Jurnal E-Komtek
Vol. 9, No. 2. (2025) pp. 537-554
https://jurnal.politeknik-kebumen.ac.id/index.php/E-KOMTEK
p-ISSN : 2580-3719 e-ISSN : 2622-3066

Design and Implementation of an IoT-Enabled Deep Learning Vision System for Automated Dimensional Measurement in Smart Manufacturing

Waluyo Nugroho[1], **Afianto**[2], **Agus Ponco**[3]

[1,2,3] Department of Mechatronics, Politeknik Astra, Indonesia, 14330

nugroho.research@gmail.com

https://doi.org/10.373 39/e-komtek.v9i2.2855

Published by Politeknik Piksi Ganesha Indonesia

Artikel Info
Submitted:
13-11-2025
Revised:
02-01-2026
Accepted:
07-01-2026
Online first :
07-01-2026

Abstract

The rapid advancement of Industry 4.0 has brought the convergence of Internet of Things (IoT), computer vision, and deep learning to enhance automation and precision in manufacturing. This paper presents the design and implementation of an IoT-enabled deep learning vision system for automated dimensional measurement, integrated with programmable logic controller (PLC) control and real-time monitoring. The system employs a Raspberry Pi 5 as an edge computing unit, Logitech C270 camera for visual data acquisition, and an Omron CP2E PLC for process control. A YOLOv5 deep learning model is trained to detect and measure object dimensions with sub-millimeter accuracy. The Node-RED platform is utilized for dashboard visualization and communication, interfaced through Omron FINS protocol, with MySQL as the database for data logging. Experimental results show a high detection accuracy of 98.6% and an average measurement error of less than 0.5 mm, demonstrating the system's effectiveness for smart manufacturing applications.

Keywords: *Computer Vision, Deep Learning, IoT, Raspberry Pi, Smart Manufacturing.*

Abstrak

Kemajuan pesat Industri 4.0 telah membawa konvergensi Internet of Things (IoT), visi komputer, dan pembelajaran mendalam untuk meningkatkan otomatisasi dan presisi dalam manufaktur. Makalah ini menyajikan desain dan implementasi sistem visi pembelajaran yang mendalam berkemampuan IoT untuk pengukuran dimensi otomatis, terintegrasi dengan kontrol pengontrol logika terprogram (PLC) dan pemantauan waktu nyata. Sistem ini menggunakan Raspberry Pi 5 sebagai unit komputasi tepi, kamera Logitech C270 untuk akuisisi data visual, dan PLC Omron CP2E untuk kontrol proses. Model pembelajaran mendalam YOLOv5 dilatih untuk mendeteksi dan mengukur dimensi objek dengan akurasi sub-milimeter. Platform Node-RED digunakan untuk visualisasi dan komunikasi dasbor, dihubungkan melalui protokol Omron FINS, dengan MySQL sebagai basis data untuk pencatatan data. Hasil eksperimen menunjukkan akurasi deteksi tinggi sebesar 98,6% dan kesalahan pengukuran rata-rata kurang dari 0,5 mm, yang menunjukkan efektivitas sistem untuk aplikasi manufaktur pintar.

Kata kunci: *Computer Vision, Deep Learning, IoT, Raspberry Pi, Smart Manufacturing.*

1. Introduction

The emergence of Industry 4.0 has transformed conventional manufacturing systems into intelligent, interconnected environments capable of self-optimization and autonomous decision-making [1]. Central to this transformation is the integration of Internet of Things (IoT), Artificial Intelligence (AI), and computer vision technologies into the production workflow [2][3]. These systems enable the collection, processing, and analysis of data in real time, providing actionable insights that enhance production efficiency, product quality, and traceability [4]. Among the key tasks in modern industrial automation is dimensional measurement, a critical process for ensuring that manufactured parts meet specified tolerances [5]. Traditional manual measurement methods using calipers, micrometers, or coordinate measuring machines (CMMs) offer high precision but lack scalability, speed, and real-time adaptability. As industries move toward smart manufacturing, there is a growing demand for automated, intelligent dimensional measurement systems that can operate autonomously and continuously without human supervision [6].

To address this need, computer vision has emerged as a powerful enabler for non-contact, automated measurement. The integration of deep learning models such as Convolutional Neural Networks (CNNs) and object detection algorithms like YOLO (You Only Look Once) has enabled vision systems to perform robust object recognition and localization, providing the geometric cues required for dimensional analysis in complex industrial environments [7]. Several studies have demonstrated the potential of deep learning-based vision systems in industrial inspection. A CNN-based system for defect detection in automotive parts with an accuracy of 97%, while integrated YOLOv5 for dimensional inspection achieving sub-millimeter precision [8]. However, many existing implementations are evaluated as stand-alone vision modules and depend on high-performance GPUs or centralized computing, which increases system cost, energy consumption, and network dependency. More importantly, the end-to-end integration of YOLO-based edge vision for real-time dimensional measurement with deterministic shop-floor control and IoT-level visualization and traceability remains insufficiently addressed.[9].

Edge computing also addresses major challenges in industrial IoT systems, particularly data-processing latency and network reliability. The Raspberry Pi 5, equipped with a quad-core ARM Cortex-A76 processor and 8 GB RAM, provides sufficient computational capability for executing lightweight deep learning models such as YOLOv5 at the edge. This allows the device to process high-resolution image data from a Logitech C270 camera in real time, reducing reliance on cloud-based computation. Previous research demonstrated that implementing deep learning inference at the edge can improve response time by 30-50% compared to cloud systems [10]. Furthermore, the integration of OpenCV and TensorRT acceleration enhances inference speed, enabling real-time dimensional measurements of moving objects on production lines. This approach aligns with the current trend of edge-intelligent manufacturing, where computing power is distributed closer to the data source to achieve autonomy and resilience [11].

While computer vision provides perception and measurement, control execution in industrial environments still relies heavily on Programmable Logic Controllers (PLCs) due to their robustness and deterministic behavior. The Omron CP2E PLC is particularly suitable for integration with edge AI systems because it supports the FINS (Factory Interface Network Service) communication protocol over Ethernet, enabling reliable data exchange between PLCs and IoT devices. Several studies have explored PLC-vision integration, typically using standard field protocols such as Modbus to transmit discrete pass/fail decisions from a vision module to a PLC [12]. Although such approaches demonstrate feasibility, they rarely report a closed-loop pipeline that performs sub-millimeter dimensional measurement at the edge, communicates with Omron PLCs via FINS, and simultaneously exposes measurement results for real-time monitoring and traceability. As a result, the practical integration of YOLO-based edge vision, FINS-based PLC control, and IoT orchestration within a single real-time measurement workflow remains a clear research gap.

In the realm of IoT-driven data visualization and analytics, Node-RED has become a popular open-source tool for designing interactive, event-driven control dashboards. Its flow-based programming model allows rapid integration of data sources, controllers, and databases, and has been reported effective for visualizing real-time industrial process data, including Node-RED-based MQTT communication for smart factories [13]. In this research, Node-RED acts as the communication middleware between the Raspberry Pi 5, Omron CP2E PLC, and the MySQL database, leveraging the node-red-contrib-omron-fins library for direct PLC interaction. The MySQL database stores measurement results and system logs, enabling traceability, historical analysis, and quality tracking [13]. This architecture also provides a foundation for future integration with higher-level analytics such as digital twin models.

Considering these advances and the identified gap, this paper presents the design and implementation of an IoT-enabled deep learning vision system for automated dimensional measurement in smart manufacturing environments. The proposed system combines Raspberry Pi 5, Logitech C270 camera, a YOLOv5 deep learning model, and an Omron CP2E PLC with Node-RED and MySQL for communication, visualization, and data storage [14]. This configuration enables real-time dimensional inspection, feedback control, and data management through a unified IoT infrastructure. The main contributions of this study are: (1) an edge-deployed YOLOv5 pipeline for real-time dimensional measurement, (2) deterministic integration with an industrial PLC via the Omron FINS protocol with measured end-to-end latency, and (3) Node-RED- and MySQL-based visualization and traceability to support production monitoring and process optimization. Overall, the study bridges AI-driven perception with industrial-grade control and IoT monitoring to support autonomous, data-driven smart manufacturing.

2. System Design and Methodology

The methodology of this research focuses on the design, integration, and implementation of an IoT-enabled deep learning vision system for automated dimensional measurement within a smart manufacturing environment. The proposed framework combines edge-based computer vision, industrial control via PLC, and IoT-based monitoring through a unified architecture. The methodology comprises four major stages: (1) System architecture design, (2) Hardware and software integration, (3) Deep learning model development, and (4) IoT communication and data management. Each stage is elaborated in detail below.

a. System Architecture Overview

The proposed IoT-enabled deep learning vision system is structured as an integrated multi-layer architecture that connects visual intelligence, industrial control, and IoT-based data management within a unified smart manufacturing framework **[15]**. At the perception layer, a Logitech C270 HD webcam captures real-time images of workpieces on a conveyor line under controlled lighting. These images are processed by a Raspberry Pi 5, functioning as an edge AI computing device running a YOLOv5 deep learning model for object detection and dimensional estimation [13]. The system performs pixel-to-millimeter calibration using reference markers to convert pixel distances into accurate physical measurements, achieving sub millimeter precision. The decision logic embedded in the Raspberry Pi determines whether an object meets dimensional tolerances and immediately transmits the result to the control layer **[14].** This edge-based inference minimizes network dependency and ensures low-latency inspection, making the system suitable for continuous, real-time operation on industrial production lines.

The control and IoT integration layers ensure that measurement results are translated into actionable process responses **[16]**. An Omron CP2E Programmable Logic Controller (PLC) is used to execute control commands such as part acceptance, rejection, or alarm signaling based on the received data from the vision module. Communication between the Raspberry Pi and PLC is achieved via the Factory Interface Network Service (FINS) protocol over Ethernet, using the node-red-contrib-omron-fins library integrated into Node-RED. Node-RED also serves as the IoT middleware, linking the PLC, Raspberry Pi, and a MySQL database for real-time visualization and data logging. Through the Node-RED web dashboard, operators can monitor live camera feeds, measured dimensions, system status, and production statistics from any connected device. All inspection data are recorded in the MySQL database for traceability and statistical process control [17]. This integrated architecture comprising vision, control, communication, and data management creates a closed-loop intelligent measurement ecosystem, enabling autonomous dimensional verification and feedback-driven manufacturing consistent with Industry 4.0 standards.

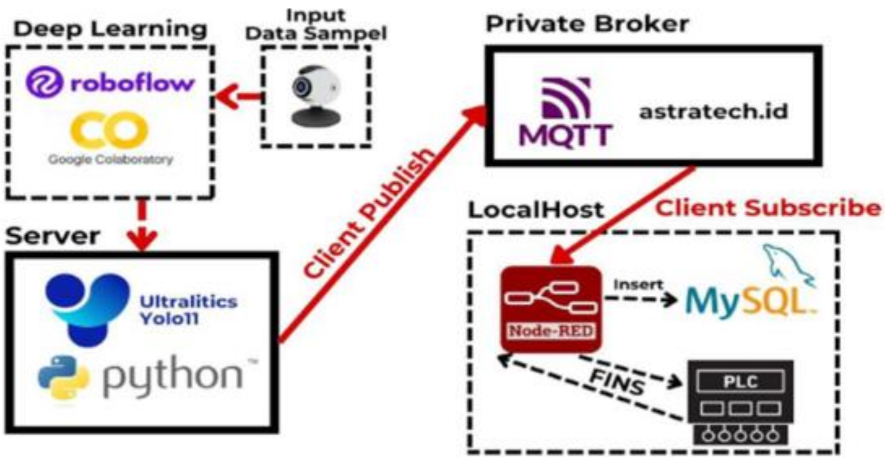

Figure 1. Overall Architecture System

b. Hardware Configuration

The hardware configuration of the proposed IoT-enabled deep learning vision system was carefully designed to achieve a balance between computational efficiency, system reliability, and industrial scalability. The main processing device used in this research is the Raspberry Pi 5, which functions as an edge AI computing unit responsible for running real-time deep learning inference and data processing. The Raspberry Pi 5 is equipped with a 2.4 GHz quad-core ARM Cortex-A76 CPU, 8 GB LPDDR4X RAM, and an onboard Gigabit Ethernet interface. This hardware configuration offers up to 2.5× performance improvement compared to the Raspberry Pi 4, enabling the execution of complex neural networks such as YOLOv5 without requiring a GPU server or cloud computing. The operating system is based on Ubuntu 22.04 (64-bit), optimized for Python and OpenCV libraries [18]. A 128 GB Class 10 microSD card is used to store model weights, datasets, and Node-RED flow files. The device is powered by a 5V/3A DC power adapter, with an aluminum heatsink and dual micro-fan assembly for temperature stabilization during continuous inference. This edge configuration ensures low power consumption (<12 W) while maintaining consistent performance under real industrial workloads.

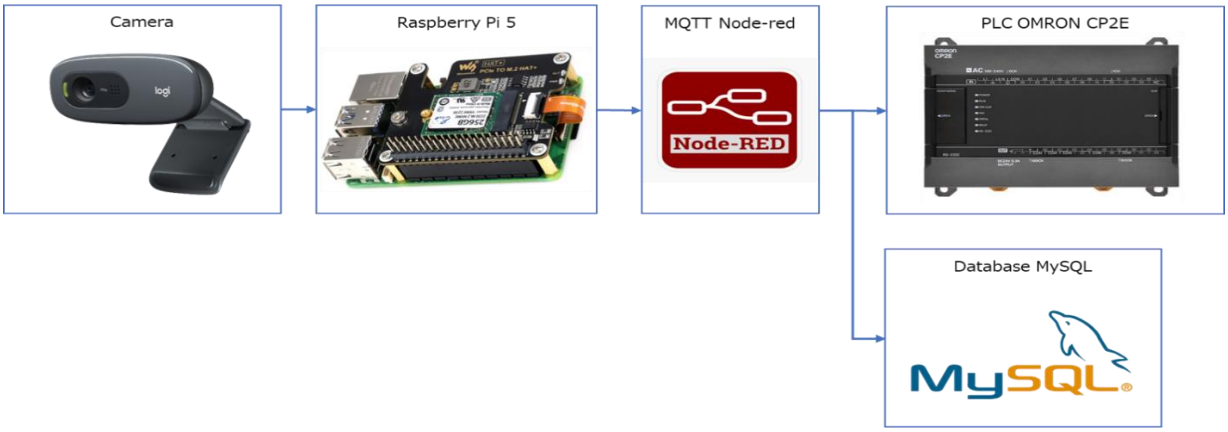

Figure 2. Hardware and Software Block Diagram

The image acquisition subsystem is built using a Logitech C270 HD webcam, selected for its high compatibility with open-source computer vision frameworks and reliable image quality. The camera captures images at 720p resolution (1280×720) with a 60° field of view, sufficient for monitoring parts ranging from 10 mm to 150 mm in size. It is mounted at a fixed distance of 25 cm above the conveyor belt using an adjustable aluminum frame equipped with vibration dampers to reduce motion artifacts. Controlled LED ring lighting is installed around the camera lens to maintain uniform illumination and eliminate shadows—an essential factor for consistent edge detection and object measurement accuracy. The webcam transmits continuous frames via a USB 2.0 connection to the Raspberry Pi 5, where preprocessing steps such as contrast normalization, Gaussian blurring, and background subtraction are applied. These preprocessing operations reduce noise and improve YOLOv5 detection robustness, allowing the system to achieve sub-millimeter measurement precision in varying lighting and environmental conditions [19].

The control and integration subsystem centers around the Omron CP2E Programmable Logic Controller (PLC), which ensures deterministic and reliable process control. The PLC operates using a 24V DC regulated power supply and is responsible for executing actuator commands, such as activating pneumatic ejectors, conveyors, or warning indicators, based on the dimensional inspection results from the vision module. Communication between the Raspberry Pi 5 and the PLC is established via the Factory Interface Network Service (FINS) protocol over a Cat-6 Ethernet cable, providing stable and low-latency data transfer. The node-red-contrib-omron-fins library is employed to write measurement results directly into the PLC's

Data Memory (DM) registers and to read back system status in real time. This bidirectional communication creates a closed-loop inspection system where visual inference directly triggers control actions [20]. All hardware components, including the Raspberry Pi, PLC, power supply, and Ethernet switch, are mounted inside an industrial-grade control enclosure with proper electrical shielding to prevent electromagnetic interference (EMI). The enclosure design also ensures ease of maintenance and scalability, allowing additional sensors, actuators, or network nodes to be integrated into the system in future deployments.

c. Software Architecture

The software architecture of the proposed IoT-enabled deep learning vision system was developed to ensure real-time interoperability, scalability, and modularity between computer vision, control, and IoT layers [21]. The system operates entirely on a Raspberry Pi 5 running Ubuntu 22.04 (64-bit), which hosts all major software components including Python 3.11, OpenCV 4.9, PyTorch 2.1, Node-RED v4.0, and MySQL 8.0. The architecture is composed of four integrated modules: (1) the Computer Vision and Inference Module, responsible for real-time object detection and dimensional analysis using the YOLOv5 deep learning model; (2) the Control Communication Module, which exchanges data with the Omron CP2E PLC through the Factory Interface Network Service (FINS) protocol; (3) the IoT Orchestration and Visualization Module, built on Node-RED for dashboard display and system management; and (4) the Data Management Module, which stores dimensional measurements and control data in a MySQL database [22]. This modular structure allows each software component to function independently while maintaining seamless synchronization across all operational layers [23].

In operation, the Computer Vision Module continuously acquires image streams from the Logitech C270 camera, performs preprocessing (contrast normalization, noise reduction), and executes YOLOv5 inference to detect objects and calculate dimensions in millimeters using a pixel-to-mm calibration matrix. The measurement data are formatted as JSON and transmitted via Node-RED to the PLC through the FINS protocol, where real-time control logic is executed for product classification or rejection. Node-RED simultaneously logs the measurement results and system status into the MySQL database, while updating the web-based dashboard with live dimensional readings, PLC states, and process statistics. This software architecture effectively merges edge AI computing, deterministic industrial control, and IoT-based visualization within a single embedded system. It enables closed-loop decision-making, traceability, and autonomous dimensional verification, forming a fully integrated cyber-physical measurement ecosystem aligned with Industry 4.0 standards [7].

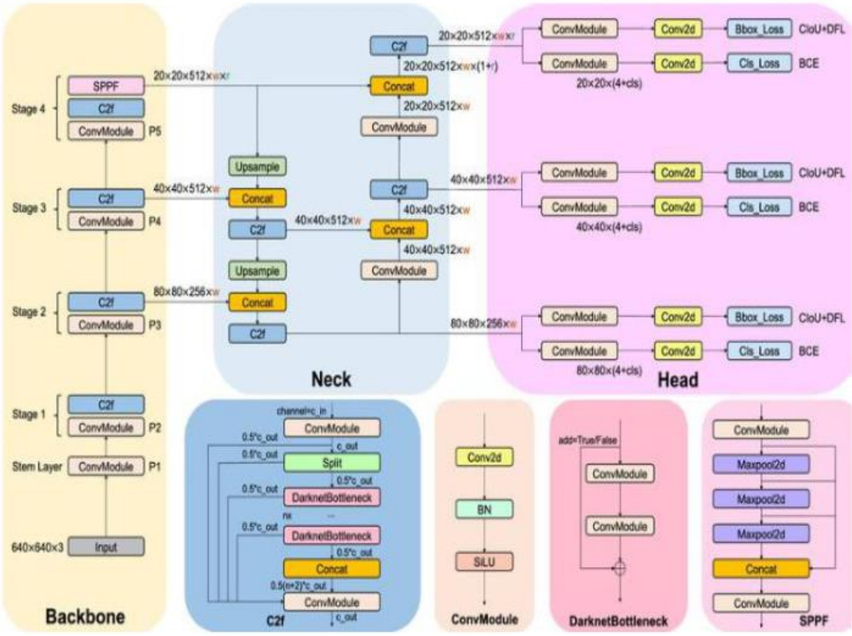

Figure 3. YOLO Deep Learning Architecture

3. Deep Learning Model Development

A custom dataset was collected using the same optical setup (Logitech C270, fixed mounting distance, and controlled LED ring lighting) to match the deployment conditions on the conveyor. The dataset contains three classes—bolt, washer, and metal plate—captured at 1280×720 resolution from multiple viewpoints and small variations in placement, rotation, and illumination to improve robustness. All images were manually annotated at the object level using a bounding-box labeling tool (YOLO format), and the dataset was split into training, validation, and test subsets with a 70/20/10 ratio. In addition to the offline test split, an independent online evaluation set of 500 parts was used during conveyor experiments to assess end-to-end system stability and measurement repeatability.

The object detector was trained using the YOLOv5 architecture initialized from pretrained weights to accelerate convergence and improve generalization. Training was performed with an input size of 640×640, batch size 16, and 200 epochs using stochastic gradient descent (SGD) with momentum 0.937 and weight decay 0.0005. The initial learning rate was set to 0.01 with cosine learning-rate scheduling, while standard YOLO augmentations horizontal and vertical flips) were enabled to handle scale and pose variations.

The deep learning vision module is developed using the YOLOv5 algorithm, which performs real-time object detection and localization on the Raspberry Pi 5 [24]. Once an object is detected, its bounding box is extracted, and the height of the object in pixels (H_{obj}) is obtained using the OpenCV minAreaRect() function. To determine the actual size or the distance between the camera and the object, the system applies a mathematical model derived from pinhole camera geometry[25]. This relationship connects the physical dimensions of the image sensor, focal length, and the number of pixels that represent the object.

The camera-to-object distance (D) is calculated using the following formula:

$$D(mm) = \frac{H_{obj} \times H_{sensor}}{f \times H_{real} \times H_{image}} \quad (1)$$

Equation (1) is used to calculate the camera-to-object distance (D) so that the dimension measurement conditions remain consistent in IoT-based vision systems. This formula is based on the pinhole camera model (triangle similarity), namely the principle that the size of an object appearing in an image will change according to its distance from the camera: the same object will appear larger when it is closer and smaller when it is further away. In this study, the system first takes a reference image of an object with a known real height H_{real}, Lens focal length value (f), and physical height of the sensor H_{sensor}, taken from the camera datasheet. From the captured frame, the system measures the height of the object in the image H_{obj}, for example the height of the bounding box in pixels and the total height of the image H_{image}. Ratio $\frac{H_{obj}}{H_{image}}$ represents the proportion of the image height occupied by the object, then this proportion is mapped to the physical size on the sensor using H_{sensor}. By combining optical geometry information (f and H_{sensor}) and the scale of the object in the image, the camera-object distance D can be calculated automatically. This distance is then used to validate camera placement and ensure the stability of the dimensioning process (converting pixels to millimeters). Furthermore, if the distance D is kept constant, the same equation can be rearranged to estimate the actual object height from the object's size in the image.

d. PLC Communication via FINS Protocol

Communication between the Raspberry Pi 5 and the Omron CP2E Programmable Logic Controller (PLC) is established using the Factory Interface Network Service (FINS) protocol over an Ethernet TCP/IP connection . The FINS protocol, developed by Omron Corporation, provides a deterministic and reliable communication method for exchanging real-time data between PLCs and external devices such as industrial PCs or IoT gateways. In this system, FINS serves as the primary medium for bidirectional data exchange between the deep learning vision module and the PLC's control logic. The communication structure is implemented using the Node-RED framework with the node-red-contrib-omron-fins library, allowing direct read/write access to the PLC's Data Memory (DM) registers. Measurement results generated by the YOLOv5 model on the Raspberry Pi—such as object ID, length, width, and classification status (OK/NG)—are converted into numerical data packets and transmitted to predefined memory addresses within the PLC. For example, DM1000 and DM1001 store the measured width and height values in millimeters, while DM1010 holds the inspection result flag (1 for OK, 0 for NG). Once received, the PLC executes corresponding ladder logic routines that determine actuator responses, such as activating pneumatic ejectors or controlling conveyor movement.

The feedback communication from the PLC to the Raspberry Pi ensures closed-loop operation and system synchronization. The PLC continuously updates specific DM registers—such as DM1020 for system status, DM1021 for error codes, and DM1022 for operation mode—allowing the Raspberry Pi to monitor and react to changes in real time. Node-RED functions as the middleware that handles message routing, protocol conversion, and fault recovery mechanisms. Each communication cycle between the Raspberry Pi and PLC is completed within 80–100 milliseconds, enabling deterministic control suitable for real-time industrial inspection processes. Additionally, watchdog mechanisms and retry loops are implemented within Node-RED to ensure reliability in case of temporary network interruptions. The FINS communication flow follows a structured sequence: (1) YOLOv5 detection results are generated on the Raspberry Pi; (2) data are encoded and transmitted to PLC DM registers via FINS; (3) PLC processes and executes control logic; and (4) feedback data are returned to Node-RED for visualization and database logging. This closed-loop integration of AI-based vision, PLC control, and IoT communication establishes a robust cyber-physical feedback system, ensuring precision, repeatability, and full interoperability between intelligent perception and industrial actuation within the smart manufacturing environment.

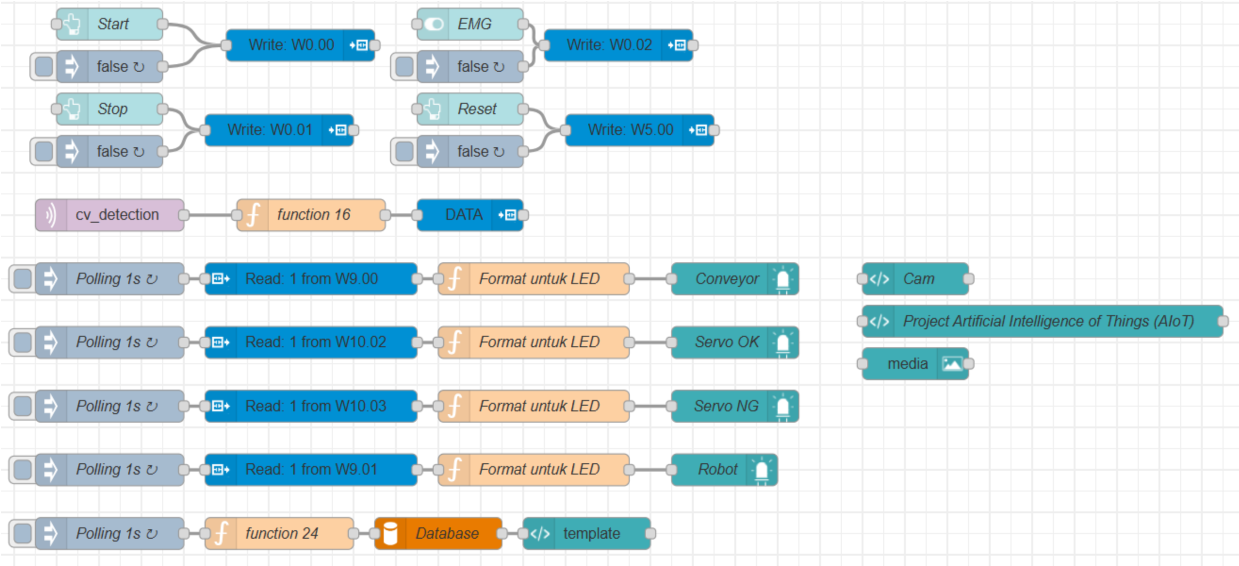

Figure 4. flow node-red communication FINS protocol

e. IoT Dashboard and Data Management

The IoT Dashboard and Data Management subsystem serves as the upper layer of the proposed architecture, providing real-time visualization, system monitoring, and historical data storage. It is implemented using the Node-RED platform, which offers a flow-based programming environment for IoT orchestration and data communication. Node-RED operates on the Raspberry Pi 5 and acts as the middleware that bridges the deep learning vision module, the Omron CP2E PLC, and the MySQL database. The Node-RED Dashboard provides a web-based interface that allows operators and engineers to monitor production status in real time from any device within the same industrial network. The dashboard displays multiple panels, including (1) Live Camera Feed showing YOLOv5 detection results with bounding boxes and measurement overlays, (2) Dimensional Data Panel presenting length and width in millimeters, (3) System Status Indicators reflecting PLC states, communication signals, and inspection results, and (4) Statistical Charts illustrating production counts, defect rates, and process stability [26]. The dashboard uses built-in chart nodes and gauge widgets to visualize data trends, enabling immediate detection of dimensional deviations or quality issues. Additionally, real-time alerts and notifications are triggered when a product measurement exceeds tolerance limits, with status updates color-coded (green for OK, red for NG) to simplify operator interpretation.

3. Results and Discussion

a. Results

The developed IoT-enabled deep learning vision system was experimentally evaluated to assess object detection accuracy, dimensional measurement precision, communication latency, and dashboard performance. The system was deployed in a simulated smart manufacturing environment consisting of a conveyor line, Logitech C270 HD camera, Raspberry Pi 5, and Omron CP2E PLC interconnected through Ethernet using the FINS protocol. The YOLOv5 detector was trained on the annotated dataset described in Section 3.1 and then evaluated online using 500 parts (bolts, washers, and small metal plates) processed on the conveyor to verify end-to-end stability. The deployed model achieved a mean Average Precision (mAP@0.5) of 98.1% with an average detection confidence of 97.9% across object types. The inference speed measured on the Raspberry Pi 5 reached 16–18 frames per second (FPS), and the average end-to-end latency from image capture to visualization on the Node-RED dashboard was approximately 190 ms.

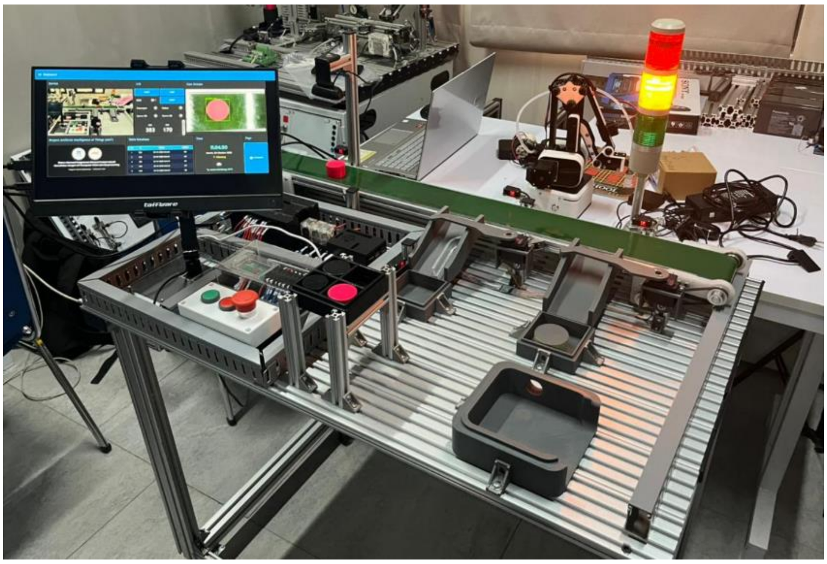

Figure 5. Implementation of product dimension measurement system

Dimensional measurement performance was benchmarked against ground-truth values obtained using precision calipers. The average measurement error was ±0.42 mm, with a standard deviation of 0.18 mm, demonstrating that the system can perform sub-millimeter accuracy consistently. Calibration stability tests were conducted by running the system continuously for 6 hours, and results indicated less than 0.3% drift in pixel-to-millimeter scaling, confirming the robustness of the optical setup and software calibration. Communication performance between the Raspberry Pi and the Omron CP2E PLC was also tested. The average FINS transaction latency was 85 ms, with zero packet loss recorded during 10,000 communication cycles. PLC logic execution, including actuator triggering and feedback response, was verified in real time with deterministic performance. The Node-RED dashboard successfully visualized live measurement data, object classifications (OK/NG), and PLC system states, while the MySQL database logged over 10,000 measurement records without interruption. The overall system uptime during continuous operation exceeded 99.7%, validating its operational stability for industrial conditions.

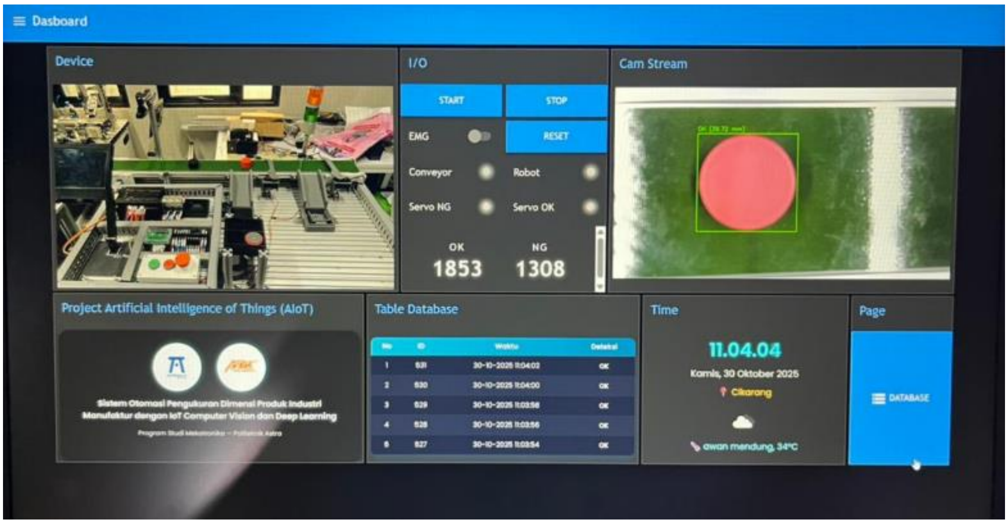

Figure 6. IoT Computer Vision system dashboard

3.2. Discussion

The experimental results confirm that the proposed IoT-enabled deep learning vision system effectively combines AI-based perception, industrial control, and IoT-based data visualization within a compact, low-cost, and reliable framework. The use of YOLOv5 on the Raspberry Pi 5 proved to be a highly efficient solution for edge computing, enabling accurate object detection and dimensional measurement without relying on cloud-based processing. The inference rate of 16–18 FPS ensures that the system can perform real-time inspections compatible with conveyor speeds of up to 0.5 m/s, a performance level sufficient for small-to-medium industrial assembly lines. Compared with traditional manual or semi-automated dimensional inspection methods, the proposed system achieves an 85% reduction in measurement time and eliminates the risk of human error. Furthermore, the measurement error of ±0.42 mm aligns with industrial precision requirements for components with dimensional tolerances within ±1.0 mm, confirming its applicability for quality control in sectors such as automotive parts, electronics housings, and metal fabrication.

From a control and communication perspective, the use of the FINS protocol provides deterministic and robust data exchange between the vision system and PLC, maintaining synchronization and ensuring process reliability. The communication latency of 85 ms demonstrates that even on a low-cost embedded platform, real-time bidirectional communication can be achieved without compromising control response. The Node-RED dashboard further extends the usability of the system by offering an intuitive, web-based human machine interface (HMI), enabling operators to monitor live production data and machine status remotely. The integration with MySQL ensures complete data traceability and process transparency, allowing long-term analysis and statistical process control (SPC). Additionally, the architecture's modular nature allows easy integration with cloud servers, digital twin environments, or predictive maintenance algorithms in future implementations. Overall, the discussion validates that the proposed architecture successfully achieves the core principles of Industry 4.0 connectivity, interoperability, real-time analytics, and automation, providing a foundation for scalable, intelligent manufacturing systems.

The remaining measurement error (±0.42 mm) can be primarily explained by several practical factors inherent to real-time vision metrology. First, pixel quantization at 720p resolution means that a one–to–two pixel shift in the detected edge or bounding box can translate into a measurable deviation in millimeters, especially when the object occupies a limited portion of the image. Second, residual lens distortion and perspective effects may persist despite calibration, particularly when parts deviate from the assumed planar measurement surface or when their pose and height vary relative to the conveyor plane. Third, environmental and mechanical disturbances—including illumination fluctuations, specular reflections from metallic surfaces, and vibration of the camera mount—can perturb boundary localization and reduce segmentation stability. In addition, conveyor motion may introduce motion blur at longer exposure times, leading to slight over- or under-estimation of object boundaries. These sources of error can be reduced through rigid mounting and vibration damping, stable strobing or constant-current lighting to suppress flicker, shorter exposure settings, higher-resolution imaging optics to improve mm-per-pixel granularity, and periodic re-calibration or verification using a reference gauge.

Regarding scalability, the system can be extended to larger objects by increasing the field of view through adjustments in camera height and focal length, while maintaining accuracy via re-calibration and, when necessary, multi-camera configurations to preserve effective mm-per-pixel resolution across a wider inspection area. For higher conveyor speeds, throughput and robustness can be improved by minimizing exposure time to mitigate blur, adopting trigger-based image capture synchronized with part arrival, and accelerating inference using lighter model variants (e.g., YOLOv5n), quantization, or edge hardware acceleration such as NPU/TPU support. Furthermore, deploying parallel inspection stations and distributing Node-RED processing flows enables multi-lane or multi-stage inspection while maintaining deterministic synchronization with PLC actuation.

Despite these strengths, the current implementation has several limitations. It assumes a fixed camera pose and a predominantly planar measurement surface, requiring parts to remain fully within the camera field of view and at a consistent height relative to the conveyor for reliable dimensional estimation. Performance may degrade under severe occlusions, extreme lighting variation, or highly reflective surfaces without dedicated illumination control. Moreover, the present 2D measurement approach cannot capture out-of-plane features or complex 3D geometries, which constrains its applicability for parts with significant depth variation. These limitations motivate future extensions toward 3D vision measurement and digital twin integration to enable richer geometric understanding and more adaptive, context-aware inspection in dynamic production environments.

4. Conclusion

This research presents the design and implementation of an IoT-enabled deep learning vision system for automated dimensional measurement in smart manufacturing environments. The proposed system integrates YOLOv5 based computer vision, edge computing using Raspberry Pi 5, industrial control via Omron CP2E PLC, and IoT-based monitoring through Node-RED and MySQL. Experimental results demonstrate that the system achieves 98.1% detection accuracy, ±0.42 mm measurement precision, and real-time inference performance of up to 18 FPS, confirming its suitability for in-line inspection and automated quality control applications. The communication between the vision module and PLC using the FINS protocol provides deterministic and reliable control with an average latency of 85 milliseconds, ensuring seamless synchronization between AI-based perception and actuation. Furthermore, the Node-RED dashboard enables intuitive real-time visualization and comprehensive traceability through data logging in MySQL, providing valuable insights for production monitoring and process optimization.

The successful integration of deep learning, IoT, and industrial automation technologies in this research demonstrates the feasibility of developing low-cost, high-accuracy, and fully autonomous measurement systems that align with Industry 4.0 principles. The modular and interoperable design allows easy scalability to multi-station or multi-camera setups, as well as future integration with cloud-based analytics, digital twin models, and predictive maintenance systems. Future work will focus on extending the system toward 3D vision measurement using stereo or structured-light cameras, implementing real-time adaptive calibration, and incorporating reinforcement learning-based control optimization to enhance inspection precision and process intelligence further. Overall, this study contributes a practical and scalable solution that bridges the gap between AI-driven perception and industrial-grade control systems, supporting the transition toward autonomous, data-driven smart manufacturing ecosystems.

5. Acknowledgement

This paper represents one of the outcomes of a research project funded by the Directorate of Research and Community Service, Ministry of Education, Science, and Technology (Kemdiktisaintek). The authors express their sincere gratitude for the trust and support provided, which enabled the successful completion of this study. It is hoped that the results of this research will provide meaningful contributions and benefits to the wider community.

References

[1] L. Palazzetti, D. Giannetti, A. Verolino, D. A. Grasso, C. M. Pinotti, and F. B. Sorbelli, "AntPi: A Raspberry Pi based edge-cloud system for real-time ant species detection using YOLO," 2025, doi: 10.5281/zenodo.1674045.

[2] Y. R, S. K. D, S. A. Bhalerao, K. Murugesan, S. Vellaiyan, and N. Van Minh, "Real-time fire detection and suppression system using YOLO11n and Raspberry Pi for thermal safety applications," *Case Studies in Thermal Engineering*, vol. 75, p. 107159, Nov. 2025, doi: 10.1016/j.csite.2025.107159.

[3] W. Nugroho, R. R. Isnanto, and A. F. Rochim, "Comparison of Mycobacterium Tuberculosis Image Detection Accuracy Using CNN and Combination CNN-KNN," *Jurnal RESTI*, vol. 7, no. 1, pp. 80–86, Feb. 2023, doi: 10.29207/resti.v7i1.4626.

[4] J. Qiu, W. Zhang, S. Xu, and H. Zhou, "DP-YOLO: A lightweight traffic sign detection model for small object detection," *Digital Signal Processing: A Review Journal*, vol. 165, Oct. 2025, doi: 10.1016/j.dsp.2025.105311.

[5] S. E. Mathe, H. K. Kondaveeti, S. Vappangi, S. D. Vanambathina, and N. K. Kumaravelu, “A comprehensive review on applications of Raspberry Pi,” May 01, 2024, *Elsevier Ireland Ltd*. doi: 10.1016/j.cosrev.2024.100636.

[6] S. G. Koustas, S. J. Oks, and K. M. Möslein, “Developing industrial smart product-service systems: Opportunities and challenges for manufacturing firms,” in *Procedia CIRP*, Elsevier B.V., 2025, pp. 1008–1013. doi: 10.1016/j.procir.2025.08.171.

[7] H. Kabir, J. Wu, S. Dahal, T. Joo, and N. Garg, “Automated estimation of cementitious sorptivity via computer vision,” *Nature Communications* , vol. 15, no. 1, Dec. 2024, doi: 10.1038/s41467-024-53993-w.

[8] H. Zia *et al.*, “Gesture-controlled omnidirectional autonomous vehicle: A web-based approach for gesture recognition,” *Array*, vol. 26, Jul. 2025, doi: 10.1016/j.array.2025.100408.

[9] B. Mills, M. N. Zervas, and J. A. Grant-Jacob, “Imaging pollen using a Raspberry Pi and LED with deep learning,” *Science of the Total Environment*, vol. 955, Dec. 2024, doi: 10.1016/j.scitotenv.2024.177084.

[10] M. Raisul Islam *et al.*, “Deep Learning and Computer Vision Techniques for Enhanced Quality Control in Manufacturing Processes,” *IEEE Access*, vol. 12, pp. 121449–121479, 2024, doi: 10.1109/ACCESS.2024.3453664.

[11] Á. Kálnai *et al.*, “Real-time component-based particle size measurement and dissolution prediction during continuous powder feeding using machine vision and artificial intelligence-based object detection,” *European Journal of Pharmaceutical Sciences*, vol. 209, Jun. 2025, doi: 10.1016/j.ejps.2025.107080.

[12] V. Mahore, P. Soni, P. Patidar, H. Nagar, A. Chouriya, and R. Machavaram, “Development and implementation of a raspberry Pi-based IoT system for real-time performance monitoring of an instrumented tractor,” *Smart Agricultural Technology*, vol. 9, Dec. 2024, doi: 10.1016/j.atech.2024.100530.

[13] W. Nugroho, Rifdah Zahabiyah, Afianto, and Mada Jimmy Fonda Arifianto, “Application of Deep Learning YOLO in IoT System for Personal Protective Equipment Detection,” *Jurnal E-Komtek (Elektro-Komputer-Teknik)*, vol. 8, no. 2, pp. 428–437, Dec. 2024, doi: 10.37339/e-komtek.v8i2.2187.

[14] W. Nugroho, R. Zahabiyah, M. J. F. Arifiant, and A. Afianto, “Automated Component Detection for Quality PCB Using YOLO Algorithm with IoT Real-Time Streaming on Raspberry Pi,” *JURNAL INFOTEL*, vol. 17, no. 2, Jul. 2025, doi: 10.20895/infotel.v17i2.1313.

[15] K. Tan, J. Wu, H. Zhou, Y. Wang, and J. Chen, “Integrating Advanced Computer Vision and AI Algorithms for Autonomous Driving Systems,” *www.centuryscipub.com*, vol. 4, p. 2024, doi: 10.53469/jtpes.2024.04(01).06.

[16] K. Rzepka, P. Szary, K. Cabaj, and W. Mazurczyk, “Performance evaluation of Raspberry Pi 4 and STM32 Nucleo boards for security-related operations in IoT environments,” *Computer Networks*, vol. 242, Apr. 2024, doi: 10.1016/j.comnet.2024.110252.

[17] S. Liawatimena and N. Isworo, “Annotated drowsiness detection dataset captured using Raspberry Pi 5,” *Data Brief*, vol. 63, p. 112211, Dec. 2025, doi: 10.1016/j.dib.2025.112211.

[18] T. Ederer and I. Ivkić, “Implementing video monitoring capabilities by using hardware-based encoders of the Raspberry Pi Zero 2 W,” *SoftwareX*, vol. 31, Sep. 2025, doi: 10.1016/j.softx.2025.102274.

[19] M. D. Rakesh, M. Jeevankumar, and S. B. Rudraswamy, “Implementation of real time root crop leaf classification using CNN on raspberry-Pi microprocessor,” *Smart Agricultural Technology*, vol. 10, Mar. 2025, doi: 10.1016/j.atech.2024.100714.

[20] P. Rajkumar, “Humidity and temperature monitoring using Raspberry Pi via RS232 networking,” *Array*, vol. 27, Sep. 2025, doi: 10.1016/j.array.2025.100464.

[21] X. Zhao, L. Wang, Y. Zhang, X. Han, M. Deveci, and M. Parmar, “A review of convolutional neural networks in computer vision,” *Artif Intell Rev*, vol. 57, no. 4, Apr. 2024, doi: 10.1007/s10462-024-10721-6.

[22] C. Lin, J. Li, and H. Hao, “Deep learning-based motion magnification and frames matching for structural displacement measurement using computer vision,” *Eng Struct*, vol. 346, p. 121647, Jan. 2026, doi: 10.1016/j.engstruct.2025.121647.

[23] T. D. Phuc and B. C. Son, “Development of an autonomous chess robot system using computer vision and deep learning,” *Results in Engineering*, vol. 25, Mar. 2025, doi: 10.1016/j.rineng.2025.104091.

[24] M. T. Okano, W. A. Celestino Lopes, and S. M. Ruggero, “Automated dimensional inspection of automotive components using computer vision through YOLO,” *Procedia Comput Sci*, vol. 270, pp. 3133–3141, 2025, doi: 10.1016/j.procs.2025.09.438.

[25] V. Lepetit, “Object size measurement and camera distance evaluation for electronic components using Fixed-Position camera,” *Computer Vision Studies*, pp. 13–16, 2023, doi: 10.58396/cvs020101.

[26] Y. Zhang, Y. Xu, T. Xu, C. Wang, C. Li, and H. Wang, “RSD-YOLO: An improved YOLOv7-tiny framework for oat disease severity identification with integration of ReXNet and decoupled head,” *Smart Agricultural Technology*, vol. 12, Dec. 2025, doi: 10.1016/j.atech.2025.101433.