WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal Homepage: www.ijrpr.com ISSN 2582-7421

PCB Fault Detection System Using Artificial Intelligence

Dr. S. P. Washimkar, Yash Motharkar, Sakshi Suryawanshi, Kundan Wadibhasme, Prajwal Agashe, Chetan Kanoje

Dept. of Electronics & Telecom. Engg. Priyadarshini College of Engineering, Nagpur, India
spwashimkar@gmail.com, yashmotharkar02@gmail.com, sakshisurya2005@gmail.com, kundanwadibhasme@gmail.com, prajwalagashe32@gmail.com, ckanoje12@gmail.com

DOI : https://doi.org/10.55248/gengpi.07.1026.34033

ABSTRACT—

Printed Circuit Boards (PCBs) are essential components of electronic systems, and the identification of manufacturing and assembly defects is important for maintaining product quality and reliability. Conventional manual inspection is time-consuming and can be affected by human error when many boards must be inspected repeatedly. This paper presents a PCB fault detection system using artificial intelligence that integrates a Raspberry Pi 5, an 8 MP USB webcam, OpenCV, and an AI-based detection model for automated inspection. The captured PCB image is preprocessed with OpenCV and analysed using YOLOv8 and based models to identify defects such as broken tracks, scratches, solder bridges, missing or misplaced components, component misalignment, short circuits, surface damage, burn marks, and corrosion. Defective regions are highlighted, and a PASS/FAIL decision is presented through a web dashboard, a 16×2 I²C LCD, green and red LEDs, and a buzzer, while inspection data and images are stored in a database for traceability. A custom interface PCB integrates power distribution, the controller, and indicator circuits, and the prototype was verified through design-rule checking, continuity, power, interface, and integrated system tests. The system is intended to reduce manual inspection effort and support consistent PCB quality monitoring in assembly lines, quality-control laboratories, and consumer, automotive, and medical electronics.

Keywords— *Printed Circuit Board (PCB), Artificial Intelligence, PCB Fault Detection, Computer Vision, OpenCV, Raspberry Pi 5, YOLOv8, Convolutional Neural Network, Image Processing, Automated Inspection.*

I. INTRODUCTION

Printed Circuit Boards (PCBs) are fundamental components of modern electronic products because they provide the physical structure and electrical interconnections required for components to operate as an integrated system. PCBs are used in consumer electronics, industrial equipment, automotive systems, and communication devices. As electronic systems become smaller and more densely populated, the dimensions of tracks, pads, solder joints, and components have also reduced, making reliable visual inspection increasingly important [1], [2].

PCB manufacturing and assembly can introduce defects such as open circuits, short circuits, missing components, incorrect components, soldering-related defects, and scratches. These defects can affect the electrical performance and reliability of the final product, so inspection at an early stage is important for identifying defective boards before they proceed to later stages [3], [4]. Image-processing and intelligent inspection techniques provide an approach for automating the identification of visible PCB defects [5], [6].

Traditional inspection has relied on visual examination and rule-based or image-processing techniques. Manual examination requires human involvement and becomes time-consuming when many boards must be inspected, while variations in illumination, board positioning, defect appearance, and operator judgment make consistent inspection difficult [2], [7]. Automated Optical Inspection (AOI) systems were therefore introduced for image-based inspection of electronic assemblies, and research has explored integrating machine learning and deep learning with optical inspection [2], [8].

Deep learning, enables computer-vision systems to learn the visual characteristics of defects directly from training images [4], [8]. Object-detection approaches additionally provide the location of a defect within the image, which is useful for highlighting defective regions during inspection.

This paper presents a PCB fault detection system that combines image acquisition, image processing, artificial intelligence, and embedded computing. A Raspberry Pi 5 is the primary processing unit and an 8 MP USB webcam acquires the PCB image. The image is processed with OpenCV and analysed by YOLOv8 and the system generates a PASS/FAIL decision. A 16×2

LCD provides local status, a green LED indicates an acceptable PCB, a red LED indicates a detected fault, and a buzzer driven through an NPN transistor gives an audible alert. The result is also available on a web dashboard, and captured images and inspection information are stored in a database. Similar research highlights challenges related to computational resources, dataset quality, defect variability, and real-time processing [4], [8], [9].

The remainder of this paper is organized as follows. Section II states the problem and objectives, Section III reviews related work, Section IV describes the proposed architecture, Section V explains the methodology, Section VI presents the implementation, Section VII discusses verification, and Sections VIII–X cover applications, limitations, and conclusions.

II. PROBLEM STATEMENT AND OBJECTIVES

A. Problem Statement

The increasing complexity and miniaturization of electronic circuits make PCB inspection an important part of electronics manufacturing and quality control. A practical automated system must address the following challenges:

- Manual inspection effort: Repeated visual examination of many boards demands significant human effort [2], [7].
- Human-dependent variation: Fatigue, attention, experience, and subjective judgment influence manual results, whereas deep-learning models can learn defect patterns from labeled data [4], [8].
- Small and localized defects: Defects occur in localized regions of tracks, solder joints, pads, and component areas; YOLO-based detectors combine localization with classification [8], [9].
- Inspection speed: The time to capture, process, and evaluate a PCB image matters, and optimized inference on constrained platforms has been studied [9].
- Lack of traceability: A PASS/FAIL decision alone is insufficient; images, defect categories, and confidence values should be retained for review.
- Need for a compact solution: Industrial systems can require specialized hardware, while efficient models for edge and embedded platforms are of growing interest [9], [10].
- Multiple defect types: Published datasets consider open circuits, short circuits, missing components, mouse bites, and spurious copper, so a system must identify more than one defect type [3], [4].

B. Primary Objectives

The primary objective is to design and develop an AI-based automated PCB fault detection system using a Raspberry Pi, camera-based image acquisition, OpenCV, and deep-learning techniques. The specific objectives are:

- Image acquisition: Automate PCB image capture using an 8 MP USB webcam connected to the Raspberry Pi 5.
- Preprocessing: Prepare captured images for AI-based analysis using OpenCV.
- AI-based detection: Detect and classify visible faults such as broken tracks, scratches, solder bridges, missing or incorrectly placed components, and component misalignment using YOLOv8 approaches.
- Defect localization: Identify and highlight defective regions so the user can locate the fault visually.
- Automatic decision: Generate a PASS/FAIL decision from the output of the AI-based inspection.
- Local indication: Provide status through a 16×2 IZC LCD, green LED, red LED, and buzzer.
- Dashboard and records: Display PCB status, defect type, confidence score, defect image, and inspection history on a web dashboard, and store data and images in a database.
- Compact prototype and evaluation: Build a compact embedded prototype around the Raspberry Pi and evaluate it using accuracy, precision, recall, confidence, and processing time.

III. RELATED WORK

Prior work on PCB inspection guides the design of the proposed system in terms of inspection methodology, learning-based detection, and embedded deployment.

A. Image-Processing-Based and Reference-Based Inspection

The survey by Ling and Isa reviewed more than 100 PCB defect-detection studies and compared the major approaches [1]. Early automated methods compare the PCB under inspection with a defect-free reference image, using image subtraction and related techniques to reveal differences. Such methods work well under controlled conditions but are affected by image displacement, rotation, illumination variation, and misalignment, so image registration is an important step [1]. Typical operations include filtering, thresholding, edge detection, morphological processing, and contour extraction. These are computationally efficient and suit embedded systems, but they need carefully selected parameters and controlled imaging conditions.

B. Machine-Learning and Deep-Learning Approaches

Machine-learning methods learn the relationship between extracted image features and defect classes, but their performance depends strongly on feature quality. Huang and Wei introduced a PCB dataset of 1,386 images with six defect categories and a reference-based approach followed [11]. Kim et al. applied a skip-connected convolutional autoencoder to six defect types: missing hole, mouse bite, open circuit, short circuit, spur, and spurious copper [12]. Transfer learning has been used for automatic optical inspection of defective PCBs [7], and Chen et al. reviewed deep-learning-based PCB defect detection [13]. Deep learning offers automatic feature learning and support for multiple defect classes, but it needs suitable datasets, computing resources, and validation.

C. YOLO-Based Detection on Embedded Platforms

Object detection identifies the approximate location of a defect in addition to its class. A YOLOv4-based framework was developed for six PCB defect types and optimized with the Intel OpenVINO toolkit for a low-power embedded platform [9], and PCBDet addresses PCB component detection on the edge [10]. A 2025 study proposed an improved YOLOv8-based model with a lightweight backbone and modified loss formulation, reporting better detection performance than its baseline under its experimental conditions [14]. This motivates the use of YOLOv8 on a Raspberry Pi 5 in the present work.

D. Research Gap

Table I compares the existing approaches. The literature shows that no single approach suits every manufacturing environment: reference-based methods are effective in controlled imaging, deep learning handles complex patterns, and embedded deployment adds a computational-efficiency requirement.

TABLE I. COMPARISON OF EXISTING PCB INSPECTION APPROACHES

Approach	Advantages	Limitations
Manual inspection	Flexible; no model training	Time-consuming; human-dependent
Reference-based image processing	Simple concept; low computation	Sensitive to alignment and illumination
Traditional machine learning	Lower computational requirement	Needs effective feature engineering
Autoencoder-based detection	Image-based anomaly detection	Depends on training and reconstruction quality
YOLO-based detection	Defect location and class with fast inference	Needs annotated data and model optimization
Embedded AI inspection	Local processing; less external dependence	Hardware limits model complexity

Many studies focus on the detection algorithm rather than the complete inspection workflow. Gaps remain in integration of acquisition, preprocessing, inference, visualization, hardware indication, and data storage; in inference on a Raspberry Pi-class platform alongside the other system functions; in inspection traceability; in clear operator feedback; and in real-world imaging conditions and defect diversity, since public datasets are limited [11]. The proposed system addresses these gaps in a single platform.

IV. PROPOSED SYSTEM ARCHITECTURE

The system follows an input–processing–output architecture built around the Raspberry Pi 5. Fig. 1 shows the block diagram. The input section comprises an 8 MP USB webcam, uniform white LED lighting, and the PCB under inspection. A 5 V DC adapter (3 A) powers the system. The central processing section performs OpenCV preprocessing, AI-based inference, and fault detection and classification. The output section consists of a 16×2 LCD, green LED, red LED, and buzzer, while a web dashboard, database, and a 32 GB microSD card (Raspberry Pi OS and project code) provide monitoring and storage over a Wi-Fi link.

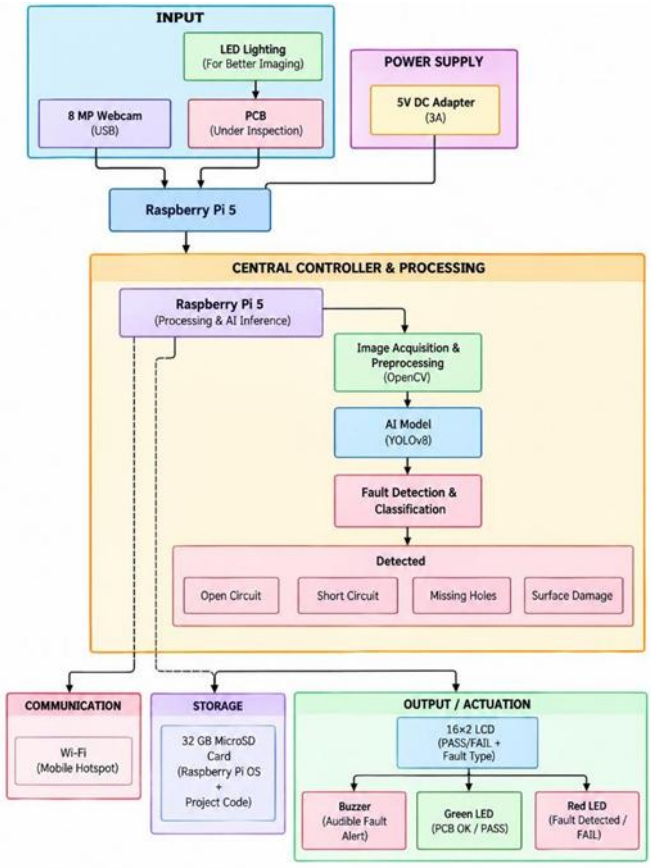


Fig. 1. Block diagram of the proposed PCB fault detection system using AI.

TABLE II. HARDWARE AND SOFTWARE COMPONENTS

Subsystem	Components
Image acquisition	8 MP USB webcam, uniform white LED lighting
Processing platform	Raspberry Pi 5, Python
Image processing	OpenCV
AI model	YOLOv8
Output indication	16×2 IZC LCD (PCF8574), green LED, red LED, buzzer with NPN driver
Power supply	230 V AC to 5 V DC (3 A) adapter
Storage and communication	32 GB microSD card, database, Wi-Fi
Monitoring	Web dashboard

A. Image Acquisition and Preprocessing

The PCB is positioned in front of a fixed 8 MP USB webcam connected to the Raspberry Pi 5, and uniform white LED lighting provides a controlled inspection environment. OpenCV then prepares the captured image for AI-based analysis. Conventional image processing and AI are not treated as competing components: preprocessing serves as the input-preparation stage for the AI model [1].

B. AI-Based Fault Detection and Classification

The AI model, incorporating YOLOv8 examines the preprocessed image with respect to the reference PCB. It identifies broken tracks, scratches, solder bridges, missing or incorrectly placed components, component misalignment, soldering defects, short circuits, surface damage, burn marks, and

corrosion. Because object detection returns both class and location, the defective region is highlighted in the output image, and a confidence score is reported for each detection.

C. Decision, Indication, and Monitoring

After analysis, the system generates a PASS/FAIL decision. The 16×2 LCD displays local status and the fault type, the green LED indicates an acceptable PCB, the red LED indicates a detected fault, and a buzzer driven through an NPN transistor sounds an audible alert. The web dashboard shows PCB status, detected defect type, confidence score, defect image, and inspection history, and the database stores inspection records and images for traceability.

D. Interface PCB and Hardware Design

A custom PCB replaces temporary jumper-wire connections with a structured circuit that provides power and ground distribution, signal routing, GPIO connections, peripheral connectors, and interface circuitry. Fig. 2 shows the connections between the Raspberry Pi 5, the IZC LCD, the indicator LEDs, and the buzzer, and Table III lists the pin assignment. Design followed a schematic-to-layout approach covering component and footprint selection, placement, track width and clearance, and routing of power, ground, and signal paths.

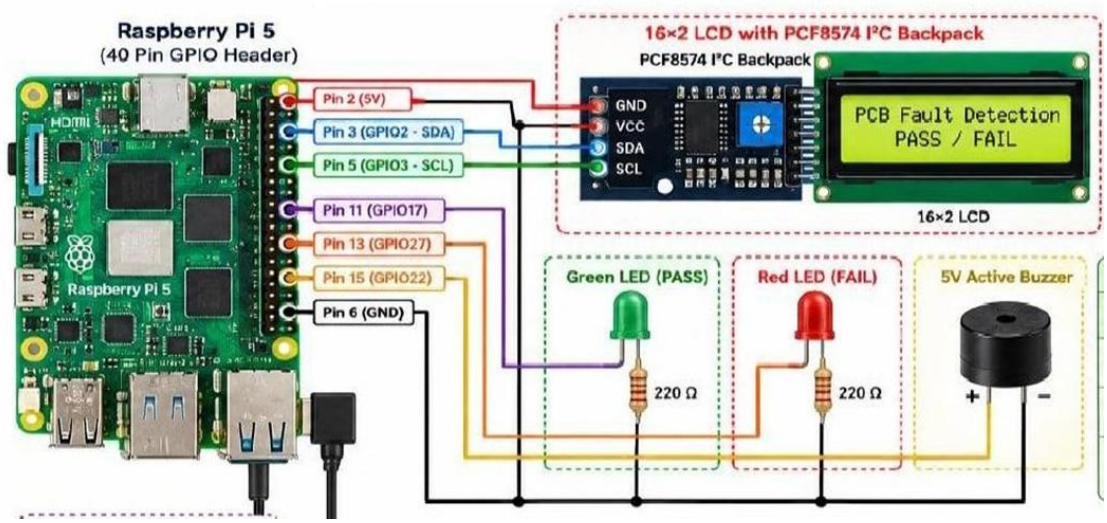


Fig. 2. Connection diagram of the Raspberry Pi 5 with the 16×2 I²C LCD (PCF8574), green and red LEDs, and the buzzer.

TABLE III. RASPBERRY PI 5 PIN ASSIGNMENT

Device	Pin	GPIO	Remark
LCD VCC	2	5 V	IZC backpack (PCF8574)
LCD GND	6	GND	Common ground
LCD SDA	3	GPIO2	IZC data
LCD SCL	5	GPIO3	IZC clock
Green LED	11	GPIO17	PASS; 220 fi to GND
Red LED	13	GPIO27	FAIL; 220 fi to GND
Buzzer	15	GPIO22	5 V active; NPN driver

A. Evaluation Parameters

The AI stage is evaluated using detection accuracy, precision, recall, confidence, and processing time. With true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) counted over the inspected boards, the metrics are:

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (1)$$

$$Precision = TP / (TP + FP) \quad (2)$$

$$Recall = TP / (TP + FN) \quad (3)$$

The total time to inspect one board is the sum of the capture, preprocessing, inference, and output stages:

$$T_{total} = t_{cap} + t_{pre} + t_{inf} + t_{out} \quad (4)$$

V. SYSTEM METHODOLOGY AND WORKFLOW

The inspection workflow, shown in Fig. 3, consists of image

acquisition, preprocessing, AI-based fault detection and classification, result generation, and storage of inspection records. The steps are as follows:

- **Step 1: PCB Positioning:** The PCB under inspection is placed in front of the camera under LED lighting.
- **Step 2: Image Acquisition:** The Raspberry Pi 5 captures the PCB image through the USB webcam.
- **Step 3: Preprocessing:** OpenCV prepares the image for AI-based analysis.
- **Step 4: AI Inference:** YOLOv8 classify faults with respect to the reference PCB.
- **Step 5: Result Generation:** A PASS/FAIL decision is produced and defective regions are highlighted.
- **Step 6: Output Indication:** The LCD, green or red LED, and buzzer present the result locally.
- **Step 7: Record Storage and Display:** The image, defect type, confidence, and result are stored in the database and shown on the dashboard.

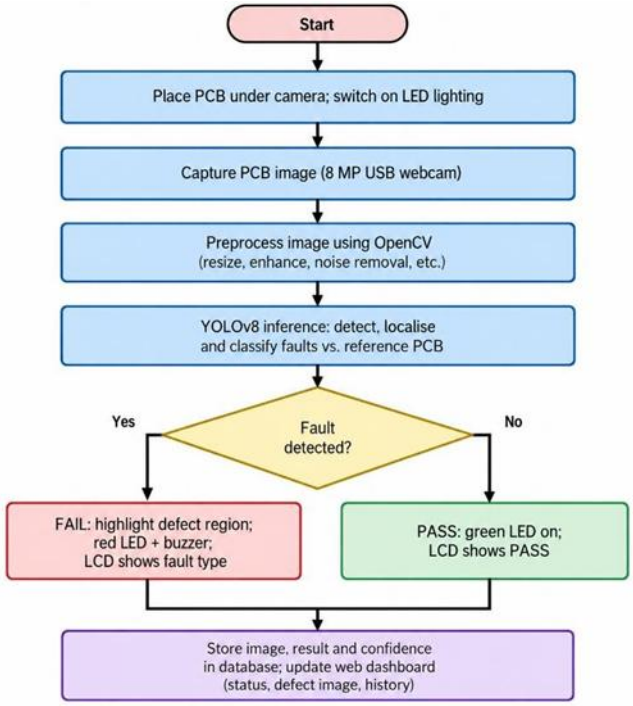


Fig. 3. Inspection workflow of the proposed system.

At the controller level, the software initializes the Raspberry Pi, GPIO, and connected devices, loads the control parameters, and then repeatedly reads the input, processes it, evaluates the required condition, generates and activates the output, and monitors it before returning to the monitoring stage.

VI. IMPLEMENTATION DETAILS

A. Hardware Development and Power Supply

The hardware is divided into power, processing, input, output, and interface sections so each can be tested independently before integration. A 230 V AC supply is converted to 5 V DC by an adapter, and the DC output is checked for voltage and polarity before the Raspberry Pi is connected. The PCB distributes power and a common ground to the controller and peripheral circuits.

B. PCB Design and Fabrication

The schematic covers the power input, ground, Raspberry Pi GPIO lines, input and output interfaces, driver and protection components, and footprints that were verified before layout. Components were placed by functional grouping with attention to connector accessibility, short signal paths, power

distribution, ease of soldering and testing, and heat-producing parts. Power traces were given widths suited to their expected current. After routing, design-rule checks covered unconnected pins, clearance, shorts, track width, component spacing, board-edge clearance, and footprint consistency. Fabrication files were then generated, and the fabricated

board was inspected for dimensions, tracks, pads, holes, connectors, solder mask, and visible defects.

C. Component Assembly

Components were mounted according to the layout, with orientation checked for polarized parts, connectors, and LEDs before soldering. The soldered board was inspected for cold joints, solder bridges, excess or missing solder, loose components, and damaged pads, followed by continuity and short-circuit checks.

D. Raspberry Pi Software

The Raspberry Pi 5 was configured with its operating environment and a Python environment. GPIO pins were set as input, output, or communication lines according to their functions, and each assignment was checked against the schematic. The GPIO assignment is kept consistent between the schematic, PCB, and software so that no mismatch causes incorrect input detection or output activation.

VII. EXPERIMENTAL VERIFICATION AND DISCUSSION

The prototype was verified through staged tests before complete operation: power testing (supply voltage, polarity, ground continuity, and power distribution), PCB continuity testing including power–ground short checks, input and output testing of each interface individually, and integrated testing of the complete system. Testing the sections separately isolates faults and reduces the risk of damage to the controller or peripherals. Table IV summarizes the verification.

A. Implementation Results

The PCB was designed according to the system schematic, and the fabricated board provided the required electrical interconnections. The Raspberry Pi 5 was integrated with the PCB, the power section delivered the required DC supply, input interfaces could be connected to the controller, output interfaces responded to the programmed control signals, and the software executed the defined control sequence.

B. Implementation Challenges

Several practical aspects required attention: organizing the many peripheral connections on the custom PCB to reduce loose wiring; component placement that balanced connector accessibility, routing space, and ease of assembly; adequate current capacity and routing for power distribution; checking the schematic and layout before fabrication; and matching the software GPIO assignments exactly to the physical PCB connections.

C. Scope of the Reported Evaluation

The results reported here are implementation-level verification of the hardware and software. Detection accuracy, precision, recall, confidence, and processing time are defined in Section IV-E as the evaluation parameters for the AI stage, and a dataset-based benchmark of the YOLOv8 on the Raspberry Pi 5 is part of the continuing work.

VIII. PRACTICAL APPLICATIONS AND IMPACT

- Electronics manufacturing: Provides an additional inspection step on assembly lines to identify visible defects before further processing.
- Quality-control laboratories: Gives a consistent procedure with stored images and inspection history for review.
- Consumer, automotive, and medical electronics: Supports PCB quality monitoring where board reliability is critical.
- Education and training: Serves as a practical platform for embedded systems, Raspberry Pi programming, PCB design, GPIO interfacing, and student projects.
- Rapid prototyping: The modular PCB can be adapted for other Raspberry Pi-based systems by changing interfaces and software.

IX. LIMITATIONS AND FUTURE WORK

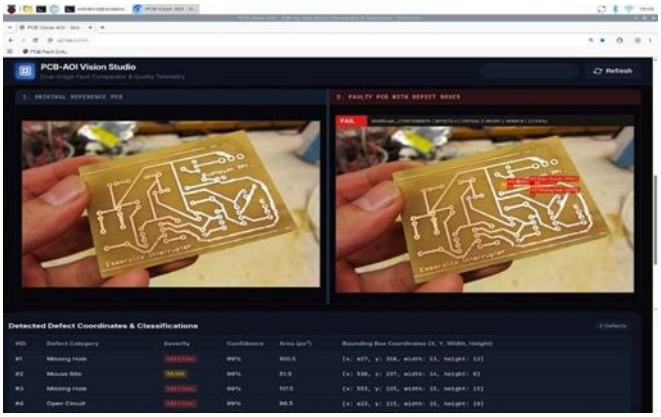
The system relies on a fixed camera and controlled lighting, and the effectiveness of the AI model depends on the quality and diversity of the training and testing images [11]. The Raspberry Pi has limited computing resources, so the model must balance detection performance and computational cost [9]. Quantitative benchmarking of the AI stage remains to be completed.

Future work includes PCB miniaturization using surface-mount components and multilayer boards; improved power

management with regulation, current monitoring, and protection circuits; wireless connectivity such as Wi-Fi, Bluetooth, or MQTT; cloud-based storage for historical analysis; a mobile application with notifications; automated self-diagnostics of power, GPIO, and interfaces; and comparison of captured images with a reference design for advanced inspection. Improved YOLO-based detectors can also be explored for real-time inspection [14].

X. RESULT

The proposed PCB-AOI Vision Studio was evaluated by comparing the original reference PCB image with the defective PCB image. The system successfully detected four defects with a confidence level of 99%. The identified defects included two missing holes, one mouse bite, and one open circuit. Among these, three defects were classified as critical and one as major based on their severity. The results demonstrate the system's potential for automated PCB defect detection, localization, and severity classification, supporting efficient quality inspection in PCB manufacturing



XI. CONCLUSION

This paper presented a PCB fault detection system using artificial intelligence that integrates a Raspberry Pi 5, an 8 MP USB webcam, OpenCV, and YOLOv8 with a PASS/FAIL decision, LCD, LED, and buzzer indication, a web dashboard, and database storage. A custom interface PCB organizes power distribution and signal connections, and the prototype was verified through design-rule checking, continuity, power, interface, and integrated tests. The system provides a compact, automated, and traceable alternative to repetitive manual visual inspection, and it forms a foundation for further extension toward benchmarked, connected, and more intelligent PCB quality monitoring.

REFERENCES

[1] Q. Ling and N. A. M. Isa, “Printed Circuit Board Defect Detection Methods Based on Image Processing, Machine Learning and Deep Learning: A Survey,” IEEE Access, vol. 11, pp. 15921–15944, 2023, doi: 10.1109/ACCESS.2023.3245093.

[2] A. A. R. M. Abu Ebayyeh and A. Mousavi, “A Review and Analysis of Automatic Optical Inspection and Quality Monitoring Methods in Electronics Industry,” IEEE Access, vol.8,pp.183192–183271,2020.

[3] W. Huang and P. Wei, “A PCB Dataset for Defects Detection and Classification,” arXiv:1901.08204, 2019, doi: 10.48550/arXiv.1901.08204.

[4] X. Wu, G. Wang, Z. Zhang, and Z. Zhang, “PCB Defect Detection Using Deep Learning Methods,” in Proc. IEEE 24th Int. Conf. Computer Supported Cooperative Work in Design (CSCWD), 2021, doi: 10.1109/CSCWD49262.2021.9437846.

[5] K. Anoop, N. Sarath, and V. Kumar, “A Review of PCB Defect Detection Using Image Processing,” Int. J. Engineering Innovation and Technology, vol. 4, pp. 188–192, 2015.

[6] L. Gaber, A. I. Hussein, and M. Moness, “Fault Detection Based on Deep Learning for Digital VLSI Circuits,” Procedia Computer Science, vol. 194, pp. 122–131, 2021.

A. L. H. de Souza Silva, G. O. Azevedo, B. J. T. Fernandes, B. L.A. Bezerra, E. B. Lima, and S. C. Oliveira, “Automatic Optical Inspection for Defective PCB Detection Using Transfer Learning,” in Proc. IEEE Latin American Conf. Computational Intelligence (LA-CCI), 2019, doi: 10.1109/LA-CCI47412.2019.9037036.

[7] V. Bardia and M. Prabu, “Printed Circuit Board Defect in Proc. 4th Int. Conf. Technological Advancements in Computational Sciences (ICTACS), 2024, doi: 10.1109/ICTACS62700.2024.10840548.

[8] M.-L. Tham, B. Y. Chong, Y. H. Tan, Y. K. Wong, S. L. Chean, and W. K. Tan, “Optimizing Deep Learning Inference to Detect PCB Soldering Defects,” in Proc. IEEE Int. Conf. Artificial Intelligence in Engineering and Technology (IICAIET), 2022, doi: 10.1109/IICAIET55139.2022.9936776.

[9] B. Li, S. Palayew, F. Li, S. Abbasi, S. Nair, and A. Wong, "PCBDet: An Efficient Deep Neural Network Object Detection Architecture for Automatic PCB Component Detection on the Edge," arXiv:2301.09268, 2023.

[10] W. Huang and P. Wei, "HRIPCB: A Challenging Dataset for PCB Defects Detection and Classification," The Journal of Engineering, vol. 2020, no. 13, pp. 303–309, 2020, doi: 10.1049/joe.2019.1183.

[11] J. Kim, J. Ko, H. Choi, and H. Kim, "Printed Circuit Board Defect Detection Using Deep Learning via a Skip-Connected Convolutional Autoencoder," Sensors, vol. 21, no. 15, p. 4968, 2021, doi: 10.3390/s21154968.

[12] X. Chen, Y. Wu, X. He, and W. Ming, "A Comprehensive Review of Deep Learning-Based PCB Defect Detection," IEEE Access, vol. 11, pp. 139017–139038, 2023, doi: 10.1109/ACCESS.2023.3339561.

[13] Improved YOLOv8-based PCB defect detection, IEEE conferencepaper,2025,doi:10.1109/ICSI64877.2025.110092 86.