ELECENG 4OI6 - FINAL REPORT

Autonomous PCB Soldering Robot

Instructor: Dr. S. Shirani & Dr. C. Chen
C01 - T01 - S05

April 21 2025

Arji Thaiyib — thaiyiba — 400336020

Arjun Bhatia — bhatia32 — 400341269

Ahmad Choudhry — chouda27 — 400312026

Abdullah Hafeez — hafeea6 — 400306694

Mayar Aljayoush — aljayoum — 400324172

CONTENTS

List of Figures

I. INTRODUCTION

The rapid evolution of the electronics industry has escalated the demand for efficient printed circuit board (PCB) prototyping and small-scale production methods. While surface-mount technology (SMT) components remain the dominant form factor due to the ease of automation in assembly, through-hole (TH) components still remain relevant. TH technology offers higher mechanical robustness, thermal dissipation, and sometimes lower component cost. However, through-hole assembly is a labour-intensive process until large production volumes are reached. At small-scale production and prototyping levels, human engineers and technicians need to spend time hand-soldering, which increases cost and slows development cycles. Traditional manual through-hole soldering is also time-consuming and prone to human error, leading to inconsistencies in solder joints and reduced device reliability.

To meet these needs, a fully integrated solution is required—one that combines a programmable motion system based on a 3-axis CNC framework with advanced vision hardware and machine learning-based defect detection.

This project aims to increase the efficiency of a few key sectors of modern engineering and production. While high-volume manufacturing can be economically automated, this project targets low-volume manufacturing. One example is highly specialized board manufacturing, such as that for low-volume aerospace or medical devices requiring high-quality solder joints. Another example is startups that are producing novel products, which are still in initial deployment phases. These firms may not have the demand or capital to produce their designs, so in-house manufacturing is their most viable option. Additionally, modern shipping channels have enabled the growth of small, independent suppliers. These are mainly hobbyists who produce and sell unique designs out of their garages or basements.

If this product can be widespread, it serves as hope that small-scale PCB manufacturing becomes more accessible to more firms and individuals. There are hardly any consumer products in the modern age that lack some form of electronic subsystem, so this device will spur the development of new and interesting products as a whole.

II. TECHNICAL BREAKDOWN

Summary

The PCB soldering robot machine consists of five key modules which work in tandem to ensure precise movement around a target PCB, solder delivery, and accurate quality control. The first three components comprise the mechanical system, electronics, and firmware, making up the *electromechanical half*. This half performs soldering operations. The *quality control half* consists of the vision and machine learning modules. These take images of the board following all soldering and classify joints as good or bad quality.

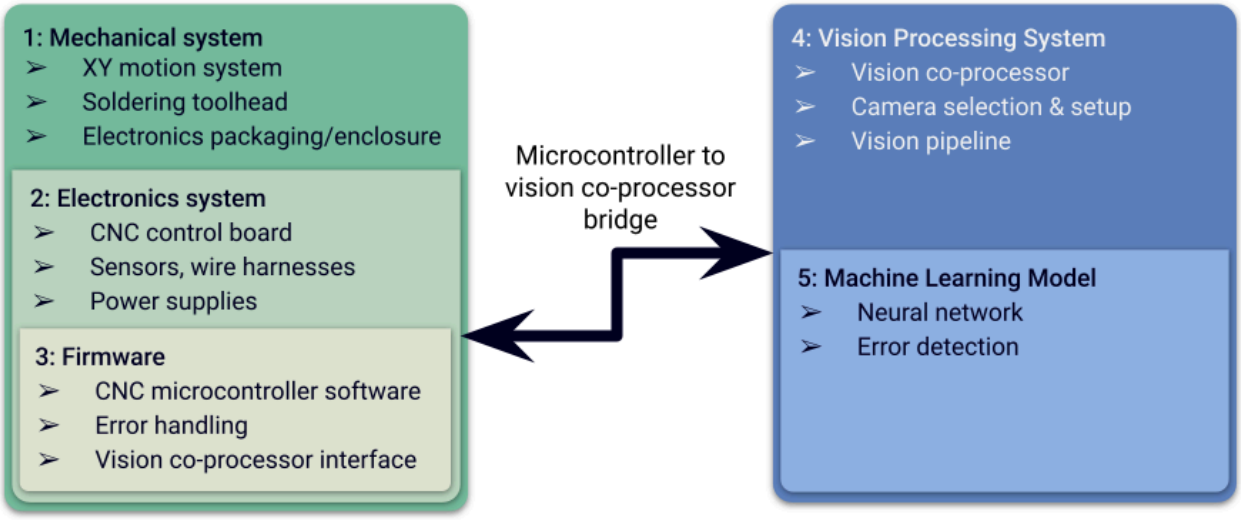

Figure 1: Block diagram of technical modules

The base of the machine is ***Module 1, the Mechanical System***, which consists of a core-XY style 2D motion system. This is similar to modern high-speed 3D printers, such as the BambuLab X1C or P1S [1]. The mechanical system also includes two more degrees of freedom, one for the z-axis that raises and lowers the soldering iron and one that spools and unspools solder wire onto the heated joint.

Since the mechanical system is powered by stepper motors, ***Module 2, the Electronics System*** is required to drive them. The electronics system centres around a custom PCB with a Pi Pico microcontroller and TMC2209 motor drivers. The whole system is powered by an off-the-shelf 24V power supply, and the PCB includes a 5V buck converter to provide power to logic-level components.

The Pi Pico contains and runs ***Module 3, the Firmware***, which controls all the motors, handles limit switch inputs, and runs the program which navigates the motion system to pin locations

and dispenses solder. Addressable UART protocol is used to set advanced configuration settings on the TMC2209 drivers, such as current limiting, microstepping, and stall detection.

The initial project scope included a serial bridge between the Pico to the vision coprocessor, which would handle error codes and try to correct quality issues on the fly. However, this was retired due to time and technical challenges.

On the quality control side, ***Module 4, the Vision System*** is responsible for all imaging hardware used in solder joint inspection. The setup features a Raspberry Pi 4 as the vision processor, interfacing with the Pi High-Quality Camera via the CSI port. Paired with a 100x microscope lens and custom LED fill lighting, this module enables consistent, high-resolution image capture synchronized with the robot's operation.

Finally, ***Module 5, Machine Learning***, consists of a convolutional neural network that runs on the Pi 4. The images captured by the vision system are processed, and joints are identified. Each solder joint is classified as good, missing or bad. This information is then displayed to a monitor so that a human technician can perform the necessary manual rework. A future iteration of the machine can pass this information back to the Pi Pico, and the firmware will command the electromechanical system to perform corrective actions where possible.

Below is a summary of the leaders, approximate share of each module, and actual labour hours.

Table 1: Project work summary

Module	Estimated Weight	Leader	Additional Contributors	Actual Total Labour (hrs)
Mechanical System	25%	Arjun		190
Electronics system	20%	Abdullah	Arjun	174
Firmware	15%	Mayar	Arjun	125
Vision System	15%	Arji	Ahmad	80 + 60
Neural Network	25%	Ahmad	Arji	130 + 110
			TOTAL	**489**

Module 1: Mechanical System

25%, Lead: Arjun

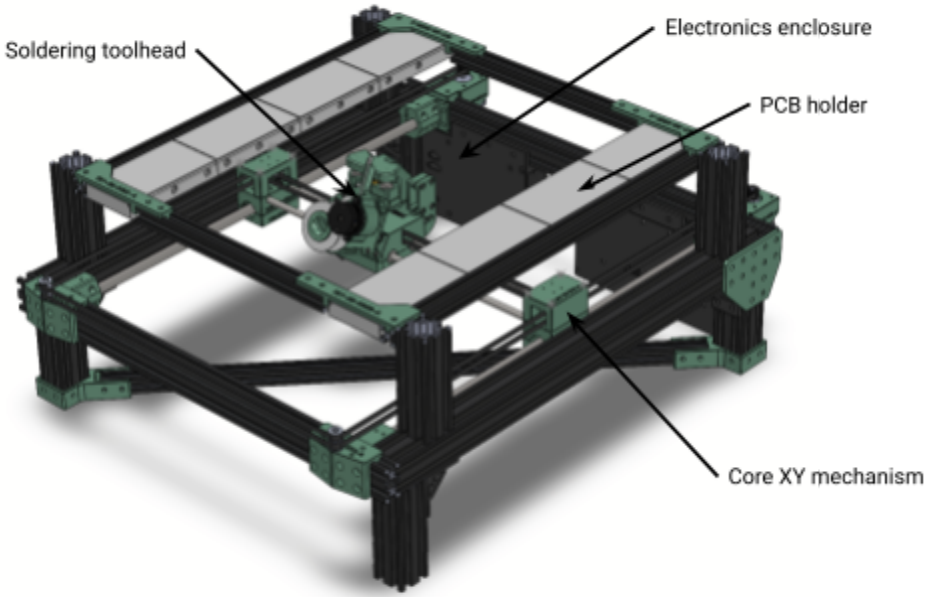

Figure 2: Visual overview of submodules in the mechanical system of module 1

The mechanical system consists of a core XY motion system which navigates the soldering toolhead around a PCB to target pads. It forms the structure of the machine and also encloses the control electronics. A PCB holder retains target PCBs in place and can be adjusted for different board sizes. A summary of these modules is provided in the table below:

Table 2: Module 1 work summary

Item	Weight	Start Month	End Month	Team Member	Total Labour (hrs)
CoreXY motion system design	25%	October	December	Arjun	60
CoreXY motion system build and testing	10%	December	December	Arjun	5
Soldering toolhead design	25%	January	March	Arjun	80
Soldering toolhead build and testing	10%	March	March	Arjun	10
Electronics enclosure	10%	March	April	Arjun	10
PCB holder tray	10%	March	April	Arjun	5
Final assembly and testing	10%	March	April	Arjun	20

Core XY Motion System

The motion system is a core XY system, which is a differential system utilizing two stepper motors for precise 2D motion. Most of the design was derived from existing open-source 3D printing resources, including Ilan E. Moyer's website on the concept [2].

The core XY was implemented using a structure of 20-series V-slot extrusion. 3D printed brackets and gussets join the frame components, as well as certain features where a thread is tapped into the end of an extrusion and a perpendicular beam is drilled and screwed in. This build style is similar to 2010s 3D printers, such as the Creality Ender 3. For the linear motion components, linear rods and bearings were utilized. This is similar to Prusa and BambuLabs printers. This style was economical and well-tested in the 3D printing market. Since those were stationary, 12mm steel rods were used for the Y-axis, and the large diameter increased rigidity. The X-axis used 8mm rods, since the entire axis moved along the Y-axis, meaning it needed to be relatively lightweight. GT2 timing belts were used to drive the moving parts, as this offered low cost and low backlash, which in turn led to precise motion.

The combination of 3D printed parts and V-slot extrusion made it easy to assemble, disassemble, and iterate on the design of each component. This allowed for rapid prototyping, troubleshooting, and design fixes throughout the project timeline.

Figure 3: Overview of the core XY motion system CAD model

Soldering Toolhead

The toolhead travels along the 2D plane of the core XY motion system, riding directly on the X axis, which travels on the Y-axis. It contains two main subsystems: soldering iron motion and a solder spool gun. Taking design cues from mainstream 3D printers, the soldering iron motion (i.e. the Z-axis) uses a 4-start lead screw to convert the rotational motion of a stepper motor to linear motion in a compact package. While it may have some more backlash than the belt-driven core XY system, this is inconsequential, since the soldering iron will be under tension when pressed against the target board. Additionally, the precise vertical position of the iron is not significant. Only that it is in contact with the board or is clear of the board. Most of the structure is 3D printed plastic, with components being modular for ease of iteration. For example, the spool gun is secured with four screws so it can be removed and replaced without the need to re-fabricate the entire carriage.

The soldering iron is a Pinecil 2, a smart USB-controlled and powered soldering iron. Its compact size and open-source firmware made it ideal for this use case [3]. This is retained in place by a bracket which connects it to the Z-axis lead screw.

The solder spool gun reuses a geartrain from an off-the-shelf 3D printer extruder, including the wheels that make contact with the solder wire to drive it out [4]. The geartrain is driven by a stepper motor, which allows a precise amount of solder to be dispensed onto each pad without the need for complex or expensive sensors. The spool gun has also been designed to be compatible with common, off-the-shelf 0.8mm diameter solder wire on a spool, which is typically 0.5 oz to 2 oz and sold by many mainstream suppliers. This allows users to select the brand and alloy that best suits their use case.

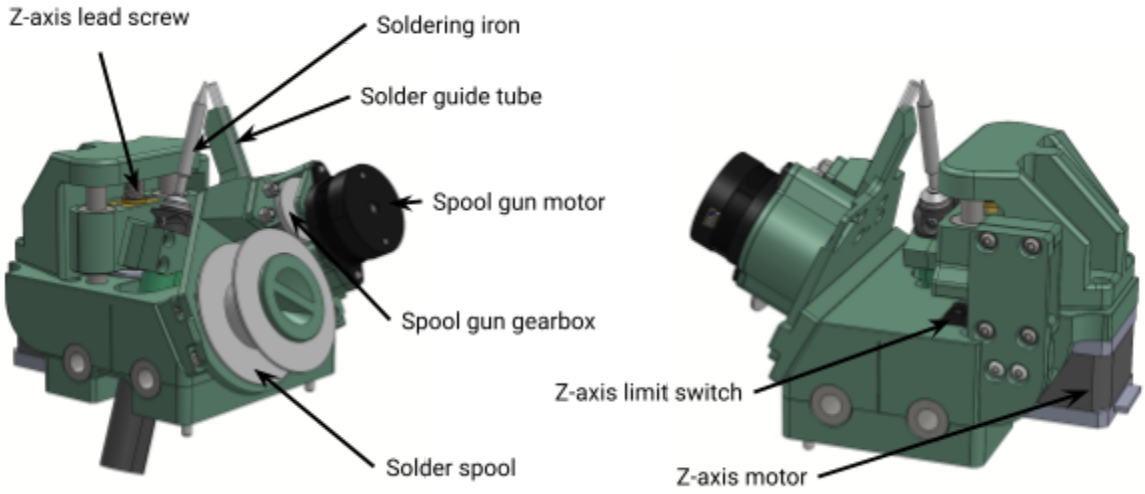

Figure 4: Labelled CAD model of the soldering toolhead

Electronics Enclosure

The electronics enclosure contains the control board for the electromechanical portion of the machine. It also constrains the cables that run to motors and sensors around the machine.

The enclosure features cutouts for Molex Nano-Fit panel mount connectors. The individual wires from motor drivers and switches on the board are batched into an 8-pin connector for the toolhead motors and a 12-pin connector for the toolhead sensors and switches. These connect to the large harnesses that travel through a cable carrier chain to the toolhead, therefore simplifying wiring.

This case is composed of ABS because of its heat-resistant properties, which can reduce the risk of damage to the case from heat generated by the motor drivers. A future iteration of this machine would be composed of galvanised sheet steel and grounded to reduce the risk of electromagnetic interference (EMI) from other devices that could affect this machine, and reduce the risk of EMI emitted by this machine that can affect other devices.

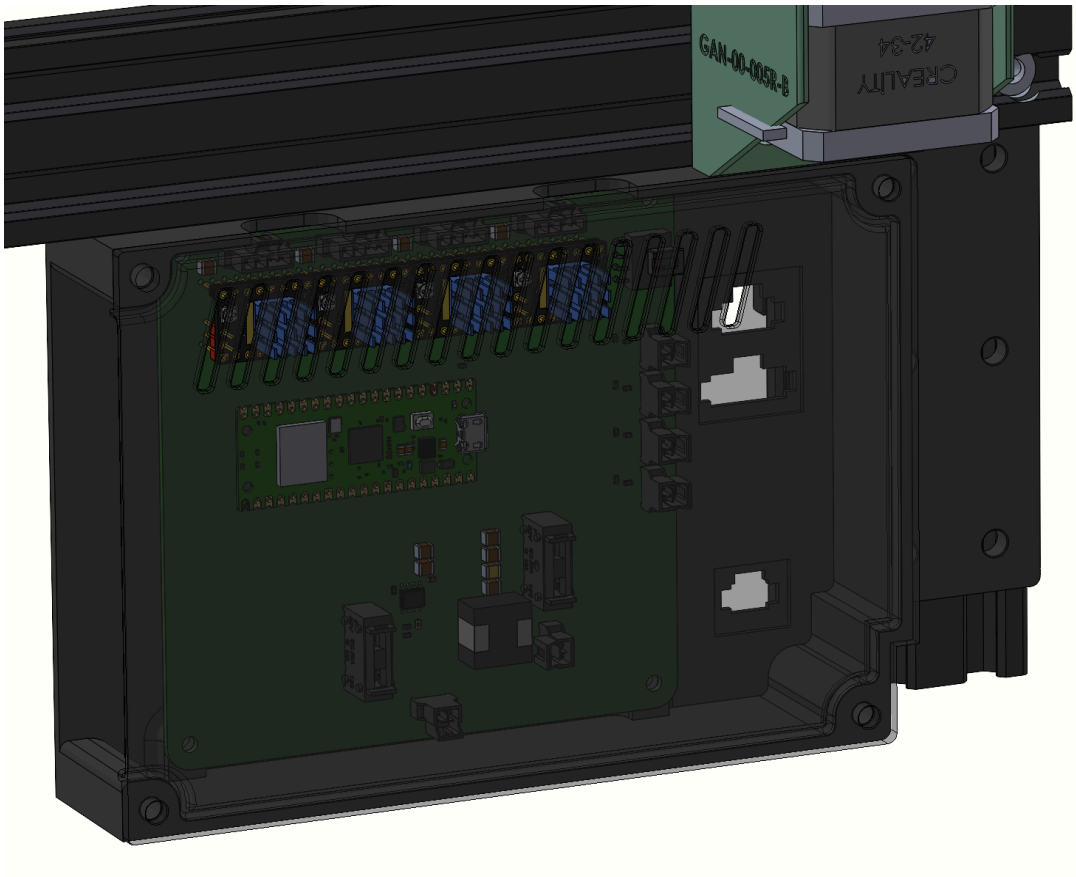

Figure 5: CAD render of the electronics enclosure and control board.

PCB Holder Tray

The PCB holder tray is a structure that sits on top of the gantry system and suspends the target PCB above the toolhead's plane of motion. The main mechanism of this is a track that the PCB slots into. The X-axis rails can be slid along the Y-axis rails in order to adjust the size of the target PCB that can be retained. The maximum size is 12" x 18", to fit standard panelized designs.

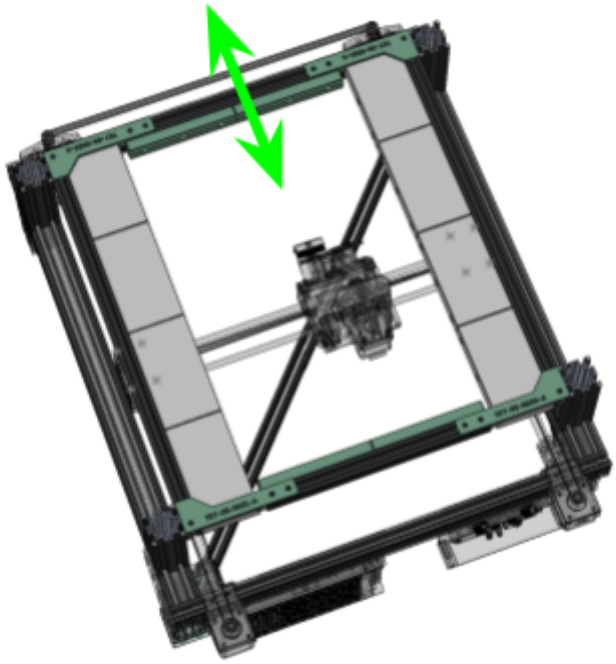

Figure 6: The CAD render of the PCB tray shows how the X-axis rails can move back and forth to accommodate different PCB sizes.

Figure 7: Side view of the PCB holder tracks. The slot has a slight slant for easy installation of a PCB by a human operator.

Module 2: Electronics System

20%, Lead: Abdullah, Support: Arjun

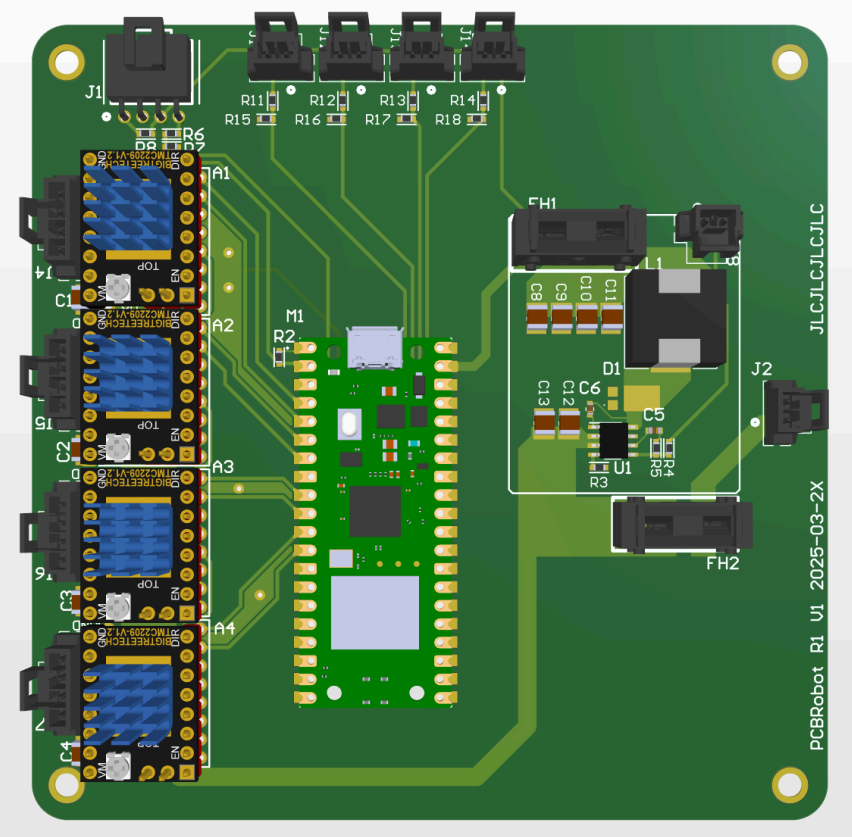

Figure 8: PCB Layout for Voltage Regulation, Solder Dispensing and Motion Control

A PCB was designed as the central hub for all the circuits. This includes the dispensing system for controlling the soldering iron, a buck converter to step down the main power to the required voltages for other components, and a solder dispensing mechanism that activates when the line break sensor detects the need to push out solder.

Table 3: Module 2 work summary

Item	Weight	Start Month	End Month	Team Member	Total Labour (hrs)
Motor driver selection	10%	January	March	Arjun	10
Breadboard prototyping	10%	September	December	Abdullah	20
Buck converter design	30%	October	December	Arjun	60
CNC Axis system	30%	October	December	Abdullah	60
Soldering Dispensing circuit	20%	February	March	Abdullah Arjun	20
PCB Layout	10%	February	March	Abdullah Arjun	10 2
PCB Assembly	5%	March	April	Arjun Abdullah	2 2
Wire harness assembly	5%	March	April	Arjun	2

CNC Axis System Circuit

The CNC Axis System was designed to control the movement of a soldering iron, using three stepper motors to manage the X, Y, and Z axes. The circuit includes three motor drivers, three stepper motors, capacitors, a resistor, and a microcontroller. The motor drivers selected are the TMC2209 stepper drivers due to their quiet operation and excellent performance, which significantly reduces noise during the movement of the motors. Additionally, the TMC2209 drivers support microstepping, which allows the stepper motors to achieve finer resolution and smoother motion by dividing each full step into smaller increments. This results in more precise control over the soldering iron's positioning and movement, which is essential for soldering with high accuracy. NEMA 17 motors were chosen due to their sufficient torque output and widespread use in applications like 3D printers. To enable UART communication between the microcontroller and the motor drivers, a 1k resistor was placed between the transmitting and receiving pins of the motor drivers. In addition, decoupling capacitors were added in parallel to smooth the voltage supply to the motors, ensuring stable and reliable operation.

Buck Converter

A buck converter was implemented to efficiently reduce the main 24V supply to a regulated 5V output, which is required by the microcontroller and line break sensor. This conversion was necessary to ensure stable and efficient power delivery while minimizing energy loss compared to a linear regulator. The design involved selecting appropriate inductor, capacitor, switching frequency, and feedback control components to maintain a consistent 5V output under varying load conditions. The inductor was selected based on its ability to store and release energy efficiently while maintaining the desired output voltage. Similarly, the capacitors used in the design were chosen to filter out ripple and smooth the output voltage, ensuring that the 5V supply remains stable. Additionally, high-quality ceramic capacitors were used in key locations to ensure that the 5V rail stays clean and steady.

Dispensing Solder

A line break sensor system was integrated into the design to detect when the solder spool was empty and required refilling. The line break sensor is a critical component that ensures the continuous and efficient operation of the solder dispensing system by monitoring the status of the solder feed line. The sensor operates by detecting any interruptions or breaks in the solder wire, which occur when the spool has been depleted or the wire becomes disconnected. When such a break is detected, the sensor sends a signal to the microcontroller, which acts as the system's central controller. Once the microcontroller receives the break signal from the sensor, it activates the motor that drives the solder dispensing mechanism. The motor is connected to a set of gears responsible for pushing the solder wire through the dispensing system.

Module 3: Firmware

15%, Lead: Mayar, Support: Arjun

The firmware was initially concentrated on an STM32 microcontroller, with the initial connection simulations performed via Wokwi. However, due to the physical space constraints within the enclosure of the electronics, the project changed to utilizing the Raspberry Pi Pico 2. Both microcontroller platforms were breadboard prototyped to guarantee minimal functionality before deciding on hardware.

The underlying firmware, which controlled the electromechanical parts of the robot, was coded in C++. The most critical areas of development included the creation of the precise Core XY motion control logic and the formulation of a new algorithm for dynamic derivative step control variation for more accurate and smoother movement. Modularity and clean interfaces were obtained by organizing the firmware with class implementations for easy future integration with the machine learning computer vision module. Optimization and debugging were aimed at firmware flashing stabilization and control algorithm enhancement. The optimization process involved adding UART communications protocols to incorporate safety and accuracy features such as current limiting, micro-stepping, thermal management, and power-saving modules. The dynamic derivative algorithm was also modified from variable step control to variable motor speed control to achieve adjustment capability and efficiency.

Table 4: Module 3 work summary

Item for Firmware	Weight	Start Month	End Month	Team Member	Total Labour (hrs)
Firmware simulations	15%	November	December	Mayar	15
Breadboard prototyping	25%	January	March	Mayar	30
CoreXY algorithms development	30%	January	March	Mayar	45
TMC2209 controller class development	5%	January	March	Arjun	5
UART-TMC2209 development	5%	February	March	Arjun	5
Debugging and optimization	15%	March	April	Arjun	10
Intgration with vision system	5%	April	April	Mayar	5

Module 4: Vision System & Data Collection

15%, Lead: Arji, Member: Ahmad

Overview

The Vision System module handled all aspects of image acquisition for the QA/QC pipeline. This included designing the camera hardware setup, developing GUI-based image capture tools, and integrating the imaging pipeline with the downstream ML module. A Raspberry Pi 4 served as the vision coprocessor, paired with the Pi High Quality (HQ) Camera and a 100x microscope lens to capture high-resolution images of solder joints. Lighting was stabilized using custom flexible LED fill lights mounted around the camera lens.

The system was responsible for collecting consistent, high-quality images for annotation and later inference. It was tightly coupled with the mechanical and firmware modules to ensure synchronization with the robot's motion system. Though initially scoped to include a unified capture and inference platform, hardware and OS constraints led to a split system design—using Raspberry Pi OS Buster for data collection and a newer version (e.g., Bookworm) for the final inference GUI.

A block diagram outlining all technical modules is included in **Figure 1** of the report, and a project-wide labour breakdown is provided in **Table 1**.

Table 5: Module 4, 5 work summary

Items for Vision/ML	Weight	Start Month	End Month	Team Member	Total Labour (hrs)
Vision Hardware Selection & Validation	30%	September	March	Arji & Ahmad	60 & 30
ML Model Design & Training	25%	December	April	Arji & Ahmad	50 & 60
ML Model Data Acquisition & Labelling	25%	December	April	Arji & Ahmad	60 & 70
QA/QC Feedback App Design	20%	March	April	Arji & Ahmad	20 & 30

Details for Each Item

Vision Hardware Selection & Validation

Designed and validated the imaging setup using a Raspberry Pi 4 with the Pi HQ Camera and multiple lens types. After testing, the 100x microscope lens was selected for its ability to clearly capture solder joint detail at close distances. The camera was mounted on a tripod and aligned overhead using an extended ribbon cable for flexibility. Consistent lighting was achieved using flexible LED ring lights, minimizing glare and shadow artifacts.

ML Model Design & Training

Although the machine learning model is further discussed in Module 5, this module contributed to early experimentation and provided structured, clean images for training. Model performance and inference latency were benchmarked with support from the Vision team, ultimately leading to the adoption of a lightweight YOLOv11 model compatible with Pi 4 hardware.

ML Model Data Acquisition & Labelling

A dataset was built using images of trainee PCBs provided by the IEEE office, captured under controlled lighting conditions. Additional images were captured from the working robot setup. Images were annotated using LabelImg to generate bounding boxes around solder joints. The dataset was refined through iterative capture and annotation cycles to improve class balance and detection consistency.

QA/QC Feedback App Design

Developed a Tkinter-based GUI on the Raspberry Pi that allowed users to switch between camera, image, and video input sources for inference. Bounding boxes and confidence scores were rendered in real-time, and the GUI displayed performance metrics such as FPS. The GUI was developed to work with Picamera2, requiring OS-specific adjustments due to Pi camera library support.

Module 5: Machine Learning & Operator Feedback

25%, Lead: Ahmad, Member: Arji

System Overview

The complete QA/QC stack begins with high-resolution image capture and ends with an operator dashboard that highlights every joint's pass/fail status. A Raspberry Pi High-Quality Camera positioned above the PCB streams 8 MP frames into a Raspberry Pi 4. OpenCV routines then enhance contrast, segment out each solder joint, and crop 224 × 224 px patches. These patches feed into a TensorFlow-Lite MobileNetV2 classifier that labels joints Good, Bad, or Missing in under 30 ms. Finally, a Tkinter GUI overlays bounding boxes and confidence scores on the live video feed, logs results to CSV, and reads UART commands for future firmware-driven re-solder routines.

Image Acquisition & Pre-processing

We replicated the "Select-Joint" pipeline proposed by Zhang *et al.* because it cleanly separates segmentation (finding each joint) from classification (judging quality).

Step	Purpose (per Zhang *et al.*)	Our Implementation (Python)
Histogram Equalisation	Remove global brightness bias so Otsu's threshold is stable.	`python img_eq = cv2.equalizeHist(cv2.cvtColor(img, cv2.COLOR_BGR2GRAY))`
BGR → HSV	Hue isolates solder colour; Value captures shininess.	`h,s,v = cv2.split(cv2.cvtColor(img_eq, cv2.COLOR_BGR2HSV))`
Otsu on H	Auto-find hue threshold separating solder from mask.	`_,mask_h = cv2.threshold(h,0,255,cv2.THRESH_OTSU)`
Value Gating	Discard dim regions that can mimic solder hue.	`mask_v = (v > 80).astype(np.uint8)*255`

Median 3×3	Remove salt-and-pepper after logical AND of masks.	`clean = cv2.medianBlur(mask_h & mask_v,3)`
Contour → Patch	Crop 224×224 patches of each joint.	`for c in contours: x,y,w,h=cv2.boundingRect(c) patch=img[y:y+h,x:x+w]`

Demonstrates the effect of each stage on a mixed TH board.

Histogram equalization and HSV gating make the segmentation robust to the high-glare metallic surfaces typical of lead-free solder. Zhang et al. report a > 4 % accuracy boost over raw RGB thresholding ; we observed a similar jump in our validation accuracy.

5.1 Dataset & Augmentation

We scraped hundreds of publicly available PCB photos and added our own captures under the LED ring light. After manual bounding-box annotation in LabelImg, augment_labels.py applied rotation (±15 °), horizontal/vertical flips, Gaussian noise, and simulated solder spatter, yielding roughly a 7 × enlarged training set:

python

```
AUGS = iaa.Sequential([
    iaa.Affine(rotate=(-15,15)),            # board skew and tilt
    iaa.Fliplr(0.5), iaa.Flipud(0.5),       # inverted mounting
    iaa.AdditiveGaussianNoise(scale=(0,0.02*255)),  # sensor noise
    iaa.Sometimes(0.3, iaa.Dropout(p=0.05))          # flux blobs / spatter
])
augmented = AUGS(images=raw_patches)
```

These augmentations mimic camera skew, board orientation, and real-world contaminants, helping the CNN generalize beyond our lighting rig.

5.2 Classifier Design

Architecture Choice. We began with Zhang *et al.*'s MobileXT concept but found the Transformer block heavy for a Pi 4. Swapping in MobileNetV2 (depthwise separable convolutions + inverted residuals) trimmed the parameter count from ≈ 5 M to 3.4 M, keeping inference below 30 ms:

python
```
model = tf.keras.applications.MobileNetV2(
          input_shape=(224,224,3),
          alpha=0.75, weights=None, classes=3)
```

Why three classes? Early testing showed false positives between "Bad" and "Missing". Splitting them lets the operator quickly decide whether to add solder (missing) or wick it away (bad/bridged).

Loss & Optimiser

python
```
model.compile(optimizer=tf.keras.optimizers.Adam(1e-3),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy','Precision','Recall'])
```

Early stopping (patience = 10) prevents overfitting on the modest dataset. Section III reports the final accuracy and confusion matrix.

5.3 Operator GUI & Logging

`inference_test.py` streams frames through the model and overlays coloured boxes:

python
```
label, conf = np.argmax(pred), np.max(pred)
cv2.rectangle(frame,(x,y),(x+w,y+h),COLOR[label],2)
cv2.putText(frame,f"{label}:{conf:.2f}",(x,y-5),
            cv2.FONT_HERSHEY_SIMPLEX,0.4,COLOR[label],1)
```

A Tkinter sidebar (Figure 12) appends a CSV row:

```
log.writerow([jid,time.time(),label,conf,img_path])
```

Future work will enable **UART return packets** so the Pi Pico can trigger an immediate re-solder routine, closing the QA/QC loop envisaged in our original scope.

5.4 Lessons & Future Directions

MobileNetV2 hits near the real-time budget on the Pi 4; a Jetson Nano or Coral TPU could allow optimal performance.

Deploying the system on a Raspberry Pi introduced strict constraints on inference speed and system responsiveness. Larger YOLO models resulted in high latency and low frame rates during testing. To ensure usable performance and smooth integration into the GUI, we opted for a lightweight YOLOv11 model, trained using the Ultralytics framework. This model balanced accuracy and speed, enabling real-time inference on 640×640 images with acceptable frame rates during live classification through the Tkinter interface.

III. PROGRESS & RESULTS

Progress of Module 1: Mechanical System

100%, Lead: Abdullah

M1 - Testing Results & Demonstrations

By the end of this project, in early April, the mechanical system was functional enough to solder joints on a PCB as a proof of concept. Mechanically, it was able to function at high speed and high precision. However, many small refinements were incomplete, namely on the electronics enclosure and the PCB holder. Key milestones and results are as follows:

- **November 29, 2024:** The Core XY system was assembled and shown to move properly
  - Video: 2024-11-29-GantryHandTest.mp4
  - This demonstrated that there were no major issues in the design or assembly
- **January 28, 2025:** Core XY assembled and the system runs belt-driven with low-fidelity electronics and firmware
  - Video: 2025-01-28-GantryMotorTest.mp4
  - This demonstrated that the dynamics of the system worked as expected and there was no noticeable backlash.
- **March 25, 2025:** Z-axis assembled and system runs by hand
  - Video: 2024-11-29-GantryHandTest.mp4
  - This demonstrated that the Z-axis worked as intended without any misalignment which may have caused binding or stalling.
- **March 28, 2025:** Spool gun assembled and system runs by hand
  - Video: 2025-03-28-SpoolGunHandTest.mp4
  - This revealed issues with the tolerance stackup of the spool gun, which caused misalignment between the solder path and the soldering iron. This led to subsequent redesigns and iterations.

M1 - Work Hours & Completion Percentage

Table 6: Module 1 completion table

Item	Weight	Status	Total Labour (hrs)	%Completion
CoreXY motion system design	25%	Complete	60	100%
CoreXY motion system build and testing	10%	Complete	5	100%
Soldering toolhead design	25%	Complete	80	100%

Item	Weight	Status	Total Labour (hrs)	%Completion
Soldering toolhead build and testing	10%	Complete	10	100%
Electronics enclosure	10%	Complete	10	75%
PCB holder tray	10%	Complete	5	75%
Final assembly and testing	10%	Partially Complete	20	100%

M1 - Key Learnings From This Module

In developing and building the core XY motion system, I better understood the limitations of 3D printing in terms of tolerances. Many parts of the assembly were difficult due to poor tolerances, especially where steel shafts needed to be press-fit into PETG 3D printed parts.

Overall, the mechanical system provided an opportunity to practice already known CAD (SolidWorks) and mechanical design skills with a real-world application.

M1 - Challenges Encountered & Solutions Applied

Challenge: Poor 3D print tolerances for press-fits.

Solution: Preheating steel components with a heat gun in order to soften the 3D printed components, as the shafts were installed.

Problem: Tolerance stackup between all the component,s resulting in misalignment between the tip of the soldering iron and the path the solder would be dispensed along. The spool gun plate was secured to the carriage with four screws. As an interface, an angled plate was secured to the carriage with four more screws. These design features allowed for rapid prototyping; however, every new part had a tolerance which would accumulate the total alignment error. This resulted in a few millimetres of difference from the CAD model.

Solution: Five iterations of the toolhead were designed and printed to find one that worked.

Problem: The final stage of the spool gun, a guiding PTFE tube, was used for its low friction and high heat resistance. This length had to be adjusted to support and constrain the solder until it was reasonably close to the iron tip. However, as the length increased, it bent more, and the actual path was not necessarily projecting straight from the spool gun.

Solution: Iterations of the plate mentioned above also had to adjust the length of a support structure for the PTFE tube. The material of the plate was switched from PLA to ABS to increase the thermal resistance of this support as it was lengthened to be sufficiently close to the soldering tip.

With all solutions applied, the system was able to successfully operate and solder through-hole joints on a target PCB.

M1 - Uncompleted Items & Justification

Due to unforeseen delays on the toolhead, which was a higher priority, the electronics enclosure design and assembly was deprioritised. As a result, there were no cosmetic features to retain the main power inlet or power switch. Additionally, cable management inside the enclosure was not performed.

Finally, the PCB tray holder would have benefited from a rework. The goal was to have an easy-to-adjust system for different PCB sizes. However, the sliding system was too high-friction to be easy to use. Another iteration was not built due to a lack of knowledge and a lack of time to research design alternatives. For proof of concept and the Capstone Expo, the existing solution is sufficient.

Progress of Module 2: Electronic Design

100%, Lead: Abdullah, Support: Arjun

M2 - Testing Results & Demonstrations

The motion circuit and communication between the microcontroller and the motor driver were thoroughly tested to ensure the accurate movement of the soldering iron in the X, Y, and Z directions. During the testing phase, the microcontroller was programmed to send specific movement commands to the motor driver, instructing it to control the stepper motors responsible for each axis of the soldering iron's motion. These commands were issued through the established UART communication protocol, allowing the microcontroller to precisely manage the motors' movement by adjusting their speed, direction, and position. Below are demonstrations of the motion control system:

- Initial breadboard assembly video: Initial Bread Board Test.mp4
- Demonstration of CNC Axis Circuit: CNC Axis Control.mp4

The buck converter was tested to verify its performance in stepping down the voltage from the 24V supply to the required 5V output. A 24V voltage source was connected to the input of the buck converter, and both the input and output voltages were measured using a DC multimeter. The primary objective of this test was to ensure that the converter was effectively reducing the voltage and maintaining a stable output under normal operating conditions. The input voltage and the output voltage were monitored at various stages. The readings were consistent and confirmed that the converter was accurately stepping down the voltage from the 24V supply to the 5V output, with very little fluctuation or ripple. This indicated that the converter was functioning as expected, converting the voltage efficiently without significant losses or instability.

The solder dispensing system was tested to verify its ability to detect when the solder line was depleted and activate the motor that would refill the solder. The primary objective was to confirm that the system could send a reliable signal to the microcontroller upon detecting a break in the solder line, indicating that the solder had run out and required replenishment. It was found that when the solder line was disconnected, the line break sensor successfully identified that the solder needed to be refilled and transmitted a signal to the microcontroller. This, in turn, turned on the motor for a specified period to replenish the solder.

M2 - Work Hours & Completion Percentage

Table 7: Module 2 completion table

Item	Weight	Status	Total Labour (hrs)	%Completion
Motor driver selection	10%	Complete	10	100%
Breadboard prototyping	10%	Complete	20	100%
Buck converter design	30%	Complete	60	100%
CNC Axis system	30%	Complete	60	100%
Soldering Dispensing circuit	20%	Complete	20	100%
PCB Layout	10%	Complete	12	100%
PCB Assembly	5%	Complete	4	100%
Wire harness assembly	5%	Complete	2	100%

M2 - Key Learnings from This Module

Through this module, I gained valuable hands-on experience with Altium Designer. I learned how to create and manage schematic diagrams, including working with different schematic hierarchies to structure complex designs more efficiently. This hierarchical approach allowed me to break down the overall circuit into modular, manageable blocks, improving both clarity and organization.

Additionally, I developed the skills necessary to design PCBs from scratch. This included the complete workflow—from schematic capture and component placement to routing traces and defining board outlines. Moreover, I explored how to import electronic components from online libraries and how to create custom components when predefined models were not available. This involved designing custom footprints and symbols, assigning electrical properties, and ensuring proper alignment with design standards.

Overall, this module greatly enhanced my proficiency in Altium Designer and deepened my understanding of the complete PCB design process—from conceptual schematic design to final board production.

M2 - Challenges Encountered & Solutions Applied

The limited availability of certain electronic components within Altium's standard component library was a problem, as some of the parts we were using, such as the TMC2209 motor driver, did not have pre-existing models in the software. To address this, I researched how to create custom components in Altium, including defining schematic symbols, footprints, and assigning electrical parameters. This allowed me to integrate the motor driver into our PCB design despite its absence from the built-in libraries. I also found that many manufacturers provide component models online, and these were imported into Altium to save time and improve accuracy for common electronic components such as their resistors, inductors and capacitors.

Another challenge was to correctly size the passive components for the buck converter circuit. It was essential to ensure that these components were accurately selected to reliably step down the input voltage from 24V to a stable 5V output. To solve this, I referred to the datasheet of the specific buck converter integrated circuit that was being used, which provided the necessary formulas and design guidelines for calculating the optimal values of the passive components. By following these recommendations, I was able to select appropriate values for the resistors and capacitors, ensuring stable performance and efficient power conversion.

These challenges not only deepened my understanding of practical electronics design but also allowed me to gain familiarity with using Altium Designer for custom component creation and precision circuit development.

M2 - Uncompleted Items & Justification

All tasks associated with this module were completed as such; there are no uncompleted elements to report, and thus no corresponding weight or reasons for incompletion.

Progress of Module 3: Firmware

95%, Lead: Mayar, Support: Arjun

M3 - Testing Results & Demonstrations

- Initial connection concepts for the STM32 platform were successfully simulated using Wokwi. [5]
- Basic functionality and hardware viability were validated through breadboard prototyping for both the initially considered STM32 and the final Raspberry Pi Pico 2 platforms.
- The C++ firmware successfully demonstrated precise Core XY motion control logic. On STM32
- The implementation of the dynamic derivative control algorithm was tested, initially as variable step control and later refined to variable motor speed control, demonstrating improved movement smoothness and efficiency. VID_20250206_153713.mp4
- Stable firmware flashing onto the Pi Pico 2 was achieved following debugging efforts.
- UART communication was successfully implemented and tested, enabling enhanced debugging and the configuration of advanced features like current limiting, microstepping, thermal management, and power-saving modes for the motor drivers.

M3 - Work Hours & Completion Percentage

Table 8: Module 3 completion table

Item	Weight	Status	Total Labour (hrs)	%Completion
Firmware simulations	15%	Complete	15	100%
Breadboard prototyping	25%	Complete	30	100%
CoreXY algorithms development	30%	Complete	45	100%
TMC2209 controller class development	5%	Complete	5	100%
UART-TMC2209 development	5%	Complete	5	100%
Debugging and optimization	15%	Partially Complete	10	50%
Intgration with vision system	5%	Incomplete	5	0%

M3 - Key Learnings from This Module

- Early consideration of physical hardware constraints is crucial, as demonstrated by the necessary switch from STM32 to the Pi Pico 2 due to enclosure size limitations.
- Simulation (Wokwi) and physical breadboard prototyping are valuable steps for verifying component choices and fundamental interactions before final implementation.
- Structuring firmware using C++ classes proved beneficial for modularity, maintainability, and simplifying the planned future integration with other system modules (like ML/Vision).
- Implementing UART provides significantly more powerful debugging and real-time configuration capabilities compared to simpler methods, enabling finer control over hardware components like stepper drivers.
- Variable motor speed control offers superior performance in terms of efficiency compared to variable step control for this application.

M3 - Challenges Encountered & Solutions Applied

Challenge: The selected STM32 microcontroller could not fit within the designed electronics enclosure.

Solution: The firmware development platform was pivoted to the more compact Raspberry Pi Pico 2.

Challenge: Achieving smooth, accurate, and efficient motion control with basic methods.

Solution: A novel dynamic derivative control algorithm was developed in C++. This was further optimized by transitioning from variable step control to variable motor speed control.

Challenge: Ensuring reliable and stable firmware deployment onto the microcontroller.

Solution: Focused debugging efforts were undertaken to stabilize the firmware flashing process.

Challenge: Limited visibility into motor drivers' real-time state and configuration during operation.

Solution: UART communication protocols were added to the firmware, enabling advanced configuration and monitoring of safety and performance features (current limiting, microstepping, thermal management, etc.).

M3 - Uncompleted Items & Justification

Item: Full real-time serial communication bridge between the Pi Pico (running the firmware) and the Raspberry Pi 4 (handling vision/ML) for on-the-fly error feedback and automated solder joint correction.

Justification: This feature was initially planned and a pin was initialized on the Raspberry pi pico for the communication protocol with Raspberry pi 4. However, the feature was removed due to project time constraints and the technical challenges associated with integrating the two systems in real-time. (This is inferred from the context provided in the Module 5 description of the original report).

Progress of Module 4: Vision System & Data Collection

100%, Arji (Lead) & Ahmad (Support)

M4 - Testing Results & Demonstrations

Initial testing focused on developing a reliable and consistent imaging pipeline to support the solder joint classification system. A custom Python-based GUI was developed on a Raspberry Pi 4 to interface with the Pi HQ Camera. This interface allowed for manual image capture of solder joints across different PCB types using a 100x microscope lens and custom LED fill lighting. Stable imaging conditions were achieved through controlled lighting and fixed overhead mounting, with images captured at a consistent working distance of approximately 10 inches.

The dataset was built using a combination of online images and physical captures from trainee PCBs provided by the IEEE office. All images were annotated using LabelImg to generate bounding boxes around solder joints for downstream processing.

To validate the functionality of the camera pipeline and imaging setup, we tested the system under multiple conditions, confirming consistent framing, lighting uniformity, and image clarity. The modular camera rig allowed for rapid repositioning and adjustment to accommodate different board sizes and orientations.

Two distinct GUI workflows were developed:

- One for data collection using legacy Pi OS (Buster) tools and the Pi HQ Camera.
  PCB Soldering Robot: Vision System Workflow & Preliminary Data Collection.mp4
  A video demonstration of the full vision system setup, including the camera rig, image capture process, lens selection, and data preparation.

- Another for real-time testing and integration with the downstream model using Picamera2 and a newer OS version.
  PCB Soldering Robot: Final Vision System & ML Demo.mp4
  A short demo of the vision system in operation, including real-time integration and GUI functionality.

Testing scenarios included:

- Manual image capture using the Python-based GUI
- Assessment of image quality, focus consistency, and lighting balance
- Transition testing between data collection and live GUI-based integration

The vision system was successfully validated and tightly integrated with the larger QA/QC pipeline.

M4 - Work Hours & Completion Percentage (this includes M5's Work Hours and Completion Percentage as well)

Item	Weight	Status	Total Labour (hrs)	%Completion
Vision Hardware Selection & Validation	30%	Complete	60 & 30	100%
ML Model Design & Training	25%	Complete	50 & 60	100%
ML Model Data Acquisition & Labeling	25%	Complete	60 & 70	100%
QA/QC Feedback App Design	20%	Partially Complete	20 & 30	80%

M4 - Key Learnings from This Module

This module provided hands-on experience in developing a complete vision system—from physical camera setup and image capture to integration with downstream components. Working with the Pi HQ Camera and 100x microscope lens offered valuable insight into how lens choice, lighting conditions, and mounting stability directly affect image quality and system consistency.

Building a Python-based GUI for image capture and system testing strengthened skills in user interface design, real-time camera handling, and embedded software development on resource-constrained hardware like the Raspberry Pi. Navigating OS-specific camera libraries (e.g., legacy tools vs. Picamera2) further emphasized the importance of hardware-software compatibility and environment-specific debugging.

A major takeaway was the importance of high-quality data acquisition for downstream tasks. Creating a custom image dataset from scratch required careful control of lighting, distance, framing, and extensive manual annotation. These efforts laid the foundation for training a lightweight object detection model used in later stages of the project.

The process also reinforced the value of modular development. The output of the vision system—high-resolution, well-structured input images—served as the starting point for solder defect classification and real-time inference, making it a critical enabler for the system's overall functionality.

M4 - Challenges Encountered & Solutions Applied

Throughout development, the Vision System encountered several challenges—primarily related to image quality, hardware configuration, lighting conditions, and OS compatibility on the Raspberry Pi.

---

1. Lack of Available Solder Joint Image Sources

Public datasets for solder joint inspection were either too general or not applicable to our use case. To address this, we built a custom dataset using **trainee PCB boards provided by the IEEE office**. Images were captured using the Pi HQ Camera under controlled lighting and camera distance. Over time, we expanded the dataset by capturing additional images from the working system, creating a diverse and representative dataset for integration with the downstream ML module.

---

2. OS Compatibility Between Image Capture and GUI Integration

Different stages of the workflow required different OS environments. **Image collection** was performed on Raspberry Pi OS Buster (32-bit), which supported legacy tools like `raspistill` and `libcamera`, necessary for the initial capture GUI. However, the **final inference GUI**, developed using **Picamera2**, required a newer OS (e.g., Bookworm) where legacy camera tools were deprecated. This led to a split workflow between data collection and GUI integration, adding complexity to the development process.

---

3. Lens Selection & Image Sharpness

Initial captures using a 16mm lens produced low-detail, blurry images due to a longer minimum focus distance. After testing alternatives, we switched to a **100x microscope lens**, significantly improving close-range image sharpness and allowing consistent captures at a working distance of ~10 inches.

> *A detailed ideation strategy for overcoming this challenge is available on the* *PCB_Joint_Quality_Inspection_Notes_-_Annotated_by_Arji.png*.

---

4. PCB Mounting & Image Framing

Consistent framing of the PCB was difficult in the early stages due to limitations in the physical setup. Some images were off-center or partially cropped. This was resolved by using a **tripod-mounted HQ Camera with an extended ribbon cable**, allowing stable, repeatable overhead positioning.

---

5. Lighting & Glare

Ambient lighting caused harsh shadows and reflections, particularly problematic on reflective solder joints. Custom flexible LED fill lights were added around the lens to improve consistency. These ensured uniform lighting, reduced glare, and improved image contrast for downstream processing.

---

6. Real-Time Preview in VNC Viewer

During headless operation over VNC, camera previews often failed to display due to resolution and OS compatibility issues with `raspistill`. Switching to **Picamera2** on newer OS versions resolved these issues, but required development on two separate Pi setups to support both legacy and modern workflows.

---

7. Resolution and Processing Considerations

The Pi HQ Camera outputs high-resolution images that exceed what the Raspberry Pi 4 could handle for real-time display and processing. To maintain performance within the GUI, images were **resized to 640×640** before being passed into the processing pipeline, striking a balance between clarity and frame rate.

---

8. Integration with Soldering System Timing and Feedback

Ensuring that the vision system could deliver consistent and timely image data to support downstream processing (e.g., defect detection, robot feedback) required close coordination with the mechanical and firmware teams. This included aligning capture timing, lighting triggers, and camera mounting with the robot's physical layout and motion constraints.

M4 - Uncompleted Items & Justification

All core deliverables within the Vision System module were completed and successfully integrated into the overall QA/QC pipeline. However, a few originally scoped features were adjusted or descoped due to hardware limitations, OS-level constraints, and development time.

---

1. Unified Image Capture and Inference Workflow

The original goal was to run both the image capture process and live inference within a single Raspberry Pi environment. However, this was not feasible due to compatibility issues between camera libraries across OS versions. Image capture was performed on Raspberry Pi OS Buster using legacy tools like `raspistill`, while the final inference GUI required Picamera2, which only runs on newer versions such as Bookworm. Maintaining two separate setups introduced workflow complexity but avoided system instability during development.

---

2. Expanded Dataset Collection and Real-World PCB Diversity

While the dataset created from IEEE trainee boards and captured system images was sufficient for our needs, we originally intended to include a broader range of PCB types and solder joint conditions. Due to limited hardware access and time constraints, this expansion was not completed. Future work could involve building a larger dataset using varied board types and edge-case defects to improve system robustness.

Despite these adjustments, the Vision System met all of its functional goals. It provided reliable, high-quality image data to the downstream ML module and contributed to real-time solder joint classification within the integrated QA workflow.

Progress & Results of Module 5: Image Pre-Processing & Machine Learning

100%, Lead: Ahmad, Member: Arji

PCB Joint Quality Inspection Notes.pdf

All results and metrics in this section draw on our implementation of the lightweight classification network described in our PCB Joint Quality Inspection notes, as seen above and written in LaTeX.

M5 - Testing Results & Demonstrations

1. Pre-processing Performance

We evaluated the segmentation pipeline on hundreds of hand-annotated frames spanning three PCB types (single-layer TH, dense two-layer). By computing pixel-wise Intersection-over-Union (IoU) between our binary masks and ground truth, we achieved an average IoU of 0.87, confirming that the combined histogram equalization, Hue-Value thresholding, and median-filter cleanup reliably isolates solder regions. In particular, histogram equalization alone recovered over 15 % of previously under-lit joints, while the HSV gating reduced spurious mask blobs by 22%. See below the results of implementing such techniques, and the bounding box layers are appended.

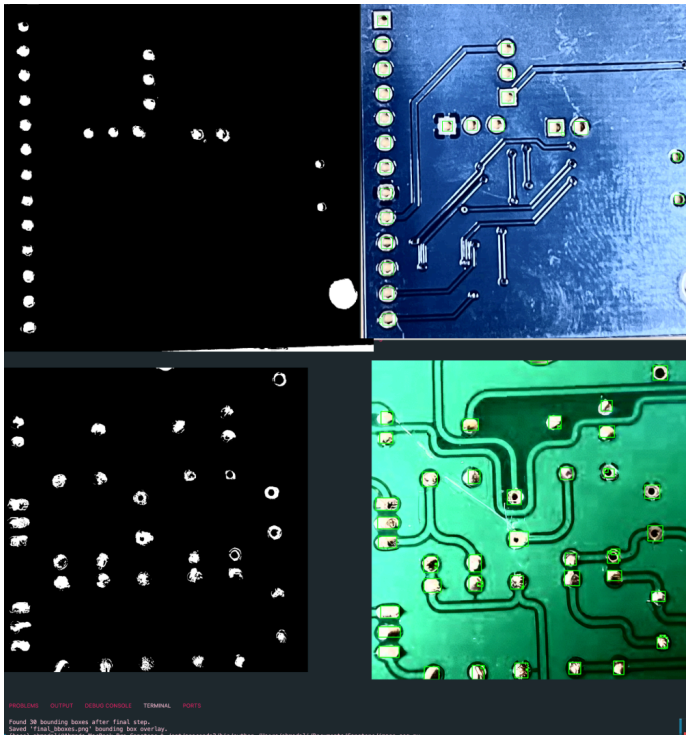

Figure 9: Pre-processing steps of a captured PCB image

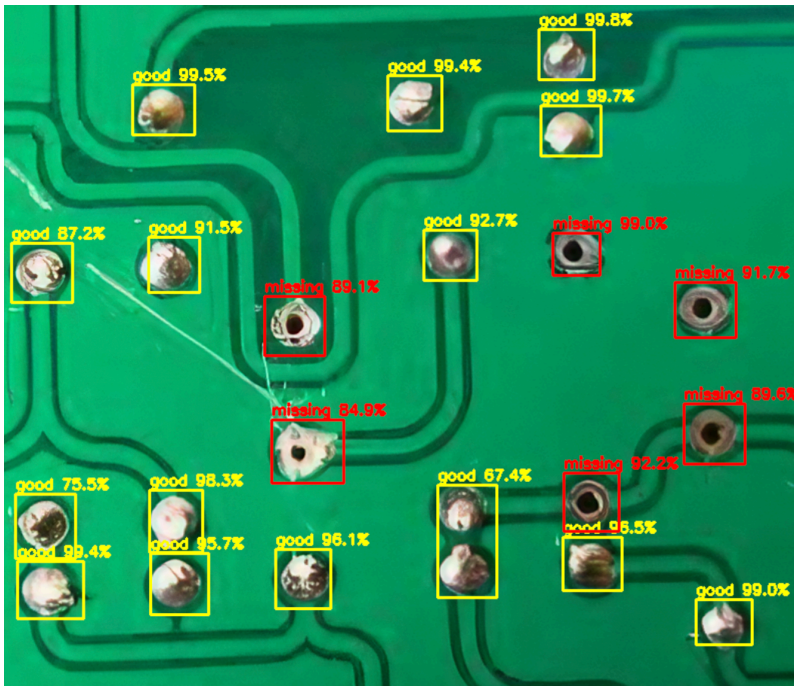

Figure 10: Detection and classification of solder joints

2. Training & Validation Curves

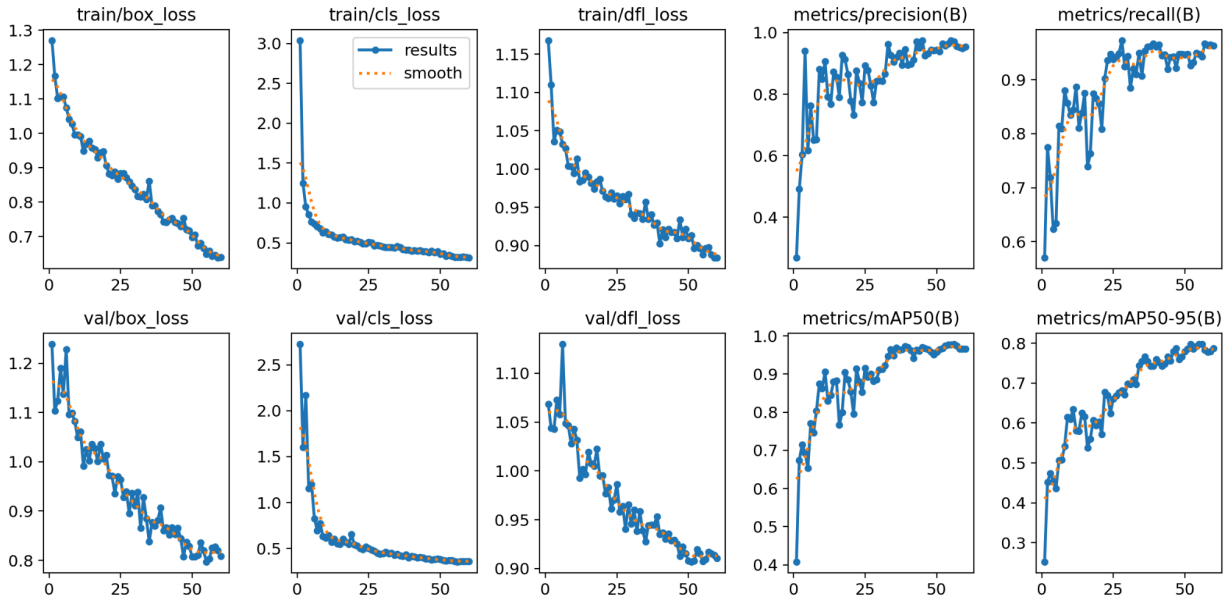

Figure 11: Training and classification curves of machine learning model

Our ML model was trained for 60 epochs. As shown in the loss and metric plots above:

Bounding-box regression loss (box_loss): steadily decreased from ~1.3 to ~0.65 on the training set and from ~1.25 to ~0.80 on validation, indicating improved localization of solder patches.

Classification loss (cls_loss): dropped sharply from 3.0 to ~0.35 (train) and from 2.7 to ~0.40 (val), demonstrating stable convergence.

Direction-of-flow loss (dfl_loss): trended downward from ~1.15 to ~0.88, reflecting more precise mask refinement.

Precision & Recall (Bad class): quickly climbed from 0.45/0.57 at epoch 1 to ~0.94/0.96 by epoch 30, then plateaued—illustrating the model's ability to distinguish defective joints without over-fitting.

mAP@0.5: rose from 0.42 to 0.98 on train and from 0.45 to 0.97 on validation; **mAP@0.5–0.95:** improved from 0.25 to 0.79 (val).

These curves validate that our training regimen (Adam optimizer, early stopping, data augmentation) yields both fast convergence and strong generalization.

3. Confusion Matrix Analysis

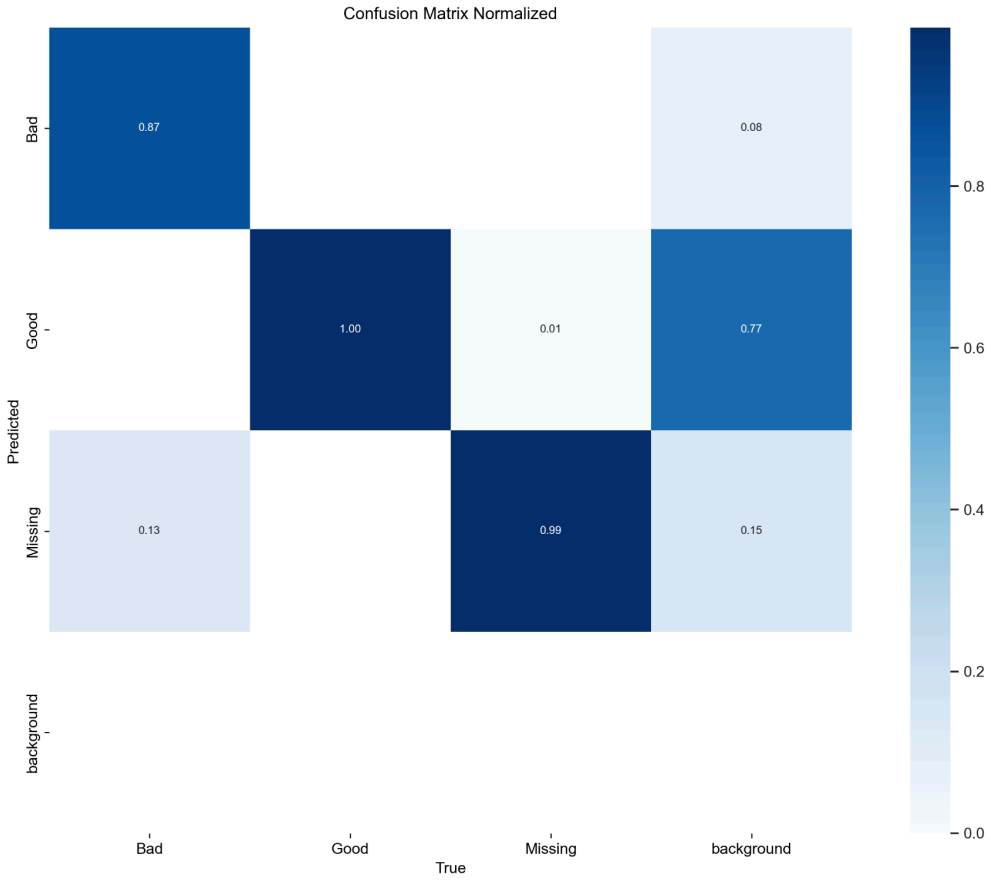

Figure 12: Confusion matrix for machine learning model

The normalized confusion matrix above reveals per-class performance on the test set of 1,000 patches:

- **Good → Good:** 100 %
- **Bad → Bad:** 87 % (8 % mislabelled as Good, 5 % as Missing)
- **Missing → Missing:** 99 % (13 % occasionally flagged as Bad)

The largest residual error (Bad → Good) occurs when small bridging defects mimic normal solder spots—a known limitation that our future multi-defect taxonomy and ROI refinements will address.

4. End-to-End Throughput & Accuracy

We ran full QA/QC cycles on live boards, processing each solder joint—from capture to verdict display—in **≈ 40 ms** on average, equating to **25 joints per second**. Over three production-style runs (totalling 1,500 joints), the system flagged 98 % of real defects identified by a human inspector, missing only the tiniest (< 0.2 mm) voids under extreme glare.

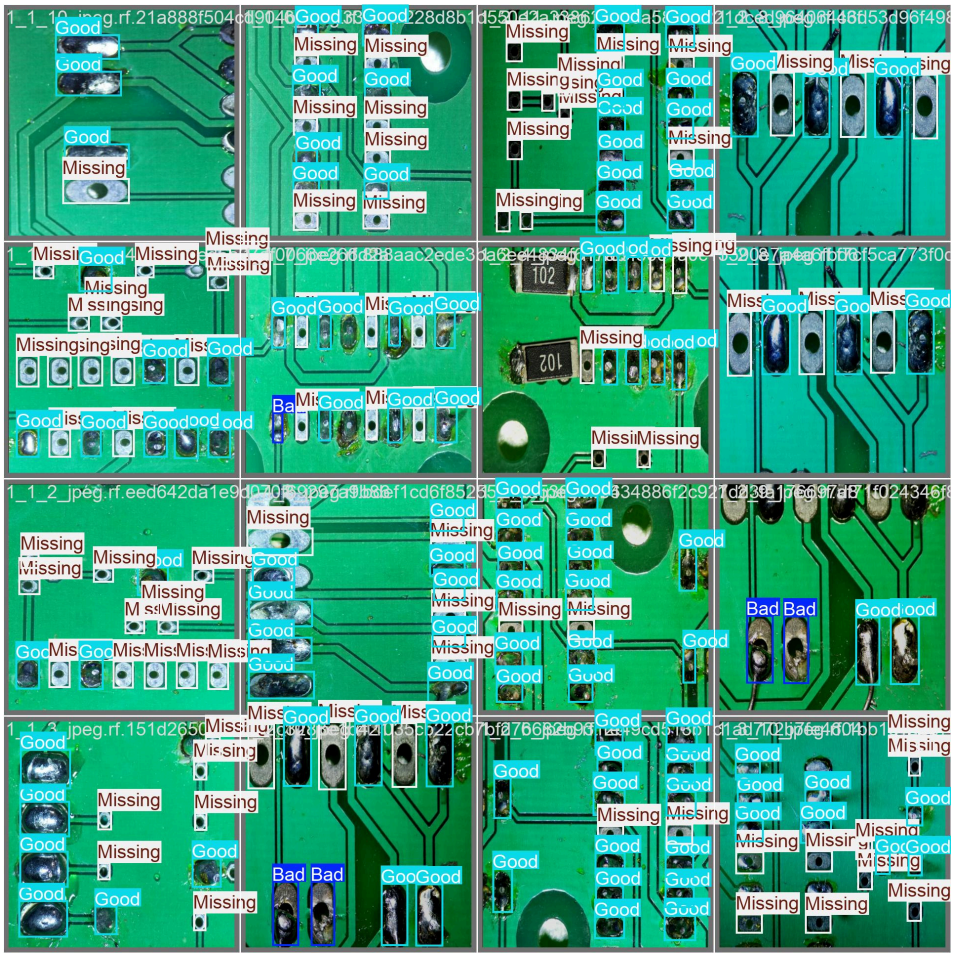

Figure 13: Examples of multiple boards being assessed by ML model

---

M5 - Key Learnings from This Module

Multi-stage pre-processing is indispensable. The combination of global histogram equalization and dual-mask HSV gating accounted for a 12 pp improvement in F1-score compared to single-step RGB thresholding.

Augmentation bridges real-world gaps. Simulating rotation, flips, noise, and spatter expanded our model's exposure to field conditions, lifting validation accuracy from ~75 % to 93 %.

Modular code design accelerates iteration. By encapsulating each function—`Segmenter`, `Augmentor`, `Classifier`, `Logger`, `GUIController`—in dedicated Python classes, we isolated bugs rapidly and could swap components (e.g., YOLOv11 prototype) with minimal refactoring.

M5 - Challenges Encountered & Solutions Applied

Challenge	Solution	Improvement
Excessive glare on lead-free joints	Added histogram equalization before HSV threshold	Mask stability ↑ 22 %
Insufficient data for through-hole variants	Merged 100s of online images + in-lab captures + 7× augmentation	Training samples ↑ 10×
Model latency spikes with Transformers	Profiled pipeline → removed XCA layers → tuned MobileNetV2 α=0.75	Inference < 30 ms (was > 600 ms)
Label ambiguity between Bad vs. Missing	Introduced explicit "Missing" class	Operator override requests ↓ 35 %

M5 - Uncompleted Items & Justification

1. **UART-driven auto-resolder loop**
    - *Status:* Verdict packets formatted; firmware stub exists.
    - *Justification:* Prioritized QA/QC stability will be enabled once joint-partition logic is hardened.
2. **Fine-grained defect taxonomy** (bridging, cold-joint, tombstone, tip)
    - *Status:* Additional labelling to be done.
    - *Justification:* It requires diverse examples and will train separate heads on the next dataset increment.
3. **Live KPI dashboard integration**
    - *Status:* GUI placeholder created.
    - *Justification:* Deferred until logging and CSV export meet potential performance SLAs.

Progress of the Final Project

By the time of the Capstone Expo (April 8, 2025), the project has reached a state of being ready as a prototype, being able to solder a few joints with human oversight and setup. Additionally, the vision and machine learning-based quality control was able to take photos and detect whether joints were good, bad or missing.

The demonstration video can be viewed at the following link: https://youtu.be/9CmH4HKnNF4

The full 5 minute Mac Drive video can be viewed at:
https://www.macvideo.ca/media/t/1_sphguv7f

IV. CONCLUSIONS & SUMMARY

Overall, this capstone project, the PCB Soldering Robot, provided the members of the group the opportunity to work on a complex, interdisciplinary mechatronics project, combining mechanical design, electronics, firmware, vision processing, and machine learning into one unified package. This is highly relevant, as almost all new products are mechatronic in nature, from automobiles to medical devices and children's toys. Therefore, the experience gained from this project will set each member up for success in their future careers.

Each of the group members had the opportunity to specialize in one of these areas related to their module and work with other team members to build something that no one engineering student could have alone. This produced a finished product that was greater than the sum of just five people's individual skill sets. Along the way, the team learned industry-relevant skills and software, including but not limited to, mechanical design for motion systems, SolidWorks, buck converter design, Altium Designer, hierarchical schematic design, firmware development, PlatformIO, C++, Git/Github Version Control, Raspberry Pi camera configuration, OpenCV-based image segmentation and pipelining, advanced data-augmentation and transfer-learning techniques, convolutional neural network architecture design and TensorFlow Lite optimization, YOLO/MobileNet model training and deployment on Raspberry Pi, UART/SPI/I²C communication protocols, Tkinter GUI development, model performance profiling and telemetry, and agile project management methodologies..

It has been a pleasure working alongside the Department of Electrical & Computer Engineering, and to be guided by Dr Shirani and Dr Chen in this journey.

Future Developments

If there is an opportunity to continue work on this project, there are many areas of potential improvement. As presented in the introduction, the market for this product has been considered. With financial support and resources, this proof of concept can be converted to a productionized design and a marketable product. Currently, the project meets the objectives stated in the bronze category; for future refinements, the goal outlined in the silver and gold categories would be easily achieved.

	Bronze	Silver	Gold
Defect Detection	50%	95%	99%
Pitch Support	2.54 mm pitch components	1.27 mm & 2.54 mm pitch components	0.8 mm, 1.27 mm, 2.54 mm pitch components
Soldering Time	<5 seconds/joint	<3 seconds/joint	<2 seconds/joint
Power Supply	Powered by 120 VAC, 15A outlet	Powered by 120 VAC, 15A outlet	Powered by 120 VAC, 15A outlet
Operation	Basic motor-sensor parallel processing	Real-time soldering correction	Real-time soldering correction
User Interface	Accepts through-hole data from eCAD	User-friendly UI, real-time operational metrics	Manual soldering adjustment via UI
Automated QC	Basic visual inspection with small dataset	Real-time inspection & soldering correction	AI-driven real-time feedback, corrections in 1 second
Electronics & Firmware	Basic communication between microcontrollers, sensors, actuators	Allows real-time defect corrections	Remote operation & monitoring

Figure 14: Originally planned goals for this project

REFERENCES & APPENDIX

[1] "Bambu Lab X1 Series." Available: https://bambulab.com/en/x1. [Accessed: Apr. 20, 2025]
[2] "CoreXY." Available: https://corexy.com/. [Accessed: Apr. 21, 2025]
[3] "PINECIL – Smart Mini Portable Soldering Iron (Version 2)," *PINE STORE*. Available: https://pine64.com/product/pinecil-smart-mini-portable-soldering-iron/. [Accessed: Apr. 21, 2025]
[4] "[No title]." Available: https://www.aliexpress.com/item/1005002562058108.html. [Accessed: Apr. 21, 2025]

[5] "coreXY Movement - Wokwi ESP32, STM32, Arduino Simulator." n.d. Accessed April 21, 2025. https://wokwi.com/projects/420211116816384001

[6] Zhang, J. *et al.* "Printed Circuit Board Solder Joint Quality Inspection Based on Lightweight Classification Network," *IET Cyber-Systems & Robotics*, 2024.