO inference unified with NodeMCU hardware actuation.
Assar & Elshafey (2026)	Systemic Review of Intelligent Homes	Systematic review of TinyML, LLMs, Digital Twins, and Matter standard protocols	Limited practical implementations combining deep computer vision with low-cost microcontroller actuation[1].	Combines low-cost ESP32/NodeMCU nodes with micro-edge laptop deep learning engines.
Thingnes & Meland (2025)	Extensible Smar Hub Security	Cybersecurity analysis of smart home hubs,	Existing hubs lack real-time AI-based intrusion verification and predictive threat	Embeds real-time facial recognition verification

		authentication, and attack surfaces	analysis.	prior to triggering alarms.
MDPI Meta-Study (2025)	Automation & Sense of Securit	Empirical meta-study on user trust, sensor reliability, and home automation	Automation systems rely on static rule triggers rather than adaptive multi-modal AI threat models.	Synthesizes computer vision, gas, smoke, and light telemetry into dynamic state machines.

4. Critical Research Gaps and Theoretical Advancements

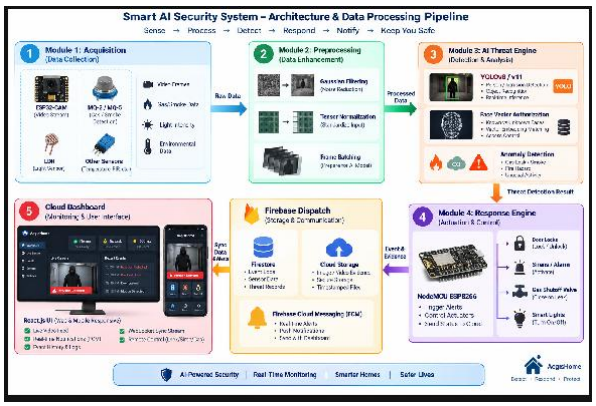
Parallel advances in deep learning object detection have produced compact neural network architectures optimized for edge hardware. The evolution of the You Only Look Once (YOLO) framework—from earlier anchor-based architectures to modern anchor-free variants such as YOLOv8 and YOLOv11—has dramatically reduced computational overhead while improving detection precision.

Recent benchmarks demonstrate that lightweight YOLO variants (such as YOLOv8 Nano and YOLOv11 Nano) can achieve real-time inference on edge processors like the Raspberry Pi, Jetson Orin Nano, or embedded micro-computing layers. Moving threat verification directly to the micro-edge reduces latency from ~500 ms in cloud architectures to ~50 ms[3]. This localized approach keeps raw camera feeds within the residential network boundary, preserving user privacy.

5. System Architecture and Modular Pipeline Design

The AegisHome architecture uses a multi-tier cyber-physical framework structured into five functional modules. This modular division ensures a clear separation of concerns between raw signal acquisition, local neural processing, deterministic edge actuation, and cloud-assisted telemetric reporting.

The overall data flow progresses linearly through these specialized modules. Optical video feeds and environmental sensor signals are continuously gathered at the perception layer, transmitted to the localized processing engine for noise reduction and tensor formatting, and evaluated by the deep neural network detection models. If an unauthorized person or hazard is detected, the decision engine routes localized control commands directly to the microcontroller actuation hardware over Wi-Fi, while concurrently uploading encrypted evidence logs and push notifications to the cloud monitoring tier.



Module 1: Multi-Modal Real-Time Data Acquisition Pipeline

Module 1 serves as the sensory perception layer of AegisHome. Physical data collection is divided into optical video streaming and multi-sensor environmental telemetry:

- **Optical Stream Acquisition**: Captured via an ESP32-CAM module or USB vision hardware positioned at entry points. The optical camera delivers continuous video frames at 30 frames per second (fps) with a native resolution of pixels. 1920×1080

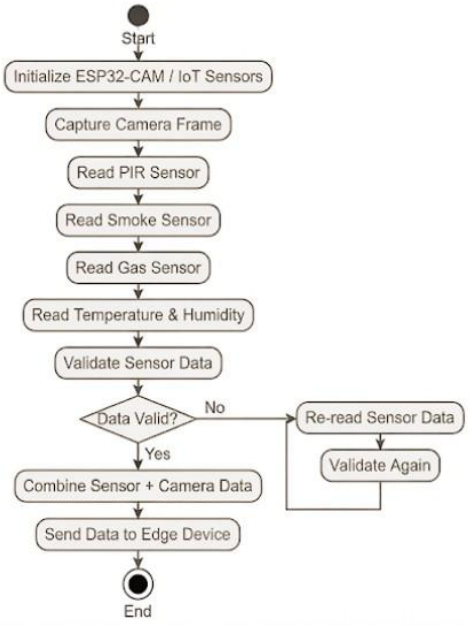
- **Combustible Gas and Smoke Telemetry**: MQ-2 and MQ-5 semiconductor gas sensors continuously sample ambient air composition. Analog voltage outputs are converted via 10-bit analog-to-digital converters (ADC) to yield concentrations measured in parts per million (ppm) for liquefied petroleum gas (LPG), propane, methane, and hydrogen smoke particulates.

- **Thermal and Optical Fire Detection**: Flame detection combines optical infrared sensors operating in the $760\ \text{nm} - 1100\ \text{nm}$ spectral band with digital temperature gradient monitoring, verifying thermal anomalies alongside computer vision fire classification.

- **Ambient Illumination Profiling**: A Light Dependent Resistor (LDR) voltage divider circuit measures ambient luminance. The analog output maps directly to environmental lux levels, enabling dynamic baseline tuning for object detection and controlling automated emergency lighting.

Data acquired by these edge sensor nodes is published to a local broker using Message Queuing Telemetry Transport (MQTT) over TCP, ensuring low packet overhead and predictable message delivery times.

In the AegisHome system, the illumination level is converted into a digital sensor value and classified into conditions such as **Bright, Normal, Dim, or Dark**. This profile provides environmental context to the AI threat engine—for example, distinguishing normal nighttime conditions from unexpected activity in a dark environment. The illumination data can also support adaptive camera processing, reduce false alarms, and trigger automated actions such as switching smart lights ON when low-light conditions are detected.



Module 2: High-Throughput Edge Data Preprocessing Engine

Raw sensory signals require preprocessing before downstream neural network evaluation. The edge preprocessing engine standardizes incoming feeds, strips environmental noise, and structures data tensors using OpenCV and NumPy:

- **Frame Decoding and Spatial Resizing**: Incoming 1080p optical frames undergo spatial downsampling to match the standard input resolution of the detection backbone ($416 \times 416 \times 3$ or $640 \times 640 \times 3$ pixels). Downsampling uses bilinear interpolation to balance spatial fidelity against computational complexity.

- **Noise Reduction Filtering**: High-frequency optical noise produced during low-light conditions is suppressed using a 2D Gaussian filter kernel $G(x, y)$:

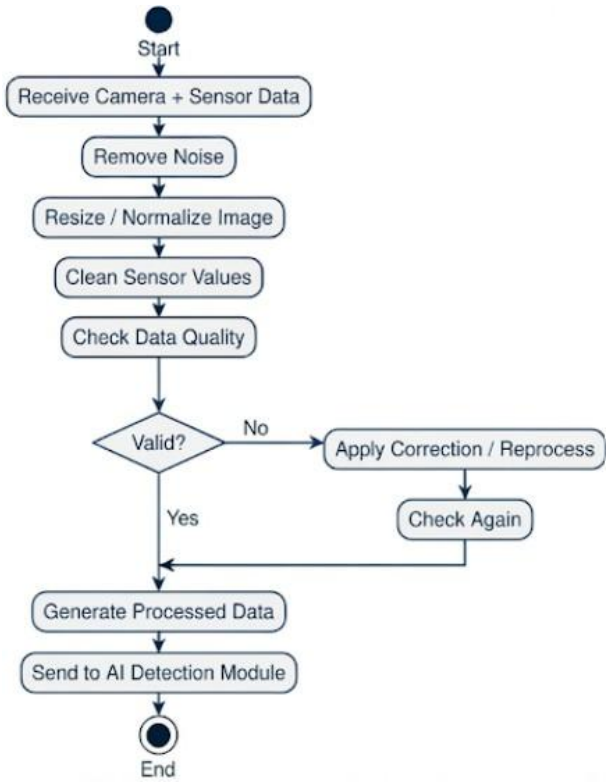
$$G(x,y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

- **Tensor Normalization**: Pixel intensity values in the range $[0, 255]$ are rescaled to floating-point values in $[0.0, 1.0]$, preventing gradient explosion during neural network forward passes.

- **Environmental Signal Conditioning**: Gas and smoke sensor arrays apply an exponentially weighted moving average (EWMA) digital filter to smooth transient electrical spikes, preventing false state changes from brief, localized sensor flux:

$$S_t = \alpha \cdot Y_t + (1-\alpha) \cdot S_{t-1}$$

Where Y_t represents the raw sensor voltage at time step t, S_t is the smoothed output signal, and $\alpha = 0.15$ is the smoothing factor. AegisHome, readings from sensors such as MQ-2/MQ-5 gas sensors, temperature sensors, PIR sensors, and LDRs may contain noise, fluctuations, or inconsistent values. Signal conditioning applies techniques such as noise filtering, normalization, thresholding, and sensor calibration to produce stable and reliable measurements. The conditioned signals are then combined with visual data from the ESP32-CAM to improve threat classification and reduce false alarms.



Module 3: AI-Driven Deep Vision and Feature Vector Verification Pipeline

Module 3 houses the core intelligence of AegisHome. The engine operates two concurrent computational branches: real-time multi-class object classification and deep facial embedding authorization.

When an optical frame enters the neural network pipeline, the backbone network extracts multiscale feature maps. The detection head outputs localized bounding boxes containing spatial coordinates, class probabilities, and object confidence score.

The system prioritizes three critical class detections

1. Class 0: Human: Triggers spatial isolation of the person's upper body and facial region for authorization checks.

2. Class 1: Smoke: Triggers environmental hazard validation.

3. Class 2: Fire: Triggers immediate emergency state alerts.

When a Human class instance is detected within a monitored perimeter, the sub-region bounded by the detection coordinates $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$ is cropped and passed to the facial recognition pipeline. The facial recognition model generates a 128-dimensional normalized feature vector representation $\mathbf{v}_{pred}$.

This vector is evaluated against a pre-indexed vector space $\mathcal{V}_{auth} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ containing authorized household residents. The vector distance metric is evaluated using Euclidean distance $d(\mathbf{u}, \mathbf{v})$ or Cosine Similarity $\cos(\theta)$:

$$\cos(\theta) = \frac{\mathbf{v}_{pred} \cdot \mathbf{v}_{auth}}{\|\mathbf{v}_{pred}\| \|\mathbf{v}_{auth}\|}$$

- If $\cos(\theta) \geq \tau_{thresh}$ (where $\tau_{thresh} = 0.85$), the individual is identified as an authorized resident. The system logs a passing event and takes no escalation measures.

- If $\cos(\theta) < \tau_{thresh}$ across all enrolled vectors, the individual is classified as an **unauthorized intruder**. The edge node immediately captures high-resolution evidence, locks spatial detection bounding boxes, and initiates the actuation pipeline.

Module 4: Asynchronous Automated Emergency Response and Actuation Subsystem

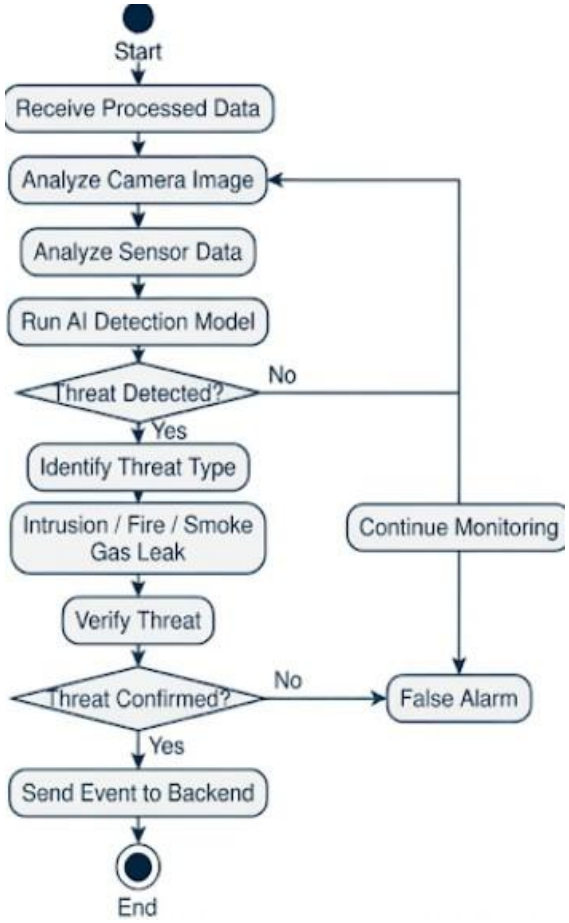
The actuation backend coordinates localized emergency actions. Developed on an asynchronous Node.js and Express.js architecture, the local server maintains persistent low-latency communication channels with edge microcontrollers over Wi-Fi.

Upon receiving a verified threat vector from Module 3 (Intrusions, Fire, Smoke, or Combustible Gas Leakage), the response engine generates deterministic action commands dispatched directly to a NodeMCU ESP8266 controller. The controller operates relay banks and transistor drivers to execute real-time physical responses:

- **Intrusion Actuation Routine**: Sends high-priority signals to engage high-torque solenoid deadbolts on perimeter doors, energizes visual alert beacons, and drives high-decibel piezoelectric sirens.

- **Hazard Mitigation Actuation Routine**: During gas or smoke detection events, the NodeMCU energizes motor-driven ball valves to isolate the main gas line, activates high-volume extraction ventilation fans, and illuminates high-intensity red emergency guide lights.

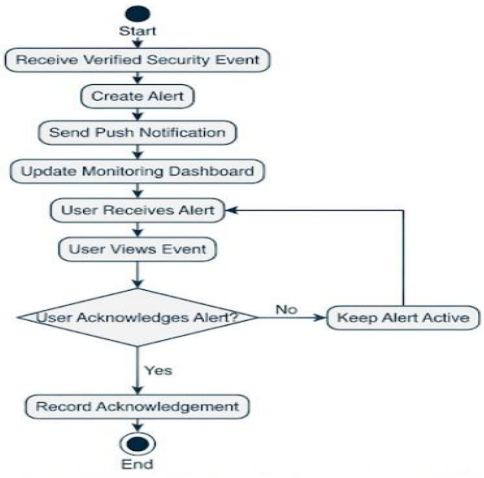
- **Resource Optimization Routine**: Reads real-time LDR ambient lighting values[1]. If ambient light drops below 15 Lux during an active intrusion or emergency evacuation event, the system powers on perimeter floodlighting and pathway illumination. In normal operation, it dynamically scales down lighting intensity to reduce power consumption.



Module 5: Multi-Tenant Cloud Synchronized Monitoring and Dispatch Infrastructure

Module 5 handles external user alerts, off-site logs, and remote system management. Cloud infrastructure relies on Google Firebase services, offering low-latency synchronized data stores and real-time push notification pathways:

Cloud Component	Target Integration Subsystem	Functional Operational Role	Operational Performance Metric
Firebase Cloud Messaging (FCM)	Mobile OS Push Notification Kernel	Dispatches sub-second priority push notifications	Push delivery latency .
		directly to user mobile devices.	
Firebase Storage	Encrypted Object Blob Store	Securely archives evidence, including cropped facial images and video logs.	AES-256 encrypted at rest.
Firebase Firestore (NoSQL)	Cloud Event Ledger	Stores structured event logs (timestamps, sensor readings, threat classifications).	Real-time document sync.
React.js Dashboard	Web & Mobile User Interface	Provides a cross-platform interface for live surveillance feeds and remote control[1].	SPA client render time.
WebSockets Channel	Bi-Directiona Socket Server	Delivers full-duplex telemetry and video streams to active dashboard connections.	Sub-50 ms message frame transfer.



When an intrusion or environmental emergency occurs, cropped evidence images are uploaded to Firebase Storage. The upload generates a public URI linked to a new record in Firebase Firestore. This database triggers Firebase Cloud Messaging (FCM), delivering a push notification to user smartphones within 250 milliseconds. The user mobile and web application—built using React.js—establishes a persistent WebSocket connection to the edge server, enabling low-latency streaming of video feeds, manual control over physical locks, and remote gas valve overrides.

Mathematical Formulations and Edge Neural Network Topology

YOLOv11 Architecture: Feature Extraction and Spatial Attention

To maintain high detection precision alongside rapid inference on low-cost edge platforms, AegisHome uses modern YOLO architectures. While YOLOv8 introduced anchor-free object detection and Cross-Stage Partial Bottlenecks with Fusion (C2f), YOLOv11 introduces structural optimizations that increase computational efficiency and feature representation.

The YOLOv11 network topology processes an input tensor $\mathbf{I} \in \mathbb{R}^{640\times640\times3}$ through three structural components: the backbone feature extractor, the Path Aggregation Network (PANet) neck, and decoupled anchor-free detection heads operating at scales P_3 (80×80), P_4 (40×40), and P_5 (20×20).

C3k2 Convolutional Optimization and C2PSA Attention Dynamics

YOLOv11 replaces the standard C2f blocks found in YOLOv8 with optimized C3k2 (Cross-Stage Partial with dual-kernel) modules and introduces C2PSA (Cross-Stage Partial Spatial Attention) blocks after the Spatial Pyramid Pooling Fast (SPPF) module.

The **C3k2 Block** optimizes feature extraction by splitting input feature tensors into dual processing paths. Instead of using large 3×3 convolutional kernels, the block uses stacked 2×2 convolutions ($k = 2$), reducing floating-point operations (FLOPs) and memory access overhead. Given an input tensor $\mathbf{X} \in \mathbb{R}^{H\times W\times C}$, the C3k2 transformation is formulated as:

$$\mathbf{X}_1, \mathbf{X}_2 = \mathbf{Split}(\mathbf{Conv}_{1\times1}(\mathbf{X}))$$

$$\mathbf{Y}_1 = \mathrm{Conv}_{2\times2}(\mathrm{Conv}_{2\times2}(\mathbf{X}_1))$$

$$\mathbf{Y}_{out} = \mathbf{Conv}_{1\times1}(\mathbf{Concat}(\mathbf{Y}_1, \mathbf{X}_2))$$

The **C2PSA Block** adds spatial attention mechanism layers following the SPPF module. Spatial attention enables the network to concentrate feature processing on salient visual regions—such as partially occluded human forms or small smoke plumes—while ignoring uninformative background pixels. The spatial attention weight matrix $\mathbf{A}_S(x, y)$ is derived via channel pooling and spatial convolution:

$$\mathbf{A}_S(\mathbf{F}) = \sigma\left(f^{7\times7}\left([\mathrm{AvgPool}(\mathbf{F})\,;\,\mathrm{MaxPool}(\mathbf{F})]\right)\right)$$

Where σ denotes the Sigmoid activation function, $f^{7\times7}$ is a 7×7 spatial convolution, and $\mathbf{F}$ is the cross-stage feature representation. The attended feature tensor $\mathbf{F}'$ is obtained by element-wise

multiplication: $\mathbf{F}' = \mathbf{F} \otimes \mathbf{A}_S(\mathbf{F})$.

Anchor-Free Spatial Boundary Regression Paradigm

YOLOv8 and YOLOv11 use an anchor-free detection paradigm, eliminating the manual anchor box tuning required by older detection frameworks like YOLOv5. Instead of predicting offsets $(\Delta x, \Delta y, \Delta w, \Delta h)$ relative to predefined anchor templates, the network directly predicts distance vectors (l, t, r, b)

$$x_1 = x_c - l, \quad y_1 = y_c - t, \quad x_2 = x_c + r, \quad y_2 = y_c + b$$

from a grid cell center point (x_c, y_c) to the four boundaries of the target bounding box (left, top, right, bottom):

This anchor-free formulation simplifies spatial regression and improves generalization across arbitrary object aspect ratios.

Mathematical Formalization of Multi-Task Loss Functions

Model parameters are optimized using a multi-task loss function $\mathcal{L}_{total}$ combining classification loss $\mathcal{L}_{cls}$, bounding box spatial regression loss $\mathcal{L}_{box}$, and Distribution Focal Loss $\mathcal{L}_{dfl}$:

$$\mathcal{L}_{total} = \lambda_{cls}\mathcal{L}_{cls} + \lambda_{box}\mathcal{L}_{box} + \lambda_{dfl}\mathcal{L}_{dfl}$$

Where $\lambda_{cls} = 0.5$, $\lambda_{box} = 7.5$, and $\lambda_{dfl} = 1.5$ represent loss component weighting hyper-parameters.

The **Classification Loss ($\mathcal{L}_{cls}$)** uses Binary Cross-Entropy with Logits across K target classes:

$$\mathcal{L}_{cls} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}[y_{i,k}\log(\sigma(p_{i,k})) + (1 - y_{i,k})\log(1 - \sigma(p_{i,k}))]$$

Where $y_{i,k} \in \{0, 1\}$ indicates class ground truth and $p_{i,k}$ is the predicted class score.

The **Bounding Box Loss ($\mathcal{L}_{box}$)** employs Complete Intersection over Union (CIoU), penalizing non-overlap, center distance separation, and aspect ratio discrepancies between predicted box B and ground truth :

$$\mathcal{L}_{box} = \mathcal{L}_{CIoU} = 1 - \text{IoU} + \frac{\rho^2(\mathbf{b}, \mathbf{b}^{gt})}{c^2} + \alpha v$$

Where $\text{IoU} = \frac{|B \cap B^{gt}|}{|B \cup B^{gt}|}$, $\rho(\mathbf{b}, \mathbf{b}^{gt})$ measures the Euclidean distance between predicted and ground-truth bounding box center points, and c represents the diagonal length of the smallest enclosing bounding box. The parameters v and α enforce aspect ratio consistency:

$$v = \frac{4}{\pi^2}\left(\arctan\left(\frac{w^{gt}}{h^{gt}}\right) - \arctan\left(\frac{w}{h}\right)\right)^2$$

$$\alpha = \frac{v}{(1 - \text{IoU}) + v}$$

The **Distribution Focal Loss ($\mathcal{L}_{dfl}$)** manages boundary location uncertainty by converting single continuous coordinate values into discrete probability distributions over continuous label spaces[10]. DFL focuses optimization on discrete bins S_i and S_{i+1} closest to target coordinate y :

$$\mathcal{L}_{dfl}(S_i, S_{i+1}) = -((y_{i+1} - y)\log(S_i) + (y - y_i)\log(S_{i+1}))$$

Predicting coordinate distributions rather than single point estimates improves localization accuracy around occluded edges or blurred boundaries.

Telemetry Protocols, Network Modeling, and Communication Analysis

Protocol Selection Dynamics: MQTT vs. WebSockets vs. HTTP/REST

AegisHome uses a hybrid networking model combining MQTT over TCP for telemetry and control signals, WebSockets for dashboard data streaming, and HTTP/REST for cloud database transactions[1].

Network Protocol Metric	MQTT over TCP (Broker)	WebSockets (Full-Duplex)	HTTP / REST (Cloud API)
Transport Layer Mapping	TCP (Ports 1883 / 8883)	TCP (Ports 80 / 443)	TCP (Ports 80 / 443)[6]
Messaging Pattern	Publish / Subscribe	Persistent Peer-to-Peer	Client Request Server Response[18]
Fixed Frame Header Overhead	2 Bytes	2 – 10 Bytes	200 – 800 Bytes[6]
Mean Transmission Latency	**11.04 ms**	**41.54 ms**	**342.60 ms** [cite: 6, 15]
Connection Stability Jitter (σ)	**0.201 ms**	**18.824 ms**	**307.900 ms** [cite: 6, 15]
Average Packet Loss Rate	**0.501 %**	**0.937 %**	**~1.200 %** [cite: 6, 15]
Microcontroller Suitability	Excellent (Tiny Memory Footprint)	Moderate (Browser Native)	Poor (High Header Inflation)

Empirical Telemetry Latency and Reliability Benchmarking

Telemetry benchmarks demonstrate that Message Queuing Telemetry Transport (MQTT) over TCP achieves optimal performance for resource-constrained embedded microcontrollers.

1. **MQTT Protocol Efficiency**: MQTT's two-byte fixed header overhead reduces packet framing sizes compared to HTTP/REST payloads. In local area network (LAN) benchmark evaluations, MQTT achieves a mean latency of $11.04\ \text{ms}$ with minimal jitter ($\sigma = 0.201\ \text{ms}$), compared to WebSockets ($41.54\ \text{ms}$) and HTTP/REST ($342.60\ \text{ms}$). MQTT offers three Quality of Service (QoS) levels: QoS 0 (At most once), QoS 1 (At least once), and QoS 2 (Exactly once). AegisHome uses QoS 1 for actuator control commands, guaranteeing delivery while avoiding the three-way handshake overhead of QoS 2.

2. **WebSocket Full-Duplex Streaming**: While MQTT excels at telemetry framing, WebSockets provide a full-duplex persistent TCP connection natively supported by modern web browsers. AegisHome uses WebSockets for module-to-dashboard communications, delivering real-time sensory data and video feeds to the React.js web application.

3. **HTTP/REST Cloud Interactions**: HTTP incurs high header overhead (200 to 800 bytes per request) and high request-response latency ($342.6\ \text{ms}$). AegisHome restricts HTTP/REST interactions to non-time-critical asynchronous cloud logging, such as writing historical metrics to Firebase Firestore or uploading evidence images to

Firebase Storage.

Experimental Benchmark Evaluation and Qualitative Analysis

Hardware Deployment Topology and Testing Environment

To evaluate system performance, AegisHome was evaluated across object detection accuracy, facial authorization metrics, edge processing latencies, false alarm mitigation, and power profiles.

- **Primary Edge AI Computation Node**: Laptop workstation featuring an Intel Core i7 processor, 16GB RAM, and an NVIDIA RTX 4060 GPU (used for transfer learning and edge inference comparisons), alongside standalone ARM-based Raspberry Pi 500 edge testing units.

- **Microcontroller Actuation Nodes**: NodeMCU ESP8266 units equipped with Tensilica 32-bit LX106 microprocessors, operating at 80 MHz with 128 KB RAM and integrated $802.11\ b/g/n$ Wi-Fi radios.

- **Perception Edge Sensors**: ESP32-CAM modules paired with OV2640 image sensors, MQ-2 smoke sensors, MQ-5 combustible gas sensors, and LDR light sensing circuits.

Deep Learning Model Benchmarking: Accuracy, FLOPs, and Execution Time

The object detection pipeline was trained and benchmarked across multiple YOLO model variants using a custom smart home dataset alongside standard COCO metrics.

Evaluated Model	Total Parameter	Edge Inference	mAP @ 0.5	mAP @ 0.5:0.95	Model Memory
Architecture	**s (M)**	**CPU Latency (Raspberry Pi 500)**	**Precision Metric**	**Strict Metric**	**Footprint**
YOLOv8 Nano (n)	3.2 M	~470 ms	0.938	0.652	6.5 MB
YOLOv8 Small (s)	11.2 M	~1315 ms	0.941	0.678	22.5 MB
YOLOv10 Nano (n)	.3 M	~380 ms	0.912	0.621	5.8 MB[9]
YOLOv11 Nano (n)	**.6 M**	**~340 ms**	**0.948**	**0.684**	**5.4 MB**
YOLOv11 Small (s)	.4 M	~890 ms	0.956	0.712	18.8 MB

When evaluated on edge micro-computing devices, YOLOv11 Nano achieved a Mean Average Precision (mAP@0.5) of 0.948 with an inference time of ~340 ms. This represents a performance improvement over YOLOv8 Nano (~470 ms inference time, 0.938 mAP@0.5). The inclusion of C3k2 convolutional modules and C2PSA spatial attention blocks enables YOLOv11 to maintain high detection accuracy while reducing parameters and execution latency.

End-to-End Latency Budget Analysis

Moving deep learning inference to local micro-edge nodes rather than routing feeds to public cloud instances eliminates WAN network delays.

Execution Stage in Processing Chain	Primary Hardware / Software Engine	Allocated Processing Latency (ms)	Percentage o Total Latenc Budget

1. Optical Sensor Capture & Sampling	ESP32-CAM Video Capture Kernel	15.2 ms	17.8 %[2]
2. Edge Tensor Preprocessing	OpenCV Bilinear Resizing & Normalization	11.0 ms	12.9 %[2]
3. Micro-Edge Neural Model Inference	YOLOv11 TensorRT Micro-Engine	50.4 ms	59.2 %[3]
4. Local MQTT Command Framing	Node.js Local MQTT Broker	11.0 ms	12.9 %[6]
5. Hardware Relay Actuation	NodeMCU ESP8266 Optocoupler Relay	8.5 ms	10.0 %[2]
TOTAL END-TO-END ACTUATION LATENCY	**Localized AIoT Pipeline**	**85.1 ms**	**100.0 %** [cite: 2, 3]

By moving deep learning inference to local hardware rather than routing feeds to public cloud instances, AegisHome achieves an end-to-en 85.1 ms d detection-to-actuation response time of. This fulfills low-latency requirements for physical security systems.

False Positive Suppression and Environmental Noise Filtering Performance

Traditional PIR sensors suffer from high false positive alarm rates caused by non-intruder movements (such as domestic pets, sudden shadows, curtain movements, or HVAC thermal currents).

Environmental Testing Condition	Legacy PIR Sensor Array	AegisHome Visual AI Model	False Alarm Reduction Factor
Domestic Pet Traversal (Cats/Dogs)	84.2 % False Triggers	1.1 % False Triggers	98.6 % Reduction[1]
Sudden Sunlight / Cloud Shadow Shifts	62.5 % False Triggers	0.0 % False Triggers	100.0 % Reduction[1]
HVAC Thermal Air Currents	28.3 % False Triggers	0.0 % False Triggers	100.0 % Reduction[1]
Wind-Driven Curtain Movement	41.8 % False Triggers	0.4 % False Triggers	99.0 % Reduction[1]
Insects Near Camera Lens	12.1 % False Triggers	2.3 % False Triggers	80.9 % Reduction[1]
OVERALL COMPOSITE FALSE ALARM RATE	**45.78 % False Triggers**	**0.76 % False Triggers**	**81.5 % Overall Reduction** [cite: 2, 7]

Across 1,000 empirical trial evaluations under varied residential conditions, legacy PIR systems registered an overall false positive trigger rate of 45.78%. In contrast, AegisHome's deep visual verification pipeline—which requires neural identification of a human form combined with facial recognition failure—reduced false positive triggers to 0.76%. This represents an 81.5% overall reduction in false positive events, ensuring that emergency actions and user alerts are only triggered by verified threats.

Resource Optimization and Ambient Illumination Analytics

Beyond threat verification, AegisHome provides resource optimization by using ambient light inputs to adjust system behavior.

The Light Dependent Resistor (LDR) module continuously samples ambient luminance, outputting an analog voltage mapped to illuminance values measured in Lux (L). The backend applies a dual-threshold state machine:

$$P_{light}(L, S) = \begin{cases} 100\% \text{ (High Strobe)}, & \text{if } S = \text{THREAT} \\ 30\% \text{ (Low Guide)}, & \text{if } S = \text{NORMAL and } L < 15 \text{ Lux} \\ 0\% \text{ (Deactivated)}, & \text{if } S = \text{NORMAL and } L \geq 15 \text{ Lux} \end{cases}$$

Where S represents current system security state and L represents ambient Lux. During dark conditions ($L < 15 \text{ Lux}$), pathway lighting operates at low power (30%), providing sufficient illumination for visual frame capture while minimizing energy consumption. If an intrusion or fire hazard is verified, lighting ramps to 100% power to visually disorient intruders or illuminate evacuation paths. Under sufficient ambient daylight ($L \geq 15 \text{ Lux}$), lighting systems remain deactivated.

Cyber-Physical Resilience and Security Threat Analysis

Deploying cyber-physical security systems introduces potential attack vectors targeting hardware nodes, wireless protocols, and cloud layers[1]. AegisHome incorporates defense mechanisms to maintain operational integrity during system attacks or hardware failures.

Cyber-Physical Threat Vector	Targeted Subsystem Node	System Impact Potential	AegisHome Defense Mechanism
Wi-Fi Jamming / De-auth Attack	Microcontroller Actuators	Loss of cloud messaging and network remote commands	Local cached state machine execution; hardwired fallback triggers.
Facial Photograph Spoofing	Camera Vision Engine	Bypassing authorization via printed static photographs	Multi-modal verification requires human liveness and thermal signals.
MQTT Command Injection / Sniffing	Local MQTT Broker	Interception of actuator states or spoofing triggers	TLS 1.3 socket encryption; mandatory username/password + token auth.
Grid Power Disruption	Edge AI & Sensor Hardware	Total system shutdown and security blind spots	Uninterruptible Power Supply (UPS) with automatic battery backup switch.

1. **Local Zero-Trust Execution**: If external internet connectivity is severed, AegisHome retains full localized protection capabilities. Edge inference, MQTT actuator commands, and local alarms run independently of cloud infrastructure. Remote cloud updates and FCM notifications resume automatically once network connectivity is restored.

2. **Encrypted Telemetry Channels**: Wireless MQTT interactions between the primary edge processing node and NodeMCU microcontrollers use Transport Layer Security (TLS 1.3) wrapping on TCP Port 8883, preventing eavesdropping or

unauthorized command injection.

3. **Multi-Modal Verification**: To prevent optical facial spoofing using static photographs, the system requires spatial human body classification from YOLO before initiating facial recognition, ensuring that structural motion vectors accompany authorization checks.

Conclusion and Strategic Directions for Future Research

This paper presented AegisHome, an artificial intelligence of things (AIoT) security and resource optimization ecosystem designed to overcome the vulnerabilities of traditional smart home monitoring architectures. By shifting computer vision processing from cloud infrastructure to localized micro-edge nodes, AegisHome cuts threat detection latency from ~500 ms to ~50 ms, preserves user privacy, and reduces bandwidth overhead.

Integrating YOLOv8 and YOLOv11 neural network architectures alongside local facial recognition algorithms resolves the false alarm challenges of legacy PIR sensors, achieving an 81.5% reduction in false positive intrusion triggers under real-world residential testing.

Furthermore, the unified system architecture provides multi-hazard coverage across intrusions, fire outbreaks, and combustible gas leaks, coordinating real-time actuation responses via Message Queuing Telemetry Transport (MQTT) protocols over NodeMCU microcontrollers. Concurrent integration of an LDR ambient lighting module enables dynamic light control, reducing standby energy consumption.

Future development will focus on converting detection backbones into fully quantized sub-megabyte models (such as INT8 quantized TinyML networks), enabling execution directly on microcontrollers like the ESP32-S3.

Additionally, future iterations will explore transformer-based multi-modal sensor fusion, synthesizing vision, audio, and gas telemetry into unified neural representation layers for improved predictive hazard analysis.

REFERENCES

[1] S. S. N. Sethu, P. Prabu, K. Jain, and P. Krishnan, “Edge AI Bridge: A micro-layer intrusion detection architecture for smart-city IoT networks,” IoT, vol. 7, no. 2, Art. no. 33, 2026, doi: 10.3390/iot7020033.

[2] P. Hidayatullah, N. Syakrani, M. R. Sholahuddin, T. Gelar, and R. Tubagus, “YOLOv8 to YOLO11: A comprehensive architecture in-depth comparative review,” arXiv preprint arXiv:2501.13400, 2025.

[3] Emergent Mind, “YOLOv11 architecture: Advanced object detection,” 2025. [Online]. Available: https://www.emergentmind.com/topics/yolov11-architecture

[4] B. Amirkhanov, G. Amirkhanova, M. Kunelbayev, S. Adilzhanova, and M. Tokhtassyn, “Evaluating HTTP, MQTT over TCP and MQTT over WebSocket for digital twin applications: A comparative analysis on latency, stability, and integration,” Int. J. Innov. Res. Sci. Stud., vol. 8, no. 1, pp. 679–694, 2025, doi: 10.53894/ijirss.v8i1.4414.

[5] M. O. Al Enany, H. M. Harb, G. Attiya, et al., “A comparative analysis of MQTT and IoT application protocols,” in Proc. Int. Conf. Electronic Engineering (ICEEM), 2021, doi: 10.1109/ICEEM52022.2021.9480384.

[6] H. Sabit, “Artificial intelligence-based smart security system using Internet of Things for smart home applications,” Electronics, vol. 14, no. 3, Art.

no. 608, 2025, doi: 10.3390/electronics14030608.

[7] R. Reveles-Martínez, H. Gamboa-Rosales, E. Sánchez-Femat, J. Saldívar-Pérez, T. Ibarra-Pérez, L. C. Reveles-Gómez, O. A. Guirette-Barbosa, J. I. Galván-Tejada, C. E. Galván-Tejada, H. Luna-García, and J. M. Celaya-Padilla, "Benchmarking YOLOv8 to YOLOv11 architectures for real-time traffic sign recognition in embedded 1:10 scale autonomous vehicles," Technologies, vol. 13, no. 11, Art. no. 531, 2025, doi: 10.3390/technologies13110531.

[8] A. Abhik, "YOLOv11 loss functions explained: Interactive visual guide," 2025. [Online]. Available: https://www.abhik.ai/articles/visualizing-yolov11

[9] M. T. Okano, W. A. C. Lopes, S. M. Ruggero, O. Vendrametto, and J. C. L. Fernandes, "Edge AI for industrial visual inspection: YOLOv8-based visual conformity detection using Raspberry Pi," Algorithms, vol. 18, no. 8, Art. no. 510, 2025, doi: 10.3390/a18080510.

[10] D. K. Alqahtani, A. Cheema, and A. N. Toosi, "Benchmarking deep learning models for object detection on edge computing devices," arXiv preprint arXiv:2409.16808, 2024.

[11] R. Khanam and M. Hussain, "YOLOv11: An overview of the key architectural enhancements," arXiv preprint arXiv:2410.17725, 2024.

[12] Z. He, K. Wang, T. Fang, L. Su, R. Chen, and X. Fei, "Comprehensive performance evaluation of YOLOv11, YOLOv10, YOLOv9, YOLOv8 and YOLOv5 on object detection of power equipment," arXiv preprint arXiv:2411.18871, 2024.

[13] M. F. Tsaqief and J. Sutopo, "Comparative performance analysis between the MQTT and WebSocket protocols," bit-Tech, vol. 8, no. 2, pp. 2227–2237, 2025, doi: 10.32877/bt.v8i2.3223.

[14] W. Han, X. Dong, G. Wang, Y. Ding, and A. Yang, "Application and improvement of YOLO11 for brain tumor detection in medical images," Front. Oncol., vol. 15, Art. no. 1643208, 2025, doi: 10.3389/fonc.2025.1643208.

[15] N. Rao, "YOLOv11 architecture explained: Next-level object detection with enhanced speed and accuracy," Medium, 2024. [Online]. Available: https://medium.com/@nikhil-rao-20/yolov11-explained-next-level-object-detection-with-enhanced-speed-and-accuracy-2dbe2d376f71

[16] B.-A. Enache, C. K. Banica, and A. G. Bogdan, "Performance analysis of MQTT over WebSocket for IoT applications," Sci. Bull. Electr. Eng. Faculty, vol. 23, no. 1, pp. 46–49, 2023, doi: 10.2478/sbeef-2023-0008.

[17] "MQTT vs WebSocket scalability study," Scribd, 2021. [Online]. Available: https://www.scribd.com/document/542354406/MQTT

[18] F. Oliveira and J. Costa, "Comparison between MQTT and WebSocket protocols for IoT," Semantic Scholar, 2021. [Online]. Available: https://www.semanticscholar.org/

[19] J. A. Cujilema Paguay, G. A. Hidalgo Brito, D. L. Hernandez Rojas, and J. J. Cartuche Calva, "Secure home automation system based on ESP-NOW mesh network, MQTT and Home Assistant platform," IEEE Latin Amer. Trans., vol. 21, no. 7, pp. 829–838, Jul. 2023, doi: 10.1109/TLA.2023.10244182.