Critical Infrastructure Monitoring using 5G, Kubernetes and AI

Quan Xiang Yeo1, a), Shih Yin Ooi1, 2, b), Heng Siong Lim3, 4, c), Ying Han Pang1, 2, d)

*1Faculty of Information Science and Technology (FIST), Multimedia University, Jalan Ayer Keroh Lama, Melaka, 75450, Malaysia*

*2Centre for Advanced Analytics (CAA), COE for Artificial Intelligence, Multimedia University, Jalan Ayer Keroh Lama, Melaka, 75450, Malaysia*

*3Faculty of Engineering and Technology (FET), Multimedia University, Jalan Ayer Keroh Lama, Melaka, 75450, Malaysia*

*4Centre for Remote Sensing and Surveillance Technologies (CRRST), COE for Robotics and Sensing Technologies, Multimedia University, Jalan Ayer Keroh Lama, Melaka, 75450, Malaysia*

*b) Corresponding author: syooi@mmu.edu.my*

*a) yeoquanxinag6@gmail.com*

*c) hslim@mmu.edu.my*

*d) yhpang@mmu.edu.my*

**Abstract.** This study aims to develop a critical infrastructure monitoring system using 5G, Kubernetes, and artificial intelligence (AI) to provide an intelligent and efficient solution to the challenges faced by current infrastructure monitoring systems, including performance optimization, resource allocation, and intelligent capabilities. As the core pillar of modern society, critical infrastructure — including power grids, water resource systems, transportation networks, and telecommunications systems — faces diverse threats, such as natural disasters, technical failures, human errors, and cyberattacks, due to its increasingly complex structures and its importance to society. This study proposes an innovative hybrid edge-cloud architecture monitoring system to address these issues.

# INTRODUCTION

In the wave of global digital transformation, the intelligent management of critical infrastructure has become an important issue in the development of modern society. Critical infrastructure, including power grids, water supply systems, transportation networks, and telecommunications, constitutes the backbone of modern society. However, in today's rapidly changing world, these systems face increasingly complex and diverse threats. Climate change exacerbates natural disasters and poses unpredictable risks to infrastructure [1]. Simultaneously, the rise of sophisticated cyberattacks and physical security breaches presents a growing concern for infrastructure operators [2].

With the rise of digital transformation and urbanization, traditional monitoring solutions cannot fulfil people's needs; combining IoT, 5G, AI, and advanced tech is becoming a trend, especially in smart cities, where using networks to monitor infrastructure can improve resource management efficiency and enhance potential threat detection and response capabilities. However, how to organically combine these technologies and efficiently realize their functions in resource-constrained edge environments is still a key issue that needs to be solved.

Based on this background, we propose an innovative critical infrastructure monitoring system that uses 5G communication technology, Kubernetes container orchestration platform and AI algorithms to build a comprehensive monitoring solution that is real-time, intelligent and efficient. Through the cloud-native architecture of Kubernetes, the system can flexibly manage resource scheduling between multiple edge nodes; the introduction of 5G communication ensures a stable connection between edge nodes and the cloud, further improving the response speed of the system; the deep integration of AI algorithms provides the system with multiple functions such as intruder detection and environmental data analysis.

# LITERATURE REVIEW

As technology advances rapidly, IoT has become increasingly crucial in monitoring critical infrastructure. By deploying connected devices to collect and process real-time data, these solutions can manage resources more efficiently and respond to complex security and environmental challenges. However, using IoT with monitoring systems creates several difficulties. The main challenges include managing various types of devices, enhancing data flow, and addressing the limited computing power available on devices with limited resources [3]. These limitations create real performance problems. IoT systems typically experience latencies of 50-70 ms [4]. At the same time, packet delivery rates in low-power networks such as LoRaWAN are typically less than 80%, far below the 99.9% reliability required for safety-critical applications [5].

5G networks with edge computing capabilities are essential for fast local data processing. 5G offers high speeds, large capacity, and low latency, which are mission critical. Edge computing processes data close to its source, reducing delays caused by sending data to distant clouds. This pairing allows for reliable, quick responses.

Studies show that this combination is effective. Edge systems can process AI tasks in 5-20ms, much faster than 100-500ms for cloud solutions [6]. Using 5G with Mobile Edge Computing (MEC) for tasks like computer vision reduces response times by 71.3% [7]. Edge AI also significantly reduces network bandwidth by 65-70% by processing data locally [6]. This is important for critical infrastructure as it saves bandwidth for essential, fast communications. However, deploying 5G-enabled edge systems presents challenges. Managing complex, hybrid environments and ensuring consistent ultra-low latency across diverse network topologies remains difficult.

Kubernetes has become the de facto standard for container orchestration, providing scalability, elasticity, and modularity through its cloud-native architecture [8]. Its ability to dynamically scale resources and support multi-tenancy is essential for critical infrastructure [8] and managing complex AI tasks [9]. This is achieved through auto-scaling mechanisms, such as the Horizontal Pod Autoscaler (HPA) and Cluster Autoscaler [8]. Lightweight Kubernetes distributions, such as k3s, are particularly well-suited for resource-constrained edge environments [10]. In comparative analyses, k3s exhibits the lowest overall resource consumption, making it a good fit for such scenarios [10]. While k3s is very efficient in terms of resource utilization, it may require enhancements for high-throughput data plane operations compared to distributions like k0s or k8s [10].

Platforms such as KubeEdge further extend Kubernetes capabilities to the edge, facilitating cloud-edge collaboration, managing IoT devices, and enabling edge autonomy where nodes can operate even with intermittent cloud connectivity [11]. KubeEdge has been successfully utilized in large-scale systems and critical infrastructure scenarios such as highway monitoring and long bridges, demonstrating its ability to handle real-time AI at the edge [11-13]. However, it consumes higher resources and has lower general scalability [10].

Despite these advances, traditional Kubernetes scaling solutions often struggle to cope with the highly dynamic and unpredictable resource requirements of AI workloads [8]. This can lead to over-provisioning of resources, resulting in increased costs or underutilization and performance degradation [8]. The concept of "edge-native infrastructure" aims to address these challenges by redesigning operating systems, orchestration, and security for edge constraints, with a focus on resiliency and remote manageability [14].

AI is important for monitoring critical infrastructure, detecting unauthorized individuals, and analyzing environmental data [15]. To analyze video and recognize faces quickly on the edge, AI models should run directly on devices. This approach reduces delays and ensures data privacy by processing data locally rather than sending it elsewhere. However, deploying advanced AI on edge devices with limited resources is a significant challenge. These devices have restricted computing power, storage space, and energy. They also need regular model updates, which adds to the difficulty [15].

# PROPOSED SOLUTION

While existing approaches typically address individual components of infrastructure monitoring in isolation—such as standalone IoT deployments with limited scalability, separate AI processing systems, or basic edge computing implementations, our proposed solution integrates these technologies into a unified, scalable framework for critical infrastructure monitoring. The system combines AI-powered intruder detection and IoT-based weather monitoring within a hybrid edge-cloud architecture, leveraging 5G for low-latency communication, K3s-based Kubernetes for scalable container orchestration, and edge computing for real-time processing to ensure reliable, responsive monitoring. Unlike previous work that focuses on single-purpose solutions or relies on cloud-edge platforms like KubeEdge that require cloud connectivity [11], our approach emphasizes edge-native processing using K3s for resource-constrained environments [10], enabling autonomous operation even with limited connectivity. Our approach emphasizes the seamless integration of the following elements: intruder detection through a multi-stage deep learning pipeline utilizing MobileNet models, environmental data processing from RS485 sensors transmitted via MQTT to edge nodes, storage in InfluxDB, and real-time visualization in Grafana. The modular design of the framework enables easy scaling and the addition of new monitoring functions, thereby addressing the limitations of traditional systems that struggle to cope with dynamic resource requirements [8]. This provides a solution that can adapt to diverse infrastructure monitoring needs. The proposed solution is divided into 3 parts: overall system architecture design, intruder detection, and a weather monitoring system.

## System Architecture Design

As illustrated in Figure 1, the proposed solution uses a hybrid edge-cloud architecture, integrating AI intruder detection and weather sensors for real-time monitoring. The solution integrates 5G connectivity for network resilience, edge computing for local processing, and Kubernetes-based orchestration for scalability and manageability. Future integrations, such as AI-driven analytics for predictive insights and a real-time alerting system, can be integrated to enhance threat detection and response capabilities further.

The architectural diagram shown in Figure 1 demonstrates how the system uses K3s, a lightweight Kubernetes distribution, for managing a cluster consisting of an edge node and multiple IoT nodes. The edge node acts as the control plane, managing the IoT nodes, which serve as worker nodes. Using a Kubernetes YAML file, a DaemonSet is deployed to ensure that the specified containers for the intruder detection system and weather sensor program run on all IoT nodes.

A diagram of a solution

Description automatically generated

**Figure 1.** Proposed solution architecture

Following the data flow depicted in Figure 1, the intruder detection and weather monitoring system are packaged as Docker images and pulled from Docker Hub. Then, weather monitoring system containers on each IoT node will collect the environmental data and use the Mosquitto Message Queuing Telemetry Transport (MQTT) broker to publish it to the edge node. The edge node runs the MQTT broker to receive weather sensor data from all IoT nodes. This data is stored in an InfluxDB instance optimized for time-series data storage. Then, using Grafana, the sensor data can be visualized to create real-time dashboards and monitor environmental conditions effectively.

### Hardware Setup

The edge node of the system is a server running Ubuntu 22.04, the control plane of the Kubernetes cluster, which is used to manage IoT nodes and runs containers such as intruder detection, weather monitoring system, and task offloading. The IoT node uses Raspberry Pi 4 Model B Rev 1.4 with 4GB RAM, and each Raspberry Pi device is responsible for local data processing and real-time monitoring.

To optimize the performance of AI models, especially for intruder detection, a Coral USB Accelerator is integrated with IoT nodes. In addition, the system deploys a set of weather sensors in Figure 2(a) to 2(c) based on the RS485 protocol to measure various environmental parameters, including humidity, temperature, noise, PM2.5 and PM10 concentrations, atmospheric pressure, light intensity, wind speed and wind direction. The data from these sensors is transmitted to the edge node through the IoT node.

Real-time video capture is done by Dahua IPC-HFW1431S1-S4 4MP Entry IR Fixed-focal Bullet Network Camera, which sends the video stream to the intruder detection container for processing via Real-Time Streaming Protocol (RTSP).  The system uses 5G routers to provide high-speed, low-latency communication between edge node and IoT nodes, ensuring seamless data transmission and real-time system response.

The IoT node hardware setup is shown in Figure 3, which shows the connection of IoT nodes, weather sensors, RTSP camera, and the 5G routers to ensure communication reliability.

|  |  |  |
| --- | --- | --- |
|  | A black device with a white label  Description automatically generated | A black object with a qr code  Description automatically generated |
| (a) | (b) | (c) |

**Figure 2.** RS485 Sensor. (a) Iradar environmental sensor; (b) Iradar gotani wind speed sensor;   
(c) Iradar gotani wind direction sensor

****

**Figure 3.** IoT node setup

## Intruder Detection

Based on the security requirements of critical infrastructure monitoring systems, we developed an AI-based intruder detection system using Python. The system adopts a multi-stage detection and recognition pipeline architecture to achieve personnel identification and intruder warning functions through deep learning models. The system consists of three core deep learning models:

1. MobileNet Single Shot Detector (MobileNet SSD) V2 Common Objects in Context (COCO) model for general object detection with mean average precision (mAP) of 25.6% [16].
2. MobileNet SSD v2 Face model specifically for face detection.
3. MobileNet Triplet model, which is trained using triplet loss function on a third of the VGGFace2 dataset for generating facial feature vectors [17].

This multi-level detection approach can effectively improve the system's accuracy and reliability. As shown in Figure 4, the system first scans the image captured by a RTSP camera to identify human-shaped targets, then locates the face within the detected human-shaped area and finally determines the identity by extracting and matching the facial feature vectors (or face embedding).

In terms of identifying registered and unregistered persons, the system has established a local facial feature database to store facial feature vectors of authorized persons. When the system detects a face, it generates a feature vector for that face and compares it with the feature vectors stored in the database. The system uses Euclidean distance to calculate similarity and sets a threshold to determine whether it is a registered person. For registered persons, the system displays their identity information and confidence level; for unregistered persons, the system marks them as "unknown" and visually distinguishes them through bounding boxes of different colors.

A diagram of a process

AI-generated content may be incorrect.

**Figure 4.** Flow chart of intruder detection

## Weather Monitoring System

In order to achieve real-time monitoring of critical infrastructure, we designed and implemented an IoT-based weather monitoring system. The system uses a multi-functional sensor array integrated with a single-board computer (Raspberry Pi) through a serial communication protocol (UART). The sensor suite can continuously collect multiple environmental parameters, including temperature, humidity, noise level, PM2.5 and PM10 particle concentrations, atmospheric pressure, light intensity, wind direction, and wind speed data. The system performs data acquisition and processing through a specially designed Python script, which implements a low-level communication protocol with the sensors, converts raw hexadecimal data into readable engineering units, and transmits data through the MQTT protocol to the edge node.

Regarding Kubernetes cluster deployment, the system uses the DaemonSet to ensure that each working node runs a weather monitoring instance. The deployment configuration is implemented through a carefully designed manifest file that defines the container image and network settings. The data pipeline architecture, illustrated in Figure 5, adopts a multi-level design. First, the sensor collects data, and Rasberry Pi reads the data. Then, the data is published through the MQTT agent on port 1883. The Telegraf agent subscribes to the relevant topics and forwards the data to the InfluxDB time series database. Finally, Grafana can visualize the data through query, supporting real-time monitoring and historical data analysis. This architecture supports monitoring and analysis across geographical locations through 5G networks.



**Figure 5.** Flow chart of weather monitoring system

# RESULTS AND DISCUSSION

From the purpose solution, it achieved real-time critical infrastructure monitoring by using 5G, Kubernetes, and AI. The solution uses a hybrid edge cloud architecture managed by Kubernetes, integrating AI models for security monitoring and IoT sensors for environmental data collection. Through containerization and cloud-native technologies, the system can deploy the containerized intruder detection and weather monitoring system to every IoT node within a single command with the help of Daemonset and YAML files.

The intruder detection system is implemented through a multi-stage deep learning pipeline based on the MobileNet family, enabling real-time personnel identification. During evaluation, the average pipeline execution time was 114.18 ms, with 95th and 99th percentiles at 145.01 ms and 164.42 ms, respectively. For the face detection submodule, the system achieved an average detection time of 35.29 ms, with the 95th percentile at 57.82 ms and the 99th percentile at 70.32 ms. Notably, 100% of face detection operations were completed under 500 ms.

The system resource usage during the intruder detection test was also evaluated. The system maintained efficient operation with an average CPU usage of 39.6%, while CPU usage remained under 70% for 96.7% of the time. The peak CPU usage was 97.6%, and the average memory usage was 21.7%, confirming the feasibility of deploying the AI inference pipeline on low-power IoT nodes, such as the Raspberry Pi, without compromising performance.

In order to register to the system, the person has to go through another Python program that uses a similar approach to capture and generate facial feature vectors and store them in a database. After storing the facial feature vectors, the system can identify the person. As shown in Figure 6, the system can recognize the registered person "yeo" at such a distance.

[](https://mmuedumy-my.sharepoint.com/:v:/r/personal/1211105012_student_mmu_edu_my/Documents/Recordings/intruder%20detection.mp4?csf=1&web=1&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJPbmVEcml2ZUZvckJ1c2luZXNzIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXciLCJyZWZlcnJhbFZpZXciOiJNeUZpbGVzTGlua0NvcHkifX0&e=tDPN4d)

**Figure 6.** Intruder detection demo video (available at https://shorturl.at/gyeOk)

The weather monitoring system deployed has successfully achieved real-time monitoring and collection of multi-parameter environmental data. Using a Python script, the sensors can collect the data through the RS485 protocol and send it through the MQTT protocol. The telegraf agent subscribed to the MQTT topic collects the data and stores it in InfluxDB. Then, using Grafana and InfluxQL, the data can be visualized, as shown in Figure 7, the Grafana dashboard showing data collected from the weather sensor.

**A graph with different colored lines

Description automatically generated**

**Figure 7.** Visualization of proposed weather monitoring system data using Grafana

This system adopts containerized deployment and uses Kubernetes for orchestration management. The intruder detection and weather monitoring system is encapsulated in their Docker container, which contains all necessary model files, dependent libraries, and operating environment. This containerization solution simplifies the deployment process while providing good scalability and maintainability. With the help of Kubernetes, the system achieves automatic expansion, load balancing, fault recovery, and update models. It maintains the system through a rolling update mechanism to ensure service continuity.

In addition, the system is deployed using the kubectl command-line tool, which supports version control and rollback functions, thereby enhancing operational and maintenance efficiency and simplifying maintenance tasks. The containerized design ensures good portability and scalability, laying a robust foundation for the future integration of AI analysis modules. The system employs a modular architecture and standardized interfaces, making it easy to integrate with other monitoring systems and providing flexibility for future expansion of monitoring functions.

High availability was verified through a series of Kubernetes-based tests, including Pod failure recovery, node drain simulation, and rolling updates. These tests demonstrated the system's self-healing capabilities, node-level disruption tolerance, and zero-downtime updates. With the Kubernetes container orchestration platform, the system achieves resilience, maintainability, and reliability in various deployment environments.

# CONCLUSION

In this study, we proposed an intelligent and scalable framework for critical infrastructure monitoring using 5G, Kubernetes, and AI. One of the main advantages of the system is its efficient deployment of specified containerized applications on all IoT nodes using Kubernetes DaemonSets, ensuring consistency and ease of management. The use of K3s, a lightweight distribution of Kubernetes, further enhances the applicability in resource-constrained environments by enabling efficient orchestration on low-power devices. The modular and containerized architecture also enables the straightforward integration of additional programs, such as predictive analytics modules and support for various sensor types, making the system scalable and adaptable to evolving monitoring needs. However, this study also has some limitations. No benchmarks were conducted against existing solutions, which limits the quantitative evaluation of performance. In addition, some important security issues, such as face vector data encryption and secure MQTT communication, were not fully addressed. In the future, we plan to implement a real-time alert system for intruder detection, predictive analytics for the weather monitoring component, and enhanced security mechanisms to ensure compliance with data protection regulations (e.g., GDPR). We will use Kubernetes HPA for scalability testing under dynamic workloads, benchmarking system performance, and integrating federated learning for model updates. Furthermore, to improve system resiliency, we will explore fallback communication mechanisms such as LoRaWAN or Wi-Fi 6 in case of 5G network failures. These improvements aim to enhance the robustness, security, and intelligence of the proposed critical infrastructure monitoring solution.

# Acknowledgments

This research was supported by ZTE-MMU Training Center.

# References

1. D. B., “The Infrastructure Imperative of Climate Change: Risk-Based Climate Adaptation of Infrastructure,” in *Climate Change - Research and Technology for Adaptation and Mitigation*, edited by J.A. Blanco, (InTech, 2011), pp. 293–324.
2. National Protection And Programs Directorate, *The Future of Smart Cities: Cyber-Physical Infrastructure Risk* (North America, 2015).
3. K. Elgazzar, H. Khalil, T. Alghamdi, A. Badr, G. Abdelkader, A. Elewah, and R. Buyya, “Revisiting the internet of things: New trends, opportunities and grand challenges,” Front. Internet Things **1**, (2022).
4. O. Khattach, O. Moussaoui, and M. Hassine, “End-to-End Architecture for Real-Time IoT Analytics and Predictive Maintenance Using Stream Processing and ML Pipelines,” Sensors 25(9), 2945 (2025).
5. D. Tamang, A. Pozzebon, L. Parri, A. Fort, and A. Abrardo, “Designing a Reliable and Low-Latency LoRaWAN Solution for Environmental Monitoring in Factories at Major Accident Risk,” Sensors (Basel) 22(6), 2372 (2022).
6. A. Ghosh, “Edge Ai Architecture: Optimizing Performance Through Distributed Intelligence,” IJITMIS 16(1), 578–591 (2025).
7. M.V.B. da Silva, M. Barbosa, A. Queiroz, and K.L. Dias, “A 5G-Edge Architecture for Computational Offloading of Computer Vision Applications,” (2025).
8. Y. Lu, S. Bian, L. Chen, Y. He, Y. Hui, M. Lentz, B. Li, F. Liu, J. Li, Q. Liu, R. Liu, X. Liu, L. Ma, K. Rong, J. Wang, Y. Wu, Y. Wu, H. Zhang, M. Zhang, Q. Zhang, T. Zhou, and D. Zhuo, “Computing in the Era of Large Generative Models: From Cloud-Native to AI-Native,” (2024).
9. “Beyond Traditional Autoscaling: The Future of Kubernetes in AI Infrastructure | DEVOPSdigest,” (n.d.).
10. D. Yakubov, and D. Hästbacka, “Comparative Analysis of Lightweight Kubernetes Distributions for Edge Computing: Performance and Resource Efficiency,” arXiv.Org, (2025).
11. L.M. Saini, P. Ajmani, V. Asha, K. Pithamber, and N. Sreevani, “KubeEdge for Scalable IoT Applications,” in 2024 7th International Conference on Contemporary Computing and Informatics (IC3I), (IEEE, Greater Noida, India, 2024), pp. 1–7.
12. “Case Studies,” KubeEdge, (n.d.). https://kubeedge.io/case-studies/.
13. C. Gutierrez, “KubeEdge: Monitoring Edge Devices at the World’s Longest Sea Bridge,” Altoros, (2022).
14. A. Rynhard, “When Kubernetes hits the edge, everything changes,” EDGEIR, (2025).
15. X. Wang, Z. Tang, J. Guo, T. Meng, C. Wang, T. Wang, and W. Jia, “Empowering Edge Intelligence: A Comprehensive Survey on On-Device AI Models,” ACM Comput. Surv. **57**(9), 1–39 (2025).
16. “Models - Object Detection,” Coral, (n.d.). https://coral.ai/models/object-detection/.
17. bergerda1, “bergerda1/PiFaceRecognition,” (2025). https://github.com/bergerda1/PiFaceRecognition.