Results in Engineering 25 (2025) 104067

Contents lists available at ScienceDirect

Results in Engineering

journal homepage: www.sciencedirect.com/journal/results-in-engineering

Research paper

Deep learning-enhanced defects detection for printed circuit boards

Van-Truong Nguyen [a], Xuan-Thuc Kieu [b,*], Duc-Tuan Chu [a], Xiem HoangVan [c], Phan Xuan Tan [d,**], Tuyen Ngoc Le [e,f]

[a] Faculty of Mechatronics, SMAE, Hanoi University of Industry, Hanoi, 11900, Viet Nam
[b] Faculty of Electronic Engineering, Hanoi University of Industry, Hanoi, 11900, Viet Nam
[c] Faculty of Electronics and Telecommunications, VNU-University of Engineering and Technology, Vietnam National University, Hanoi, 10000, Viet Nam
[d] College of Engineering, Shibaura Institute of Technology, Tokyo, 135-8548, Japan
[e] Department of Electronic Engineering, Ming Chi University of Technology, Taipei 24301, Taiwan
[f] Center for Reliability Engineering, Ming Chi University of Technology, Taipei 24301, Taiwan

ARTICLE INFO

Keywords:
Printed circuit board
U-NET
Defect detection
Scratches defect
Computer vision

ABSTRACT

Printed circuit boards (PCBs) are an important component of electronic devices. Therefore, ensuring the quality of such PCBs in the manufacturing process is crucial. Especially, cracks or scratches appearing on the PCB surface pose a significant hurdle, due to their minuscule size, making them the most challenging to address. In this work, we present a real-time automated algorithm for defects inspection of printed circuit boards (PCBs) in different lighting conditions. First, the Oriented FAST and Rotated BRIEF (ORB) algorithm extracts features from the input images, then the Brute-force matching method matches these features with the ORB features template. Next, the input images are calibrated to match the size and orientation of the template data by the RANSAC (Random Sample Consensus) algorithm. Finally, the defective areas on the PCB surface are segmented by using the U-NET (i.e., a type of convolutional neural network (CNN)) model. The proposed algorithm is tested in three different lighting conditions: low light, normal light, and high light conditions. Experimental studies are conducted on a representative PCB to evaluate the defect detection capacity of the proposed algorithm and the experimental results show that the proposed system works well in the three different lighting conditions with an accuracy of up to 97%, the detection speed is 12 frames per second (FPS).

1. Introduction

Printed circuit boards (PCBs) are a critical part of the majority of electronic devices. It has been used in a wide range of different fields, like the medical, industry, automobile, and many others. PCBs are electric circuits, they are thin and solid plates made of laminated materials, fiberglass, or composite epoxy. These materials serve as a physical foundation for chips and electronic components [1]. PCBs have conductive channels to provide power to electronic devices attached to them. The rising demand for electronic devices and industrial electronics has resulted in a significant growth in PCB manufacture. Hence, the number of produced PCBs is increasing sharply. With such a large number of PCBs, quality management in the manufacturing process is challenging. Detecting defects on PCBs performed by humans or traditional algorithms is not effective. The detection results are influenced by the subjective of humans or other conditions such as changes in lighting, and position. Besides, in the eras of technology, electronic devices tend to decrease their size to optimize performance as well as serve many other purposes [2]. This leads to the need for more sophisticated and dedicated PCBs. In these PCBs, only a small scratch can lead to incorrect functionality because it can interrupt or break the flow of electrical connections within the circuit [3]. Therefore, detecting defects automatically on the PCB surface is very important and attracts many companies and scientists to resolve them.

In the last decades, examining the quality of PCB has received many investigations. Traditional methods such as the template-matching method [4,5], and the image subtraction method using OpenCV [6] have been used to detect defects on PCBs. Unfortunately, these approaches are susceptible to variations in transition and rotation. Moreover, they are limited to specific types of defects of PCBs and fail to detect missing or undersized components. The Canny edge detection [7] is a common edge algorithm for detecting defects on PCB. This algorithm works well

* Corresponding author.
** Principal corresponding author.
E-mail addresses: thuckx@haui.edu.vn (X.-T. Kieu), tanpx@shibaura-it.ac.jp (P.X. Tan).

https://doi.org/10.1016/j.rineng.2025.104067
Received 10 September 2024; Received in revised form 30 November 2024; Accepted 16 January 2025

Available online 20 January 2025

in low-light conditions. However, the disadvantage of the Canny edge detection method is complex computation and sensitive to noise [8]. The reference-free path-walking method [9] is a useful method for detecting PCB faults without reference design. However, if measurements are not performed accurately, accuracy is reduced. In addition, it takes a long time for measurement. X-ray computed tomography (XCT) [10,11] is a method widely used for detecting defects on PCBs. This method uses X-rays to create cross-sectional images of the PCB. Hence, the detailed structure of PCB can be examined more accurately. However, this method has many limitations such as it requires high cost, and is time-consuming and complex in evaluating image results.

In recent years, stand out of deep machine learning has made defect detection more efficient. Compared with the traditional PCB defect detection methods, deep learning models have higher efficiency and accuracy. For example, the YOLO [12,13] algorithm provides good detection precision and speed for object detection. However, it preserves a drawback is that it faces difficulty in detecting smaller and tiny objects. A approach based on Faster RCNN [14,15] is proposed to detect tiny defects of PCB and achieved high precision. It is the good detection algorithm due to its high accuracy and robustness. Faster RCNN can detect smaller objects better than YOLO. Nevertheless, Faster RCNN takes a longer time to train, so it is not efficient for real-time applications. Mask RCNN [16] is a robust and effective algorithm. This algorithm is used to segment and determine the exact positions and sizes of objects in the image. The drawbacks of this method are complex architecture and slow speed [17]. Despite its high performance in identifying scratches on the surface of PCBs, deep learning approaches face difficulty in defining all characteristics of a specific defect. This phenomenon is caused by the heterogeneity in the shape of the scratches. In addition, the changes in lighting conditions also affect the detection performance of deep learning algorithms [18].

In this study, to deal with a common error on PCBs: cracks and scratches, a new system is proposed to provide higher accuracy in PCB defect detection. Firstly, the ORB (Oriented FAST and Rotated BRIEF) algorithm is used to extract the features of input images [19], and then the Brute-force matching method [20,21] matches these features with ORB features in the template data. Next, the input images are calibrated to match the size and orientation of the template data by the RANSAC (Random Sample Consensus) algorithm [22–24]. The U-NET model [25] is incorporated for the purpose of segmenting scratch defects. In conclusion, the proposed system has three major contributions, as highlighted below:

1. The existing PCB defect detection system focuses mainly on detecting the number of defects as much as possible and increasing the accuracy in normal conditions. There are few research evaluate and improve the performance in different light conditions. In this paper, a system is proposed to detect cracks and scratches on the surface of PCBs in different lighting conditions.
2. According to the experimental results, although the system works with very few training images, it brings productive performances in detecting defects in PCB. The highest accuracy of the proposed system reached 97% in ideal conditions.
3. In this study, to optimize production efficiency and ensure the quality of printed circuit boards (PCBs), the proposed system can work in real-time. Additionally, the detailed information and the functional operation of the system are shown automatically on the HMI display screen. Therefore, the workforce and manual resources are decreased dramatically.

The remaining paper is organized as follows. Section 2 describes in detail the materials and methods used in the proposed system; Section 3 discusses the achieved result of the proposed algorithm; Section 4 shows the conclusion and gives out the task that needs improvement in the future.

2. Materials and methods

2.1. System architect

Overall, there are three basic stages in this system. The initial phase is pre-processing including five smaller steps. The first step involves adjusting the images captured by the camera. Then, the ORB algorithm extracts features from the input images and reference images. Next, the Brute-Force matching method matches the ORB features with similar ORB feature templates. Hence, the best edge key point is found. If the matching results are not good, these images are re-calibrated and re-matched until getting the best outcome. The PCBs in this error-detection processing are rotated in a different direction from the PCBs template. Subsequently, the input images are calibrated to match the size and orientation of the template data by the RANSAC algorithm. More specifically, by utilizing the RANSAC algorithm in conjunction with the best key points, the transform matrix is calculated to address the size and orientation discrepancies of the PCBs.

The second stage involves identifying and generating ROI errors in the PCBs. Initially, the input images and template data obtained in the first stage are applied image subtraction algorithm based on the YUV color space. Next, the results are treated to noise reduction through the utilization of the OPENING method and thresholding method. These methods allow the creation of ROI errors in the PCBs. The ROI errors are evaluated based on three criteria: area, length-width ratio, and consistency. If the ROI regions do not meet all three criteria, the PCBs contain no error. In contrast, if the ROI errors satisfy all three criteria, the third step is carried out.

In the last stage, all kinds of ROI errors found in the second step are segmented by using the U-NET models to determine the scratches of PCBs. Fig. 1 shows the flowchart of the proposed algorithm.

2.2. Process of defect detection board

Nowadays, some pinhole cameras introduce significant distortion to images. There are two main kinds of distortion. It is radial distortion and tangential distortion. Radial distortion causes straight lines to appear curved. Fig. 2 presents the distortion of the Arduino board.

The border of the chessboard is not a straight line. This problem is solved by using a camera calibration [26]. First, the distortions are written as mathematical functions. These functions describe the displacement of pixels in the images with radial distortion:

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{1}$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{2}$$

with tangential distortion:

$$x_{corrected} = x + [(2p_1 xy + p_2(r^2 + 2x^2)] \tag{3}$$

$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2 xy] \tag{4}$$

whereas distortion coefficients $= (k_1, k_2, p_1, p_2, k_3)$

Besides, the intrinsic and extrinsic parameters of the camera play a crucial role. An intrinsic matrix is a coefficient matrix. It provides camera parameters such as focal length (f_x, f_y), optical centers (c_x, c_y) [27]. Therefore, these camera parameters (or called camera matrix) depend on each type of camera, and this matrix is calculated by the matrix P:

$$\mathbf{P} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

Extrinsic parameters depend on rotation and translation vectors. These vectors transform the coordinates of 3D points into an image coordinate. After the input images are calibrated, the matching between the input images and the template images proceeded by determining the optimal transform matrix between the two images [28]. Assume H is a

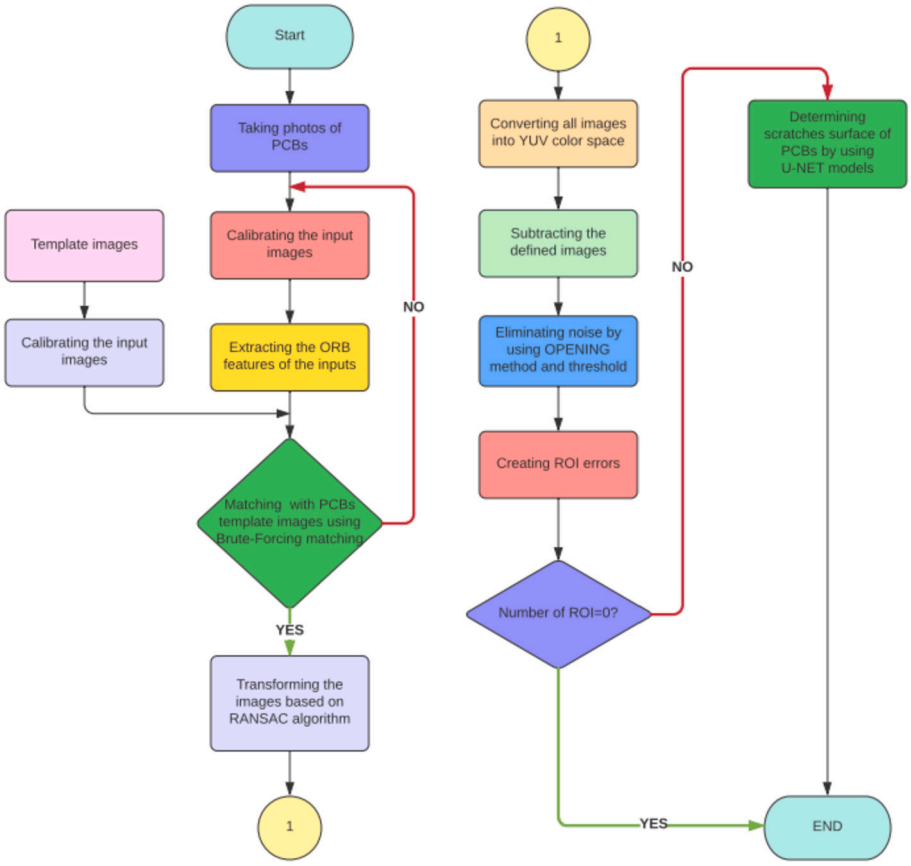

Fig. 1. Flowchart of the proposed algorithm.

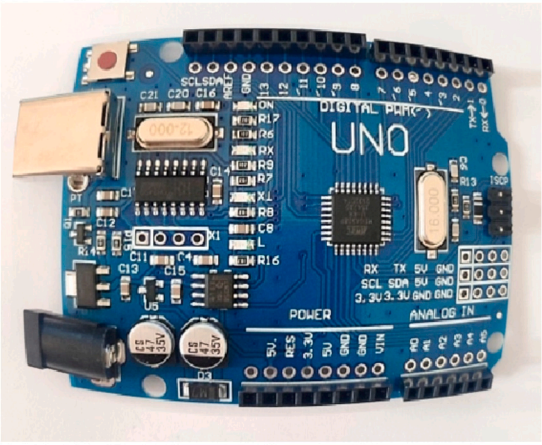

Fig. 2. Distortion of Arduino Uno board.

3x3 matrix. The H matrix is used to adjust the rotation direction and scale so that the two images fit perfectly. The transformation equation in Homography space is defined as follows:

$$\begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} = H \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{6}$$

whereas:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{7}$$

To calculate the **H** matrix, it is necessary to use at least nine known corresponding points in the two images [29]. By using the feature points with the ORB algorithm, the coefficients of the matrix are calculated. After the computation process, the **H** matrix is found. However, the distance values between the feature vectors are treated equally. This results in the incorrect matching of some point pairs and results in outliers. The RANSAC algorithm is applied to solve this problem. First, the RANSAC algorithm randomly selects a subset of data points to compute the matrix **H** [30]. Then the RANSAC finds the transformation matrix by eliminating outliers and finding the best-fitting model. The final model is the model that is obtained with the largest number of inliers during the iterations N. When the optimal transformation matrix **H** is found, we then apply the matrix **H** to transform the input PCB image into the template PCB image as shown in Fig. 3. The number of iterations N is chosen based on the probability ρ as follows:

$$1 - (1 - \partial^K)^N = \rho \tag{8}$$

Using the same inlier ratio ∂ and sample size N, a smaller subset cardinality K can achieve a higher success probability ρ. To fit the model during sampling, the RANSAC algorithm always requires a minimal K. And thus with some manipulation:

$$N = \frac{log(1-\rho)}{log[1-(1-\upsilon)^K]} \tag{9}$$

where υ is the probability of observing outliers.

After the input images are transformed, these result images and the PCB template to YUV color space conversion are computed in order to reduce the influence of luminance. The results that have been transformed are known as the YUV dataset [22]. The subsequent procedure involves the computation of the disparity for each channel in the YUV dataset. The disparity is determined by using the following equation:

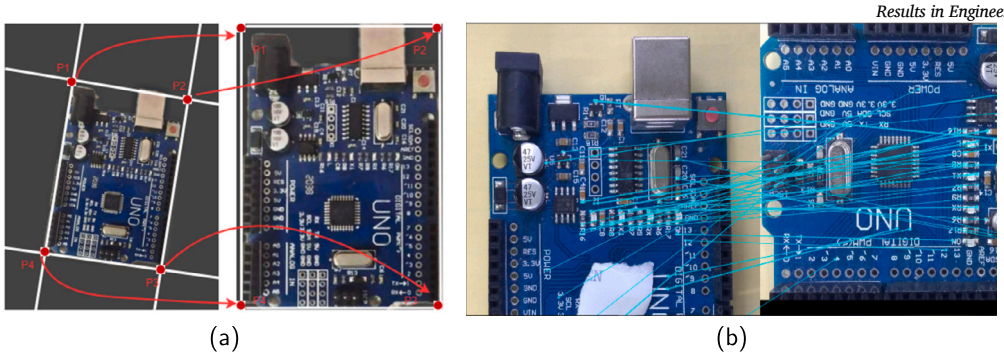

Fig. 3. Re-orientation images and the matching key point in RANSAC algorithm. (a) Re-orientation of re-ordered corners to form an upright segmented view. (b) Matching key points among two images.

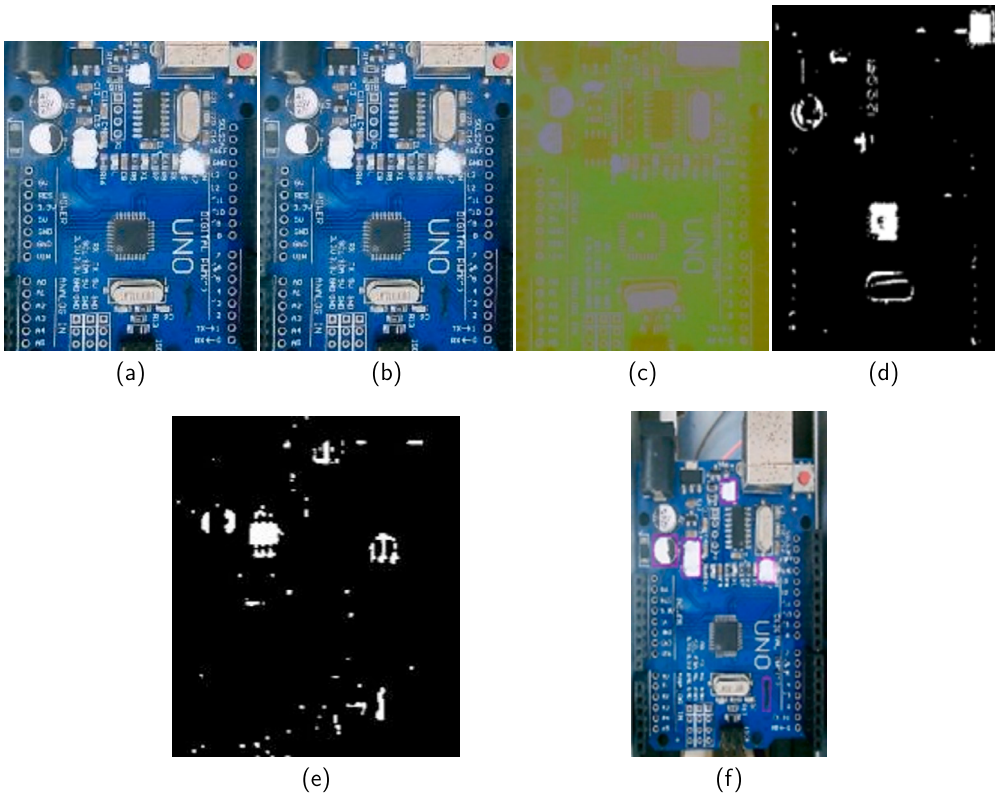

Fig. 4. The process of finding error areas in Arduino Uno board. (a) Input image. (b) Result image after transformation. (c) YUV color space. (d) The binary image. (e) Result of OPENING method. (f) Result image with error areas.

$$E = |I_c - T_c| \tag{10}$$

where E is the absolute difference between input image values I_c and template images pixel value T_c.

To segment error areas, the output images are converted to "blobs" or binary images. Due to the noise degrades the quality of an image and affects the accuracy of image processing, the OPENING method is used to address the noise issue on the results. The process of OPENING consists of two sequential operations: dilation and erosion [31]. Dilation is an operation that involves expanding the boundaries of objects in an image, and Erosion is shrinking the boundaries of objects in an image. This OPENING operation is effective in noise reduction while preserving the structure of the main objects. Three types of thresholds are used defined:

$$wh = Thresh_{area} \tag{11}$$

$$\frac{wh}{s} = Thresh_{solid} \tag{12}$$

$$\frac{w}{h}, \frac{h}{w} = Thresh_{scale} \tag{13}$$

where w is the width of the blob. h is the height of the blob. s is the area of the blob. Fig. 4 shows the results for finding error areas in the Arduino Uno board of the proposed system.

2.3. U-NET architecture

In this paper, to segment the scratches on the surface of circuits, the U-NET model [25] is used to solve the task. U-NET architecture is structured by an encoder and decoder part [32]. The encoder part is used to extract features like other CNN models, and the decoder part restores the size of the feature map. In addition, the skip connection is combined to give out an output image. U-Net does not include fully connected layers and primarily relies on convolutional layers [33]. Only the valid part of each convolution is used. Hence, input images of U-NET are used with any size. U-NET architecture is shown in Fig. 5. The input images are resized into 512x512 pixels, and the output model is a segmentation

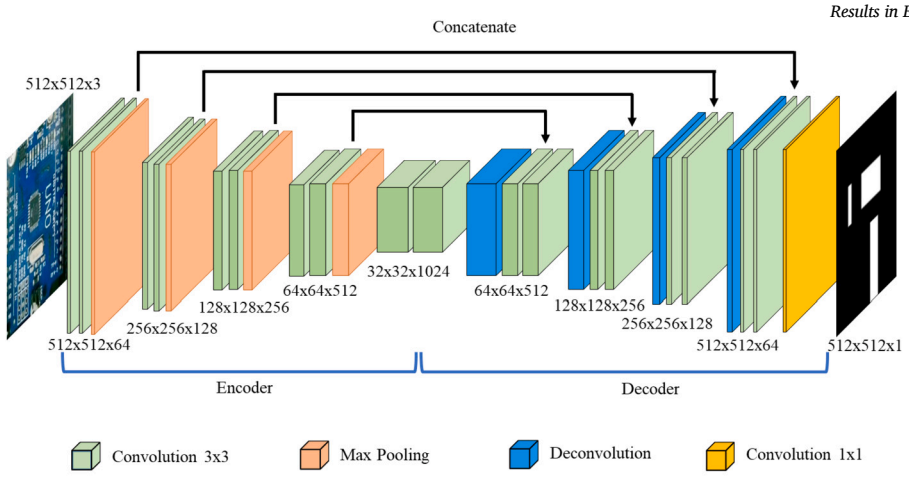

Fig. 5. U-NET architecture.

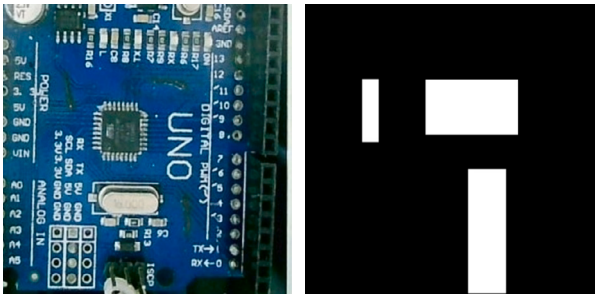

Fig. 6. The defecting of detection in PCB. (a) input image. (b) the segmentation mask.

map. The U-Net architecture incorporates the process of concatenating feature maps in order to increase the number of channels. The process of training the model necessitates a smaller number of data sets and exhibits the ability to converge on a limited amount of data. The U-NET uses a loss function for each pixel of the image. Firstly, pixel-wise softmax is applied on the resulting image, and then the cross-entropy loss [34]. The softmax is defined by:

$$p_k(x) = exp(\frac{a_k(x)}{\sum_{k'=1}^{K} exp(a_{k'}(x))}) \quad (14)$$

where $a_k(x)$ is the activation in the feature channel at the pixel position x. K is the number of classes.

The cross-entropy then is used to compute the degree of difference between the model's prediction and the true label at each pixel [35]:

$$E = \sum_{x\in\Omega} \omega(x) log(p_{l(x)}(x)) \quad (15)$$

where l represents the actual label of each pixel and $\omega(x)$ is a weight map.

To account for the varying frequency of pixels from a particular class in the training data set, the weight map must be recalculated [36]. Therefore, the network concentrates on the small separation borders. Fig. 6 displays the result of segmentation.

2.4. Hardware structure

The depiction of the hardware configuration of the proposed system is shown in Fig. 7.

Step 1: The images of the printed circuit boards (PCBs) are captured using a 2D camera with a resolution of 5 megapixels.

Step 2: the images are transmitted to the Jetson Nano Developer Kit, which serves as the primary processing unit responsible for executing image processing algorithms. The configuration of the Jetson Nano uses a CPU (Quad-core ARM Cortex-A57), a GPU (128-core NVIDIA GPU), a RAM (4GB LPDDR4), a Storage (Onboard storage (typically a microSD card or M.2 SSD)).

Step 3: After the image processing has been completed, the primary controller (Arduino Mega 2560R3) receives signals from sensors and gives out control signals to the Motor Drive Module in order to regulate the line.

The system is comprised of five primary components as shown in Fig. 8(a), including the control box, the conveyor belts, the image processing box, the vacuum handling system, and the HMI display monitor. Fig. 8(b) shows the main components inside the control box including the circuit breaker, switching mode power supply, bucking circuit, relay, and Arduino board. The conveyor system is partitioned into two distinct belts. The first conveyor belt is used to transport the printed circuit boards (PCBs) into the image processing box, which contains a camera and lighting subsystem to enhance accuracy. The second conveyor belt is applied for the correct PCBs to move them out of the image processing box and transport them to other stages. Conversely, if any faults are detected in the PCBs, they are then recovered by the picking system and relocated to an alternative box. The error PCBs are picked by a vacuum handling system that is driven by a stepper motor, a pneumatic cylinder, and a vacuum gripper. The processing box is equipped with a Human-Machine Interface (HMI) monitor. This monitor serves to display information on PCBs to users.

To show detailed information about PCBs, the interface for users is designed on PyQT Open Library. The largest area in the interface displays the image from the webcam. The other areas present the errors, the processing results, and the control panel. The user interface of the system is shown in Fig. 9(a) and Fig. 9(b).

3. Results and discussion

The proposed system is tested with the Arduino Uno SMD board in low light, normal light, and high light conditions. In addition, the results are compared with the YOLOv5 [34], YOLOv3 [35], Faster RCNN [12, 13], and Mask RCNN [14] in the same conditions, as mentioned above to demonstrate the performance.

3.1. Dataset preprocessing

The experiment utilized our PCB dataset. The dataset comprises 2500 images, including 700 images of good PCB, 600 error images in low light, 600 error images in normal light, and 600 error images in high light conditions. The type of PCB used in this study is Arduino Uno SMD

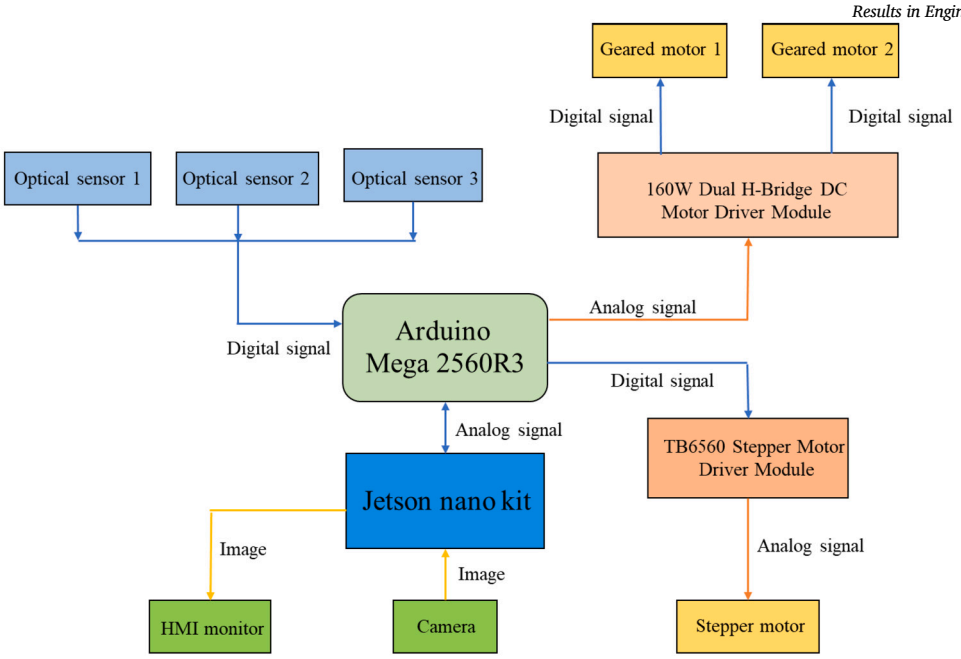

Fig. 7. The hardware structure of the proposed system.

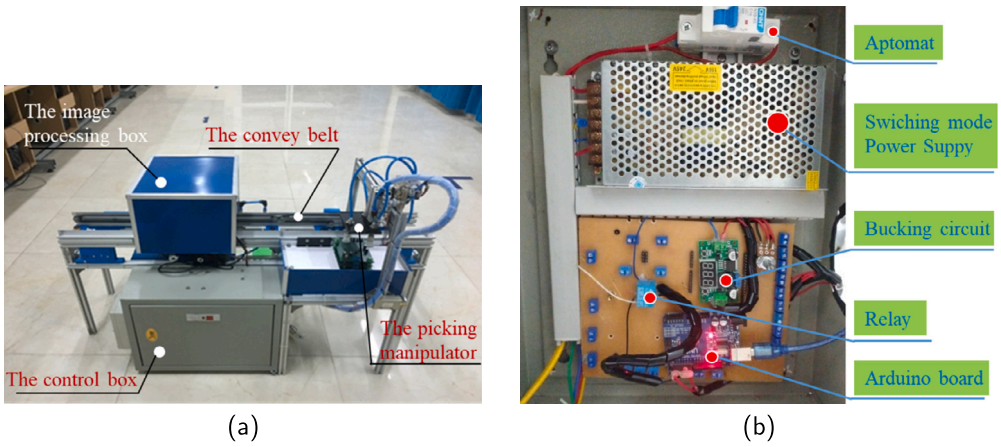

Fig. 8. Illustration of the experimental setup. (a) The appearance of the proposed system. (b) The components inside the control box.

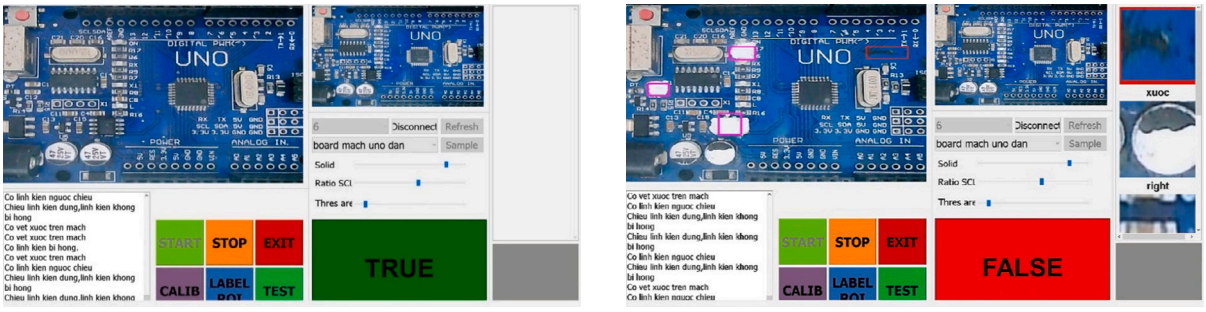

Fig. 9. User interface of the proposed system. (a) The user interface of the correct PCB. (b) The user interface of the incorrect PCB.

board and it contains the image of PCB with scratches on the surface. The dataset was then divided into 80% for training and 20% for test, respectively. Due to the lack of real PCB with scratches, we created PCB with scratches by using a pen to draw and Photoshop software to draw the scratch errors. Fig. 10 shows some examples of shape scratches in datasets.

3.2. Evaluation metrics

In this paper, the Mean Average Precision (mAP) is used to evaluate the models as it can effectively evaluate the recognition effect of the model. This parameter is composed of Precision and Recall. The Precision and Recall is calculated as follows:

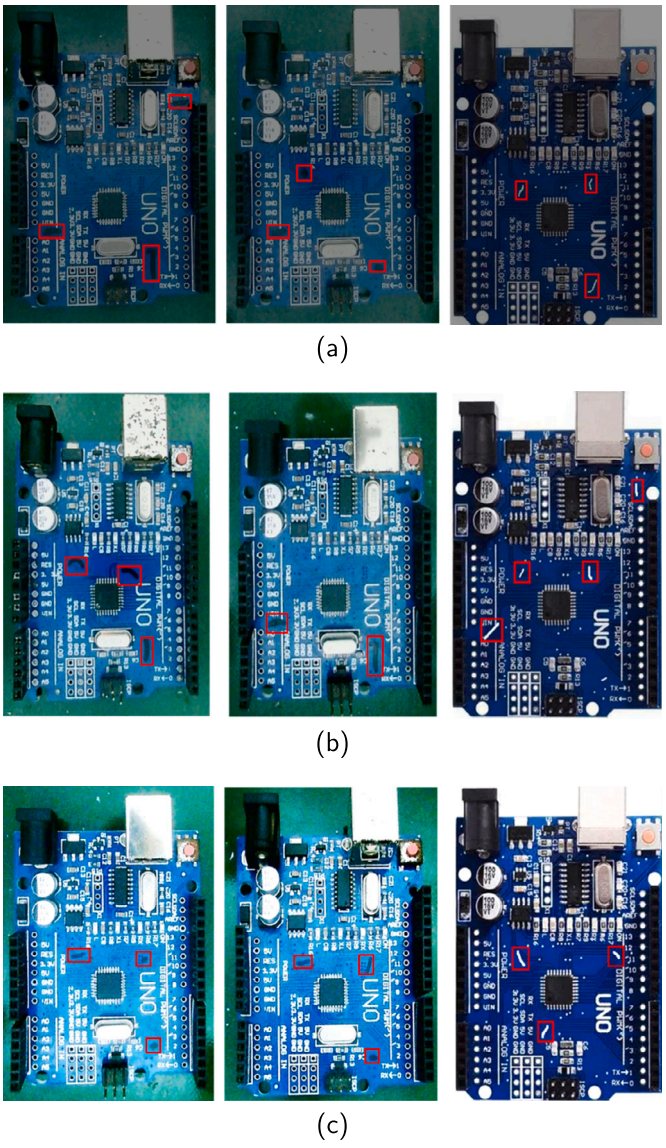

Fig. 10. Example of scratches on PCB dataset (a) Low light conditions. (b) Normal light conditions. (c) High lightconditions.

$$Precision = \frac{TP}{TP + FP} \tag{16}$$

$$Recall = \frac{TP}{TP + FN} \tag{17}$$

where Precision measures the accuracy of positive predictions, Recall measures the completeness of positive predictions, TP is True Positives, FP is False Positives, FN is False Negatives. Next, average precision (AP) is defined as follows:

$$AP = \sum_{k=1}^{k=n-1} [R_{k+1} - R_k] P_{interp}(r_{k+1}) \tag{18}$$

where n is the number of thresholds. R_k is the Recall, P_{interp} is Interpolated precision.
The mAP is calculated based on the AP:

$$mAP = \frac{1}{k} \sum_{k=1}^{k=n} AP_k \tag{19}$$

where n is the number of classes, AP_k is the AP of class k.

3.3. Analysis of results

The five algorithms are trained on Google Colab. Fig. 11 shows the performance of five systems when IoU is 0.5. The U-NET architecture's training results gained noticeable performance rather than the YOLOv3, YOLOv5, Faster RCNN, and Mask RCNN networks. To be more exact, in the U-NET model, after 120 epochs, the mAP tends to converge. At the end of training the mAP is 0.975. Whereas the YOLOv5 network tends to converge after 170 epochs, and mAP is 0.937 at the end. YoloV3's mAP is the lowest at 0.787. The mAP of Faster RCNN and Mask RCNN is 0.963 and 0.941, respectively. However, the training time of Faster RCNN and Mask RCNN is the longest with 9.2 hours and 12.5 hours, respectively.

Table 1 represents the training parameters of YOLOv5 [37], YOLOv3 [38] and U-NET. The YOLOv5, YOLOv3, and U-NET have the same number of trains, number of tests, batch size, maximum training, model checkpoint, and input image size. In the U-NET network, the pre-trained is applied to reduce training time. Hence, the training time of the U-NET model is only 55 minutes.

To evaluate the performance, the proposed system is tested with 500 PCB in different light conditions. The intensity of light is changed in the image processing box when it comes in. The intensity of light is measured by the Extech EA30 device. In this study, the system is tested in three intensities including 80 lux (low light), 400 lux (normal light), and 2000 lux (high light). Take a board with three scratches for example, in normal light conditions, all three scratches are detected by the proposed method, YOLOv5, YOLOv3, Faster RCNN, and Mask RCNN. Fig. 13 shows the results of five algorithms in normal light conditions. However, in low light conditions and high light conditions, the YOLOv3, and YOLOv5 often miss out on some defects while the proposed method, Faster RCNN, Mask RCNN still detects all scratches on PCB. For example, in low light conditions, the YOLOv5's bounding boxes do not fit with error areas like the proposed system. Whereas, the YOLOv3 only detects two scratches. In high light conditions, the YOLOv5 detects an electronic part as an error, and YOLOv3 only detects one scratch. The proposed system, Faster RCNN, and Mask RCNN still detect all of the scratches. Fig. 12 and Fig. 14 show the results of the five methods in low-light conditions and high-light conditions respectively.

The comparison results are displayed in Table 2. In general, the detection rate of the proposed method is considerably higher than the other algorithms. In specific, in normal light conditions, the proposed approach reaches an accuracy of 97%. At the same condition, the highest accuracy of YOLOv3, YOLOv5, Faster RCNN, and Mask RCNN is only 83,8%, 93.2%, 95.2% and 93.2% respectively. In complex conditions, the accuracy of the proposed system, and the other algorithms have a downward trend. For example, in low light conditions, the proposed method, YOLOv5, YOLOv3, Faster RCNN, and Mask RCNN declined to 90.6%, 86.4%, 76%, 91.8%, and 90.4%, respectively. In high light conditions, the accuracy of the proposed system is decreased at 91.2%, while YOLOv5 is 87.8%, YOLOv3 is 79.6%, Faster RCNN and Mask RCNN is 91%. Although the Faster RCNN and Mask RCNN can detect all the scratches in this example, the FPS of these two algorithms is only 2-3 FPS. Therefore, it is not suitable for real-time application.

4. Conclusion

This paper presents a novel defect detection system for assessing the quality of printed circuit boards (PCBs) across various scenarios including low light, normal light, and high light conditions. Specifically, the proposed system is developed by integrating ORB, Brute-force matching, RANSAC algorithm to transform the image based on the template image, and U-NET model is used to detect error areas. Besides, the interface of the system is designed and developed by adding PyQT libraries. Training results show that the proposed system achieves the highest mAP of 0.975 compared to the other methods. The comparison in terms of accuracy and speed between the proposed method with the YOLOv5, YOLOv3,

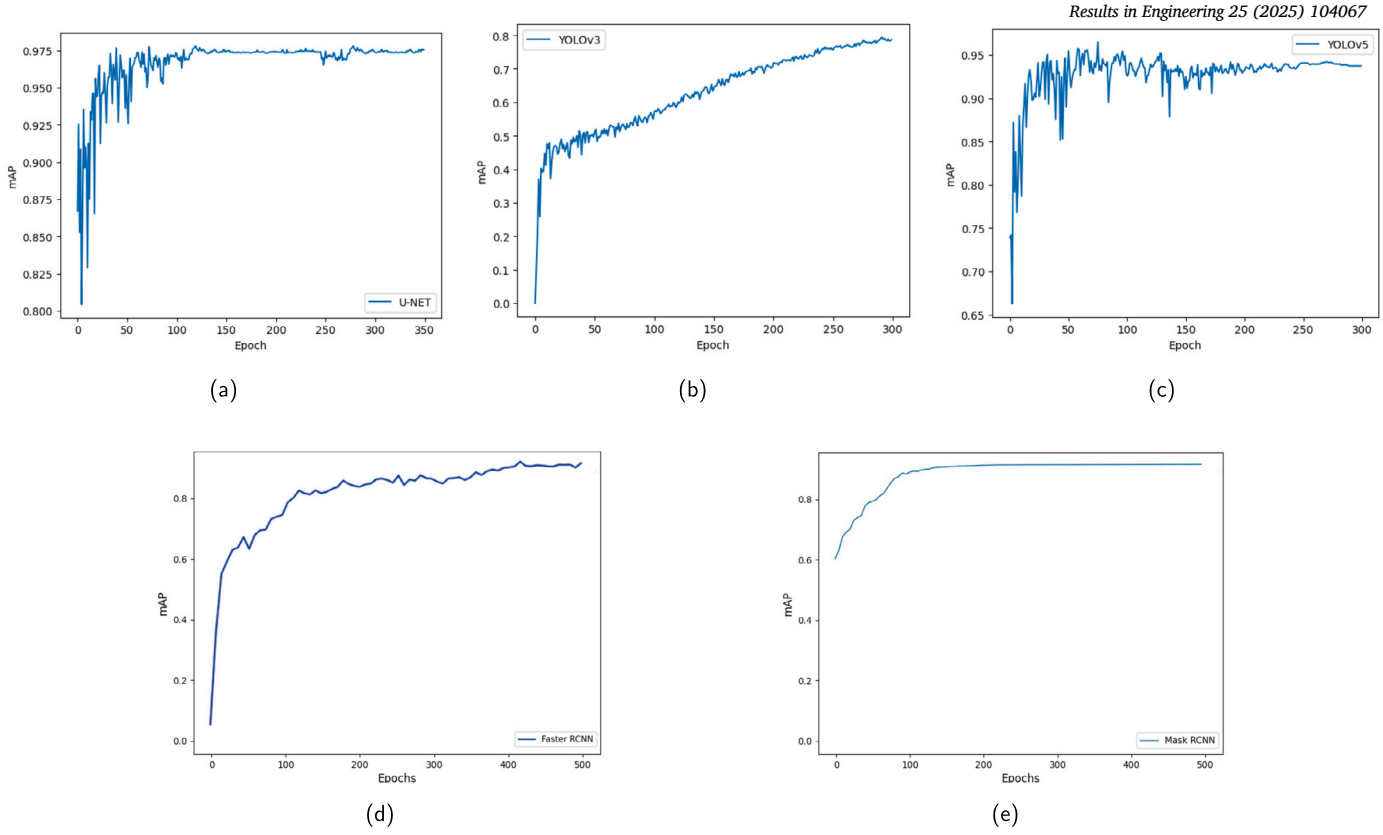

Fig. 11. Training result of the proposed system, YOLOv3 and YOLOv5 when IoU is 0.5. (a) The proposed system. (b) The YOLOv3. (c) The YOLOv5.

Table 1
The training parameters of the YOLOv5, YOLOv3, Faster RCNN, Mask RCNN and U-NET model.

Training parameters	YOLOv5 [37]	YOLOv3 [38]	Faster RCNN [14,15]	Mask RCNN [17]	Our
Input image size	512x512	512x512	512x512	512x512	512x512
No. of train	2000	2000	2000	2000	2000
No. of test	500	500	500	500	500
Pre-train	Yes	No	Yes	Yes	Yes
Batch size	16	16	16	16	16
Maximum training	500	500	500	500	500
Model checkpoint	1 epoch	1 epoch	1 epoch	1 epoch	1 epoch
Epochs	300	300	500	500	350
Time training	4.5 hours	3 hours	9.2 hours	12.5 hours	55 minutes

Table 2
Performance comparison of five models in Arduino Uno SMD board.

Lighting conditions	Our		YOLOv5 [37]		YOLOv3 [38]		Faster RCNN [14,15]		Mask RCNN [16]	
	Acc (%)	FPS	Acc (%)	FPS	Acc (%)	FPS	Acc (%)	FPS	Acc (%)	FPS
Low light conditions (80 lux)	90.6%	12	86.4%	15	76%	14	91.8%	3	90.4%	2
Normal light conditions (400 lux)	**97%**	12	93.2%	15	83.8%	14	95.2%	3	93.2%	2
High light conditions (2000 lux)	91.2%	12	87.8%	15	79.6%	14	91%	3	91%	2

Faster RCNN, and Mask RCNN demonstrates the higher performance of the proposed algorithm in changing light conditions. Specifically, the experiments show that the proposed system is able to successfully detect scratches in three light conditions including low light, normal light, and high light with an accuracy of 90.6%, 97%, and 91.2% respectively. The FPS of this approach is around 12. In the future, the proposed method will be improved to detect more types of defects on PCB like solder joints, and component errors.

CRediT authorship contribution statement

Van-Truong Nguyen: Writing – review & editing, Writing – original draft, Supervision, Project administration, Methodology, Formal analysis, Data curation, Conceptualization. **Xuan-Thuc Kieu:** Writing – review & editing, Supervision, Software, Resources, Data curation, Conceptualization. **Duc-Tuan Chu:** Writing – original draft, Visualization, Validation, Software, Investigation, Data curation, Conceptualiza-

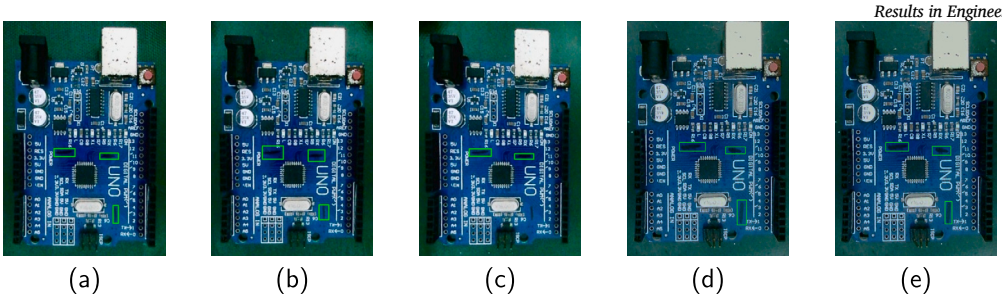

Fig. 12. Results in low light conditions. (a) The proposed system. (b) YOLOv5. (c) YOLOv3. (d) Faster RCNN. (e) Mask RCNN.

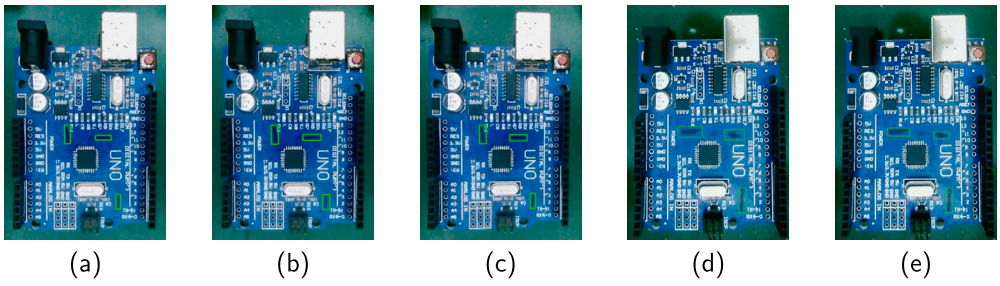

Fig. 13. Results in normal light conditions. (a) The proposed system. (b) YOLOv5. (c) YOLOv3. (d) Faster RCNN. (e) Mask RCNN.

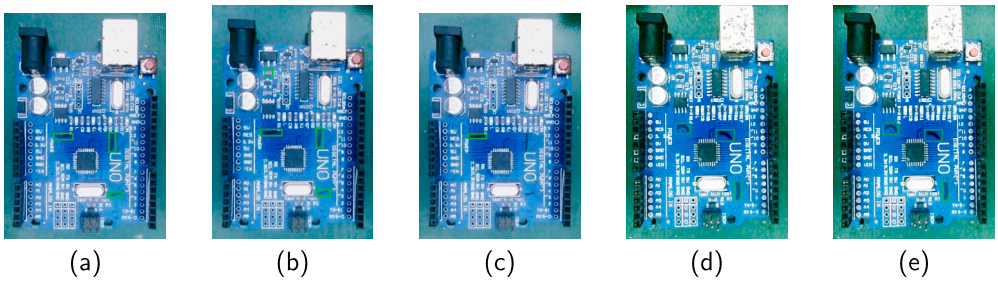

Fig. 14. Results in high light conditions. (a) The proposed system. (b) YOLOv5. (c) YOLOv3. (d) Faster RCNN. (e) Mask RCNN.

tion. **Xiem HoangVan:** Visualization, Resources, Project administration, Methodology, Investigation. **Phan Xuan Tan:** Writing – review & editing, Software, Resources, Methodology. **Tuyen Ngoc Le:** Writing – review & editing, Software, Resources, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] A. Atintoh, W. Kpobie, N. Bonfoh, M. Fendler, F. Addiego, P. Lipinski, Multiscale characterization of the mechanical behavior of a printed circuit board (PCB), Mater. Today Commun. 34 (2023) 104968.
[2] M.A. Khater, High-speed printed circuit boards: a tutorial, IEEE Circuits Syst. Mag. 20 (3) (2020) 34–45.
[3] M. Liu, F. Ye, X. Li, K. Chakrabarty, X. Gu, Board-level functional fault identification using streaming data, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 40 (9) (2020) 1920–1933.
[4] C.F.J. Kuo, C.H. Tsai, W.R. Wang, H.C. Wu, Automatic marking point positioning of printed circuit boards based on template matching technique, J. Intell. Manuf. 30 (2) (2019) 671–685.
[5] Q. Kong, Z. Wu, Y. Song, Online detection of external thread surface defects based on an improved template matching algorithm, Measurement 195 (2022) 111087.
[6] X. Li, G. Liu, S. Sun, C. Bai, Contour detection and salient feature line regularization for printed circuit board in point clouds based on geometric primitives, Measurement 185 (2021) 109978.
[7] W. Liu, X. Yang, X. Yang, H. Gao, A novel industrial chip parameters identification method based on cascaded region segmentation for surface-mount equipment, IEEE Trans. Ind. Electron. 69 (5) (2021) 5247–5256.
[8] J. Xu, Y. Liu, Y. Wu, Automatic defect inspection for monocrystalline solar cell interior by electroluminescence image self-comparison method, IEEE Trans. Instrum. Meas. 70 (2021) 1–11.
[9] W. Jin, W. Lin, X. Yang, H. Gao, Reference-free path-walking method for ball grid array inspection in surface mounting machines, IEEE Trans. Ind. Electron. 64 (8) (2017) 6310–6318.
[10] H. Villarraga-Gómez, E.L. Herazo, S.T. Smith, X-ray computed tomography: from medical imaging to dimensional metrology, Precis. Eng. 60 (2019) 544–569.
[11] N.Q. Nguyen, S.F. Su, Q.V. Tran, V.T. Nguyen, J.T. Jeng, Real time human tracking using improved CAM-shift, in: Joint 17th World Congress of International Fuzzy Systems Association and 9th International Conference on Soft Computing and Intelligent Systems, 2017, pp. 1–5.
[12] Y. Lu, B. Yang, Y. Gao, Z. Xu, An automatic sorting system for electronic components detached from waste printed circuit boards, Waste Manag. 137 (2022) 1–8.
[13] V.T. Nguyen, A.T. Nguyen, V.T. Nguyen, H.A. Bui, A real-time human tracking system using convolutional neural network and particle filter, in: Intelligent Systems and Networks, 2021, pp. 411–417.
[14] R. Ding, L. Dai, G. Li, H. Liu, TDD-net: a tiny defect detection network for printed circuit boards, CAAI Trans. Intell. Technol. 4 (2) (2019) 110–116.
[15] V. Sudharshan, P. Seidel, P. Ghamisi, S. Lorenz, M. Fuchs, J.S. Fareedh, P. Neubert, S. Schubert, R. Gloaguen, Object detection routine for material streams combining RGB and hyperspectral reflectance data based on guided object localization, IEEE Sens. J. 20 (19) (2020) 11490–11498.
[16] H. Wu, W. Gao, X. Xu, Solder joint recognition using mask R-CNN method, IEEE Trans. Compon. Packag. Manuf. Technol. 10 (3) (2019) 525–530.
[17] R.S. Zimmermann, J.N. Siems, Faster training of Mask R-CNN by focusing on instance boundaries, Comput. Vis. Image Underst. 188 (2019) 102795.

[18] D.M. Tsai, Y.H. Chou, Fast and precise positioning in PCBs using deep neural network regression, IEEE Trans. Instrum. Meas. 69 (7) (2019) 4692–4701.
[19] I.A. Soomro, A. Ahmad, R.H. Raza, Printed circuit board identification using deep convolutional neural networks to facilitate recycling, Resour. Conserv. Recycl. 177 (2022) 105963.
[20] Y. Wang, W. Liu, F. Li, H. Li, W. Zha, J. He, G. Ma, Y. Duan, A fast template matching method based on improved ring projection transformation and local dynamic time warping, Optik 216 (2020) 164954.
[21] C. Zhao, Z. Cao, J. Yang, K. Xian, X. Li, Image feature correspondence selection: a comparative study and a new contribution, IEEE Trans. Image Process. 29 (2020) 3506–3519.
[22] L. Bai, X. Yang, H. Gao, A novel coarse–fine method for ball grid array component positioning and defect inspection, IEEE Trans. Ind. Electron. 65 (6) (2017) 5023–5031.
[23] E. Shojaedini, M. Majd, R. Safabakhsh, Novel adaptive genetic algorithm sample consensus, Appl. Soft Comput. 77 (2019) 635–642.
[24] V.T. Nguyen, H.A. Bui, A real-time defect detection in printed circuit boards applying deep learning, Eureka 2 (2022) 143–153.
[25] W. Du, H. Shen, J. Fu, Automatic defect segmentation in X-ray images based on deep learning, IEEE Trans. Ind. Electron. 68 (12) (2020) 12912–12920.
[26] Y. Zhang, X. Zhao, D. Qian, Learning-based distortion correction and feature detection for high precision and robust camera calibration, IEEE Robot. Autom. Lett. 7 (4) (2022) 10470–10477.
[27] Z. Zhu, X. Wang, Q. Liu, F. Zhang, Camera calibration method based on optimal polarization angle, Opt. Lasers Eng. 112 (2019) 128–135.
[28] D. Ghosh, N. Kaabouch, A survey on image mosaicing techniques, J. Vis. Commun. Image Represent. 34 (2016) 1–11.
[29] T. Nguyen, S.W. Chen, S.S. Shivakumar, C.J. Taylor, V. Kumar, Unsupervised deep homography: a fast and robust homography estimation model, IEEE Robot. Autom. Lett. 3 (3) (2018) 2346–2353.
[30] S. Ma, P. Guo, H. You, P. He, G. Li, H. Li, An image matching optimization algorithm based on pixel shift clustering RANSAC, Inf. Sci. 562 (2021) 452–474.
[31] B. Boustani, A. Javaherian, M. Nabi-Bidhendi, S. Torabi, H.R. Amindavar, Mapping channel edges in seismic data using curvelet transform and morphological filter, J. Appl. Geophys. 160 (2019) 57–68.
[32] H. Zunair, A.B. Hamza, Sharp U-net: depthwise convolutional network for biomedical image segmentation, Comput. Biol. Med. 136 (2021) 104699.
[33] M.E. Gegundez-Arias, D. Marin-Santos, I. Perez-Borrero, M.J. Vasallo-Vazquez, A new deep learning method for blood vessel segmentation in retinal images based on convolutional kernels and modified U-net model, Comput. Methods Programs Biomed. 205 (2021) 106081.
[34] Q. Li, W. Jia, M. Sun, S. Hou, Y. Zheng, A novel green apple segmentation algorithm based on ensemble U-net under complex orchard environment, Comput. Electron. Agric. 180 (2021) 105900.
[35] X.Y. Zhou, G.Z. Yang, Normalization in training U-net for 2-D biomedical semantic segmentation, IEEE Robot. Autom. Lett. 4 (2) (2019) 1792–1799.
[36] S. Kumar, P. Jayagopal, Delineation of field boundary from multispectral satellite images through U-net segmentation and template matching, Ecol. Inform. 64 (2021) 101370.
[37] J. Yang, Z. Liu, W. Du, S. Zhang, A PCB defect detector based on coordinate feature refinement, IEEE Trans. Instrum. Meas. 72 (2023) 1–10.
[38] S.H. Chen, C.C. Tsai, SMD LED chips defect detection using a YOLOv3-dense model, Adv. Eng. Inform. 47 (2021) 101255.