Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 7, No. 3, June 2026, Page. 2790-2797
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.3.5731

Evaluation of Image Transmission Strategies on Edge Server-Based Centralized Object Detection Systems

Firmansyah Achmad Adam[*1], Bambang Harjito[2], Fajar Muslim[3]

[1,2,3]Informatics Study Program, Sebelas Maret University. Indonesia

Email: [1]adamfirman078@gmail.com

Received : Feb 8, 2026; Revised : Feb 10, 2026; Accepted : Feb 11, 2026; Published : Jun 15, 2026

Abstract

Urban waste management in smart city development requires efficient and stable visual monitoring systems. Utilization of edge devices such as Raspberry Pi is often constrained by limited computational power for complex computer vision models, making edge server architecture a relevant solution. This study evaluates the performance of image transmission from a Raspberry Pi to a centralized server for YOLOv8 object detection by comparing MJPEG streaming and HTTP POST-based periodic snapshot methods. Evaluation metrics included median latency (p50), jitter, and tail latency (p95 and p99). The results indicate that MJPEG streaming provides more stable latency compared to snapshots, particularly at tight transmission intervals. The transmission interval proved to have a significant effect on inference pipeline stability, while image resolution showed no observable impact on latency distribution under the evaluated conditions. This research recommends selecting appropriate transmission strategies to maintain the reliability of visual monitoring systems. These findings provide practical guidance for designing reliable centralized visual monitoring systems in resource-constrained edge environments.

Keywords : *End-To-End Latency, HTTP POST Snapshot, MJPEG Streaming, Raspberry Pi, Server-Based Object Detection.*

1. INTRODUCTION

Population growth in urban areas increases the complexity of waste management, which, when handled manually, often leads to operational inefficiencies. Although IoT technologies and edge devices like Raspberry Pi have been employed for real-time monitoring, the resource constraints of these devices limit their ability to efficiently execute complex computer vision and deep learning workloads [1], [2]. Consequently, an edge server architecture that separates data acquisition at the edge from inference at the server has become a primary choice [3].

Edge–cloud and edge server architectures have been widely adopted to overcome the computational limitations of edge devices by offloading deep learning inference tasks to more powerful centralized servers, enabling scalable and efficient visual analytics systems [4], [5], [6]. Recent performance evaluations also show that end-to-end latency and throughput of object detection pipelines vary significantly across edge devices and deployment settings, reinforcing the importance of system-level evaluation beyond model accuracy [7]. Recent studies further highlight that collaborative inference and adaptive partitioning strategies can significantly affect end-to-end latency and system efficiency in edge–server environments [8], [9]. Bandwidth-aware edge–cloud collaboration strategies have also been proposed to adapt transmission and inference decisions dynamically, which can affect latency stability under constrained networks [10].

However, research empirically evaluating the impact of HTTP-based image transmission strategies on end-to-end latency and system stability in resource-constrained devices remains limited.

Vol. 7, No. 3, June 2026, Page. 2790-2797
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.3.5731

Most studies focus more on model accuracy than on data flow consistency. Characteristics such as tail latency and jitter are crucial factors determining the quality of service in visual monitoring systems. This study aims to evaluate the performance of MJPEG streaming and periodic HTTP POST snapshots, and to analyze the influence of transmission intervals on system stability [11], [12], [13].

Previous studies on edge-based object detection systems primarily emphasize model accuracy, computational efficiency, or edge–cloud collaboration strategies. However, limited attention has been given to the behavior of image transmission mechanisms at the application layer, particularly regarding latency stability, jitter, and tail latency in centralized inference architectures. Moreover, comparative evaluations between continuous MJPEG streaming and discrete HTTP-based snapshot transmission under varying image delivery intervals remain underexplored. This lack of empirical evaluation makes it difficult to assess the suitability of each transmission strategy for latency-sensitive object detection systems.

Continuous image and video transmission mechanisms such as MJPEG and WebRTC-based streaming have been widely explored for resource-constrained devices, showing that sustained streaming can improve transmission stability and reduce latency fluctuation compared to discrete transmission methods [14], [15].

The contributions of this study are threefold. First, this work provides an empirical comparison between MJPEG streaming and periodic HTTP snapshot transmission in a centralized object detection system. Second, the impact of image transmission intervals on latency stability, jitter, and tail latency is systematically analyzed, which are key indicators of reliability in latency-sensitive visual systems [16]. Third, this study presents experimental evidence on the behavior of end-to-end latency distribution in server-based inference pipelines using Raspberry Pi as an edge device [17].

2. METHOD

The experimental configuration used in this study is summarized in Table 1.

Table 1. Experimental Setup

Parameter	Value
Edge Device	Raspberry Pi 5 (4 GB RAM)
Server OS	Windows 11
Detection Model	YOLOv8n.pt
Image Resolution	320p, 640p
Transmission Interval	0.2 s, 0.5 s, 1 s
Samples of Scenario	40
Network type	Wi-Fi (Iconnet 20 Mbps)

This study utilizes an edge server architecture design with a Raspberry Pi serving as the edge node for frame acquisition via a webcam and performing JPEG compression. The centralized server runs an inference service using FastAPI and the YOLO-based object detection model [18]. The workflow begins with the edge device sending images to the server via the HTTP protocol. On the server side, images are decoded into OpenCV arrays and processed by YOLOv8 to generate bounding box coordinates and confidence scores [19]. All latency data is recorded from the moment the frame is sent until the response is received back by the edge device. Testing scenarios involved variations in intervals (0.2s, 0.5s, and 1s) and image resolutions (320p and 640p). Each scenario was executed with 40 samples to capture latency distribution characteristics rather than point estimates. Percentile-based metrics were selected to represent both typical performance (p50) and worst-case behavior (p95 and p99), which are critical for evaluating stability in latency-sensitive systems [20], [21], [22].

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 7, No. 3, June 2026, Page. 2790-2797
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.3.5731

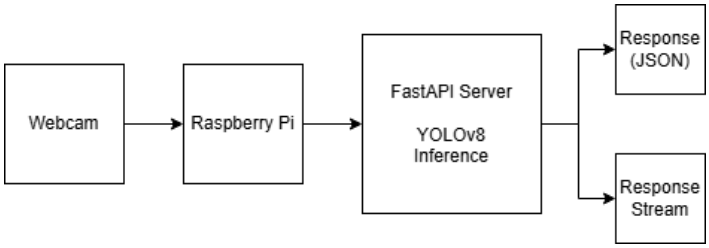

Figure 1. Edge server-based centralized object detection architecture

End-to-end latency is defined as the time difference between the moment an image is transmitted from the edge device and the moment the inference result is received by the client, as formulated in Equation (1). Latency measurement is calculated using timestamp logging at both the client and server sides. The total latency includes image transmission, server-side processing, and inference execution, while network and processing delays are implicitly captured within this measurement.

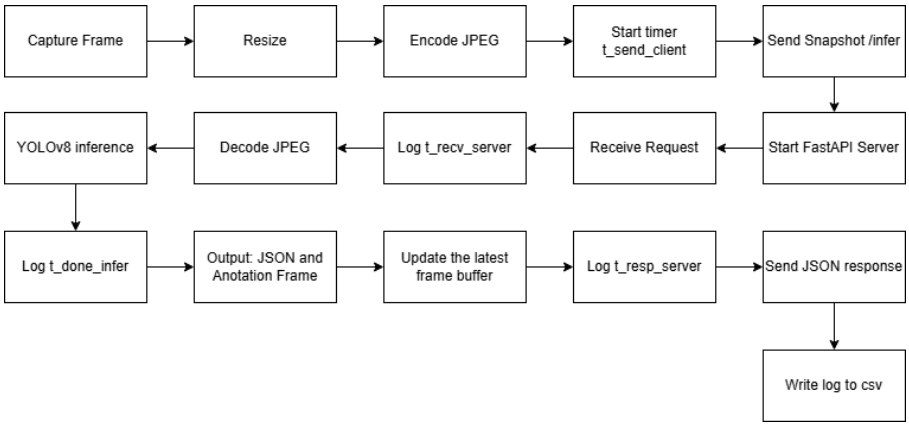

Figure 2. Image transmission and inference workflow

End-to-End Latency

$$L = t\_response - t\_request \quad (1)$$

Median Latency (p50)

$$p50 = median(L) \quad (2)$$

Tail Latency

$$p95 = percentile(L, 95) \quad (3)$$

$$p99 = percentile(L, 99) \quad (4)$$

Jitter

$$J = sqrt(\,(1/N) * \Sigma\,(Li - \mu)^2\,) \quad (5)$$

3. RESULTS

This section presents the experimental results of end-to-end latency evaluation for centralized object detection using MJPEG streaming and periodic HTTP snapshot transmission. Performance is analyzed using percentile-based metrics, including median latency (p50), tail latency (p95 and p99), and jitter under different transmission intervals and image resolutions.

Vol. 7, No. 3, June 2026, Page. 2790-2797
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.3.5731

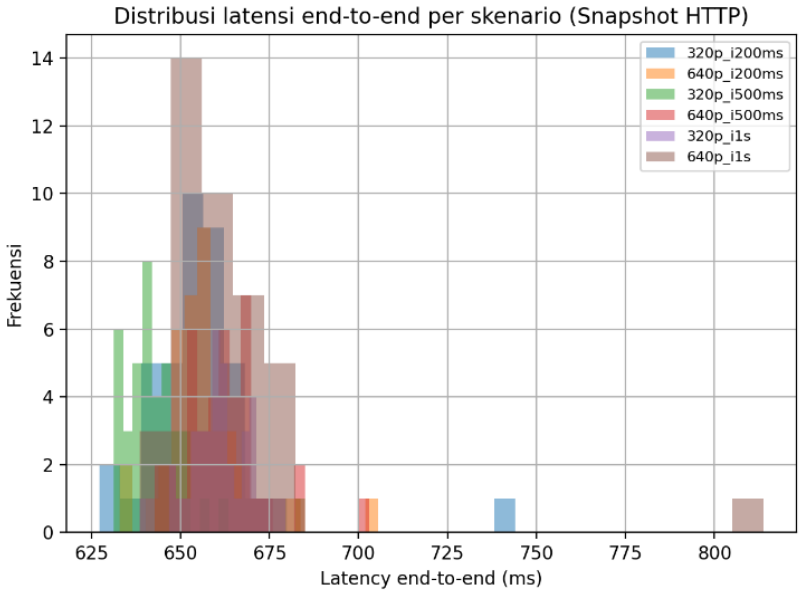

Figure 3. Median end-to-end latency (p50) under varying transmission intervals for Snapshot HTTP and MJPEG streaming at 320p and 640p resolutions

Figure 3 presents the median end-to-end latency (p50) for both transmission strategies across different transmission intervals and image resolutions. The measured median latency values range from 641 ms to 662 ms across all evaluated scenarios.

The results show that changes in image resolution from 320p to 640p do not produce a noticeable difference in median latency for either MJPEG streaming or HTTP snapshot transmission. Similarly, variations in transmission interval only slightly affect p50 latency values, indicating that median latency alone is insufficient to fully describe system performance stability.

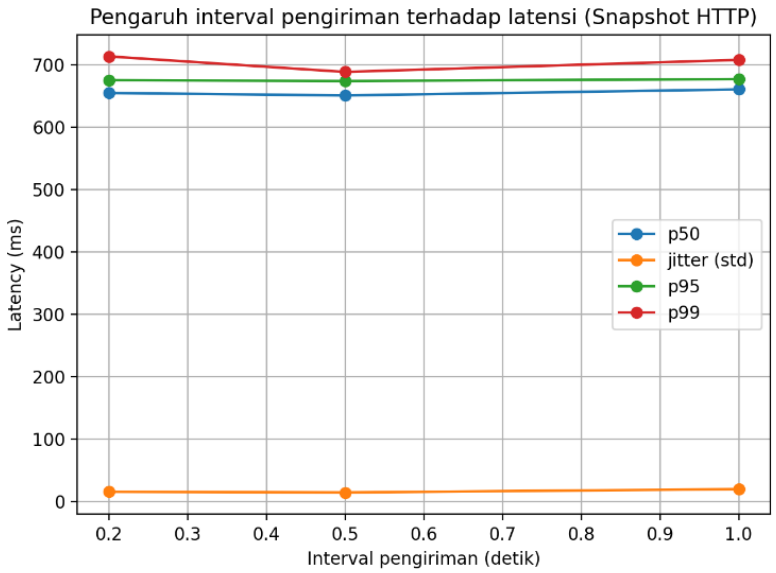

Figure 4. Tail latency (p95 and p99) comparison across transmission intervals for Snapshot HTTP and MJPEG streaming

Tail latency results, represented by the 95th percentile (p95) and 99th percentile (p99), are illustrated in Figure 4. Compared to snapshot-based transmission, MJPEG streaming consistently demonstrates lower p95 and p99 latency values across all tested transmission intervals.

Vol. 7, No. 3, June 2026, Page. 2790-2797
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.3.5731

At longer transmission intervals (1 s), both methods exhibit increased tail latency; however, the increase is more pronounced for HTTP snapshot transmission. These results indicate that worst-case latency behavior is more sensitive to transmission strategy and interval than to image resolution.

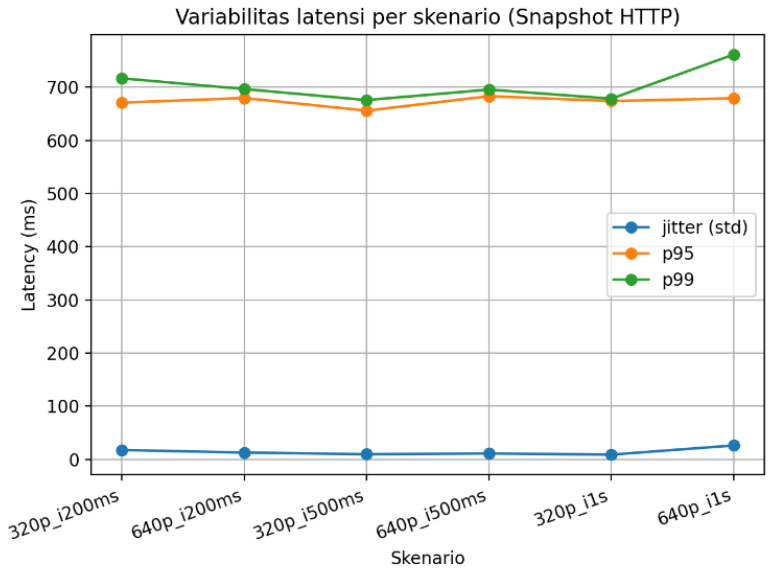

Figure 5. Jitter comparison across transmission strategies under different image resolutions and transmission intervals

Figure 5 shows the jitter comparison for both transmission methods under varying resolutions and transmission intervals. MJPEG streaming produces lower jitter values compared to HTTP snapshot transmission, particularly at shorter transmission intervals (0.2 s and 0.5 s).

The snapshot-based approach demonstrates higher latency fluctuations, indicating reduced temporal consistency between consecutive inference requests. This behavior is observed consistently across both image resolutions, further confirming that jitter is largely independent of payload size under the evaluated conditions. Nevertheless, this interpretation is based on empirical observation, and further investigation is required to isolate the exact contributing factors.

Several limitations should be considered when interpreting the results of this study. The experiments were conducted under a homogeneous network environment, which may not fully represent real-world conditions with fluctuating bandwidth and packet loss. Additionally, the evaluation was limited to a single object detection model and a fixed number of samples per scenario. Future studies should extend the evaluation to heterogeneous network settings, larger workloads, and different inference models to improve generalizability.

4. DISCUSSION

The experimental results demonstrate that the choice of image transmission strategy significantly influences end-to-end latency stability in centralized object detection systems. Although median latency (p50) values across all scenarios remained within a comparable range, the variability characteristics, such as tail latency (p95 and p99) and jitter, differed substantially between MJPEG streaming and periodic HTTP snapshot transmission. Similar behavior has been reported in prior studies on continuous inference workloads, where sustained request patterns contribute to more stable processing times [14], [15].

The observed reduction in latency variability under MJPEG streaming is likely related to the inference pipeline remaining in an active or warm state, reducing initialization overhead between

Vol. 7, No. 3, June 2026, Page. 2790-2797
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.3.5731

consecutive requests. MJPEG streaming consistently exhibited lower latency variance and reduced tail latency compared to snapshot-based transmission. This behavior indicates that continuous streaming helps maintain the inference pipeline in an active or warm execution state on the server side. As a result, initialization overheads and resource reallocation delays between consecutive inference requests are minimized. Similar observations have been reported in prior studies on sustained inference workloads, where continuous request patterns lead to more predictable processing times and improved system stability.

Transmission interval was found to be a critical factor affecting latency stability. Shorter intervals (0.2 s and 0.5 s) produced more consistent latency distributions than the 1-second interval for both transmission methods. At longer intervals, the inference service is more likely to enter an idle state, which introduces sporadic latency spikes when new requests arrive. This phenomenon explains the increased p95 and p99 values observed under lower transmission frequencies and highlights the importance of request continuity for latency-sensitive visual monitoring applications.

Interestingly, increasing image resolution from 320p to 640p did not result in a measurable impact on latency distribution. This suggests that, under the evaluated conditions, network transmission time and server-side inference overhead dominate the end-to-end latency rather than payload size. This finding implies that system designers may prioritize image quality improvements without significantly compromising latency, provided that the inference pipeline and network conditions remain stable. This observation is consistent with studies reporting that system bottlenecks in edge-based detection are often dominated by compute pipeline behavior and networking conditions rather than pixel-level payload differences under moderate bandwidth [7].

From a system design perspective, these findings emphasize that transmission strategy selection is as important as model optimization in centralized object detection architectures. While snapshot-based HTTP transmission may reduce bandwidth usage, it introduces higher latency variability that can negatively affect real-time monitoring reliability. In contrast, MJPEG streaming offers improved temporal consistency, making it more suitable for applications requiring stable response behavior rather than minimal average latency.

Nevertheless, this study has several limitations. The experiments were conducted in a controlled and homogeneous network environment, which may not fully represent real-world conditions with fluctuating bandwidth, congestion, or packet loss. Furthermore, the evaluation was limited to a single object detection model and a fixed workload size. Future research should extend the analysis to heterogeneous network scenarios, larger-scale workloads, and alternative communication techniques such as learned image compression and timeliness-aware transmission to further validate generalizability and reduce tail latency under constrained links [23], [24]. In addition, semantic communication approaches that transmit task-relevant representations rather than full images may further improve bandwidth efficiency for IoT visual monitoring systems [25].

5. CONCLUSION

This study evaluated the impact of image transmission strategies on end-to-end latency stability in an HTTP-based edge server centralized object detection system using Raspberry Pi as an edge device. Experimental results demonstrate that MJPEG streaming consistently yields lower latency variability compared to periodic HTTP snapshot transmission, particularly under shorter transmission intervals.

The findings indicate that transmission frequency plays a critical role in maintaining inference pipeline stability, as higher request rates help preserve a warm execution state on the server side, thereby reducing latency spikes and jitter. In contrast, image resolution showed no significant influence on latency distribution under the evaluated conditions, suggesting that system performance is more sensitive to transmission behavior than payload size.

Vol. 7, No. 3, June 2026, Page. 2790-2797
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.3.5731

These results provide practical insights for designing reliable visual monitoring systems that rely on centralized inference, especially in resource-constrained edge environments. Future work should extend this evaluation to heterogeneous network conditions, larger-scale workloads, and alternative communication protocols to further validate the generalizability of the proposed findings.

ACKNOWLEDGEMENT

Acknowledgement is only addressed to funders or donors and object of research. Acknowledgement can also be expressed to those who helped carry out the research.

REFERENCES

[1] S. Wan et al., "Insights into the urban municipal solid waste generation during the COVID-19 pandemic from machine learning analysis," Sustainable Cities and Society, vol. 100, Jan. 2024, doi: 10.1016/j.scs.2023.105044.

[2] [2] A. A. Ravindran, "Edge Computing Systems for Streaming Video Analytics: Trail Behind and the Paths Ahead," 2023, doi: 10.20944/preprints202308.0383.v1.

[3] [3] X. R. Huang, S. S. Yang, W. S. Chen, Y. Q. Zhang, C. T. Lee, and L. B. Chen, "An IoT-Based Smart Trash Cans Monitoring System," in 2021 IEEE 10th Global Conference on Consumer Electronics (GCCE), 2021, pp. 623–624, doi: 10.1109/GCCE53005.2021.9621358.

[4] [4] G. Liu et al., "An adaptive DNN inference acceleration framework with end–edge–cloud collaborative computing," Future Generation Computer Systems, vol. 140, pp. 422–435, Mar. 2023, doi: 10.1016/j.future.2022.10.033.

[5] [5] H. Kim, J. S. Choi, J. Kim, and J. H. Ko, "A DNN partitioning framework with controlled lossy mechanisms for edge-cloud collaborative intelligence," Future Generation Computer Systems, vol. 154, pp. 426–439, May 2024, doi: 10.1016/j.future.2024.01.006.

[6] [6] G. Rong, Y. Xu, X. Tong, and H. Fan, "An edge-cloud collaborative computing platform for building AIoT applications efficiently," Journal of Cloud Computing, vol. 10, no. 1, Dec. 2021, doi: 10.1186/s13677-021-00250-w.

[7] [7] D. G. Lema, R. Usamentiaga, and D. F. García, "Quantitative comparison and performance evaluation of deep learning-based object detection models on edge computing devices," Integration, vol. 95, Mar. 2024, doi: 10.1016/j.vlsi.2023.102127.

[8] [8] Y. Wang, G. Zhong, Y. Duan, Y. Cheng, M. Yin, and R. Yang, "Efficient and privacy-preserving deep inference towards cloud–edge collaborative," Applied Soft Computing, vol. 180, Aug. 2025, doi: 10.1016/j.asoc.2025.113381.

[9] [9] Y. Ma, Y. Wang, and B. Tang, "Joint Optimization of Model Partitioning and Resource Allocation for Multi-Exit DNNs in Edge-Device Collaboration," Electronics, vol. 14, no. 8, Apr. 2025, doi: 10.3390/electronics14081647.

[10] [10] Z. Cao, Y. Cheng, Z. Zhou, Y. Chen, Y. Hu, A. Lu, J. Liu, and Z. Li, "Edge-Cloud Collaborated Object Detection via Bandwidth Adaptive Difficult-Case Discriminator," IEEE Transactions on Mobile Computing, 2024, doi: 10.1109/TMC.2024.3474743.

[11] [11] A. Ghotbou and M. Khansari, "VE-CoAP: A constrained application layer protocol for IoT video transmission," Journal of Network and Computer Applications, vol. 173, Jan. 2021, doi: 10.1016/j.jnca.2020.102855.

[12] [12] A. Ghotbou and M. Khansari, "Comparing application layer protocols for video transmission in IoT low power lossy networks: an analytic comparison," Wireless Networks, vol. 27, no. 1, pp. 269–283, Jan. 2021, doi: 10.1007/s11276-020-02453-6.

[13] [13] Y. Jiang, P. Zhao, C. Zhao, and J. Lin, "Towards bandwidth efficient edge–cloud collaborative deep learning with Data Importance driven Compression," Neurocomputing, vol. 650, Oct. 2025, doi: 10.1016/j.neucom.2025.130835.

[14] [14] B. Diallo, A. Ouamri, and M. Keche, "A Hybrid Approach for WebRTC Video Streaming on Resource-Constrained Devices," Electronics, vol. 12, no. 18, Sep. 2023, doi: 10.3390/electronics12183775.

Vol. 7, No. 3, June 2026, Page. 2790-2797
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2026.7.3.5731

[15] [15] J. Nakazato, K. Nakagawa, K. Itoh, R. Fontugne, M. Tsukada, and H. Esaki, "WebRTC over 5G: A Study of Remote Collaboration QoS in Mobile Environment," Journal of Network and Systems Management, vol. 32, no. 1, Mar. 2024, doi: 10.1007/s10922-023-09778-5.
[16] [16] J. Ma, B. Shang, H. Song, Y. Huang, and P. Fan, "Reliability Versus Latency in IIoT Visual Applications: A Scalable Task Offloading Framework," IEEE Internet of Things Journal, vol. 9, no. 17, pp. 16726–16735, Sep. 2022, doi: 10.1109/JIOT.2022.3148115.
[17] [17] Y. Bandung et al., "Performance Evaluation of CoAP Communication Method Extensions in Internet of Video Things," Journal of Advances in Information Technology, vol. 16, no. 7, pp. 1017–1029, 2025, doi: 10.12720/jait.16.7.1017-1029.
[18] [18] L. Lu, Z. Chen, R. Wang, L. Liu, and H. Chi, "Yolo-inspection: defect detection method for power transmission lines based on enhanced YOLOv5s," Journal of Real-Time Image Processing, vol. 20, no. 5, Oct. 2023, doi: 10.1007/s11554-023-01360-1.
[19] [19] Z. Lei, Y. Zhang, J. Wang, and M. Zhou, "Cloud-Edge Collaborative Defect Detection Based on Efficient Yolo Networks and Incremental Learning," Sensors, vol. 24, no. 18, Sep. 2024, doi: 10.3390/s24185921.
[20] S. Nazir and M. Kaleem, "Object classification and visualization with edge artificial intelligence for a customized camera trap platform," Ecological Informatics, vol. 79, Mar. 2024, doi: 10.1016/j.ecoinf.2023.102453.
[21] N. Saha, P. Paul, K. Ji, and R. Harik, "Performance evaluation framework of MQTT client libraries for IoT applications in manufacturing," Manufacturing Letters, vol. 41, pp. 1237–1245, 2024.
[22] C. Caiazza, V. Luconi, and A. Vecchio, "Energy consumption of smartphones and IoT devices when using different versions of the HTTP protocol," *Pervasive Mob. Comput.*, vol. 97, Jan. 2024, doi: 10.1016/j.pmcj.2023.101871.
[23] M. Naseri et al., "Deep Learning-Based Image Compression for Wireless Communications: Impacts on Reliability, Throughput, and Latency," arXiv:2411.10650, 2024.
[24] X. Yang, Z. Wang, Z. Qin, and X. Tao, "Timeliness-Aware Joint Source and Channel Coding for Adaptive Image Transmission," arXiv:2509.19754, 2025.
[25] G. Ma, H. Tong, N. Yang, and C. Yin, "Attention-based UNet enabled Lightweight Image Semantic Communication System over Internet of Things," arXiv:2401.07329, 2024.