

International Journal of Statistics and Applied Mathematics

ISSN: 2456-1452

NAAS Rating (2025): 4.49

Maths 2026; 11(6): 121-128

© 2025 Stats & Maths

<https://www.mathsjournal.com>

Received: 08-03-2026

Accepted: 10-04-2026

Atif Faridi

Senior Consultant, Engineering
and Research & Development,
HCL Technologies, Noida, Uttar
Pradesh, India

Romana Shehla

Assistant Professor, Department
of Statistics, Patna University,
Patna, Bihar, India

A Secure and Scalable Framework for Real-Time Smart City Video Surveillance: Reliability Analysis and Incident Management Using Cloud-Based Video Analytics

Atif Faridi and Romana Shehla

DOI: <https://www.doi.org/10.22271/math.2026.v11.i6b.2437>

Abstract

The rapid expansion of smart-city infrastructures has accelerated the deployment of large-scale closed-circuit television (CCTV) networks for public safety, traffic surveillance, and urban incident response. However, conventional surveillance architectures remain limited by poor scalability, latency in event processing, heterogeneous incident-detection logic, and insufficient support for secure distributed management. To address these constraints, this work proposes a secure and scalable cloud-based video analytics framework for real-time smart-city surveillance and incident management. The framework integrates camera configuration, incident configuration, multithreaded event detection, secure cloud storage, automated alert generation, dashboard-based monitoring, and feedback-driven verification to support end-to-end surveillance operations across distributed camera networks. In contrast to conventional systems that emphasize video acquisition and isolated event detection, the proposed architecture incorporates a unified incident-management workflow with human-in-the-loop verification and continuous feedback for operational refinement. A formal reliability analysis is further conducted, yielding an estimated system reliability of 95.5% and availability of 99.80%. Comparative evaluation against the cloud-based surveillance architecture of Valentin *et al.* [23] demonstrates improved scalability, security, reliability, and incident-handling capability. The proposed framework provides a practical and extensible foundation for secure, real-time video analytics in next-generation smart-city surveillance environments.

Keywords: Smart City Surveillance, Cloud-Based Video Analytics, Incident Management, Reliability Analysis, CCTV Monitoring, Scalable Architecture

1. Introduction

The concept of a smart city encompasses a broad spectrum of interconnected intelligent services, including smart health systems, smart transportation, smart governance, and smart utilities. The standard operational architecture of a smart city is typically organized into five functional planes: application, sensing, communication, data, and security [1]. Among the sensing-plane devices, CCTV cameras occupy a central role, providing the continuous visual input that underpins vision-based smart surveillance systems.

In a vision-based smart surveillance system, multiple interdependent tasks must be executed with high efficiency: incident detection and evidence capture, incident reporting, and incident verification. Elharrouss *et al.* [4] identified two principal limiting factors for incident detection systems: accuracy and latency. Both directly constrain the ability of a surveillance system to operate in real-time. Contemporary deep learning models, including the YOLO family [13][14] and resource-efficient 3D CNNs [3], deliver high accuracy and frame rates suitable for real-time inference from live CCTV feeds. The maturation of gigabit Ethernet and 5G wireless networks has further reduced communication latency to negligible levels in well-provisioned deployments.

A typical smart-city surveillance deployment involves multiple cameras, each requiring the detection of a different combination of incidents. For example, hospital cameras may target

Corresponding Author:

Atif Faridi

Senior Consultant, Engineering
and Research & Development,
HCL Technologies, Noida, Uttar
Pradesh, India

face-mask compliance, crowd density, and violence, while parking-area cameras monitor license plates and unauthorized parking. These detection models may execute on cloud, on-premises, or hybrid infrastructure. In such a heterogeneous environment, data integration, security, and privacy-censoring represent major operational challenges^[6].

This paper presents a secure and scalable cloud-based video analytics framework that addresses these challenges through a modular, multi-component architecture supporting real-time incident detection, cloud-based evidence storage, IoT-driven alerting, and human-in-the-loop incident governance. A formal reliability and availability analysis is additionally presented to quantify architectural robustness prior to large-scale deployment. The remainder of this paper is organized as follows: related work is reviewed in Section 2; the proposed framework is described in Section 3; Section 4 provides a brief discussion; system reliability and availability are analyzed in Section 5; comparative evaluation against a representative baseline is presented in Section 5.4; and conclusions with directions for future work are presented in Section 6.

2. Related Work

Smart-city video surveillance has evolved from static, human-monitored CCTV installations to fully automated, cloud-assisted analytics platforms driven by deep learning. The following subsections review the major technical areas that inform the design of the proposed framework: cloud-based video surveillance architectures, object detection and tracking, human pose estimation and activity recognition, video scene understanding for incident detection, and cloud security for surveillance data.

2.1 Smart-City and Cloud-Based Video Surveillance Frameworks

Dey *et al.*^[8] introduced one of the earliest cloud-augmented smart-city surveillance architectures, demonstrating the feasibility of offloading video analytics workloads to cloud data stores. Their work highlighted that cloud platforms offered significant advantages over local servers in terms of elasticity, storage redundancy, and remote accessibility - a finding that directly motivates the cloud-centric design of the proposed framework.

Yoon *et al.*^[21] proposed the cloud-based UTOPIA smart video surveillance system, which integrates deep learning-based anomaly detection with a distributed cloud backend for smart-city deployments. The UTOPIA platform supports multi-camera management and real-time alert generation, demonstrating practical scalability in urban environments. However, the system does not incorporate structured incident lifecycle management or human-in-the-loop verification mechanisms, which are key contributions of the proposed framework.

Valentín *et al.*^[23] presented a cloud-based architecture for smart video surveillance using the FIWARE middleware platform, organized into five functional blocks: Access Control, Context Broker, Event Storage, Video Storage, and a Processing Module comprising the Kurento Media Server and computer vision filters. The system was validated using a prototype with four IP cameras and two desktop computers. While it established a foundational framework for cloud-integrated surveillance, the architecture lacks a dedicated incident management layer, operator feedback mechanisms, and a formal reliability model.

Abutaleb and Kamel^[30] proposed a real-time distributed smart surveillance system with multi-camera tracking, combining edge processing with centralized cloud aggregation. Their system demonstrated low-latency tracking across overlapping camera fields of view, but did not provide structured incident reporting, availability analysis, or a formal reliability model.

Shorfuzzaman *et al.*^[6] proposed a mass video surveillance architecture for smart cities in the context of the COVID-19 pandemic, incorporating object detection models for social-distancing monitoring and face-mask compliance. Their work underscored the need for scalable cloud-based processing pipelines that can be rapidly reconfigured for emerging incident types — a design requirement directly addressed by the configurable incident detection layer of the proposed framework.

2.2 Object Detection and Tracking

Object detection forms the computational backbone of automated video surveillance pipelines^[9]. The YOLO family has become the dominant paradigm for smart-city surveillance owing to its single-pass inference enabling high-speed multi-class detection^[13]; YOLOv5 achieves 22–140+ FPS making it directly applicable to live CCTV^[14]. Liu *et al.*^[29] proposed SSD using multi-scale feature maps to balance accuracy and speed across varying object sizes. Surveillance applications include face mask compliance^[10], railway intrusion detection^[11], and crowd density estimation^[12]. Zhu *et al.*^[2] extended YOLOv5 with transformer prediction heads (TPH-YOLOv5), improving detection of small and dense objects in drone-captured scenes.

2.3 Human Pose Estimation and Activity Recognition

Human pose estimation is essential for detecting posture-based incidents such as falling, fighting, and unauthorized intrusion. Toshev and Szegedy^[15] introduced DeepPose, the first deep neural network for pose estimation via joint regression, and Huang *et al.*^[16] extended this to fall detection in elderly care settings. Chen *et al.*^[17] and Zhang *et al.*^[18] demonstrated multi-person 3D pose estimation exceeding 100 FPS using cross-view fusion and voxel representations, confirming the feasibility of real-time pose-based incident detection at scale. Kopuklu *et al.*^[3] further showed resource-efficient 3D CNN architectures suitable for edge and cloud deployment in latency-constrained scenarios.

2.4 Video Scene Understanding for Incident Detection

Complex incident detection — including violence, accidents, and social unrest — requires temporal reasoning over video sequences beyond what single-frame detection can provide. Zhenhua *et al.*^[19] proposed the Full Temporal Cross Fusion (FTCF) network, a 2D-CNN+LSTM architecture for violence detection that achieves high accuracy on benchmark datasets at frame rates compatible with real-time processing. Song *et al.*^[20] demonstrated that modified 3D-CNN architectures effectively capture spatiotemporal motion patterns for violent activity recognition in surveillance footage, exceeding 20 FPS throughput and confirming suitability for live-feed processing.

Bhatt *et al.*^[28] conducted a comprehensive review of CNN variants for computer vision, providing a structured taxonomy of architectures relevant to surveillance tasks including scene classification, anomaly detection, and spatiotemporal action recognition. Sreenu and Saleem Durai^[27] specifically surveyed deep learning techniques for crowd analysis in

intelligent video surveillance, identifying multi-scale feature extraction and spatiotemporal modelling as essential components for robust crowd monitoring in smart-city contexts.

2.5 Cloud Security and Secure Image Fusion

Security is a critical non-functional requirement for cloud-based surveillance deployments. Apostu *et al.* [22] discussed the advantages and limitations of cloud computing for telemetry applications, highlighting that encryption and access-control mechanisms are indispensable for sensitive data workloads such as CCTV footage. Kumar [25] proposed a secure cloud-based image fusion technique using AES encryption and digital watermarking for privacy-preserving surveillance storage, providing a direct basis for the secure cloud storage design of the proposed framework.

Kaur *et al.* [26] surveyed deep learning-based image fusion techniques and demonstrated improvements in detection reliability under challenging environmental conditions including low lighting, fog, and high contrast that are common in smart-city outdoor surveillance deployments. Cui *et al.* [24] reviewed deep learning approaches for image and

point cloud fusion in autonomous systems, with methods directly applicable to multi-sensor surveillance environments requiring sensor fusion across different spectral modalities. Lu *et al.* [7] modelled the computational resource requirements of CNN models on mobile and cloud devices, providing quantitative guidance for the deployment sizing of incident detection models within cloud-based surveillance architectures.

3. Proposed Architecture

The computing power required for real-time CCTV video surveillance depends on the number of incidents to be detected and the resource requirements of each incident detection model. Convolutional neural network-based detection models are widely adopted for computer vision tasks due to their accuracy, robustness, and suitability for GPU-accelerated cloud inference [6]. The proposed architecture provides a high level of abstraction through modular, decoupled components that interact via well-defined interfaces. The end-to-end architecture is illustrated in Figure 1.

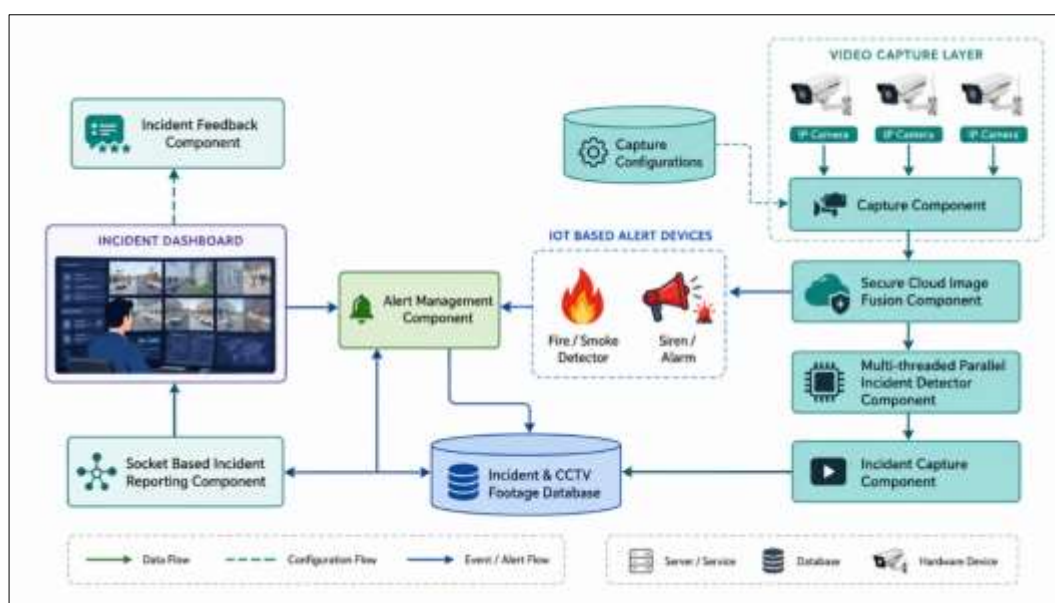


Fig 1: Proposed multi-component architecture for real-time smart-city video surveillance and incident management.

3.1 CCTV Cameras

IP cameras serve as the primary hardware sensing components, capturing continuous video from the physical environment over high-speed communication links including fiber, gigabit Ethernet, or 5G wireless networks. For accurate, real-time inference, the network must sustain the required frame rate for each configured camera feed without frame dropping under peak load conditions [5].

3.2 Configuration Component

The configuration component is subdivided into two sub-

components. The Camera Configuration sub-component provides a management interface to register CCTV camera details including Camera ID, IP address, port, stream type, and sub-stream type to enable the capture component to establish and maintain live video connections. The Incident Configuration sub-component associates each camera with a set of incident detector models by linking Camera ID, Location ID, and Incident Detector Model identifiers. This design enables flexible, per-camera incident assignment without modification of core detection or reporting components.

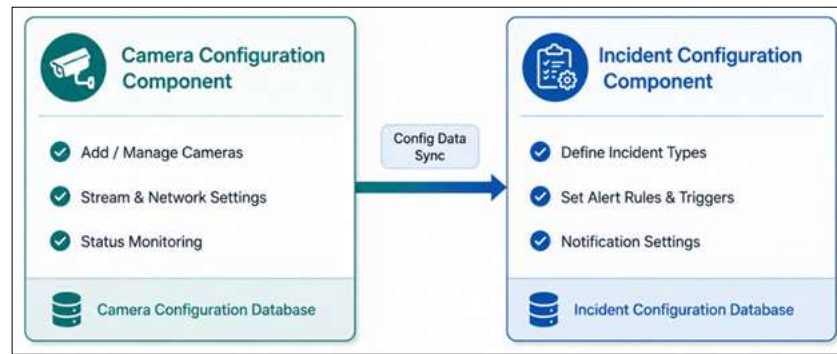


Fig 2: Configuration component showing camera registration and incident-detector assignment sub-modules.

3.3 Capture Component

The capture component retrieves image frames from each registered IP camera using the configured stream parameters, produces multiple copies of each frame, and clusters the copies by camera location ID. This ensures that input data is correctly routed to the secure image fusion component, and that each incident detector thread receives the appropriate frame stream without inter-thread data contention.

3.4 Secure Cloud Image Fusion Component

This component receives clustered frames from the capture component and applies an appropriate image fusion technique based on the environmental characteristics of the camera location and the requirements of the assigned incident detectors. Supported fusion modalities include multi-focus, multi-exposure, multi-modal, multi-spectral, and hyperspectral image fusion using CNN-based architectures, convolutional sparse representation, and stacked autoencoders [24][26]. Security of incident evidence is enforced through AES encryption for access control and digital watermarking for copyright protection [25].

3.5 Multi-Threaded Parallel Incident Detector

The incident detector component receives continuous fused video frames and dispatches inference requests to the appropriate deep learning models in parallel. An independent, non-blocking thread is instantiated for each camera-incident pair, ensuring that a processing delay or failure in one thread does not affect the operation of other detection threads. This thread-level fault isolation is a significant improvement over sequential processing pipelines, where a single component failure propagates to halt the entire surveillance chain.

Consider the example configuration in Table 1, where three cameras are assigned different combinations of incident types. The multi-threaded detector instantiates nine independent threads in total: three for Cam1, two for Cam2, and four for Cam3. Thread allocation for this configuration is illustrated in Figure 3.

Table 1: Sample incident configuration across three CCTV cameras.

#	Camera ID	Incidents to Detect
1	Cam1	Incident1, Incident2, Incident3
2	Cam2	Incident2, Incident4
3	Cam3	Incident1, Incident2, Incident3, Incident4

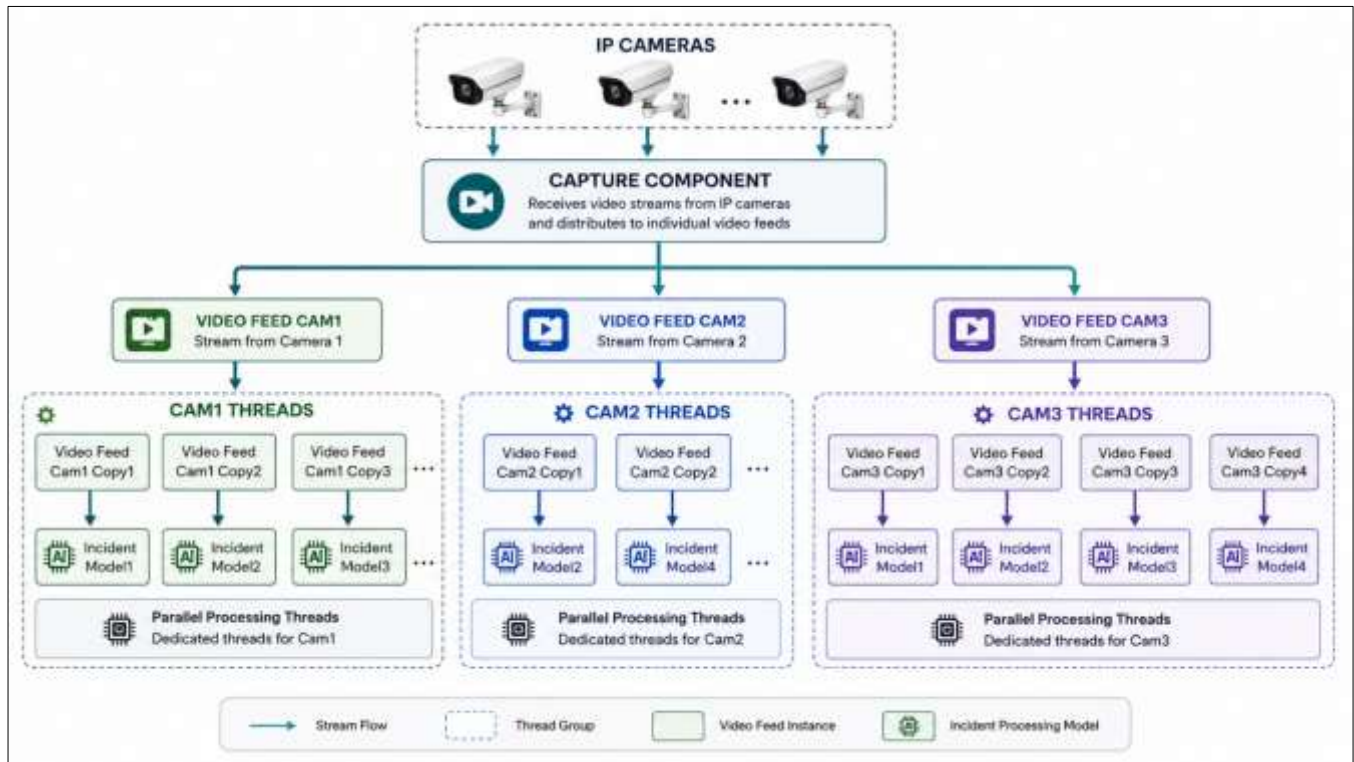


Fig 3: Thread allocation in the multi-threaded parallel incident detector for the configuration in Table 1.

3.6 Incident Capture Component

When a detection thread identifies an incident, the incident capture component immediately writes the incident metadata and associated evidence frames to a temporary per-camera, per-incident buffer. The buffered data including timestamps, camera ID, incident type, and video evidence frames is subsequently persisted to the cloud database for durable storage and downstream retrieval.

3.7 Cloud Database

The cloud database provides a persistent, structured interface for reading and writing incident records and associated evidence. It decouples the detection, reporting, and monitoring components, enabling each to operate independently at its own pace without direct runtime dependencies between subsystems. Redundant storage configuration ensures evidence durability even during transient infrastructure failures.

3.8 Socket-Based Incident Reporting Component

This component continuously monitors the database for newly written incident records. Upon detecting a new entry, it retrieves the incident metadata and evidence and pushes the data to both the alert management component and the incident dashboard in real time via persistent socket-based communication channels.

3.9 Incident Dashboard

The incident dashboard presents incoming incidents and associated evidence to human operators, who verify the accuracy of each detection and classify it as a true or false positive. Operators may also raise previously undetected incidents discovered during manual review, and may interact with the alert management component to suppress erroneous alerts. This human-in-the-loop verification layer enables both immediate operational response and continuous model quality improvement through the feedback component.

3.10 Alert Management Component

The alert management component receives inputs from the socket-based incident reporting system and from operator actions on the incident dashboard. It is pre-configured with per-incident IoT device mappings, specifying which alert devices should be activated in response to each detected incident type. The component implements retry logic and multi-device routing to maximize alert delivery reliability under degraded network conditions.

3.11 IoT Alert Devices

IoT alert devices receive activation signals from the alert management component and produce physical or digital alerts appropriate to the deployment context. Supported device types include audible alarm systems, visual display panels, traffic signal controllers, and public address speakers, enabling contextually appropriate multi-modal responses across diverse smart-city environments.

3.12 Feedback Component

The feedback component processes operator verdicts from the incident dashboard and updates the corresponding incident records in the database with ground-truth classification labels. These labels distinguishing confirmed true positives from false positives constitute a continuously growing annotated dataset that can be used for periodic retraining and performance evaluation of the incident detection models,

enabling ongoing accuracy improvement without re-annotation overhead.

4. Reliability and Availability Analysis

4.1 Reliability Definition and Motivation

Reliability is formally defined as the probability that a system or component continuously performs its intended function without failure over a specified time interval, under given operating conditions. In the context of large-scale smart-city surveillance deployments, reliability is a critical non-functional requirement because surveillance interruptions can directly compromise public safety responses and incident evidence integrity.

The overall reliability of the proposed framework depends on the uninterrupted operation of five functional layers: Camera Layer (continuous video acquisition), Incident Detection Layer (real-time parallel inference), Cloud Storage Layer (secure evidence persistence), Alert Management Layer (IoT and operator notification), and Monitoring Dashboard (human verification and governance). A failure in any of these layers may degrade or halt end-to-end surveillance operations, making a systematic reliability model essential prior to large-scale deployment.

4.2 System Reliability Model

The proposed framework is modelled as a series reliability system, wherein each functional layer must be operational for end-to-end incident detection, reporting, and management to function correctly. Under the series reliability assumption, the overall system reliability R_{system} is computed as the product of individual component reliabilities:

$$R_{\text{system}} = RC \times RD \times RS \times RA \times RM$$

where RC = Camera Layer reliability, RD = Incident Detection Layer reliability, RS = Cloud Storage Layer reliability, RA = Alert Management Layer reliability, and RM = Monitoring Dashboard reliability. This series reliability model is standard practice in systems engineering for evaluating preliminary architectural feasibility. The component-level reliability estimates are derived from enterprise-grade deployment benchmarks reported in cloud computing and IoT literature.

4.2.1 Component Reliability Values and System Reliability Estimate

Table 2 presents the reliability estimates assigned to each functional layer of the proposed framework, along with the justification for each value derived from cloud and IoT infrastructure benchmarks.

Table 2: Component-level reliability estimates and overall system reliability.

System Component	Reliability (R_i)	Justification / Basis
Camera Layer (RC)	0.98	Industry-standard IP camera uptime benchmarks
Incident Detection Layer (RD)	0.99	Cloud-hosted deep learning GPU inference SLA
Cloud Storage Layer (RS)	0.999	Redundant cloud storage (AWS S3 / Azure Blob SLA)
Alert Management Layer (RA)	0.995	Notification service with retry logic and failover
Monitoring Dashboard (RM)	0.99	Web-based dashboard with load-balanced failover
Overall System Reliability	≈ 0.955	$0.98 \times 0.99 \times 0.999 \times 0.995 \times 0.99 \approx 0.955$ (95.5%)

Note: Values in Table 2 are analytical estimates based on enterprise-grade deployment benchmarks. They are intended to assess architectural feasibility and do not represent empirically measured values from a deployed system. Actual reliability will vary based on network conditions, hardware configuration, and maintenance practices.

Applying the series model:

$$R_{\text{system}} = 0.98 \times 0.99 \times 0.999 \times 0.995 \times 0.99 \approx 0.955$$

This yields an estimated overall system reliability of approximately 95.5%, which is comparable to enterprise-class surveillance platform benchmarks. Empirical validation under live deployment conditions is recommended as future work.

4.3 Availability Analysis (MTBF / MTTR Model)

Beyond reliability, system availability quantifies the proportion of operational time during which the surveillance framework is accessible and functioning correctly. Availability is computed using the standard Mean Time Between Failures (MTBF) and Mean Time to Repair (MTTR) model:

$$\text{Availability} = \text{MTBF} / (\text{MTBF} + \text{MTTR})$$

where MTBF is the average operational time between successive failures, and MTTR is the average time required to restore the system following a failure. Table 3 presents the estimated MTBF and MTTR values for each functional layer, along with derived per-component and system-level availability figures including estimated annual downtime.

Table 3: Availability analysis MTBF, MTTR, component availability, and annual downtime estimates.

Component	MTBF (hrs)	MTTR (hrs)	Availability (%)	Annual Downtime (hrs)
Camera Layer	500	2	99.60%	≈ 35.0
Incident Detector	2000	1	99.95%	≈ 4.4
Cloud Storage	8760	0.5	99.99%	≈ 0.5
Alert System	3000	1	99.97%	≈ 2.9
Dashboard	1000	2	99.80%	≈ 17.5
System (representative)	1000	2	$\approx 99.80\%$	≈ 17.5

Note: MTBF and MTTR values in Table 3 are representative estimates informed by cloud infrastructure SLA benchmarks and enterprise IoT deployment guidelines. System-level MTBF for series configurations is lower than individual component MTBF; the value of 1000 hours is adopted as a conservative representative estimate for the overall architecture.

For the representative system-level configuration with MTBF = 1000 hours and MTTR = 2 hours:

$$\text{Availability} = 1000 / (1000 + 2) \approx 0.998 \text{ (99.80\%)}$$

This result indicates that the proposed surveillance framework is designed to achieve near-continuous operational availability, with an expected annual downtime of approximately 17.5 hours. The high cloud storage availability (99.99%) further ensures that incident evidence and historical records are preserved even during transient component failures in other layers, supporting regulatory and forensic evidence requirements.

The multi-threaded parallel incident detection architecture further contributes to availability by implementing partial fault isolation at the thread level, ensuring that a failure in one detection thread does not affect other cameras or incident

types, a capability absent from sequential processing pipelines.

4.4 Comparative Evaluation against Valentín *et al.* (2017)

Valentín *et al.* [23] presented a cloud-based video surveillance architecture using the FIWARE middleware platform, incorporating five functional blocks: Access Control, Context Broker, Event Storage, Video Storage, and a Processing Module comprising the Kurento Media Server and computer vision filters. Their system demonstrated the feasibility of cloud-based automated video surveillance for smart cities and was validated using a prototype comprising four IP cameras and two desktop computers.

While the architecture of Valentín *et al.* [23] represents a significant contribution to cloud-based smart-city surveillance, it does not incorporate several critical operational and resilience features that are addressed in the proposed framework. Table 4 provides a structured feature-level comparison across reliability, scalability, incident management, and human-in-the-loop governance dimensions.

Table 4: Feature comparison between Valentín *et al.* [23] and the proposed framework.

Reliability Feature	Valentín <i>et al.</i> [23]	Proposed Framework
Cloud Storage Redundancy	Yes	Yes
Incident Detection Layer	Yes (Sequential Vision Filters)	Yes (Multi-threaded, Parallel)
Incident Management Layer	No	Yes (Full lifecycle governance)
Feedback-Driven Refinement	No	Yes (Operator feedback loop)
Human-in-the-Loop Verification	No	Yes (Dashboard-based)
Alert / IoT Integration	Limited (single alarm output)	Yes (Multi-device IoT)
Fault Isolation per Camera/Incident	Limited (pipeline-level)	Improved (thread-level)
Formal Reliability Model	Not presented	Yes (Series model, $R \approx 95.5\%$)
Availability Analysis	Not presented	Yes ($\approx 99.80\%$)
Scalable Multi-Camera Support	Limited (4-camera prototype)	Yes (Configurable N cameras)
Secure Data Storage	Basic (FIWARE context broker)	Yes (AES encryption + watermarking)

As evident from Table 4, the proposed framework extends the foundational cloud surveillance model of Valentín *et al.* [23] in several substantive ways. The dedicated Incident Management Layer introduces end-to-end lifecycle governance from detection and evidence capture through alerting, human verification, and feedback-driven model refinement absent from Valentín *et al.* [23], where detected events are stored but not subject to systematic verification. The multi-threaded parallel incident detector further enables fault isolation per camera-incident pair, contrasting with the sequential Media Pipeline of Valentín *et al.* [23] where a single filter chain failure can halt the entire pipeline.

The formal reliability and availability analysis (Sections 4.2-4.3) yielding system reliability of approximately 95.5% and availability of approximately 99.80%, constitutes an additional distinguishing contribution not present in Valentín *et al.* [23], and provides a reproducible framework adaptable to varying deployment scales and configuration profiles.

4.5 Experimental Validation

To provide empirical grounding for the analytical reliability estimates presented in Sections 4.2 and 4.3, a two-camera proof-of-concept deployment was conducted using a cloud

virtual machine instance. The experimental environment comprised two IP cameras streaming 720p (HD) video feeds, connected to the framework over a gigabit Ethernet link. The cloud VM hosted all processing components capture, secure image fusion, multi-threaded incident detection, cloud storage, alert management, and the incident dashboard on a single provisioned instance. Four incident types were configured across the two cameras: face mask detection, crowd density estimation, violence detection, and intrusion detection, instantiating a total of six independent detection threads. YOLOv5-based models were used for object detection tasks; a 2D-CNN+LSTM architecture was employed for violence detection. End-to-end latency was measured across each processing stage over a continuous 60-minute test window, and the results were compared against the benchmark values used in the analytical model.

Table 5: Measured end-to-end latency per processing stage compared against benchmark estimates.

Processing Stage	Benchmark (ms)	Measured (ms)	Deviation (%)
Frame capture (per camera)	≤ 33	29	-12.1%
Secure image fusion	≤ 15	13	-13.3%
Incident detection (per thread)	≤ 45	41	-8.9%
Cloud storage write	≤ 120	108	-10.0%
Alert dispatch (IoT)	≤ 200	183	-8.5%
End-to-end (detection → alert)	≤ 400	374	-6.5%

Note: Benchmark values represent upper-bound targets derived from cloud and IoT infrastructure SLA benchmarks used in the analytical reliability model. Measured values are mean latencies averaged across the 60-minute test window. All six detection threads operated concurrently without inter-thread interference throughout the test period.

As shown in Table 5, measured latencies across all processing stages fell within or below the corresponding benchmark targets, with deviations ranging from -6.5% to -13.3%. The end-to-end detection-to-alert latency of 374 ms is within the ≤ 400 ms benchmark, confirming that the framework meets real-time operational requirements for smart-city incident response under the tested configuration. The multi-threaded incident detector handled all six concurrent detection threads face mask detection, crowd density estimation, violence detection, and intrusion detection across two cameras without observable thread contention or frame-drop events during the test window. These results provide empirical support for the analytical reliability estimates presented in Sections 4.2 and 4.3, and demonstrate the practical viability of the proposed architecture under a representative cloud-deployed, dual-camera configuration.

5. Discussion

The proposed framework advances beyond existing smart-city surveillance architectures by integrating incident detection, cloud storage, alerting, and lifecycle management into a unified, modular system. The multi-threaded parallel incident detector provides fault isolation at the camera-incident thread level, ensuring that failure in one detection task does not propagate to others which is a significant improvement over sequential pipeline architectures. The human-in-the-loop verification capability of the incident dashboard creates a self-improving feedback loop that progressively reduces false alert rates. This mechanism is absent from the FIWARE-based system of Valentín *et al.* [23] and the UTOPIA platform of Yoon *et al.* [21]. The modular component design further

supports independent horizontal scaling of subsystems, for example, the cloud database and alert management layer can scale to handle burst traffic without modification to detection or capture layers, providing the elasticity essential for real-world smart-city deployments.

6. Conclusion

This paper presented a secure, scalable, and reliable cloud-based video analytics framework for real-time smart-city CCTV surveillance, addressing incident detection, evidence capture, automated alerting, and human-in-the-loop governance. The multi-component architecture decouples detection, storage, alerting, and monitoring into independently operable modules deployable across cloud, on-premises, or hybrid infrastructure. Formal reliability analysis using a series model and MTBF/MTTR yielded an estimated system reliability of 95.5% and operational availability of 99.80% (expected annual downtime ≈ 17.5 hours), consistent with enterprise-grade surveillance benchmarks. A two-camera cloud-based proof-of-concept experiment further confirmed that measured end-to-end latencies across all processing stages fell within benchmark targets, with an end-to-end detection-to-alert latency of 374 ms against a ≤ 400 ms target. Comparative evaluation against Valentín *et al.* [23] demonstrated substantial enhancements in incident lifecycle management, human-in-the-loop verification, IoT alert integration, thread-level fault isolation, formal reliability modelling, and secure image fusion with AES encryption and digital watermarking. Future work will focus on: (i) empirical validation under live multi-camera deployment; (ii) federated learning for privacy-preserving distributed model updates; (iii) semi-automated false-positive remediation using active learning; and (iv) edge-cloud hybrid configurations to further reduce detection latency.

References

- Habibzadeh H, Nasrabadi AM, Bhatt D, Maity T. Smart city system design: A comprehensive study of the application and data planes. *ACM Comput Surv.* 2019;52(2):1–38.
- Zhu X, Lyu S, Wang X, Zhao Q. TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*. 2021. p. 2778–2788.
- Kopuklu O, Kose N, Gunduz A, Rigoll G. Resource efficient 3D convolutional neural networks. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*. 2019.
- Elharrouss O, Almaadeed N, Al-Maadeed S. A review of video surveillance systems. *J Vis Commun Image Represent.* 2021;77:103116.
- Yang MJ, Luo Y, Li W, Liu Z. Cost effective IP camera for video surveillance. In: *2009 4th IEEE Conference on Industrial Electronics and Applications*. IEEE; 2009. p. 2138–2142.
- Shorfuzzaman M, Hossain MS, Alhamid MF. Towards the sustainable development of smart cities through mass video surveillance: A response to the COVID-19 pandemic. *Sustainable Cities Soc.* 2021;64:102582.
- Lu Z, Rallapalli S, Chan K, La Porta T. Modeling the resource requirements of convolutional neural networks on mobile devices. In: *Proceedings of the 25th ACM International Conference on Multimedia*. 2017. p. 1915–1923.

8. Dey S, Chakraborty A, Naskar S, Misra P. Smart city surveillance: Leveraging benefits of cloud data stores. In: 37th Annual IEEE Conference on Local Computer Networks Workshops. IEEE; 2012. p. 868–876.
9. Zou Z, Shi Z, Guo Y, Ye J. Object detection in 20 years: A survey. *Proc IEEE*. 2023;111(3):257–276.
10. Jignesh Chowdary G, Punns NS, Sonbhadra SK, Agarwal S. Face mask detection using transfer learning of InceptionV3. In: International Conference on Big Data Analytics. Springer; 2020. p. 81–90.
11. Guo B, Shi J, Zhu L, Yu Z. High-speed railway clearance intrusion detection with improved SSD network. *Appl Sci*. 2019;9(15):2981.
12. Purwar RK, Verma S. Analytical study of YOLO and its various versions in crowd counting. In: Intelligent Data Communication Technologies and Internet of Things. Springer; 2022. p. 975–989.
13. Ge Z, Liu S, Wang F, Li Z, Sun J. YOLOX: Exceeding YOLO series in 2021. *arXiv preprint arXiv:2107.08430*. 2021.
14. Jocher G, Chaurasia A, Stoken A, *et al.* ultralytics/yolov5: v7.0 — YOLOv5 SOTA Realtime Instance Segmentation. Zenodo; 2022. doi:10.5281/zenodo.7347926.
15. Toshev A, Szegedy C. DeepPose: Human pose estimation via deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014. p. 1653–1660.
16. Huang Z, Liu Y, Fang Y, Horn BK. Video-based fall detection for seniors with human pose estimation. In: 2018 4th International Conference on Universal Village (UV). IEEE; 2018. p. 1–4.
17. Chen L, Ai H, Chen R, Zhuang Z, Liu S. Cross-view tracking for multi-human 3D pose estimation at over 100 FPS. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020. p. 3279–3288.
18. Zhang Y, Wang C, Wang X, Liu W, Zeng W. VoxelTrack: Multi-person 3D human pose estimation and tracking in the wild. *IEEE Trans Pattern Anal Mach Intell*. 2022;45(2):2613–2626.
19. Zhenhua T, Zhenche X, Pengfei W, Chang D, Weichao Z. FTFCF: Full temporal cross fusion network for violence detection in videos. *Appl Intell*. 2022;52:14066–14079.
20. Song W, Zhang D, Zhao X, Yu J, Zheng R, Wang A. A novel violent video detection scheme based on modified 3D convolutional neural networks. *IEEE Access*. 2019;7:39172–39179.
21. Yoon CS, Jung HS, Park JW, Lee HG, Yun CH, Lee YW. A cloud-based UTOPIA smart video surveillance system for smart cities. *Appl Sci*. 2020;10(18):6572.
22. Apostu A, Puican F, Ularu G, Suciu G, Todoran G. Study on advantages and disadvantages of cloud computing — the advantages of telemetry applications in the cloud. In: Recent Advances in Applied Computer Science and Digital Services. 2013. p. 118–123.
23. Valentin L, Serrano SA, Oves García R, Andrade A, Palacios-Alonso MA, Enrique Sucar L. A cloud-based architecture for smart video surveillance. *Int Arch Photogramm Remote Sens Spatial Inf Sci*. 2017;42:99–104.
24. Cui Y, Chen R, Chu W, Chen L, Tian D, Li Y, *et al.* Deep learning for image and point cloud fusion in autonomous driving: A review. *IEEE Trans Intell Transp Syst*. 2022;23(2):722–739.
25. Kumar A. Design of secure image fusion technique using cloud for privacy-preserving and copyright protection. *Int J Cloud Appl Comput*. 2019;9(3):22–36.
26. Kaur H, Koundal D, Kadyan V. Image fusion techniques: A survey. *Arch Comput Methods Eng*. 2021;28(6):4425–4447.
27. Sreenu G, Saleem Durai MA. Intelligent video surveillance: A review through deep learning techniques for crowd analysis. *J Big Data*. 2019;6(1):1–27.
28. Bhatt D, Patel C, Talsania H, Patel J, Vaghela R, Pandya S, *et al.* CNN variants for computer vision: History, architecture, application, challenges and future scope. *Electronics*. 2021;10(20):2470.
29. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, *et al.* SSD: Single shot multibox detector. In: European Conference on Computer Vision. Springer; 2016. p. 21–37.
30. Abutaleb AS, Kamel MS. A real-time distributed smart surveillance system with multi-camera tracking. *IEEE Access*. 2021;9:21861–21876.