PCBShot: An Assisted Image Acquisition Method for PCB Damage Detection With Mobile Devices

1st Lucas Cabral
Systems and Databases Laboratory
Federal University of Ceará
Fortaleza, Brazil
lucas.cabral@lsbd.ufc.br

2nd Lucas Sena
Systems and Databases Laboratory
Federal University of Ceará
Fortaleza, Brazil
lucas.sena@lsbd.ufc.br

3rd João Pedro Santiago
Systems and Databases Laboratory
Federal University of Ceará
Fortaleza, Brazil
pedro.santiago@lsbd.ufc.br

4th Joaquim Bento Cavalcante Neto
Systems and Databases Laboratory
Federal University of Ceará
Fortaleza, Brazil
joaquimb@dc.ufc.br

5th Yuri Lenon
Systems and Databases Laboratory
Federal University of Ceará
Fortaleza, Brazil
yuri@dc.ufc.br

6th Javam Machado
Systems and Databases Laboratory
Federal University of Ceará
Fortaleza, Brazil
javam.machado@lsbd.ufc.br

***Abstract*—Identifying damages in Printed Circuit Boards is a critical task for quality assurance and repair inspection workflows. Image processing mobile applications, with embedded deep learning, assist technicians in detecting damages in this task, increasing accuracy and agility. However, the performance of such applications is highly dependent on the ability of the user in taking adequate photos. We propose an automatic capture method named PCBShot, that assists users of mobile applications of PCB damage detection to take better photos, enhancing the detection performance. Our method uses classical image processing algorithms to detect if a target PCB is inside a virtual guideline, ensuring that the position and distance are appropriate. Then, a photo is automatically captured, the background is cropped and the image is sliced into four quadrants for resolution preservation. The damage detection is performed in the slices. We evaluate our method through a real-life mobile application used in repair centers of an electronics manufacturer, comparing the detection performance with the manual image acquisition, without further assistance. Our results show that our method largely surpasses the manual acquisition, as it allows the capture of higher-quality images due to framing assistance with image processing methods, eliminating noisy backgrounds and preserving resolution.**

***Index Terms*—pcb damage detection, image acquisition, mobile application, automatic capture**

I. Introduction

The proliferation of mobile devices such as smartphones and tablets has significantly impacted modern life by facilitating communication, daily planning, multimedia consumption, and gaming. In particular, smartphones are replacing desktop computers and laptops as the primary computing platforms due to their mobility, connectivity, and application diversity [1], [2]. With ongoing advancements, these devices have seen increases in both versatility and performance, alongside enhancements in the number of their embedded sensors. These improvements open new avenues for utilizing mobile devices in specialized image processing tasks in various contexts, including industrial applications [3], agriculture [4] and healthcare [5].

One such task that leverages mobile devices' portability and connectivity is the automatic detection of damage in printed circuit boards. A printed circuit board (PCB) is an essential component in electronic devices, providing both mechanical support and electrical connections for various electronic parts, thus forming the backbone of their internal structure. Thus, evaluating their integrity during the quality assurance process and repair inspections in warranties programs is of great importance [6].

Great advances were obtained in this task using deep neural networks trained to perform deep object detection [7], [8]. Deep object detection is a computer vision technique that uses deep learning algorithms to locate and classify objects within an image or video, delimiting them by bounding boxes. This approach combined with mobile-based solutions not only facilitates on-the-go inspections but also integrates seamlessly into existing quality assurance and repair workflows, allowing for real-time damage assessment and immediate corrective actions. Thus, these applications hold significant potential for improving efficiency and reducing costs in industries reliant on electronic components.

However, deep damage detection in PCBs through mobile devices presents several challenges. For effective image analysis on these devices, it is essential to maintain high image quality, including factors like focus, illumination, and proper alignment of the object being analyzed [9]. Manual image acquisition involves the user directing the camera toward the object of interest, which can lead to inconsistencies and variations in image quality [10]. Moreover, the inherent portability and use in varied environments pose challenges such as motion blur and inconsistent lighting, which can degrade image quality.

One significant challenge in PCB damage detection is that, due to the numerous small components on PCBs, damages tend to be very subtle and small. Detecting such small objects is inherently difficult due to factors like low image resolution, noise, and insufficient detailed information about their appearance [11]–[13]. This challenge is exacerbated when photos are taken from a long distance away from the PCB [14]. Additionally, many deep object detectors have fixed resolutions for input images, typically lower than those of modern mobile device cameras. Consequently, images must be resized before inputting them into the detector, resulting in resolution loss. Apart from issues with small damages, the presence of objects and noise in the background of the image (i.e., regions not part of the PCB) can lead to erroneous damage detection.

To obtain optimized results with such applications, a user must take photos at an adequate distance and minimize the presence of background in the photo. Thus, the detection performance in mobile applications is highly dependent on the user's ability to take adequate photos without assistance [10]. In contrast, assisted capture employs algorithms to guide users in adjusting the image, ensuring the object is centered and properly scaled. This approach can significantly enhance image quality and improve object detection accuracy in mobile applications. [5], [15].

Considering the provided context, we introduce an assisted capture method designed to aid users in PCB damage detection applications using mobile devices, named **PCBShot**. Our method employs classical image processing techniques to segment the PCB, guiding users to position the board within a designated area displayed in the camera preview on the mobile device screen. Once the board fits within the guideline, a photo is automatically taken. Subsequently, the background is cropped, and the image is divided into four quadrants, each independently processed by a trained deep damage detector. This approach not only minimizes background interference through cropping but also enhances the size and detail of small damages in the quadrants, which would otherwise be obscured due to resolution constraints of object detection models. While initially developed for PCB damage detection, this solution holds potential for broader mobile applications in scenarios requiring top-down photography of different objects.

We assess the efficacy of PCBShot using a real-world industrial mobile application deployed in repair centers of a multinational electronics manufacturer. We compare the detection performance between images obtained with our method and those captured without additional assistance. Our findings clearly demonstrate that our method significantly outperforms manual capture, underscoring its effectiveness.

The remainder of this work is organized as follows: Section II presents related work that supports this study. Section III describes our proposed method. Section IV outlines the methodology of our work. Section V presents the results of our experiments. Finally, Section VI provides the conclusion and future directions.

II. Related Works

Many works found in literature propose mobile applications for image analysis, harnessing the portability, connectivity and computing power of modern smartphones. Mobile applications facilitate immediate analysis and actionable insights on-site, improving efficiency and reducing the time and labor required for manual inspections. Particularly, the task of object detection enables numerous practical and innovative solutions across various domains.

In the domain of agriculture, various solutions have been proposed for crop disease detection [4], [16], pest detection [17], and growth monitoring [18]. The portability of mobile devices ensures that these technological benefits are accessible even in remote or underserved areas. In the healthcare domain, numerous solutions assist in the image-based diagnosis of pathologies such as skin cancer [19], cataracts [20], and oral cavities [21], providing valuable support by enabling the rapid identification of medical issues without the need for specialized equipment. In industrial scenarios, mobile device-based object detection has been applied for safety monitoring, equipment tracking, and quality control [22].

It has been demonstrated that image quality aspects such as focus, resolution, non-uniform lighting, viewing distance, and noise affect the performance of deep object detection [23], [24]. Since mobile devices are predominantly used handheld and in unconstrained scenarios, such image quality issues are likely to arise during photo capture [9]. Despite the increasing number of mobile image processing applications, the volume of research papers addressing the challenges involved in the image acquisition stage for such applications remains limited.

In [1], an approach is proposed that utilizes pose estimation methods to guide users in correctly positioning their devices for image capture. The paper also emphasizes the importance of integrating sensor data and positioning information for relative motion estimation to improve the image analysis of documents on mobile devices. The automation of capture relies on device-object alignment, image quality, motion, and environmental factors.

A method for automated image focus assessment is proposed in [5] for segmenting dermatological lesions using a mobile application. The results demonstrated that the method not only effectively distinguishes focused from non-focused images but also enables real-time processing and provides user feedback. This capability significantly enhances the application's accuracy during operation.

The research in [25] presents a smartphone image acquisition application for remotely monitoring vineyard insect traps and assessing pest infestations. The proposed methodology addresses various aspects of image capture quality and suitability, including focus validation, shadow and reflection detection, trap type identification, trap segmentation, and perspective correction.

Finally, in [26], an investigation comparing fixated ultrasonography with freehand acquisition is conducted to evaluate image quality and measurement repeatability. The research offers quantitative evidence that fixated probe acquisition can enhance both image quality and measurement consistency in musculoskeletal sonography during exercise.

To the best of our knowledge, our work is the only one proposing a method to assist in capturing high-quality images for PCB damage detection. Our approach addresses specific issues identified during the use of a real-life mobile application. While related works have focused on challenges such as focus, lighting, and stabilization in handheld photo capture, our method emphasizes enhancing object framing, background removal, and preserving resolution.

III. PCBShot

PCBShot is an automatic capture method designed to assist users of a real-world mobile application of PCB damage detection used by technicians of a large hardware company in its inspection process at repair centers. It features a series of image processing algorithms to automatically enable the image acquisition process, without requiring a user to capture the images manually.

During the image capture process, our method continuously process each frame in a loop while the user adjusts the position of the camera towards the PCB. The user must fit the PCB in an on-screen guideline at a minimum distance, which is measured by the image processing steps. Once the PCB is correctly positioned, the image is automatically captured. Then, the region of the board is cropped and sliced into four quadrants. The four images are then sent to an object detection model.

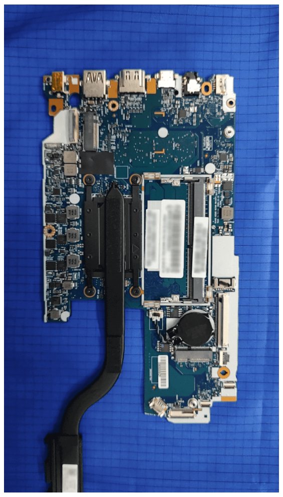
The rationale behind PCBShot is twofold: firstly, to enhance sensitivity by instructing users to capture photos at optimal distances and preserving damage resolution through slicing. Otherwise, resolution would degrade, as object detection models typically require input images with fixed resolutions lower than those of modern mobile device cameras. Secondly, cropping aims to eliminate false detections caused by background interference. Figure 1 illustrates the potential impact of these issues on performance.

Our method relies solely on classical image processing techniques instead of learning-based methods for three key reasons. These techniques typically have a lower computational cost, making them suitable for mobile devices with limited processing power. Additionally, they avoid the need for data collection and labeling, which can be expensive. Finally, the method has been effective in its specific use case, serving as a reliable baseline.

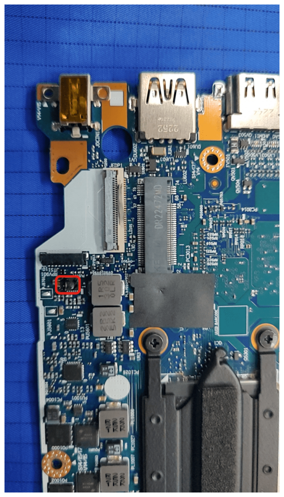
Below is a detailed description of each step as illustrated in Figure 2.

1) **Image received from camera:** The application begins with the reception of an image captured by the camera sensor, that is displayed in the camera preview. This image is the primary input for the processing system. A virtual reference guideline is displayed on-screen that a user must fit the PCB in.
2) **Grayscale:** After reception, the color image is converted to a grayscale image. This conversion is essential to reduce data complexity and facilitate subsequent processing.
3) **Gaussian blur:** Next, the grayscale image undergoes a Gaussian blur filter. This filter helps to smooth the image, reducing noise and enhancing the main contours [27].
4) **Morphological dilatation:** Morphological dilation is a process used to expand the regions of interest or objects present in the image [28]. Dilation helps to reduce noise by filling in small holes or gaps in the objects within the image. This is particularly useful in images of PCBs, where objects have irregular boundaries or small gaps.
5) **Otsu thresholding:** The smoothed and dilated image is then subjected to the Otsu thresholding method [29]. This technique segments the image into areas of interest by finding an optimal threshold value that minimizes the intra-class variance of the black and white pixels. This creates a binary image where pixels are classified as either objects or backgrounds.
6) **Find Contours in image:** Contours can be understood as a curve of continuous points with the same color or intensity. The contours of objects in the image are then found through the contour detection process proposed by [30].
7) **Obtain Bounding Boxes (Bbox) around detected contours:** Next, each of the found contours has its corresponding bounding box determined [31].
8) **Find bbox with largest area:** The largest contour is then identified and segmented [32], and ideally, it corresponds to the PCB that will be further analyzed for any damage.
9) **PCB is fully inside guideline?:** Check the acceptance conditions for automatic image capture. If the bounding box with the largest area is completely contained within the predefined guideline and the area of the detected bounding box exceeds a certain threshold, then, proceed to the next step. If not, the process returns to Step 1, as the camera is continually showing images. The threshold was determined empirically through try and error to be 65% of the guideline area. This value ensures a minimum distance of good quality is maintained, while also providing a margin that makes the task manageable for the user.
10) **Capture is triggered:** The capture is triggered, and the image is saved.
11) **Crop:** The image is cropped into the reference guideline, thus reducing the presence of strange objects in the background that create noise.
12) **Slice:** Next, the image is sliced into four quadrants. This step is necessary because the damage detection model used in the application is designed to work with a fixed input resolution, optimized for accuracy and communication constraints. Since this resolution is lower than that of typical mobile device cameras, sending the entire image to the model results in a loss of detail. By slicing the image into quadrants, the original resolution is preserved, which enhances detection performance.
13) **Object detection:** Each of the four resulting images from the previous step is sent to our trained damage detection model.

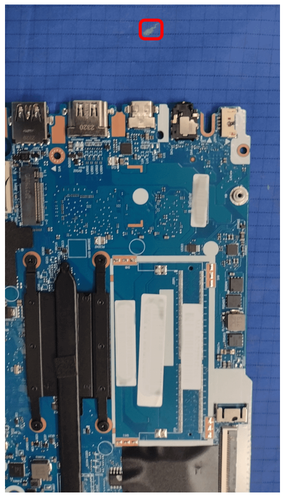
With this method, we anticipate higher photo quality, en-



(a) The detector fails to detect a burned component when the photo is taken far from the PCB.



(b) When a photo of the same PCB in Figure 1a is taken closer, the burned component is successfully detected.



(c) The detector incorrectly detected a scratch in the background as damage.

Fig. 1: Examples of image acquisition issues that are mitigated by PCBShot.

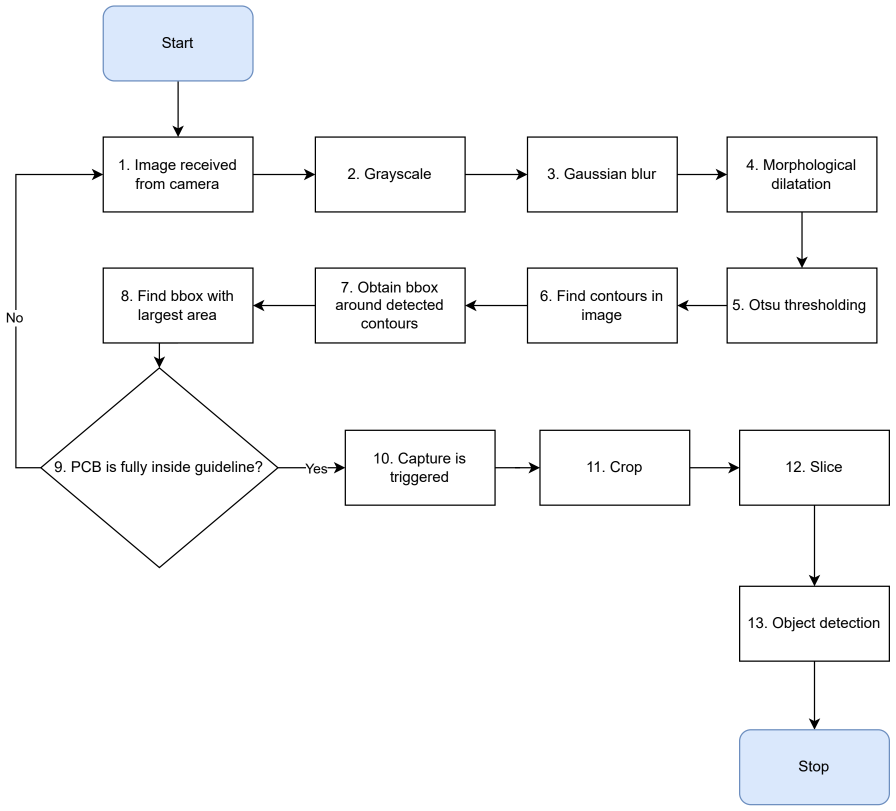
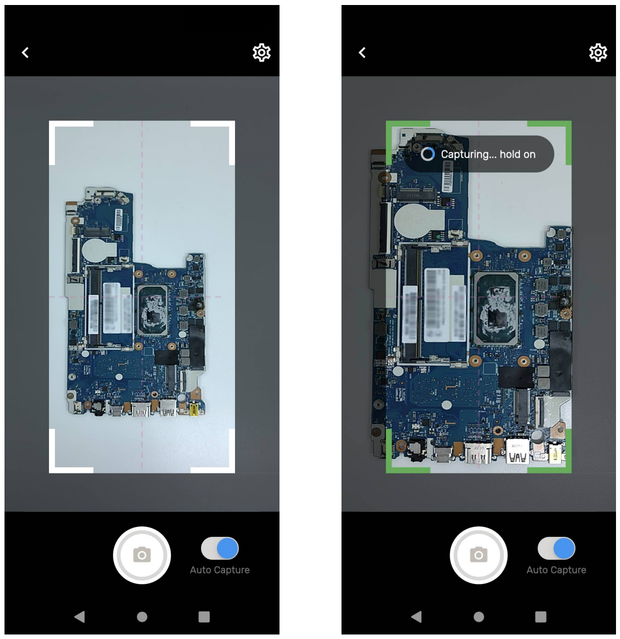

Start
1. Image received from camera
2. Grayscale
3. Gaussian blur
4. Morphological dilatation
5. Otsu thresholding
6. Find contours in image
7. Obtain bbox around detected contours
8. Find bbox with largest area
9. PCB is fully inside guideline?
No
Yes
10. Capture is triggered
11. Crop
12. Slice
13. Object detection
Stop

Fig. 2: Flowchart - PCBShot.

hancing the performance of deep learning models in detecting damage on PCBs. Figure 3 shows the visual feedback during the application of PCBShot. While this solution was designed with a specific mobile application in mind, it could potentially be applied to other image processing applications for PCB damage detection or even in different fields. This includes scenarios where photos are taken with mobile devices from a top-down view of different objects, such as other electronic devices or documents.



(a) Guideline to assist with the correct alignment of the PCB.

(b) Board identified, capturing, and removing the background.

Fig. 3: Visual feedback on the operation of our automatic capture method.

A. Limitations

PCBShot relies on classical image processing techniques that have some limitations that users should be aware to achieve optimal results. The method is highly dependent on PCB segmentation using Otsu thresholding. If the thresholding fails to effectively separate the PCB from the background, the method will not work. This issue can arise if the PCB is placed on a surface with low contrast. Additionally, the method assumes the PCB is the largest object in the scene, so it will also fail if larger objects are present. Lastly, there is a possibility that the slicing step could split a damaged area, potentially leading to misdetection due to information loss. However, in all the experiments we conducted, no misdetection occurred when damage was split.

IV. Materials and Methods

We evaluate our method by using a real-life mobile application of deep damage detection in PCBs. We compare the results obtained with photos of damaged PCBs captured with PCBShot and photos of the same PCBs captured without additional assistance, under the same lighting, focus, and background conditions. We then evaluate the detection performance of the obtained results.

A. Mobile Application of Damage Detection in PCBs

The development of PCBShot arises from a practical need observed during the use of a mobile application for detecting damage to PCBs. This application is used by technicians at repair centers of a large electronics manufacturer, during inspection of PCBs in warranties programs. In this application, when the user takes a photo of a PCB, this image is sent to a remote deep object detector model through an API.

All images sent to the API are resized to HD resolution (1280 x 720 pixels). Through empirical evaluation, it was found that this resolution presents an optimum trade-off between detection performance and computational and communication costs. After the model processes the image, the result is sent back to the user, displaying the detected damage on the screen and generating reports.

The deep learning architecture used in this mobile application for object detection is Mask R-CNN [33] with the Swin Transformer [34] as the backbone. This architecture yielded superior results as reported in [8], which compared various deep learning architectures for PCB damage detection.

The dataset used to train the model consists of a collection of 4101 photos of damaged PCBs, collected from repair centers worldwide. The photos were taken with smartphone cameras for registration of damage in the warranty inspection process, without a rigorous standard procedure. The photos were captured in diverse conditions of illumination, background, camera quality, distance, and position of PCBs, making it a diverse and challenging dataset. Each image in the dataset has at least one annotation made by experts, that corresponds to the location and class of damage, totaling 8642 annotations.

B. Materials

We implemented PCBShot with OpenCV [35] and Python for image processing. The images were captured using a Moto Edge 30 Pro smartphone. The primary camera configuration consists of three lenses: a 50 MP wide-angle lens with an aperture of f/1.8, a 50 MP ultra-wide-angle lens with an aperture of f/2.2, and a 2 MP depth sensor with an aperture of f/2.4.

We conducted our experiments in a set of 10 damaged notebook motherboards, with a total of 28 damage distributed among them. Five of these damage were inflicted by regular use, while the remaining were induced intentionally for these experiments. The induced damage was carefully performed to be similar to regular damage. The types of damage include burns, cracks, scratches, liquid spillage, mold, and corrosion.

We cataloged every damage and categorized them by type and size. We divided the damage into small ($<$ 1 cm) and non-small ($>=$ 1 cm). There are a total of 15 small damages and 13 non-small damages. We are particularly interested in the performance of the small damage, as our method has features that may improve the detection of these kinds of damages.

C. Capture Methods

We compared the automatic capture of PCBShot with two approaches of manual capture at the distances of 20cm and 30cm. For each board, we take photos using each of the capture methods, at the same conditions of lighting, focus, and background. As the background found in real-life repair centers can be noisy and diverse, at each PCB we use a different surface as a background that replicates real-life scenarios. We use 5 types of background, each used in two PCBs: the plain white surface of the workbench; the surface of the workbench with tools, keyboard, and mouse near the PCB; the surface of the workbench with the edge of the workbench visible; bubble wrap; and anti-static bag. All these scenarios were found in real-life.

1) Manual Capture: Manual capture is a very simple process. Using the smartphone's native camera, the photo can be taken manually by the user. The mandatory requirement is that the camera be positioned above the board, parallel to it, and the capture must be able to register the entire PCB.

We conducted two manual image capture processes: one at a distance of 20cm above the board's surface (M20) and another at a distance of 30cm from the surface (M30). We take these two approaches to better understand the impact of the distance on the detection performance. The distance of 20cm is usually enough to frame the entire PCB in the photo and including minimal information of the background, thus increasing the probability of true positives and decreasing the ratio of false positives due to background interference. Therefore, is a more adequate distance. However, it was observed that users usually take photos at a distance of 30cm, which results in lower performance.

Our method is devised to assist users in taking pictures at optimum distances, but also conduct further processing by cropping the background and slicing the image in four. By experimenting with these two manual approaches, we compare PCBShot with a scenario where the user takes a photo from an inadequate distance (M30) and a scenario where the user takes a photo at a good distance (M20). In the last scenario, the distance should be similar to the PCBShot, so we evaluate the additional processing steps of crop and slice. Figure 4 illustrates these scenarios.

After the image is captured, the captured image is sent to the damage detection model through the aforementioned API, without undergoing pre-processing. The model then performs inference on the received image and displays the detected damage results for this image. We manually identify the true positives (TP), false positives (FP), and false negatives (FN), and record them in a table according to each board that already has its damages pre-cataloged. This process is repeated for all images of boards captured manually during the experiment.

2) PCBShot: Our method was implemented in the mobile application to automatically capture PCBs on the workbench. The user only needs to hold the camera and, if our image processing verifies that the image meets the acceptance conditions, the capture is performed automatically, reducing the user's effort in obtaining a good image capture of the PCBs. We use the application to take photos with our method of each PCB and record the results of TP, FP, and FN. As with the manual capture method, the results of TP, FP, and FN are also recorded in a table for each board according to their ground truths.

D. Evaluation Metrics

To evaluate the performance of PCBShot relative to manual capture, we use the well-known Precision, Recall, and F1-Score metrics.

Precision measures the proportion of true-positive predictions among all positive predictions made by the model. Precision can be obtained using the formula shown in Equation (1):

$$Precision = \frac{\text{True Positive}}{\text{True Positive + False Positive}}. \tag{1}$$

Recall is a metric that measures the ability of a model to correctly identify all relevant objects within an image. It focuses on the proportion of truth-positive detection of all the present objects. Recall can be obtained using the formula shown in Equation (2):

$$Recall = \frac{\text{True Positive}}{\text{True Positive + False Negative}}. \tag{2}$$

F1-Score is the harmonic mean of precision and recall, providing a balance between both. The formula to F1-Score is shown in Equation (3):

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}. \tag{3}$$

V. Results

Table I summarizes our experimental results regarding the F1-Score metric. It can be seen that our method surpasses both manual capture methods by a large margin. It improves the performance by 66% when compared with manual capture at a distance of 30cm and by 25% when compared with manual capture at a distance of 20cm. We highlight that it was observed that photos taken at distances near 30cm occurred frequently during the use of the application before the implementation of our method. In this sense, PCBShot guides the users to take photos at adequate distances, that are near 20cm. However, even though the M20 method captured photos at a adequate distance, our method still surpass it in performance due the crop and slice operations.

As mentioned before, the deep object detectors used in the experimented mobile application receives input images at a fixed HD resolution. The mobile camera used in the experiment has Full-HD resolution, so the images captured with the manual capture methods must be resized to fit the HD resolution, resulting in proportional a loss of resolution of 55.56%. With the PCBShot method, each slice can be

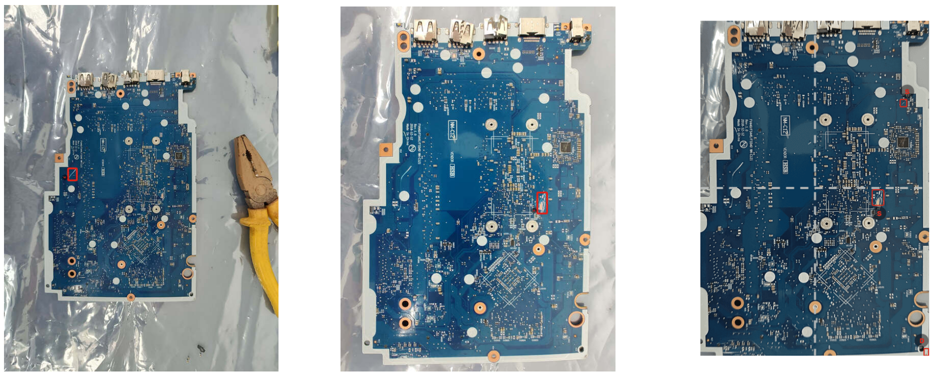

(a) Capture with M30. (b) Capture with M20. (c) Capture with PCBShot.

Fig. 4: Examples of each method of capture, applied in the same PCB. The damage detection was also performed in each scenario. The PCBShot method in this PCB detected 3 of the 5 existing damage in the board, while the other methods detected one each.

TABLE I: Result of F1-Score for the three experimented methods. Our method surpass the manual capture from far and close distances by a large margin.

Capture Method	F1
M30	0.439
M20	0.583
PCBShot	**0.729**

processed in its full resolution, without loss of information, which resulted in the observed performance.

We also compared the Recall of each method, grouping by the size of the existing damages. These results are presented in Table II. Once again, the PCBShot method surpassed both manual methods in all scenarios. In particular, small damages are more challenging to all methods, and in that scenario, our method achieved the largest improvement over the others, with a proportional difference of 200% and 80% for the methods M30 and M20 respectively.

TABLE II: Recall by capture method and by size of the damages.

Damages		Recall		
Size	Count	M30	M20	**PCBShot**
Non-Small ($>$ 1cm)	13	0.461	0.615	**0.923**
Small ($\leq$ 1cm)	15	0.200	0.333	**0.600**
Total	28	0.321	0.464	**0.750**

Finally, we compare the precision of each metric, which can be seen in Table III. A significant increase in recall often accompanies a decrease in precision. Notably, the M20 method achieved the highest precision, surpassing PCBShot by 10%. The M30 method performed comparably to our approach. This could be attributed to resolution loss in manual methods, inadvertently filtering out noise by downscaling textures that might trigger false positive detections, thereby reducing detector sensitivity. Despite a slight decrease in precision, the significantly higher F1 score and recall of our method indicate very favorable results.

It is important to emphasize that in PCB damage detection, maximizing recall ensures that damages aren't overlooked, thereby maintaining product quality, preventing costly recalls, and complying with strict regulatory standards. False positives are typically less problematic since users can spot and correct detected damage, whereas false negatives may go unnoticed. While precision remains crucial, the potential impact of undetected defects makes recall the primary metric in this scenario. Therefore, we consider the resulting balance between precision and recall crucial for ensuring robust product integrity and operational excellence.

TABLE III: Results of precision of each acquisition method.

Capture Method	Number of Predictions	Precision
M30	13	0.692
M20	14	**0.785**
PCBShot	31	0.709

VI. Conclusion

This work presents an application for the automatic capture of PCB photos using computer vision and image processing methods to improve the image quality in detecting damage to PCBs, being useful in electronic manufacturer warranty programs, manufacturing quality, and general repairs. PCB damage detection is a task hindered by small objects, as many damages can be very tiny, such as damage to integrated circuits, capacitors, small connectors, CPU pins, and other small components, or small surface damages like scratches on the board. Another factor that hinders damage detection in PCB is noisy environments such as background noise, causing false detection. Experiments with our application show that our method can capture higher-quality images due to framing assistance with image processing methods, eliminating noisy backgrounds and preserving resolution.

In future work, we aim to improve our automated capture method to enhance the quality of the images. One potential approach is to implement techniques that improve image focus during capture, as blurriness can hinder damage detection on PCBs. We also plan to explore machine learning-based methods for more robust PCB segmentation across varying scenarios. Additionally, we intend to evaluate our method in detecting damage on other objects, such as notebooks, keyboards, computer monitors, and other electronic devices.

Acknowledgment

This research received partial funding from Lenovo as part of its R&D investment under the Brazilian Informatics Law.

References

[1] K.-F. Henning, A. Fritze, E. Gillich, U. Mönks, and V. Lohweg, "Stable image acquisition for mobile image processing applications," in *Digital Photography XI*, vol. 9404. SPIE, 2015, pp. 177–186.

[2] A. M. Da Costa, A. O. De Sà, and R. C. Machado, "Data acquisition and extraction on mobile devices-a review," in *2022 IEEE International Workshop on Metrology for Industry 4.0 & IoT (MetroInd4. 0&IoT)*. IEEE, 2022, pp. 294–299.

[3] J. Lei, X. Gao, Z. Feng, H. Qiu, and M. Song, "Scale insensitive and focus driven mobile screen defect detection in industry," *Neurocomputing*, vol. 294, pp. 72–81, 2018.

[4] A. Picon, A. Alvarez-Gila, M. Seitz, A. Ortiz-Barredo, J. Echazarra, and A. Johannes, "Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild," *Computers and Electronics in Agriculture*, vol. 161, pp. 280–290, 2019.

[5] D. Moreira, P. Alves, F. Veiga, L. Rosado, and M. J. M. Vasconcelos, "Automated mobile image acquisition of macroscopic dermatological lesions." in *HEALTHINF*, 2021, pp. 122–132.

[6] V. A. Adibhatla, H.-C. Chih, C.-C. Hsu, J. Cheng, M. F. Abbod, and J.-S. Shieh, "Defect detection in printed circuit boards using you-only-look-once convolutional neural networks," *Electronics*, vol. 9, no. 9, 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/9/1547

[7] J. P. Santiago, V. Farias, L. Sena, J. P. P. Gomes, and J. Machado, "Real-time detection of customer-induced damage in printed circuit boards using mobile devices and YOLO detectors," *Learning & Nonlinear Models*, vol. 22, no. 2, pp. 17–31, 2024.

[8] D. Alves, V. Farias, I. Chaves, R. Chao, J. P. Madeiro, J. P. Gomes, and J. Machado, "Detecting customer induced damages in motherboards with deep neural networks," in *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2022, pp. 1–8.

[9] C. Morikawa, M. Kobayashi, M. Satoh, Y. Kuroda, T. Inomata, H. Matsuo, T. Miura, and M. Hilaga, "Image and video processing on mobile devices: a survey," *the visual Computer*, vol. 37, no. 12, pp. 2931–2949, 2021.

[10] L. Zu, Y. Zhao, J. Liu, F. Su, Y. Zhang, and P. Liu, "Detection and segmentation of mature green tomatoes based on mask r-cnn with automatic image acquisition approach," *Sensors*, vol. 21, no. 23, p. 7842, 2021.

[11] Y. Liu, P. Sun, N. Wergeles, and Y. Shang, "A survey and performance evaluation of deep learning methods for small object detection," *Expert Systems with Applications*, vol. 172, p. 114602, 2021.

[12] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1222–1230.

[13] Y. Bai, Y. Zhang, M. Ding, and B. Ghanem, "Sod-mtgan: Small object detection via multi-task generative adversarial network," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 206–221.

[14] L. Cabral, V. Farias, L. Sena, I. Chaves, J. P. Pordeus, J. P. Santiago, D. Sá, J. Machado, and J. P. Madeiro, "An active learning approach for detecting customer induced damages in motherboards with deep neural networks," *Learning & Nonlinear Models*, vol. 21, no. 2, pp. 29–42, 2023.

[15] J. Kaur and W. Singh, "Tools, techniques, datasets and application areas for object detection in an image: a review," *Multimedia Tools and Applications*, vol. 81, no. 27, pp. 38 297–38 351, 2022.

[16] A. A. Ahmed and G. H. Reddy, "A mobile-based system for detecting plant leaf diseases using deep learning," *AgriEngineering*, vol. 3, no. 3, pp. 478–493, 2021.

[17] J.-W. Chen, W.-J. Lin, H.-J. Cheng, C.-L. Hung, C.-Y. Lin, and S.-P. Chen, "A smartphone-based application for scale pest detection using multiple-object detection methods," *Electronics*, vol. 10, no. 4, p. 372, 2021.

[18] T. Napier and I. Lee, "Using mobile-based augmented reality and object detection for real-time abalone growth monitoring," *Computers and Electronics in Agriculture*, vol. 207, p. 107744, 2023.

[19] C. A. Hartanto and A. Wibowo, "Development of mobile skin cancer detection using faster r-cnn and mobilenet v2 model," in *2020 7th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*. IEEE, 2020, pp. 58–63.

[20] V. Agarwal, V. Gupta, V. M. Vashisht, K. Sharma, and N. Sharma, "Mobile application based cataract detection system," in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*. IEEE, 2019, pp. 780–787.

[21] P. S. SM, M. Shariff, D. Subramanyam, M. Varun, K. Shruthi, and A. Poornima, "Real time oral cavity detection leading to oral cancer using cnn," in *2023 International Conference on Network, Multimedia and Information Technology (NMITCON)*. IEEE, 2023, pp. 1–7.

[22] N. Rane, "Yolo and faster r-cnn object detection for smart industry 4.0 and industry 5.0: applications, challenges, and opportunities," *Available at SSRN 4624206*, 2023.

[23] A. C. Bergstrom and D. W. Messinger, "Image quality and object detection performance of convolutional neural networks," in *Pattern Recognition and Tracking XXXIV*, vol. 12527. SPIE, 2023, pp. 159–177.

[24] Y. Hao, H. Pei, Y. Lyu, Z. Yuan, J.-R. Rizzo, Y. Wang, and Y. Fang, "Understanding the impact of image quality and distance of objects to object detection performance," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 11 436–11 442.

[25] P. Faria, T. Nogueira, A. Ferreira, C. Carlos, and L. Rosado, "Ai-powered mobile image acquisition of vineyard insect traps with automatic quality and adequacy assessment," *Agronomy*, vol. 11, no. 4, p. 731, 2021.

[26] H. M. Heres, M. Sjoerdsma, T. Schoots, M. C. Rutten, F. N. van de Vosse, and R. G. Lopata, "Image acquisition stability of fixated musculoskeletal sonography in an exercise setting: A quantitative analysis and comparison with freehand acquisition," *Journal of Medical Ultrasonics*, vol. 47, pp. 47–56, 2020.

[27] R. C. Gonzales and P. Wintz, *Digital image processing*. Addison-Wesley Longman Publishing Co., Inc., 1987.

[28] E. R. Dougherty, "An introduction to morphological image processing," in *SPIE*. Optical Engineering Press, 1992.

[29] N. Otsu *et al.*, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[30] S. Suzuki *et al.*, "Topological structural analysis of digitized binary images by border following," *Computer vision, graphics, and image processing*, vol. 30, no. 1, pp. 32–46, 1985.

[31] D. Yang, B. Peng, Z. Al-Huda, A. Malik, and D. Zhai, "An overview of edge and object contour detection," *Neurocomputing*, vol. 488, pp. 470–493, 2022.

[32] Z. Wang, E. Wang, and Y. Zhu, "Image segmentation evaluation: a survey of methods," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5637–5674, 2020.

[33] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017. [Online]. Available: http://arxiv.org/abs/1703.06870

[34] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[35] G. Bradski, "The opencv library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000.