p-ISSN 2085- 6989
e-ISSN 2654 - 4733
Elektron Jurnal Ilmiah Volume 14 Nomor 2 Desember 2022 **TERAKREDITASI RISTEKDIKTI No. SK : 28/E/KPT/2019**
DOI : https://doi.org/ 10.30630/eji.14.2.295

Pembuatan Alat Inspeksi Visual Jalur PCB menggunakan Pengolahan Citra

Rangga Ade Juliano[1*], Efrizon[2], Hendrick[3] Laxsmy Devy[4], Suryadi[5], Yul Antonisfia[6]
[1,23456] Jurusan Teknik Elektro Politeknik Negeri Padang, Jl. Limau Manih Padang, 25164, Indonesia
*email: efrizon@pnp.ac.id

Abstrak— Pada penelitian ini dikembangkan alat pemeriksa kecacatan pada PCB untuk menggantikan penglihatan manusia agar lebih mudah dan dapat menghemat biaya. Alat ini dilengkapi dengan bantuan Webcam Logitech c920 dan mikroprosessor Raspberry Pi 3b+ yang digunakan untuk menyimpan dan menjalankan program yang telah dibuat pada *software* pemrograman Phyton, sehingga alat ini dapat digunakan secara *portable*. Dengan kedua teknologi tersebut dapat dimanfaatkan *Image Processing* untuk mendeteksi objek dengan *library* OpenCv serta Google Colab. Alat pendeteksi kecacatan PCB dengan bantuan *Image Processing* menggunakan metode YOLO *Convolutional Neural Network* untuk membantu menentukan kerusakan jalur pada PCB. Algoritma *You Only Look Once* (YOLO) dengan lima klasifikasi pendeteksi yaitu *short, open circuit, missing hole, mouse bite,* dan *spur*. Dari hasil penelitian diperoleh hasil bahwa algoritma YOLO sudah dapat mendeteksi kelima klasifikasi tersebut dengan nilai mAP@0.5 *short* 90.67%, *open circuit* 97.86%, *Mouse Bite* 94.43%, *Missing Hole* 96.09%, dan *spur* 97.56%.

Kata Kunci: PCB, Webcam Logitech c920, Raspberry Pi 3b+, *Image Processing, YOLO CNN*

Abstract— In this study, a PCB defect checking tool was developed to replace human vision to make it easier and save costs. This tool is equipped with the help of a Logitech c920 Webcam and a Raspberry Pi 3b+ microprocessor which is used to store and run programs that have been created in the Python programming software, so this tool can be used portable. With these two technologies, Image Processing can be used to detect objects using the OpenCv library and Google Colab. The PCB defect detection tool with the help of Image Processing uses the YOLO Convolutional Neural Network method to help determine the path damage on the PCB. You Only Look Once (YOLO) algorithm with five detection classifications, namely short, open circuit, missing hole, mouse bite, and spur. The results showed that the YOLO algorithm was able to detect the five classifications with values of mAP@0.5 short 90.67%, open circuit 97.86%, Mouse Bite 94.43%, Missing Hole 96.09%, and spur 97.56%.

Keywords: PCB, Logitech c920 Webcam, Raspberry Pi 3b+, Image Processing, YOLO CNN

I. PENDAHULUAN

Perkembangan teknologi informasi saat ini telah berkembang sangat pesat yang memberikan manfaat besar kepada manusia, termasuk teknologi *image processing*. Teknik *image processing* ini dapat dikembangkan untuk ilmu yang lebih luas, salah satunya yaitu untuk mendeteksi kerusakan pada suatu benda, misal *printed circuit board* (PCB). Sistem inspeksi otomatis yang akurat terbukti membantu dalam pemeliharaan kualitas. Sistem seperti itu mengatasi keterbatasan inspeksi manual untuk sejumlah besar PCB. Inspeksi PCB visual otomatis dapat memberikan informasi cacat yang cepat dan kuantitatif dan karenanya dapat terbukti menjadi aset dalam proses manufaktur[1].

Ilmu yang berhubungan erat dengan penentuan kualitas produk, jenis dan kuantitas secara visual, bidang ilmunya adalah *computer vision*. *Computer Vision* umumnya membahas tentang klasifikasi objek dan deteksi objek pada gambar. *Computer Vision* yang digunakan umumnya terdari dari pengumpulan gambar, *pre-processing*, *training* dan *deploy*[2]. Setelah semua proses tersebut dilalui maka akan dihasilkan model yang akan digunakan untuk klasifikasi atau deteksi gambar yang baru. Pembuatan model didalam *computer vision* biasanya didasari pada metode *Convolutional Neural Network* (CNN).

Dengan metode pengolahan data atau citra yang tanpa pemrograman secara eksplisit membuat salah satu cabang *artificial intelligence*, yaitu *machine learning* memiliki fungsi yang lebih spesifik dalam mengolah data gambar, diantaranya dengan menggunakan pemprosesan secara *deep learning*[3].

Algoritma *You Only Look Once* (YOLO) merupakan suatu algoritma yang mendeteksi objek dengan membagi citra menjadi beberapa *grid*. *Feature map* dari keluaran YOLO menghasilkan bbox, skor objektif, dan skor kelas[4]. YOLO adalah salah satu metode deteksi objek tercepat dengan kinerja yang baik dan akurasi tinggi[5][6][7].

Beberapa penelitian yang berkaitan telah dilakukan sebelumnya, menggunakan metode *template matching* dan *euclidean distance* Nugroho,dkk[8] mengembangkan alat deteksi kerusakan jalur PCB, hasil penelitian menunjukkan alat dapat mendeteksi jalur putus pada *PCB*. Penelitian serupa juga dilakukan oleh Mustika Sonsank.dkk[9] mengembangkan

Received 14 November 2022; Revised 30 November 2022; Accepted 06 Desember 2022

aplikasi untuk mendeteksi cacat pelebaran dan penyempitan jalur lapisan tembaga yang mengakibatkan adanya jalur hubung singkat sehingga PCB tidak berfungsi, hasil aplikasi menunjukkan bahwa metode deteksi kerusakan trek PCB dapat mendeteksi dengan kerusakan *template matching* pada jalur PCB. Nur Kurniasari, dkk menggunakan metode *convolutional neural network* untuk mendeteksi jalur putus pada PCB [10] dengan 6 jenis kerusakan jalur PCB, hasil penelitian juga menunjukkan metode yang digunakan rata-rata 80% dapat mengidentifikasi jenis kesalahan.

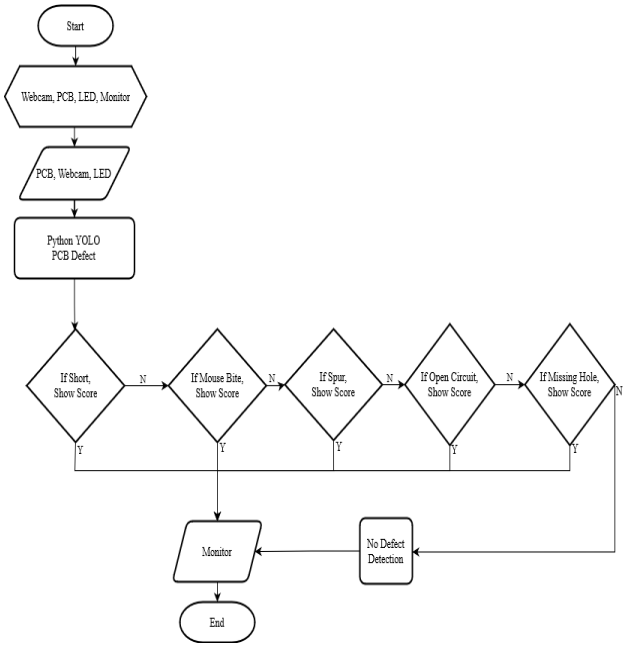
Pada penelitian ini kami membuat alat untuk deteksi kerusakan pada PCB untuk kegiatan pratikum pengawatan dan teknologi PCB, perbedaan dari penelitian sebelumnya alat ini dapat mengidentifikasi 5 kerusakan PCB. Miniaturisasi adalah kontribusi dari penelitian ini. Batasan masalah disini dengan studi lainnya adalah hanya bisa mendeteksi kerusakan pada jenis pcb polos. Diharapkan dengan adanya alat ini mahasiswa dan dosen dapat mengembangkan materi perkuliahan dan kegiatan praktikum dapat berjalan efektif.

II. METODE

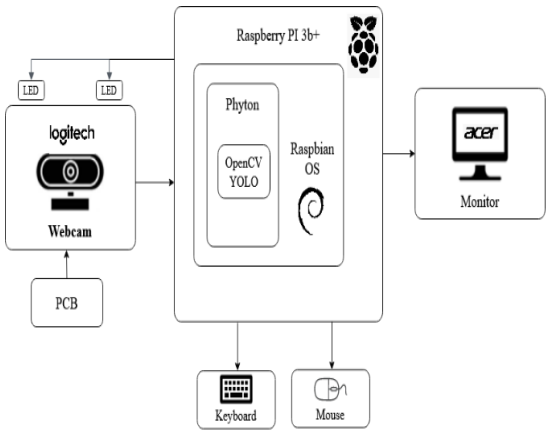
A. Perancangan Sistem

Secara keseluruhan alat ini tersusun atas bagian-bagian penting yang saling berhubungan satu sama lain yaitu perangkat keras (*hardware*) dan perangkat lunak (*software*). Kedua bagian ini harus saling sinkron satu sama lain agar maksud dan tujuan dari pembuatan alat ini tercapai dan sesuai dengan yang diharapkan. Bagian *hardware* terdiri dari perangkat Raspberry Pi 3b+, Webcam Logitech c920, LCD Monitor Acer. Bagian *software* terdiri dari pemrograman dan *interface*.

Alat pendeteksian kerusakan jalur PCB ini menggunakan Webcam yang akan diaktifkan menggunakan program python dengan bantuan OpenCV untuk memanggil Webcam yang terhubung dengan Raspberry Pi, adapun LED yang terhubung serial dengan USB untuk pencahayaan. PCB yang terletak dihadapan Webcam akan ditangkap gambarnya, kemudian diproses dan dideteksi jalur PCB oleh algoritma YOLO untuk menentukan kemungkinan adanya kecacatan atau kerusakan pada jalur PCB tersebut. Hasil deteksi kemudian akan ditampilkan pada monitor dalam bentuk tampilan UI. Flowchart sistem dan blok diagram ditunjukkan pada Gambar 1 dan Gambar 2.



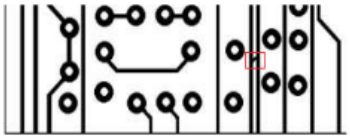
Gambar 1. *FlowChart* Sistem



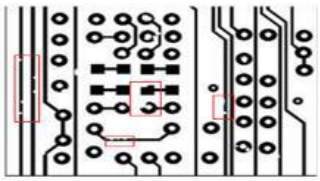
Gambar 2. Blok Diagram Sistem

B. Pembuatan *Dataset*

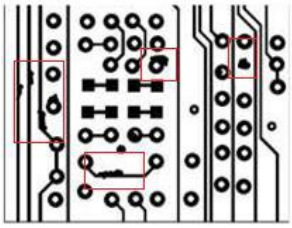
Pembuatan *dataset* dimulai dengan menentukan klasifikasi atau *class* yang akan dideteksi. Klasifikasi yang dibuat adalah *short*, *missing hole*, *spur*, *open circuit*, dan *mouse bite*.



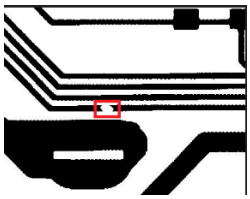
Gambar 3. *Short*



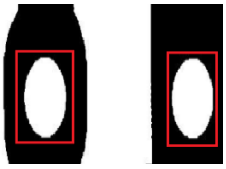
Gambar 4. *Mouse Bite*



Gambar 5. *Spur*



Gambar 6. *Open Circuit*



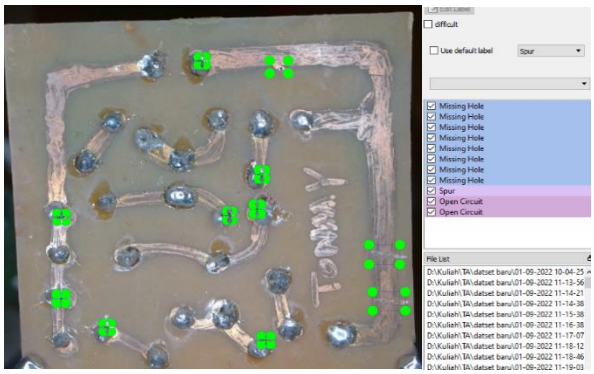
Gambar 7. *Missing Hole*

Pengumpulan data gambar sebanyak 260 gambar dengan berbagai posisi dan kumpulan PCB. *Dataset* yang dikumpulkan berupa gambar dengan format gambar keseluruhan memakai format (.jpg).

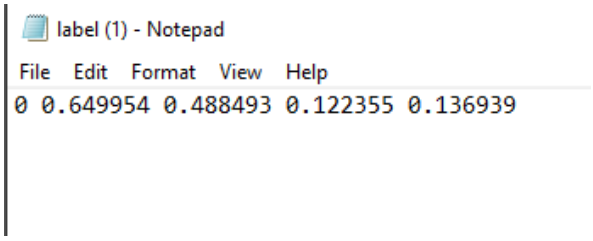
Gambar 8. *Dataset* gambar

C. Pelabelan

Proses pelabelan ini dimulai dengan membuat *bounding box* beserta nama kelas pada setiap objek. *Dataset* yang terkumpul dapat dilabelkan dengan kategori *short* sebanyak 43 label, *Mouse Bite* sebanyak 188 label, *Spur* sebanyak 264 label, *Open Circuit* 388 label, *Missing Hole* sebanyak 833 label. Pelabelan ini menggunakan format YOLO dimana data citra akan diubah menjadi data *array* yang nantinya akan digunakan untuk membuat algoritma YOLO.



Gambar 9. Pelabelan



Gambar 10. Data *Array*

D. *Training* Data

Training data ini dilakukan setelah didapatkan anotasi dari kelas-kelas objek yang dideteksi, maka digunakan Google Colab dan untuk lebih spesifik digunakan *yolo training* pada *network* dengan OpenCV. Metode konvolusi berulang pada YOLO ini menerapkan wilayah untuk melokasi objek yang dideteksi, yang berarti bahwa model diterapkan ke beberapa wilayah dalam sebuah gambar dan kemudian model menghitung skor ke gambar di beberapa lokasi dan skala daerah. Skor tinggi dari suatu gambar dianggap sebagai objek terdeteksi.

Di sisi lain, YOLO menggunakan total pendekatan yang berbeda alih-alih memilih beberapa wilayah, menerapkan jaringan saraf tunggal ke gambar penuh, memprediksi *bounding boxes*, dan probabilitas untuk setiap wilayah. *Bounding boxes* ini dibobot oleh probabilitas yang diprediksi dan probabilitas gambar yang tinggi dianggap sebagai objek yang terdeteksi. Hal ini dikarenakan YOLO memindai gambar hanya satu kali untuk membuat prediksi dibandingkan algoritma lain yang membutuhkan banyak pemindaian dan banyak waktu, jadi YOLO ini lebih cepat, efisien, dan praktis.

Fasilitas yang diperlukan pada proses *training* model ini adalah fasilitas GPU. *Framework* yang digunakan adalah *Darknet* dengan *pre-trained weight* yaitu *darknet53.conv.74*. Pada tahapan awal pelatihan, harus dilakukan konfigurasi awal pada beberapa aspek penting seperti *file Make* dengan mengaktifkan penggunaan pada OpenCV, GPU (*Graphics Processing Unit*), dan CUDNN. Konfigurasi selanjutnya adalah pemberian nilai parameter yang sesuai pada gambar 6. Hasil pelatihan model ini adalah *file* dengan format (.cfg) dan (.weights) yaitu *file* yang berisi urutan

konfigurasi dan bobot yang berkaitan dengan *Sequence Alignment and Modeling.*

```
!sed -i 's/batch=1/batch=260/' cfg/yolov3-tiny_training.cfg
!sed -i 's/subdivisions=1/subdivisions=65/' cfg/yolov3-tiny_training.cfg
!sed -i 's/max_batches = 500200/max_batches = 10000/' cfg/yolov3-tiny_training.cfg
!sed -i '610 s@classes=80@classes=5@' cfg/yolov3-tiny_training.cfg
!sed -i '696 s@classes=80@classes=5@' cfg/yolov3-tiny_training.cfg
!sed -i '783 s@classes=80@classes=5@' cfg/yolov3-tiny_training.cfg
!sed -i '603 s@filters=255@filters=30@' cfg/yolov3-tiny_training.cfg
!sed -i '689 s@filters=255@filters=30@' cfg/yolov3-tiny_training.cfg
!sed -i '776 s@filters=255@filters=30@' cfg/yolov3-tiny_training.cfg
```

Gambar 11. Konfigurasi yolov3-tiny

E. Evaluasi Model

Terdapat beberapa metoda untuk mengevaluasi kinerja sistem, yaitu *recall*, presisi, *F1 score*, *Intersection over Union*, *mean Average Precision*, dan Akurasi.

Recall didefnisikan sebagai rasio dari jumlah total contoh positif yang diklasifikasikan dengan benar dibagi dengan jumlah total contoh positif. *Recall* yang tinggi menunjukkan bahwa kelas dikenali dengan benar (*Fn* sedikit), seperti terlihat pada persamaan 1.

$$Recall = \frac{Tp}{Tp+Fn} \qquad (1)$$

Nilai presisi didapatkan dengan cara membagi jumlah total contoh positif yang diklasifikasikan dengan benar dengan jumlah total contoh positif yang diprediksi, seperti terlihat pada persamaan 2.

$$Presisi = \frac{Tp}{Tp+Fp} \qquad (2)$$

Dimana *True positive* (*Tp*) merupakan nilai aktual bernilai positif dan diprediksi positif juga, sedangkan *True negative* (*Tn*) merupakan nilai aktual bernilai negatif dan diprediksi negatif juga. *False positive* (*Fp*) merupakan nilai aktual bernilai negatif tetapi diprediksi positif, sedangkan *False negative* (*Fn*) merupakan nilai aktual bernilai positif tetapi diprediksi negatif.

Kondisi dimana *recall* tinggi dan presisi rendah artinya *detector* besar contoh positif dikenali dengan benar (*Fn* rendah) tetapi ada banyak positif palsu (*Fp* tinggi). Sedangkan kondisi *Recall* rendah dan presisi tinggi artinya kehilangan banyak contoh positif (*Fn* tinggi) dengan nilai positif palsu yang sedikit (*Fp* rendah).

F1 Score merupakan perbandingan rata-rata presisi dan *recall* yang dibobotkan sesuai dengan persamaan 3.

$$F1 = 2 \text{ x } \frac{Recall\ x\ Precission}{Recall + Precission} \qquad (3)$$

Intersection over Union (IoU) merupakan metrik evaluasi untuk mengukur keakuratan *detector* objek pada *dataset* tertentu. IoU dapat digunakan dengan ketentuan *bounding box* dari *ground-truth bounding box* pada *dataset* objek dan juga prediksi *bounding box* atau *detection results* pada *dataset* objek. IoU merupakan perbandingan antara *ground-truth bounding box* dengan *detection-results bounding box* pada model.

Mean Average Precision (mAP) merupakan nilai rata-rata dari *Average Precision* (AP) yang membentuk metrik evaluasi untuk mengukur kinerja dari sebuah deteksi objek.

Untuk mendeteksi atau melihat akurasi pada model yang telah dibuat akan diuji coba dengan 56 gambar acak dan akan di analisis dengan menggunakan persamaan dengan persamaan 4.

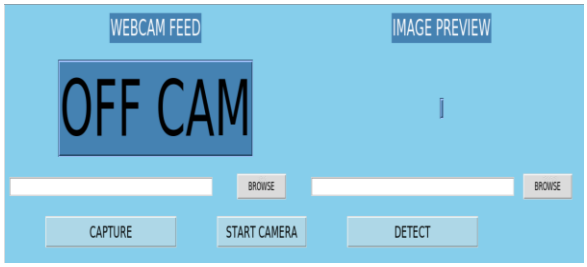
$$Akurasi = \frac{\text{Jumlah data yang benar}}{\text{Jumlah data keseluruhan}} \text{ x } 100\% \qquad (4)$$

F. Pemrograman dan Tampilan UI

Pemrograman Python dibuat pada platform Raspbian. Alasan menggunakan bahasa pemrograman Python adalah karena Python merupakan bahasa pemrograman yang mudah dipelajari dan memiliki struktur yang sederhana serta keyword yang sedikit. Selain itu juga mudah diaplikasikan karena penulisan sintaksnya lebih sederhana dibandingkan dengan bahasa pemrograman lainnya untuk masalah yang sama. *Library* yang digunakan dapat dilihat dibawah ini.

1. import cv2
2. import numpy as np
3. import tkinter as tk
4. from tkinter import *
5. from PIL import Image,ImageTk
6. from datetime import datetime
7. from tkinter import messagebox, filedialog

Tampilan UI dari pemrograman yang dibuat dapat dilihat pada gambar 7. Tampilan UI menggunakan pustaka standar dari Phyton yaitu Tkinter. Python bila dikombinasikan dengan Tkinter menyediakan cara cepat dan mudah untuk membuat aplikasi UI. Tkinter menyediakan antarmuka berorientasi objek yang kuat ke toolkit UI Tk.



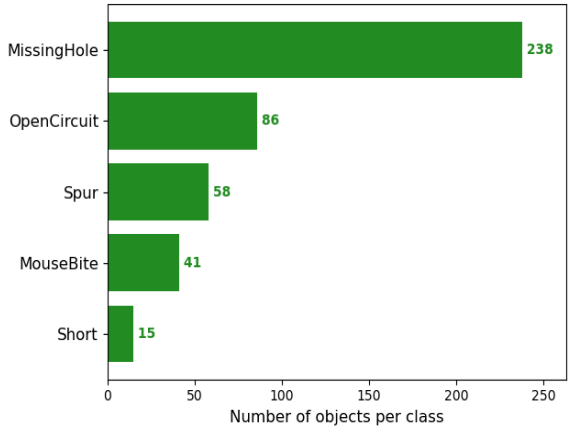
Gambar 12. Tampilan UI

III. HASIL DAN PEMBAHASAN

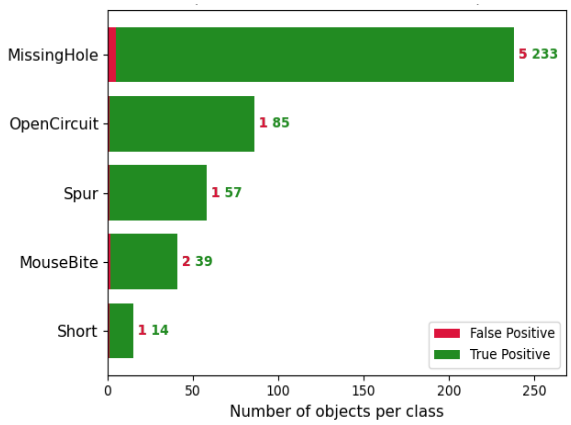
A. Evaluasi Model Kecacatan PCB

Pada gambar 8 menampilkan hasil deteksi objek/*ground truth* dari 56 gambar yang telah diuji dari 5 kelas, yaitu *Missing Hole* sebanyak 238, *Open Circuit* sebanyak 86, *Spur* sebanyak 58, *Mouse Bite* sebanyak 41, dan *Short* sebanyak 15. Pada gambar 9 menampilkan hasil objek deteksi, terdapat *False Positive* dan *True Positive* pada masing-masing kelas, yaitu pada *Missing Hole* dari 238 objek terdapat *False Positive* sebanyak 5 objek dan *True Positive* sebanyak 233 objek. Kemudian pada *Open Circuit* dari 86 objek terdapat *False Positive* sebanyak 1 objek dan *True Positive* sebanyak 85 objek. Setelah itu pada *Spur* dari 58 objek terdapat *False Positive* sebanyak 1 objek dan *True Positive* sebanyak 57 objek. Lalu, pada *Mouse*

Bite dari 41 objek terdapat *False Positive* sebanyak 2 objek dan *True Positive* sebanyak 39 objek, dan pada *Open Circuit* dari 15 objek terdapat *False Positive* sebanyak 1 objek dan *True Positive* sebanyak 14 objek.



Gambar 13. *Ground Truth*



Gambar 14. *Detection Results*

Pada tabel 1 ditunjukan hasil evaluasi model, dari *testing* 56 gambar dan hasil rata-rata akurasi mAP dengan *confidence* 0.5 didapat sebesar 95.32%.

Tabel 1. Evaluasi Performa Model Cacat PCB

Class	*Ground Truth*	*Detection Results*		*Average Precision* (%)	*Data Testing*	mAP@0.5 (%)
		False Positive	*True Positive*			
Open Circuit	238	5	233	97.86	56	95.32
Mouse Bite	86	1	85	94.43		
Missing Hole	58	1	57	96.09		
Spur	41	2	39	97.56		
Short	15	1	14	90.67		

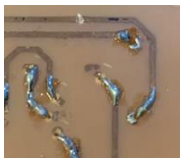
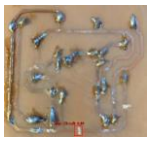
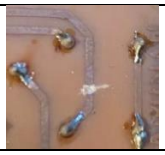
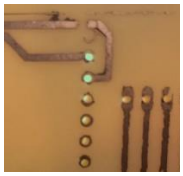
B. Uji Sistem

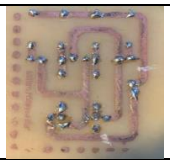
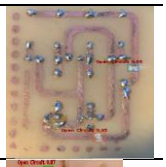
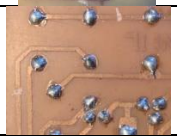
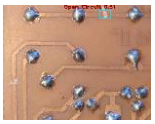
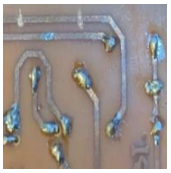
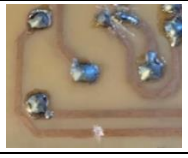
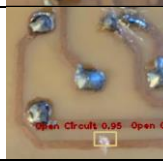
Data akan di ambil oleh Webcam yang diletakan di *Stand* Webcam dilihat pada gambar 10. Data yang telah diambil akan diektraksi dan diolah di Raspberry Pi dan akan ditampilkan pada monitor dalam bentuk UI.

Gambar 15. Tampilan Pengujian Sistem Alat

Lalu dilakukan sampling 25 data acak. Pengujian dilakukan secara langsung di kampus Politeknik Negeri Padang dan data *sampling* yang telah disiapkan, sebagai contoh disini ditampilkan 12 data gambar yang dideteksi, seperti yang ditampilkan pada tabel 2.

Tabel 2. Hasil Uji Sistem

No	Data Gambar Uji	Label (Nilai Confidence)	Hasil
1		*Open Circuit* 0.78, *Mouse Bite* 0.72, *Missing Hole* 0.98, *Missing Hole* 0.71, *Missing Hole* 0.52	
2		*Open Circuit* 0.81	
3		*Mouse Bite* 0.52, *Open Circuit* 0.59	
4		*Open Circuit* 0.82	
5		*Missing Hole* 0.88, *Missing Hole* 0.99, *Missing Hole* 0.95, *Missing Hole* 0.97, *Missing Hole* 0.99, *Missing Hole* 0.96, *Missing Hole* 0.99, *Missing Hole* 0.91 *Missing Hole* 0.97, *Missing Hole* 0.97	

6		*Open Circuit* 0.95, *Open Circuit* 0.55	
7		*Open Circuit* 0.87, *Open Circuit* 0.80, *Missing Hole* 0.98, *Missing Hole* 0.52	
8		*Open Circuit* 0.59, *Open Circuit* 0.60	
9		*Open Circuit* 0.51	
10		*Open Circuit* 0.97, *Open Circuit* 0.96	
11		*Spur* 0.91	
12		*Open Circuit* 0.95, *Open Circuit* 0.87	

Dari 12 contoh gambar diatas, data tabel 2 tersebut akan dilakukan evaluasi dengan persamaan-persamaan 1–4. Tabel 3 adalah hasil evaluasi dari data acak yang dilakukan.

Tabel 3. Hasil Evaluasi Data Acak

Data Acak	Presisi (%)	*Recall* (%)	*F1 Score* (%)	Akurasi (%)
25	58.7	65.4	61.7	85.45

Berdasarkan dari Tabel 3, dari 25 data acak yang diuji didapatkan nilai presisi 58.7%, recall 65.4%, F1 score 61.7%, dengan akurasi 85.45%.

IV. KESIMPULAN

Untuk pendeteksian kerusakan PCB dibangun dengan *Deep Neural Network* (DNN) dan algoritma YOLOv3 menggunakan *Jupyter Notebook* pada Google Colab. Kerusakan PCB dapat membaca ketika prediksi *confidence* > 4.9, dan tidak dapat membaca ketika *confidence* < 5.0. Untuk *dataset* terdiri dari total 260 gambar PCB dengan kategori *class short* sebanyak 43 label, *Open Circuit* sebanyak 388 label, *Spur* sebanyak 264 label, *Missing Hole* sebanyak 833 label, *Mouse Bite* sebanyak 188 label. Pada beberapa hasil perbandingan masih terdapat kotak yang dianggap cacat, hal ini disebabkan karena intensitas cahaya dan posisi letak PCB yang berubah-ubah pada waktu pengambilan gambar. Evaluasi *testing* dengan 56 gambar yang mirip dengan gambar pelabelan didapatkan hasil mAP@0.5 *short* 90.67%, *open circuit* 97.86%, *Mouse Bite* 94.43%, *Missing Hole* 96.09%, dan *spur* 97.56%.

Saran untuk pengembangan selanjutnya adalah dapat mengumpulkan *dataset* yang lebih banyak lagi agar proses pendeteksian lebih akurat lagi. Menggunakan peralatan tambahan, seperti *conveyor* untuk nantinya mejadi penyortir kerusakan-kerusakan PCB dan juga dapat membuat tambahan jenis kerusakan PCB lainnya.

REFERENSI

[1] 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 (Basel)*, vol. 9, no. 9, p. 1547, Sep. 2020, doi: 10.3390/electronics9091547.

[2] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2019). Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision*, *128*(2), 261–318. DOI: https://doi.org/10.1007/s11263-019-01247-4.

[3] J. T. Terpadu, I. Arifin, R. Fakhran Haidi, and M. Dzalhaqi, "PENERAPAN COMPUTER VISION MENGGUNAKAN METODE DEEP LEARNING PADA PERSPEKTIF GENERASI ULUL ALBAB," *Jurnal Teknologi Terpadu*, vol. 7, no. 2, pp. 98–107, 2021, [Online]. Available: https://journal.nurulfikri.ac.id/index.php/jtt

[4] JIWOONG, C., DAYOUNG, C., HYUN, K. & LEE, H.-J., 2019. Gaussian YOLOv3: An Accurate and Fast Object Detector Using Localization Uncertainty for Autonomous Driving. Seoul, IEEE International Conference on Computer Vision.

[5] FANG, W., WANG, L. & REN, P., 2020. Tinier-YOLO: A Real-Time Object DetectionMethod for Constrained Environments. IEEE Access, Volume 8, pp. 1935 - 1944.

[6] A. Albar, H. Hendrick, and R. Hidayat, "Segmentation Method for Face Modelling in Thermal Images," *Knowl. Eng. Data Sci.*, vol. 3, no. 2, p. 99, 2020, doi: 10.17977/um018v3i22020p99-105.

[7] R. Park and J. Jo, "Reference class-based improvement of object detection accuracy," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 10, no. 4, pp. 1526–1535, 2020, doi: 10.18517/ijaseit.10.4.12792.

[8] W. Nugroho, "Deteksi Kerusakan Jalur PCB (*Printed Circuit Board*) Menggunakan Metode Template Matching," 2014.

[9] M. Metode, P. T. Karya, and M. Nugraha, "Penerapan Metode Template Matching Dalam Menganalisa Cacat Pada Keping PCB" Jurnal Vokasional Teknik Elektronika & Informatika, vol. 1, no. 2, pp. 88–93, 2019.

[10] N. Kurniasari and J. P. Sugiono, "DETEKSI JALUR YANG TERPUTUS PADA RANGKAIAN LISTRIK DALAM PCB MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK (CNN)," 2021.