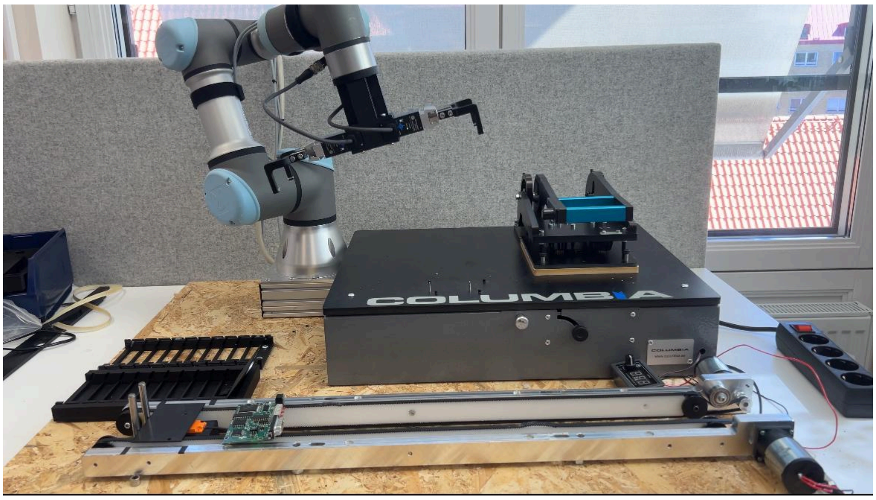


Automation of a PCB teststation with UR3

An Applied Study in Collaborative Robotics and Test Process Engineering

Degree project within the Bachelor of Science in Mechanical Engineering program

Alexander Forslind

Alexander Smith

Department of Industrial and Materials Science

CHALMERS UNIVERSITY OF TECHNOLOGY

Gothenburg, Sweden 2025

www.chalmers.se

Automation of a PCB teststation with UR3

An Applied Study in Collaborative Robotics and Test Process Engineering

Qestit Systems

Department of Industrial and Materials Science

Chalmers University of Technology

SE-412 96 Göteborg

Sweden

Telephone +47 (0) 31-772 1000

Abstract

Manual testing of printed circuit boards (PCBs) is a time consuming and repetitive process that can lead to ergonomic strain and inefficient productivity. This thesis explores how collaborative robots can be used to automate the PCB testing process in a lab scale environment. The objective was to design, build and evaluate a prototype test station using the UR3 collaborative robot to perform automated pick and place, handling, test initiation and sorting of PCBs.

The system was developed in two parallel configurations. The first used the UR3s built in interface called Polycope and interacted directly with the other components through the robots I/Os. The second setup combined a Raspberry Pi with Python programming and Real Time Data Exchange (RTDE) to enable external control, vision based positioning and more dynamic system behavior. A fixed camera mounted above the conveyor was used to implement edge detection for identifying the PCB's orientation while capacitive sensors and PWM-regulated motors supported component detection and transport.

To support the physical integration of components, a number of custom designed mechanical parts were prototyped using 3D printing. These included gripper mounts, sensor housings, camera holders, claws and sorting trays, all tailored to match the layout and functional needs of the station.

Testing showed that the UR3 robot could reliably automate the core steps of the test process. The PolyScope-based system provided high repeatability under fixed conditions, while the Python-based setup allowed for greater flexibility and modularity. Some challenges remained in the implementation of vision-based positioning, particularly in terms of distinguishing the PCB from the conveyor background. However the overall system fulfilled the defined functional goals and demonstrated the possibility of using a collaborative robot to improve efficiency, reduce physical workload and increase consistency in test workflows.

Keywords: Collaborative robot, UR3, PCB, Automation, PolyScope, RTDE, vision system, Python, Raspberry Pi, edge detection, robotic integration

Acknowledgements

We would like to express our sincere gratitude to our academic supervisor Hao Wang for his continuous support and guidance throughout the project. His feedback and encouragement have been invaluable to the successful completion of this thesis.

We also wish to thank Kristoffer Lagerström, our company supervisor at Qestit, for providing us with the opportunity and resources to carry out this project.

Furthermore, we are grateful to all colleagues at Qestit who assisted us during the project with orders, technical advice and for fostering a positive and collaborative work environment. Qestit support has been crucial in facilitating our practical work.

We would also like to extend a special thanks to Sven Ekered, who led our Automation Technology course at Chalmers University of Technology. His course inspired the choice of this project and has provided important theoretical foundations and practical insights that have been important throughout the work. Sven generously lent us parts that aided our development and was always available to answer questions during the project.

Table of Content

1. Introduction

This chapter provides an overview of the project, explaining why it is being carried out, what it aims to achieve and which specific questions the project group are trying to answer through the work. It also outlines what is included in the project and what lies outside of its boundaries. The work is carried out in collaboration with Qestit, a company specializing in quality testing and draws inspiration from the Automation Technology course at Chalmers University of Technology.

1.1 Background

In the field of electronics, printed circuit boards (PCBs) play a crucial role and they must be tested carefully before they can be used in any product. These tests ensure that the board functions correctly and meets quality standards. Today much of this testing is still done manually. An operator places the PCB into a testing machine, starts the test, waits for the result and then sorts the board into a ‘pass’ or ‘fail’ tray/pile. This manual process is time consuming, repetitive, physically tiring and can lead to mistakes if the operator is stressed or tired.

Manual testing also limits how fast a company can produce and test its products. If testing takes a long time it can slow down the entire production line. In addition, doing the same task over and over again can lead to injuries like wrist or back problems. This makes manual testing not only ineffective but also unergonomic for the workers. As products become more complex and demand increases companies are looking for smarter and more effective ways to handle testing.

Qestit Systems is a company that focuses on quality control in production and R&D. Their job is to provide tailored test solutions and systems to other companies to make sure they meet the right standards and to reduce defects. Recognizing the challenges inherent in manual PCB testing, Qestit aims to develop an automated test station. Qestit has set up a test fixture in their lab that mimics how PCB testing is done in real life. In partnership with the project group they launched this project to see if a collaborative robot (cobot) can automate this process. Cobots are made to work safely next to people, making them a good choice for this kind of task. This approach aligns with the broader Industry 4.0 trend towards smart manufacturing and increased automation.

1.3 Aim and Purpose

The purpose of this project is to explore whether collaborative robots can help improve the PCB testing process. Manual testing takes time, depends on the operator’s focus and skills and can lead to strain or injuries. As companies try to increase their output, this way of working becomes a bottleneck. Large industrial robots are one possible solution, but they are often too big, complex and expensive for smaller labs or setups. Cobots, like the UR3 are smaller, easier to program and include built in safety features that make them better suited for this kind of environment.

Even though cobots are used more often in industry, their role in PCB testing is still limited. One challenge is how they can manage small adjustments and precise handling when boards are not always placed exactly the same way. This creates a need to test how well cobots can work together with sensors and cameras to carry out such tasks.

The aim of the project is to build a working prototype of an automated PCB test station using the UR3 cobot. The station should be able to pick up circuit boards, place them correctly for testing and sort them based on test results. The idea is to automate key steps in the process and see whether this type of setup can reduce physical strain, increase testing capacity and work more efficiently than a manual solution.

If the prototype works well Qestit could show that this type of solution is worth exploring further in real production settings.

1.4 Research Questions / Major Goals

- Can a collaborative robot be used to automate the PCB testing process in a small lab setup?
- How can a test station be automated?

These questions are at the center of the project. By answering them the project hopes to give a clearer picture of how useful and practical cobots can be in this type of work.

1.5 Scope and Delimitations

This project is carried out in a small lab setting and is not meant to be used in a real factory at the moment. It is focused on building and testing a prototype to see if the idea works. In the project only one real PCB is available and the rest of the boards used for testing are 3D-printed models. This means the setup is quite simple compared to a full production environment.

The test fixture provided and used in the project is not a working fixture which means that pass/fail signals need to be imitated in code. It also does not try to handle large numbers of boards or run for long periods without supervision. Instead, it focuses on just a few key parts: Picking and placing PCBs using sensors to detect signals and using a camera/mechanical pins to make sure the board is in the right place.

Qestit provided the project group with a UR3 collaborative robot and an earlier thesis work that could be stripped for parts to build the prototype from scratch. This means that other components or robots might be more suitable for a fully developed test station, but the work had to be adapted to the available equipment. The sensor used in the project was recommended by our supervisors at Chalmers before the project started and became a natural choice based on available guidance and resources. For this reason, the pre-study is limited to the components used in the prototype and does not include a broader comparison of alternative robots or sensors.

Because of time, space and equipment limitations only one system setup was created and tested. Even though the scope is limited, the results can help show what works well and what challenges need to be solved before moving toward a more advanced and fully automated version of the system.

1.6 AI Disclaimer

In this project Github Copilot has been used as a correction tool, to compress/rewrite and given recommendation when writing the code. For example when choosing which filters to use and what parameters to focus on changing to improve the effectivity.

2. Pre-Study and Theoretical Framework

To develop an efficient and reliable solution it is important to understand how technical systems and principles work and which parts can be used together. The purpose of the pre-study is to build a theoretical foundation by searching for similar cases, finding components that solve the project's problems and to get an understanding on how each part contributes toward the objectives.

This pre-study is directly linked to the thesis goals of automating a test station thereby reducing cycle time, lowering manual workload and creating a system that can operate without human supervision. By exploring the technologies required to achieve this, such as robot integration, sensor technology, control systems and automated workflows, a framework is established for how the test station should be developed and validated.

2.1 Collaborative Robotics and Communication

Collaborative robots or cobots, have become increasingly relevant in industrial automation due to their ability to safely interact with humans and perform tasks without the need for physical barriers. This section introduces the general characteristics, advantages and limitations of collaborative robots, with a focus on their applicability in testing environments. It also presents real world use cases and outlines the role of the UR3 robot as a representative model within this category.

2.1.1 Collaborative Robots in Industrial Testing

Collaborative robots are designed to work side by side with human operators without the need for protective fencing or cages. Their built-in safety features, compact size and intuitive programming interfaces make them especially suitable for tasks such as pick-and-place, light assembly and repetitive motion handling. The UR3 model is often used in applications that require precision, flexibility and ease of use.

In various industries collaborative robots have been shown to reduce takt time, improve consistency and relieve operators from physically demanding tasks. For instance, the company Creating Revolutions implemented a UR3 to automate drilling, soldering and assembly, resulting in a failure rate below 1% and a fivefold increase in production capacity

(Universal Robots, n.d.-a). The Italian company MARKA automated its bottle capping process using UR3 robots and achieved a return on investment within one year (Universal Robots, n.d.-b). Similarly, Albrecht Jung GmbH in Germany used UR3 and UR5 cobots for component assembly in digital radios, improving both speed and handling accuracy (Universal Robots, n.d.-c).

These examples demonstrate the effectiveness of collaborative robots in production environments and highlight their potential in applications where precision, flexibility and reliability are required.

2.1.2 UR3 Collaborative Robot

The UR3 CB from Universal Robots is a compact six axis collaborative robot designed for light duty automation tasks in confined spaces. It has a maximum payload of 3 kg and a reach of 500 mm, making it well suited for applications such as testing, light assembly and pick-and-place operations in laboratory or bench-top environments (Universal Robots, 2016.). The robot has a repeatability of ±0.1 mm.

Weighing only 11 kg, the UR3 is portable and easy to integrate into small setups. It can be mounted on any flat surface (even upside down or on the wall) using four M6 bolts through its standard ISO flange. The robot can be mounted on custom bases to ensure appropriate reach within confined workstations. The robots compact footprint allows for efficient use of limited workspace without sacrificing access to key operations.

Programming can be done directly via the Polyscope interface, which is accessible on the robots touchscreen pendant. This graphical interface enables users to define movements and logic by writing code in URscript. The UR3 CB series also supports external control via TCP/IP protocols such as Real-Time Data Exchange (RTDE), which makes it possible to communicate with the robot through other devices like Raspberry Pi-based and other computers.

An overview of the UR3 robots specifications, including its dimensions, mounting interface and axis configuration, is provided in Appendix 1. Figure 1 below also offers a visual reference of the robots design and structure.

Figure 1. The UR3 collaborative robot from Universal Robots. The image shows the robot's compact form factor and six joint configuration. Source: Universal Robots (2016).

The UR3 robot is equipped with a control box that houses the main power supply, communication ports and I/O(Inputs and Outputs) interfaces. This box is installed separately from the robot arm and connects to the system's other components via standardized cable routing. See Figure 2 for an external view and the internal terminal layout.

The control box provides the following I/O capabilities:

- Digital Inputs (DI): Used to receive binary signals (e.g., from sensors or switches).
- Digital Outputs (DO): Used to send binary control signals (e.g., to grippers or relays).
- Configurable I/O (CI/CO): Can be set as either input or output depending on system needs.
- Analog Inputs/Outputs (AI/AO): Used for devices requiring variable signals such as voltage or current.

These interfaces can be programmed directly via the Polyscope interface or accessed externally through TCP/IP. This flexibility makes it possible to design modular automation systems that combine internal robot logic with external control and feedback.

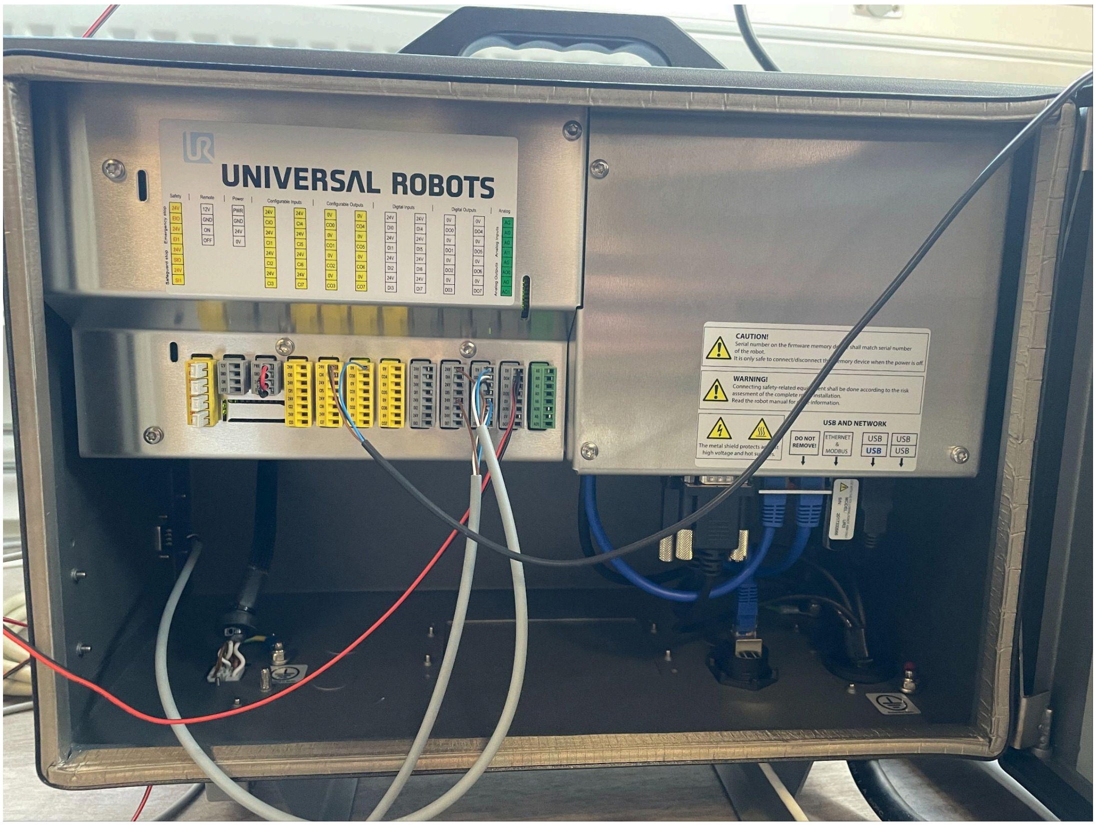


Figure 2. Internal I/O terminal layout inside the UR3 control box.

2.1.3 Advantages and limitations of collaborative robots

This section discusses the strengths and weaknesses of collaborative robots in industrial settings, particularly in relation to testing environments.

As previously mentioned, collaborative robots are designed to work safely alongside humans without the need for physical safety zones making them particularly suitable for test environments. They are easier to program and reconfigure than traditional industrial robots which makes them perfect for prototype builds. It also simplifies integration into existing production lines and enables quick adaptations to changes in production (Medical Design Briefs, 2023). Industrial robots are often too big, require much more security training and are more advanced to program to adjust to new changes on the production line.

In examples such as Instron's CT6 system, cobots are used to improve repeatability and data integrity in testing applications (Instron, 2023).

At the same time, cobots have limitations that need to be considered. Since less security training is needed to operate they need to have more strict security settings to remove the risks of hurting an operator. Therefore they are generally slower, smaller and have lower payload capacity compared to industrial robots which makes them unsuitable for applications requiring high throughput or heavy lifting. Industrial robots are often optimized for those types of tasks and are commonly used in high volume production where short cycle times are prioritized, according to Kassow Robots (n.d.) and Wevolver (2022).

Cobots on the other hand are well suited for low- to medium-volume production, lightweight work and in situations where safe interaction with humans is essential. This trade off makes them especially suitable for subtasks such as assembling small components, sorting or testing, where the need for precision, delicate handling and high repeatability outweighs the demand for maximum speed or strength (International Federation of Robotics, 2021).

One way to understand these trade offs is to compare manual and automated test stations. In manual test stations the operator is fully responsible for the whole testing process which includes pick and place, testing and the sorting based on the outfall from the test machines. This leads to several limitations such as high workload, risk of repetitive strain injuries and variation in test quality due to human error. In automated solutions with cobots many of these problems are eliminated.

Universal Robots' webinar (2021) presents examples of cobots being used for testing, assembly and dispensing. The result was a clear improvement in both cycle time and accuracy. Similarly to the result from Universal Robots webinar, Instron's CT6 system describes how cobot testing improves repeatability and data integrity.

2.1.4 Relevant safety standards for cobot integration

This section presents the key international safety standards that govern collaborative robot design and implementation.

Collaborative robots often work in direct cooperation with humans, which places higher requirements on how safety is handled. Although many safety standards for industrial robots are not fully adapted for cobots or do not need to be implemented, they are often used as a guide when evaluating risks and designing cobot systems.

The main ISO standard for robots is ISO/TS 15066 which is a supplement to ISO 10218. It specifies the limits for force and pressure during physical contact between the robot and another object. It also defines different types of collaborative modes such as monitored stop, speed and separation monitoring and power and force limiting (Swedish Standards Institute, 2016).

ISO 10218-1 and ISO 10218-2 are originally written for conventional industrial robots but they are widely used as a foundation for safety requirements, risk assessments and system integration in cobot applications (Swedish Standards Institute, 2025a, 2025b).

ISO 13849-1 is one of the most important standards when it comes to the safety of control systems. It focuses on the reliability of safety related parts such as emergency stop circuits and zone monitoring and is commonly used in the design of robot cells (Swedish Standards Institute, 2016).

These standards form the theoretical foundation for planning work stations, evaluating safety in collaborative robotics and are particularly important when designing systems that operate without physical barriers.

2.2 Robot communication

Communication between the robot and the control system is a central part of designing an automated test station. The UR3 robot supports multiple programming and communication methods which makes it possible to adapt the system depending on the level of complexity, flexibility and external integration required. Programming can be performed either directly through the robots built in interface or via communication protocols that allow control from an external system.

2.2.1 Internal programming in PolyScope

UR3 robots can be programmed through Universal Robots interface, PolyScope, which is accessed via the teach pendant. The simple interface allows the user to define movement sequences and logic through a point-and-click method without needing to write code manually (Universal Robots, n.d.-d). Programs created in PolyScope are stored on the robots internal controller and can be configured to run automatically when the system starts, which is suitable for static and repetitive operations.

PolyScope also supports the use of URScript, a text based scripting language that offers deeper access to the robots internal functions and parameters. Although URScript is automatically generated when using PolyScope it can also be written manually to customize motion control or system behavior (Universal Robots Forum, n.d.). However, URScript has limitations in more advanced applications such as real time coordination with some external sensors or vision systems where dynamic interaction is required. In such cases external control through TCP/IP may be more suitable.

2.2.2 Real-Time Data Exchange (RTDE)

To enable fast, reliable and real-time communication between the UR3 cobot and an external control system the Real-Time Data Exchange (RTDE) protocol can be used. RTDE is built on TCP/IP and allows continuous exchange of structured data between the robot's controller and external devices such as a Raspberry Pi or a PC. Examples of the type of data that can be transferred and adapted in real time include tool positions, digital input and output signals and various internal variables (Universal Robots, n.d.-e; Universal Robots Forum, n.d.-f).

RTDE can be used to coordinate communication between the UR3 and the other system in the setup. For instance, a test sequence could be managed by sending external signals to the robot, which then triggers a gripping action or waits for test results. In the same way, the robots status could be sent to the rest of the system to synchronize movement, update new coordinates based on the last position and control logic.

The RTDE protocol is flexible in how it can be configured. Users can choose which variables should be transmitted and which should be read which makes it suitable for custom applications. RTDE supports several programming languages, including Python which is helpful since much of the guides and provided frameworks from the Universal robot community uses python. In addition, Universal Robots provides documentation and a developer portal where there are many active user forums that can be useful during integration and troubleshooting (Universal Robots, n.d.-e; Universal Robots Forum, n.d.). Python is also a widely known language and supports more than 137 000 libraries (naukri, n.d) which makes it very flexible and appealing to industries to have in their production line.

2.3 Vision Systems in automation solutions

This section presents the fundamental concepts and technologies behind vision systems used in industrial automation. It explains how edge detection, camera configurations and vision-guided robotics enable precision and adaptability in processes such as positioning, inspection and quality control.

Vision systems play a crucial role in modern automation solutions, especially in contexts where high precision, repeatability and adaptability are required. By integrating camera-based image analysis with edge detection algorithms and object recognition robots can position themselves accurately and handle components even when the placement of objects varies from cycle to cycle.

Vision systems are widely used to support dynamic positioning, quality control and process control in applications such as part alignment, inspection and sorting. One of the key technologies within vision-based automation is edge detection.

2.3.1 Edge Detection and Positioning

Edge detection is a fundamental method in machine vision used to identify shapes and boundaries within an image field. Using algorithms such as Sobel, Canny, or Prewitt edges can be extracted so that a computer can calculate an object's position in a coordinate model (Milvus, n.d.; Sciotex, n.d.). Using the detected edges the system can determine the orientation, center point and alignment of an object with high accuracy.

This technique is well suited for industrial applications where fast and reliable results are required, such as picking up components from a conveyor belt. In automated systems that handle variation in object placement between cycles, edge detection helps ensure consistent pick up without relying on mechanical fixtures or predefined positions. This is especially valuable in electronics assembly and testing where tolerances are tight and contact surfaces must be approached with precision (HD Vision Systems, n.d.).

By interpreting visual information frame by frame, a robot equipped with edge detection can maintain a high level of repeatability and minimize the risk of mechanical errors, supporting continuous operation in environments where downtime is costly.

2.3.2 Vision System Configurations

Most collaborative robots do not include a built-in vision system but can be connected to external cameras mounted above or beside the work area, either in an "eye-in-hand" or "eye-to-hand" configuration. The camera sends image data to an external control system, which then calculates the coordinates of each component. Based on this input, the robot adjusts its path before picking up the board, eliminating the need for mechanical positioning or fixtures. See Figure 3 below.

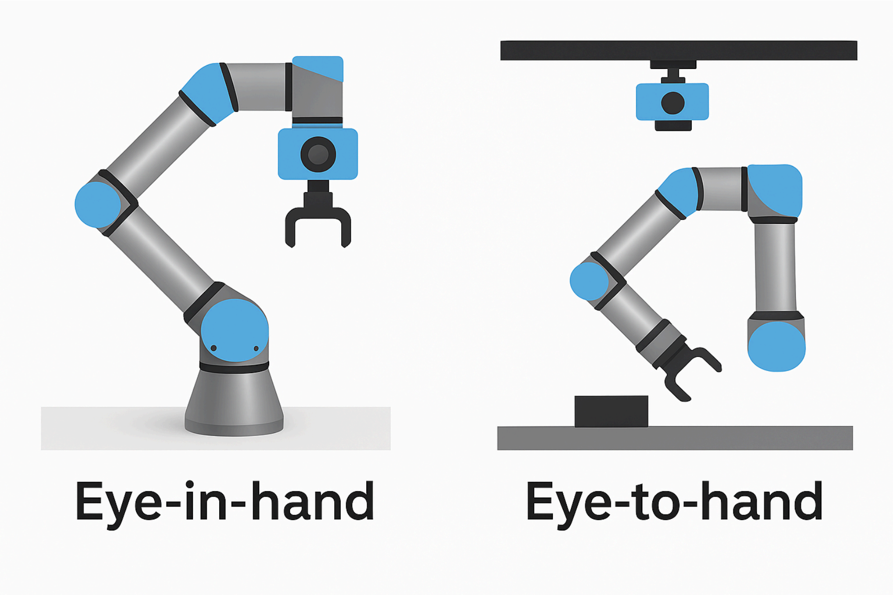


Figure 3. Comparison of the Eye-in-hand (left) and Eye-to-hand (right) vision configurations.

This type of solution is commonly used in commercial applications, such as the TM5-900 from Techman Robot which integrates a camera directly into the robot's wrist to enable flexible picking from a conveyor belt (Techman Robot, n.d.; Tech Briefs, 2017).

An eye-in-hand configuration, where the camera is mounted on the robot arm, provides greater flexibility. The same camera can be used throughout the test cycle. However, this configuration also presents some challenges. Since the camera is constantly moving, coordinate transformation becomes more complex and time-consuming. Additionally, the camera is more exposed to physical shocks or gradual misalignment due to vibrations or contact with operators, which may affect accuracy over time (Ensenso, n.d.; Bogue, 2022).

Eye-to-hand configuration uses a fixed camera mounted above or beside the work area. This setup is generally more stable and can offer higher positional accuracy because it operates within a constant reference frame. However, its field of view is limited to a single subtask or area which reduces flexibility. It is typically best suited for applications where object location is predictable and confined to a well-defined zone (Bogue, 2022; Wikipedia, ”Vision-guided robot systems”, n.d.).

2.3.3 Applications of Vision in Automation

In industrial production vision systems are used for more than just positioning. They are also applied to identify defects, measure dimensions or sort objects based on color, shape or pattern (HD Vision Systems, n.d.). The combination of 2D and 3D vision, depth information and AI-based analysis enables more advanced applications such as adaptive quality assurance or autonomous adjustment of process parameters.

In summary vision systems enhance the autonomy, precision and adaptability of automated processes. Whether used for positioning, inspection or quality assurance, they are a key enabler of smart and flexible production environments.

2.4 Object Handling and Transport in Robotic Systems

Automated systems often require components to be transported, positioned and manipulated with high precision and reliability. This section presents key technologies that enable such functionality, including motor control through pulse width modulation (PWM), object detection using capacitive sensors and mechanical optimization through the use of dual grippers.

2.4.1 Conveyor Control and PWM

Pulse width modulation (PWM) is a common method for controlling the speed of DC motors in automation systems and is often appearing in conveyor belts, fans and pumps. By turning the voltage to the motor on and off very quickly and adjusting how long the signal stays on during each cycle (known as the duty cycle), it is possible to regulate the motor speed in a smooth and energy efficient way (Electronics Tutorials, n.d.).

PWM is especially ideal for conveyor belts in small systems where the speed has to be regulated based on the workflow. Livinti and Mazen (2015) showed in an experiment that using PWM to control a DC motor driving a conveyor belt gave stable and responsive results.

Their setup used an Arduino and LabVIEW to send PWM signals to an H-bridge circuit and they were able to regulate the belt speed effectively across a range of duty cycles.

Thanks to its low cost, ease of implementation and reliability a PWM is a good fit for test stations and other automation systems where simple adjustable motion control is needed.

2.4.2 Capacitive sensors in industrial automation

Capacitive sensors are often used within industrial automation to detect the presence of both solid and fluid materials without any physical contact. Capacitive sensors generate an electric field between two electrodes. When an object with a different dielectric constant than the surrounding moves in proximity to the sensor the electric field and the capacitance changes which triggers a signal in the sensor (Texas Instruments, 2014).

Unlike inductive sensors, which only work with metallic objects, capacitive sensors can sense a wide variety of materials including some types of plastic, glass, wood, liquids and powders. This makes them very useful in applications where the variety of objects is large, such as in electronics test stations.

According to RealPars (2023) capacitive sensors can be used in a wide range of areas, some of these are the following:

1. Level detection of liquids, such as water, oil and chemicals
2. Detection of non-metallic materials such as plastic, wood, glass, ceramics and paper
3. Touchless controls in user interfaces, such as touch buttons
4. Proximity sensors in industrial applications
5. Detection of objects through non-metallic containers
6. Counting products in production lines
7. Quality control by detecting variations in material or thickness

The advantages of using capacitive sensors are that they are versatile, reliable and can be used to detect changes without any physical contact which makes it suitable for many different industrial and commercial applications.

2.4.3 Dual Grippers and Cycle Time Optimization

Using dual grippers in robotic systems is a proven method for reducing cycle times and increasing productivity in automated flows. For example with two grippers a robot can pick up a new object while placing a finished one and therefore eliminating the need for unnecessary returns or put downs thus saving valuable time per cycle.

In addition to shortening cycle times the double grippers also contribute to a more stable and continuous flow, increased repeatability and the possibility of unmanned operation for longer periods of time (OnRobot, n.d.-c; Robotiq, n.d.). It is a solution that is mechanically simple to implement but has a major impact on the overall system efficiency.

Industrial applications clearly demonstrate the benefits of this strategy. An example of this is the Danish company Osvald Jensen managed to reduce the cycle time in its CNC machining from 27 to 15 seconds by using dual RG2 grippers from OnRobot, an improvement of approximately 44 percent (OnRobot, n.d.-a). The Swedish company FT-Produktion was also able to halve its cycle time and save over 500 working hours in two months by implementing a robot cell with dual grippers (OnRobot, n.d.-b).

2.5 Control and Prototyping Platforms

This section presents two key technologies frequently used in industrial automation development and prototyping: the Raspberry Pi platform for control applications and 3D printing for rapid component fabrication and iterative testing.

2.5.1 Raspberry Pi in automation systems

Raspberry Pi is a small, cheap and powerful single board computer. The Raspberry Pi has had a breakthrough within prototype building and industrial automation thanks to its flexibility and low cost. It is primarily used as an alternative to PLC systems in simpler automation solutions.

Through the Raspberries GPIO pins it can read and send digital signals which makes it possible to effectively communicate with other units and parts. The Raspberry’s strong compatibility with Linux based OS and support for programming languages makes it easy to adapt and customize the system for the desired use (Control.com, n.d).

The industry has shown a big interest in Raspberry Pi. According to Predictable Designs (n.d.) are Raspberry Pi used in everything from monitoring production lines to energy management systems. The Raspberry Pi community offers resources and support for both private and industrial use, help with choosing the right components and help with coding and.

2.5.2 3D-printed components in prototyping

Additive manufacturing, in the form of 3D printing has become a key tool in product development and especially in prototyping. The technology enables fast and cost effective production of components with complex geometries directly from a digital CAD model. This is extra useful in projects where continuous iteration and adaptation are required.

Several sources highlight the role of 3D printing in streamlining prototyping within automation. Protolabs (n.d.) emphasizes that additive manufacturing shortens lead times, reduces development costs and simplifies testing during early stages of product development. Replique (2024) demonstrates how 3D printing is used to produce customized grippers and end effectors tailored to specific robot applications. Similarly, UltiMaker (n.d.) points out that the ability to modify and reprint components directly in the production environment increases flexibility and speeds up iteration cycles.

3D printing is often used to develop lightweight, application specific parts that meet the payload and dimensional constraints of collaborative robots. This flexibility allows engineers to test multiple solutions in short timeframes, improving both technical performance and development efficiency.

3. Method

This chapter describes the method used for developing, assembling and evaluating the automated test station. It outlines the system architecture, hardware setup, software control and testing procedures that were followed throughout the project.

The aim is to provide a step-by-step understanding of how the solution was built and verified with a clear link to the theoretical framework presented in Chapter 2. Each section corresponds to a specific phase or subsystem in the development process, from system overview and hardware configuration to software programming, calibration and final testing.

3.1 System Overview

This project focuses on the practical implementation of a collaborative robot system designed to automate a circuit board test station. Two different system configurations were developed, both enabling automated pick-and-place, testing and sorting of printed circuit boards. The first configuration was written in python and executed through a Raspberry Pi and the second configuration through the UR3 internal interface polyscope. The system design was based on the integration of several key subsystems: A UR3 robot, a vision system with camera integration, a dual gripper solution, a Raspberry Pi-based control platform, capacitive sensors and a motorized conveyor belt controlled via PWM.

3.1.1 General System Design

The test station was structured as a modular lab setup to allow flexible prototyping and rapid iteration. A block diagram illustrating the Raspberry Pi-based configuration is provided in Figure 4.

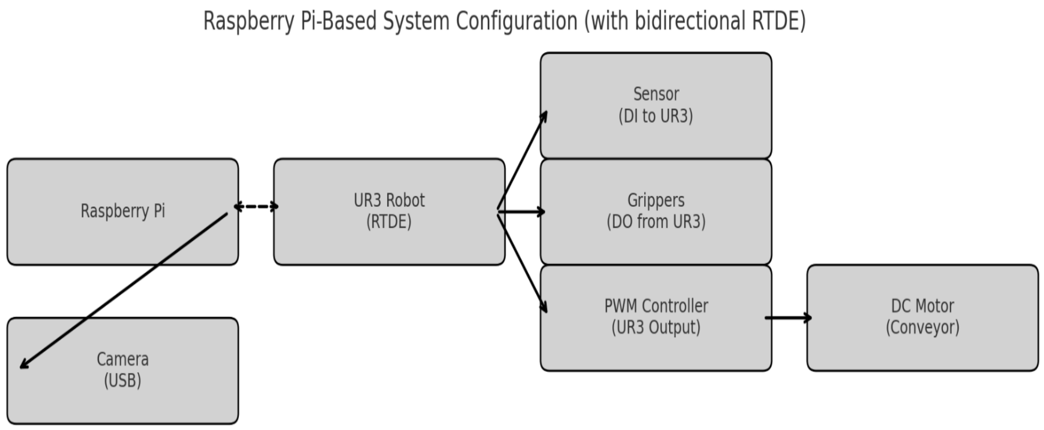


Figure 4. Block diagram of the Raspberry Pi configuration.

The block diagram shows the relationships between the cobot, camera, sensors, motor and Raspberry Pi. The Pi communicates with the UR3 via RTDE, while all I/O is handled by the robot. Only the vision system is connected directly to the Raspberry Pi via USB. The robot arm was mounted on a stable wooden platform positioned next to the test fixture and conveyor system, ensuring that it could reach the pickup zone and sorting tray without interfering with other components.

To control and communicate with the various subsystems a Python script was using the robots RTDE port. The python script was written on a computer and transferred to the Raspberry Pi. The RTDE protocol enabled precise synchronization between the robots actions and external events such as conveyor movement or sensor input. For testing code and to work from distance the python script and polyscope code was run through Oracle VirtualBox which is a virtual machine. By running the code in a virtual machine the code could be verified to be both safe and to work properly.

A camera was installed above the conveyor to detect the position of the PCB before the robot performed pick-and-place operations. The camera provided vision which the code processed and calculated into coordinates which was sent to the robot. Capacitive sensors were placed in the fixture to confirm correct PCB positioning and to simulate test pass/fail signals.

In parallel, a fully functional control sequence was also developed entirely in PolyScope, shown in Figure 5.

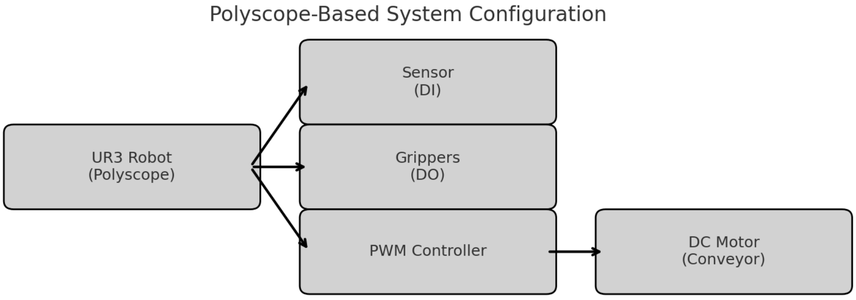


Figure 5. Block diagram of the Polyscope configuration. The UR3 handles all control, using digital I/O signals for the sensor and grippers and built-in PWM for motor regulation.

This version used fixed metal pins installed at the end of the conveyor to physically align each PCB in a consistent pickup position. The UR3 robot then performed the full test cycle using only digital I/O signals and its built-in PWM motor control.

The station was designed to operate autonomously with the help of a mechanical solution to the problem with rotated PCBs that can occur. When a PCB was detected by the capacitive sensor the robot would retrieve it from the conveyor where it had been aligned by the pins mounted at the end of the conveyor. The robot then places the PCB in the test fixture, waits for a simulated test result and then sorts it into a pass or fail tray depending on the signal received.

3.2 Prototyping

This section describes the physical development of the automated test station. Each component was prototyped and refined over time to ensure compatibility with the overall design and functionality of the system. The following subsections detail how each mechanical and electrical subsystem was developed and integrated.

3.2.1 UR3 Robot and Test Station

The UR3 robot and the test fixture were mounted on a robust wooden baseplate to ensure stability. This base served as the physical foundation for the entire project and offered enough surface area to integrate all essential subsystems. Its dimensions were selected to match the working area of the UR3 robot ensuring reach to the conveyor belt, the test fixture and the designated sorting areas. The wooden platform allowed for attachment of sensors, cabling and mounts which made it well suited for development and reconfiguration during the prototyping phase. It also made it easy to move the whole project when needed for a demo.

3.2.2 Conveyor Belt and Motor

The conveyor system used in the test station originated from an older student thesis provided by the company about 10 years ago. The setup consisted of two parallel guide rails, a smooth fabric belt and a pair of 24V DC motors (model RS 440-313) mounted on smooth, toothless metal drive pulleys. The frame was secured directly to the wooden base plate using screws. The spacing between the rails was intended to stabilize circuit boards during transport while also allowing room for sensors between them.

During early testing some issues with the original drive system became apparent. The combination of a smooth belt and toothless pulleys resulted in insufficient grip under certain conditions and could cause PCBs to drift slightly sideways. These minor alignment errors affected the robots ability to perform consistent pickups. As testing progressed the condition worsened due to increasing wear on the belt, which further reduced traction and led to more frequent slippage and uneven motion. Eventually, one of the belts failed completely and broke in half.

Following the failure both belts and pulleys were replaced. The new configuration used a rubber timing belt with tooths together with toothed metal drive wheels. This significantly improved the grip and ensured consistent and repeatable positioning of the PCBs. The original motors and guide rails remained unchanged.

To enable adjustable speed and synchronize motion with the robot a PWM controller (model HW-687) was implemented. As described in section 2.4.1, pulse width modulation enables voltage control through duty cycle adjustment, allowing smooth starts and stops without feedback loops.

The PWM controller was connected to digital output 4 on the UR3 robot enabling the conveyor to be controlled directly through robot interface or code (python and polyscope). The robot triggered the PWM signal which in turn regulated power to the motors in real time.

3.2.3 Grippers and Mount

Initially the robot was equipped with only a single gripper. However both the project team and the company agreed early on that the station should be optimized using a dual gripper configuration. This is also supported by the pre-study in section 2.4.3. Since commercially available dual gripper mounts were either extremely expensive or poorly designed in terms of angle and compatibility with the project's system, a custom solution was developed. Two prototypes of a 3D-printed mount were created to hold two Festo electric grippers. One prototype was designed to be more robust with an appealing and integrated look. While the second focused on being as minimal and cost-efficient as possible.

3.2.4 Capacitive Sensor

A capacitive sensor (ifm KQ6001) was implemented to detect the presence of a PCB as it arrived at the pickup position. The sensor was mounted beneath the conveyor, centered between the guide rails and adjusted to ensure stable detection of circuit boards without interfering with the PCBs.

The sensor output was wired to the I/O digital input 4 on the UR3 robot. During implementation the input signal was tested using both manual and program-triggered movement of test boards to verify if it was in the right spot to detect PCBs. The sensor was integrated into the two types of codes to serve as a condition for initiating pick up sequences, ensuring that the robot only attempted gripping when a board was properly in place.

3.2.5 Positioning Pins and Mechanical Adjustments

To ensure correct positioning without a vision system for the polyscope configuration and early stages of python, two fixed metal pins were mounted at the end of the conveyor. These mechanical pins aligned each PCB in a known pickup location.

3.2.6 3D Printed Components

Several custom components were developed using fused deposition modeling (3D printed) with PLA filament. These included mounts for the dual gripper system, camera holder, a sensor case and pass and fail trays. All components were designed using Catia V5 and prepared for printing in PrusaSlicer, as the available printer at the company was a Prusa model.

During initial testing dimensional deviations between CAD and printed parts became clear. As a result the designs were iteratively adjusted with slightly increased dimensions to ensure proper fit. Many of the components, particularly those used for mounting or casing, required multiple prototype versions before achieving a reliable final geometry.

Material selection focused on balancing durability with ease of prototyping. PLA was chosen due to its availability and sufficient mechanical strength for the low-stress conditions in the test station. Critical parts such as the gripper mount were reinforced with higher infill settings and thicker walls to ensure structural integrity during repeated operation, while the capacitive sensor casing was used with a low infill and less material.

During prototyping an issue was discovered where the 3D-printed PCB models failed to activate the capacitive sensor. Since the models were made from PLA, a non-conductive plastic they lacked the capacitive properties required for detection by our sensor. This was not initially considered as the project expected to use multiple real PCBs which would naturally trigger the sensor. When no additional real boards could be acquired, a simple workaround was implemented by gluing a small metal ring to the underside of each printed PCB. This modification allowed the sensor to detect them reliably without requiring any changes to the hardware.

3.3 Robot Programming

This section describes how the UR3 robot was programmed using two different control strategies, one based on external Python code and another using the built-in PolyScope interface. The programming efforts aimed to establish a reliable pick-and-place sequence, simulate testing and enable sorting depending on testing status. Additionally a virtual environment was used to test and verify logic before deploying to the physical robot.

3.3.1 Python-Based Control System

The Python-controlled system was written on a computer in the program visual studio code and transferred on to a Raspberry Pi. The communication was using the Real-Time Data Exchange (RTDE) interface to send commands and receive data from the UR3. Python libraries that were used were Sys, time, threading, random, kivy, socket, cv2, numpy , os, time and tkinter.

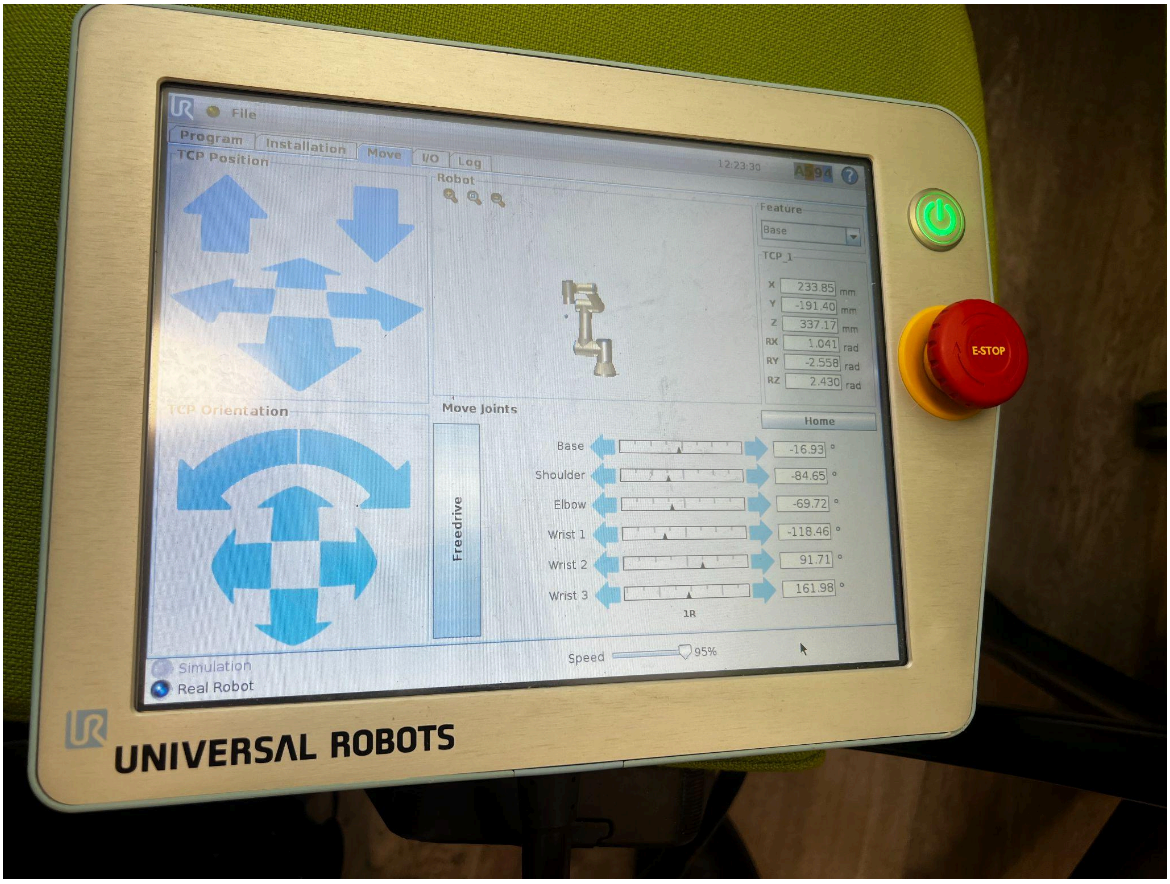
The test cycle was divided into smaller segments and functions for each segment were created. The segments were the “conveyor_loop” which controlled the conveyor and waited for the sensor status to change, “first_chip” which performs the the placement of the first PCB when the test station is empty, “last_chip” which empties the test fixture when no more PCBs are detected and “all_chip” which runs constantly from the “first_chip” function is called until the “last_chip” function is called. A link to the repository is provided for full view to the code [https://svn.addq4.se:8009/svn/SVNRoboticAutomation/trunk/].

In addition to the RTDE-based communication with the robot, a graphical user interface (GUI) was developed using the Kivy framework in Python. The interface allowed operators to trigger key sequences or run/pause the program with a button click, monitor system status and control the state of individual subsystems during testing.

In the Python-based configuration, the Raspberry Pi 5 served as the central processing unit for the control logic and vision system. It communicated with the UR3 robot via the RTDE protocol to synchronize movements and system state. The Pi also received visual input from the USB camera and processed the image stream in real time using OpenCV. All other hardware interactions such as sensor readings and motor control were routed through the robots digital ports.

3.3.2 Polyscope-Based Control System

The PolyScope interface used for robot programming includes both manual motion control and a structured programming platform. As seen in Figure 6, the interface provides direct control over the robots movement using arrows for linear and rotational axis adjustment via the teach pendant. This makes it possible to position the robot precisely when setting up waypoints.



Feature: Base

TCP_1

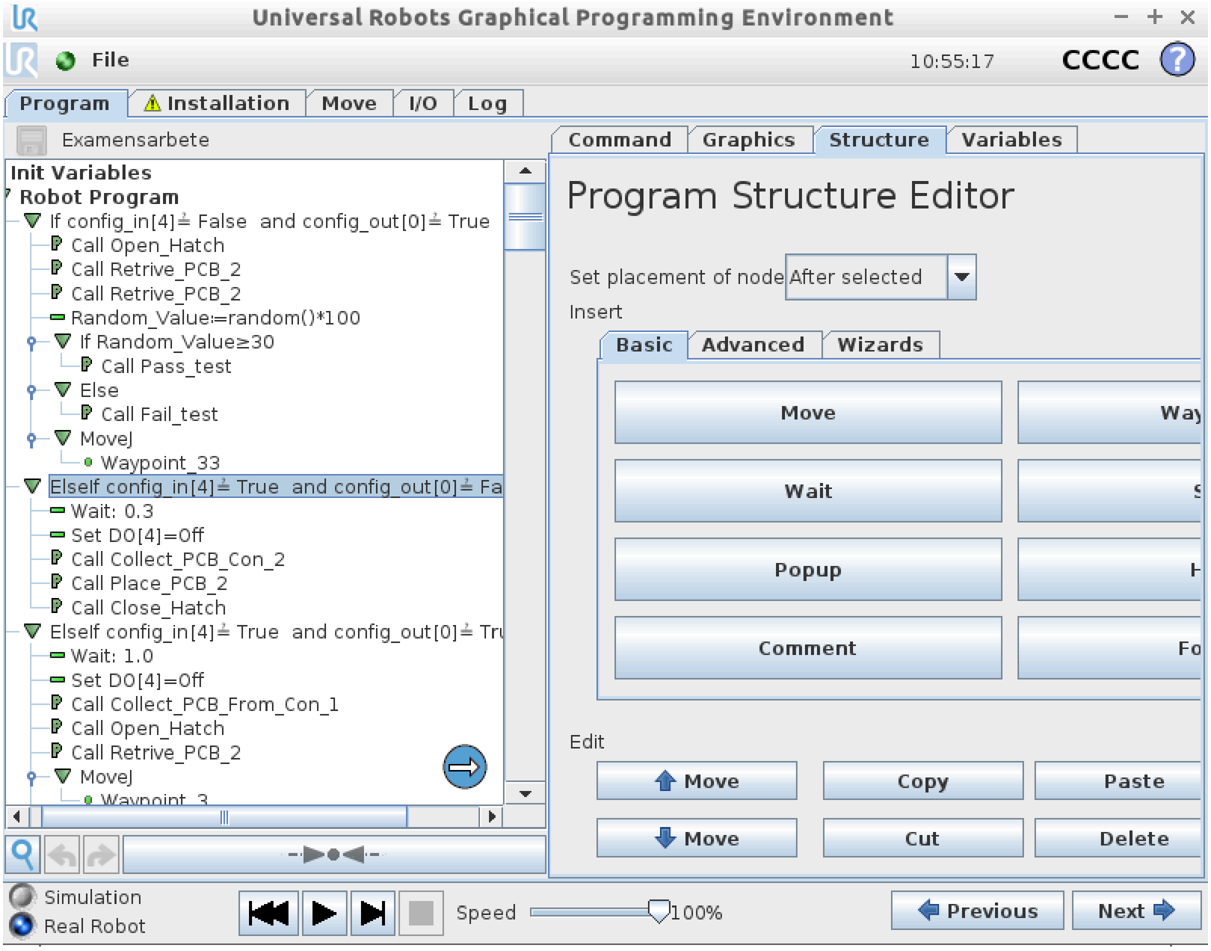
X 233.85 mm
Y -191.40 mm
Z 337.17 mm
RX 1.041 rad
RY -2.558 rad
RZ 2.430 rad

Base -16.93 °
Shoulder -84.65 °
Elbow -69.72 °
Wrist 1 -118.46 °
Wrist 2 91.71 °
Wrist 3 161.98 °

Speed 95%

Figure 6. PolyScope's manual movement interface for positioning and orientation of the robot's tool center point.

After waypoints and control logic are added, the interface transitions into a program tree view, shown in Figure 7, where each movement, wait command and signal operation is displayed in a hierarchical structure. One useful feature is the ability to create subprograms, which makes it easier to reuse recurring sequences and keep the main program clean and readable. This structured approach enables the robot to execute a complete test sequence without external input, if not asked for.



```
Init Variables
Robot Program
  If config_in[4]≐ False  and config_out[0]≐ True
    Call Open_Hatch
    Call Retrive_PCB_2
    Call Retrive_PCB_2
    Random_Value≔random()*100
    If Random_Value≥30
      Call Pass_test
    Else
      Call Fail_test
    MoveJ
      Waypoint_33
  ElseIf config_in[4]≐ True  and config_out[0]≐ Fa
    Wait: 0.3
    Set DO[4]=Off
    Call Collect_PCB_Con_2
    Call Place_PCB_2
    Call Close_Hatch
  ElseIf config_in[4]≐ True  and config_out[0]≐ Tru
    Wait: 1.0
    Set DO[4]=Off
    Call Collect_PCB_From_Con_1
    Call Open_Hatch
    Call Retrive_PCB_2
    MoveJ
      Waypoint_3
```

Figure 7. Program structure editor showing the waypoint sequence, conditional logic and signal operations used in the full test cycle.

This version of the system depended entirely on mechanical alignment. The PCBs position was fixed using the positioning pins described in **section 3.2.5**, which made it possible to use hard coded coordinates for pickup and placement without relying on the vision system.

All of the UR3s digital and configurable I/O signals were accessed and managed directly through the I/O tab in the PolyScope interface. As shown in Figure 8, this view provided an overview of the current state of each signal and allowed for quick manual toggling during setup and testing. It also made it easy to verify whether connected components such as sensors, actuators and the conveyor motor were functioning correctly.

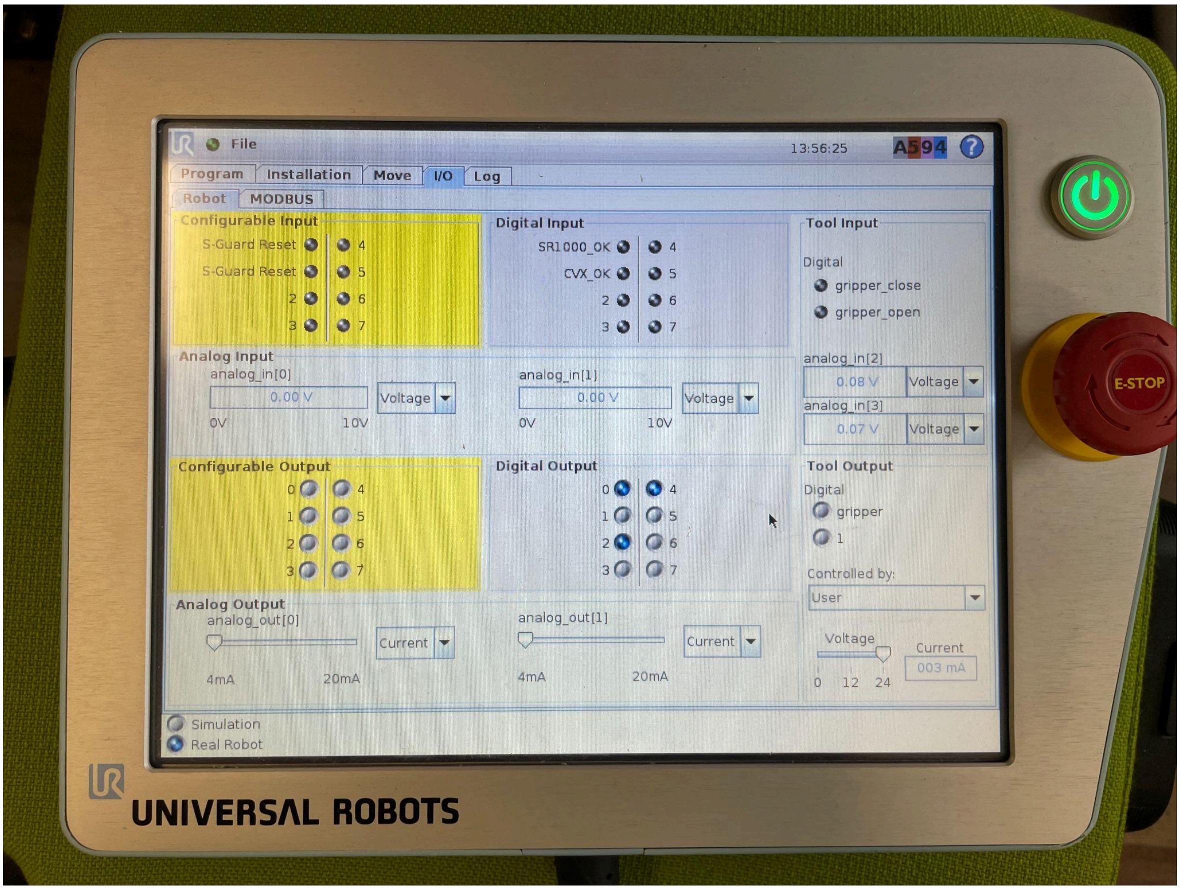


Figure 8. I/O overview tab in PolyScope used for monitoring and manually testing signals connected to the robots control box.

3.3.3 Virtual Machine Usage (Oracle VirtualBox)

A virtual machine running Oracle VirtualBox was set up to simulate the UR environment. This was used to test PolyScope programs offline and validate I/O logic before deploying the code to the real robot. By doing offline programming errors could be reduced and made the online programming safer.

3.4 Vision System

This section describes how the camera system was developed. The focus is on how the camera is used to detect the position and orientation of printed circuit boards, how different methods were applied in real time and how the pixel-based image coordinates were calibrated to match the robots workspace. No defect classification or quality inspection is performed, the system is limited to position detection and geometric alignment.

3.4.1 Camera Integration and Setup

The USB camera was mounted in a fixed position above the conveyor belt, an eye-to-hand configuration. This setup provided a stable reference frame independent of the robot's motion which made it suitable for consistent image analysis. The camera was connected directly to the computer and used exclusively in the Python-controlled version of the system. As shown in Figure 9, the camera was positioned to capture the entire pickup area of the conveyor. The system used OpenCV to process the live video stream and extract image data in real time.

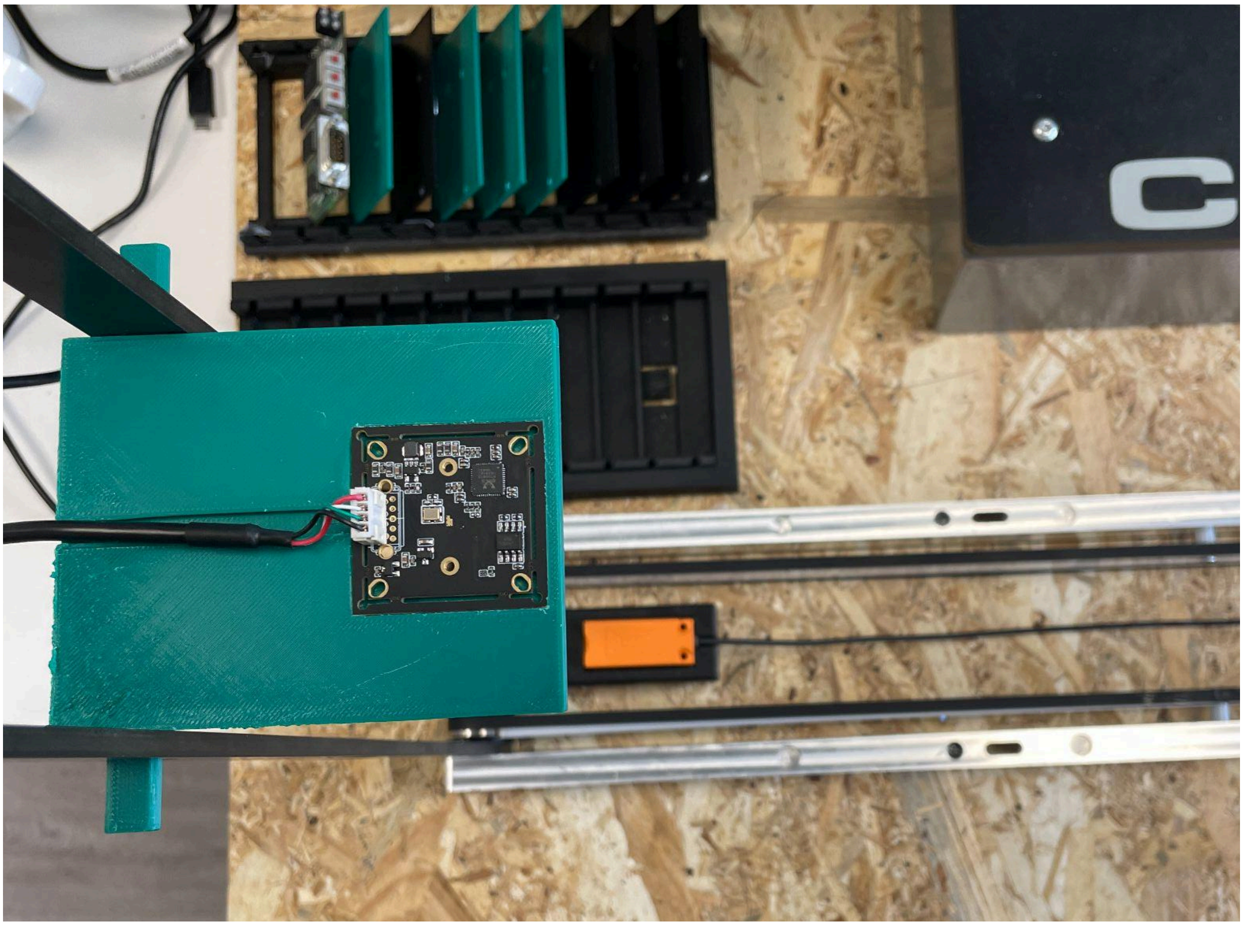


Figure 9. Camera mount and top-down view of the conveyor showing the visual field and pickup zone

3.4.2 PCB Detection and Object Localization

To identify PCB different methods were tested. For the positioning of the PCB all methods used the same procedure. The first method involved the Edge detection algorithm and one filter for size and one for shape. The second method used an image of the camera view without any PCB present which then was used to remove the background to only detect objects that were not in the background image. The third method that was tried was to use filters for size, shape, color and to look for contours. When a rectangular contour was identified in the video stream, the algorithm computed the geometric center and angle of the object which was then calculated into robot coordinates and used as the pickup coordinate for the robot.

3.4.3 Manual Calibration and Coordinate Mapping

To translate image pixels into robot coordinates the code had to be calibrated. A checkerboard was printed out and the parameters for the checkerboard were entered in the code. To map pixels to robot coordinates the robot was positioned on the four corners of the checkerboard, both the robot coordinates and pixels were entered in the code, OpenCV's "findhomography" was then used to map the robot coordinates and the pixels together.

The transformation parameters were then implemented in the Python control system, allowing the robot to adjust its TCP (Tool Center Point) coordinates based on real-time camera input.

3.5 Testing and Validation

This section outlines how different parts of the system were tested individually, how the complete workflow was validated and which methods were used to verify correct function and behavior. The purpose was to ensure that all subsystems operated as expected both independently and when integrated into the full automation sequence.

3.5.1 Unit Testing of Subsystems

Each subsystem was tested separately during development to confirm that core functions worked correctly before full integration. The UR3 robot was tested using both PolyScope and RTDE to verify accurate motion, I/O responses and gripper function.

The conveyor motor and PWM control were evaluated by varying duty cycles and direction commands to ensure smooth start-up and stop behavior. The capacitive sensor was tested with a real PCB, metal objects and the printed prototype PCB to confirm its detection range and sensitivity.

The USB camera was verified by changing the camera from the computer webcam to use the USB camera, by testing a simple OpenCV script and by testing the camera focus and resolution from different distances.

3.5.2 Integration Tests

Once each component passed unit testing, integration tests were conducted to evaluate the performance of the complete automated sequence. These tests included:

- Adjusting the PCB position with the mechanical pins
- Activating the gripper
- Placing the PCB in the test fixture
- Simulating a pass/fail test signal
- Sorting the board to the appropriate tray

The system was run fully automatic with the exception that the PCB had to be placed on the conveyor and removed from the trays. Each cycle was monitored for timing, synchronization, precision and consistency. Both the PolyScope-only configuration and the Python-controlled version were tested in this way to compare performance.

3.5.3 Observations and Verification Methods

During testing the results were logged manually using a checklist and video documentation. Key parameters such as pickup accuracy, conveyor response, sensor triggers and gripper performance were observed. When errors occurred the system was paused and diagnostic feedback from the robot or Python terminal was reviewed to identify root causes.

4. Results and Analysis

This chapter presents the final outcome of the implemented automation solution. Each section mirrors the structure of the methodology chapter to provide a clear comparison between what was planned and how the system performed. Key results related to system integration, prototyping, programming, vision system accuracy and functional validation are detailed. Where relevant, performance metrics and qualitative observations are supported by figures, photos, or video documentation. By following the structure of the development process, this chapter aims to demonstrate how the theoretical and technical choices led to a functional and verifiable solution.

4.1 Result of System Overview

The final system layout successfully integrated all functional modules described in the method. Figure 10 shows the complete test station in its operational configuration, including the UR3 robot, conveyor system, test fixture, vision system and sorting trays. The physical setup was optimized to ensure full reachability and minimize interference between components.

The UR3 robot was mounted on a wooden base giving it sufficient reach to access the pickup area, test fixture and both sorting trays. The conveyor system was positioned in parallel with the robot's base and powered via a PWM-controlled DC motor.

The block diagram in Figure 1 (see Chapter 3.1.1) represents the Raspberry Pi configuration which functioned reliably during testing. Capacitive sensors embedded in the conveyor belt confirmed when a PCB was in position. The vision system has not yet been successfully integrated into the system. A randomized event in python simulated a pass/fail signal which in turn triggered the sorting logic.

In the PolyScope configuration (Figure 2, see Chapter 3.1.1) mechanical pins ensured that all PCBs were placed in a consistent pickup position. This allowed the robot to complete the full cycle using internal I/O signals without needing vision support. Although less adaptable to part variation and wrongly positioned incoming PCBs the setup proved stable and suited for simpler applications involving uniform geometries.

To provide a visual reference for the final prototype, Figure 10 includes all key hardware components in their final mounted positions. Table 1 lists each component and its function, with numbering corresponding to the figure.

Table 1. List of components used in the final test station prototype. The numbering corresponds to the labels in Figure 10, showing each component's physical placement in the complete setup.

No.	**Component**	**Description**
1	UR3 robot and pendant	Central actuator for all pick-and-place operations
2	Festo EHPS-16A grippers	Dual parallel electric grippers with individual I/O control
3	Dual gripper mount	3D-printed bracket attaching both grippers to the robot flange
4	Conveyor belt	Transports PCBs to the pickup area
5	24V DC motor (RS 440-313)	Powers the conveyor via a toothed belt drive system
6	PWM controller (HW-687)	Regulates conveyor speed by adjusting the voltage duty cycle
7	Vision system (DFRobot 8000K USB camera)	Fixed overhead camera used for positional analysis
8	Raspberry Pi 5	Control unit for vision, logic and robot communication
9	Test fixture (dummy fixture)	Static placeholder used to simulate electrical PCB testing and timing
10	Capacitive sensor (ifm KQ6001)	Detects board presence and triggers test signals
11	PCB trays	Collect pass/fail boards after testing
12	Positioning pins	Align PCBs for the Polyscope configuration
13	HMI/GUI	Display with the interacting interface of the code
14	PCB (Printed Circuit Board)	Test object handled and evaluated

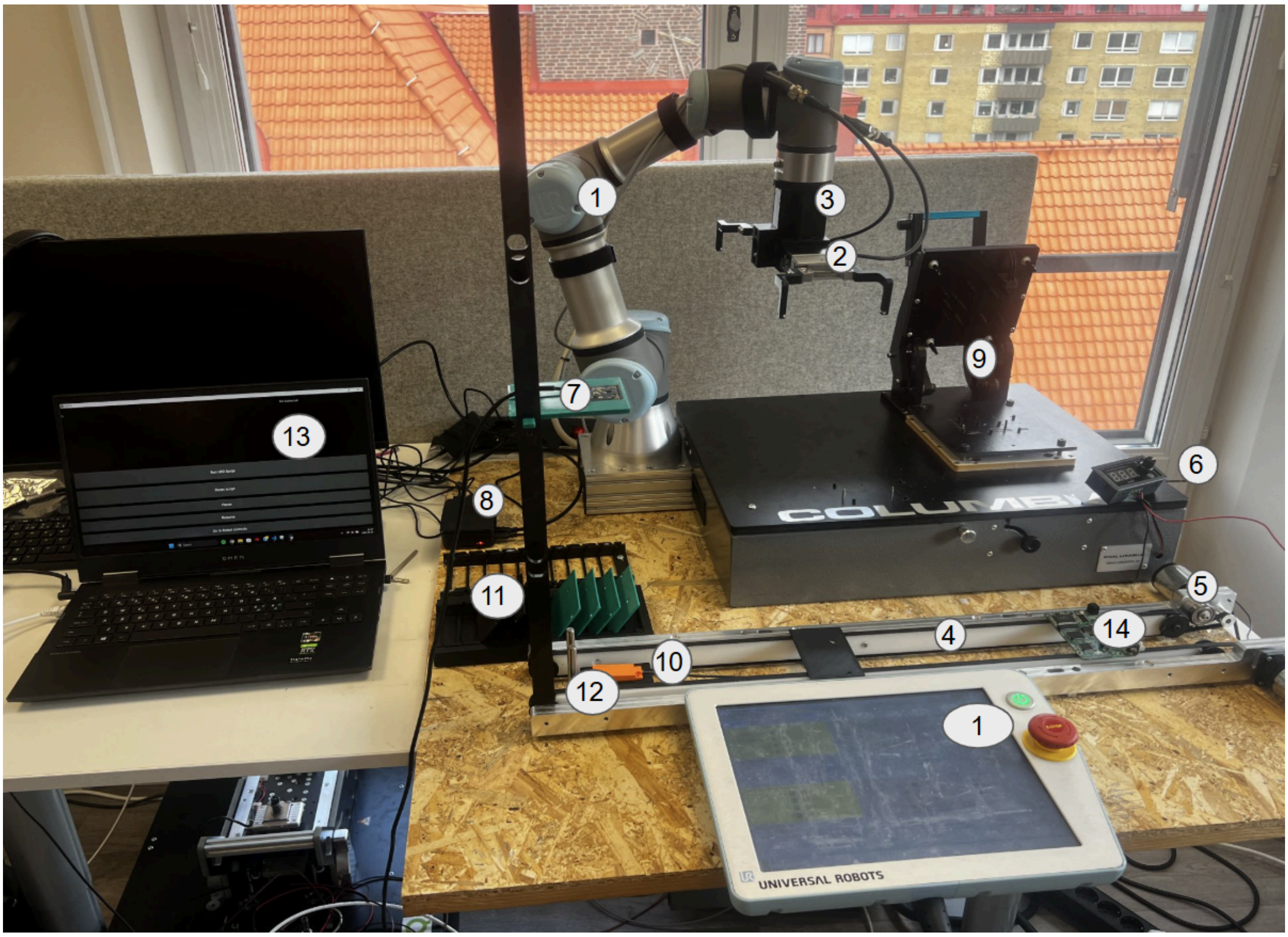


Figure 10. Photograph of the complete test station setup with numbered component references as listed in Table 1.

The coordination between the PolyScope-based and the RTDE-controlled Python systems functioned effectively during all test runs. This confirmed the successful synchronization between internal robot behavior and external control logic. The modularity of the system design proved beneficial as individual components could be adjusted or replaced without disrupting the overall structure.

A walkthrough video of the complete setup and operational cycle is available at the following link: https://youtu.be/S91IOP_09rE?si=_Mxy_j4toNEN_HhL.

This integrated design fulfilled the goal of creating a semi-autonomous test station capable of performing the full testing workflow with minimal human input supporting the theoretical concepts discussed in Section 2.2 on robot communication and modular integration.

4.2 Result of Prototyping

This section presents the outcome of the prototyping process, focusing on the performance and reliability of each developed component. The results reflect how well each subsystem functioned in practice and how they contributed to the system's overall capability and stability.

4.2.1 UR3 Robot and Test Station

The completed assembly of the UR3 robot and test station provided a stable and functional platform for automated PCB testing. The wooden baseplate ensured structural rigidity and reduced vibrations during operation, which helped maintain precise robot movements.

All key areas such as the conveyor, test fixture and sorting zones were within the robots working range which confirmed the initial layout design. The base also allowed easy mounting of sensors and components, supporting iterative development and quick adjustments during testing.

Its portability made it possible to relocate the entire setup for demonstrations without affecting alignment or functionality. Overall, the station proved reliable and adaptable as a foundation for further system integration.

4.2.2 Conveyor Belt and Motor

The conveyor system functioned reliably after the initial hardware updates. The original belt had to be replaced due to breakage but the new timing belt and toothed drive wheels provided improved traction and stability for circuit board transport.

The motors responded well to control signals and produced smooth and consistent motion under varying speed conditions. The PWM controller allowed precise adjustment of the motor speed by modifying the input signal which is important to control the flow of products on the conveyor belt.

Mechanical alignment between the conveyor rails and the test station remained stable throughout testing. The spacing between rails allowed for successful integration of sensors without interfering with board transport.

Control through the UR3 robots digital output worked as intended and signal transmission to the PWM module enabled coordinated motion between robot and conveyor.

A photo of the final conveyor setup is shown in Figure 11, illustrating the timing belt, toothed drive wheels and how the motor is mounted to the baseplate.

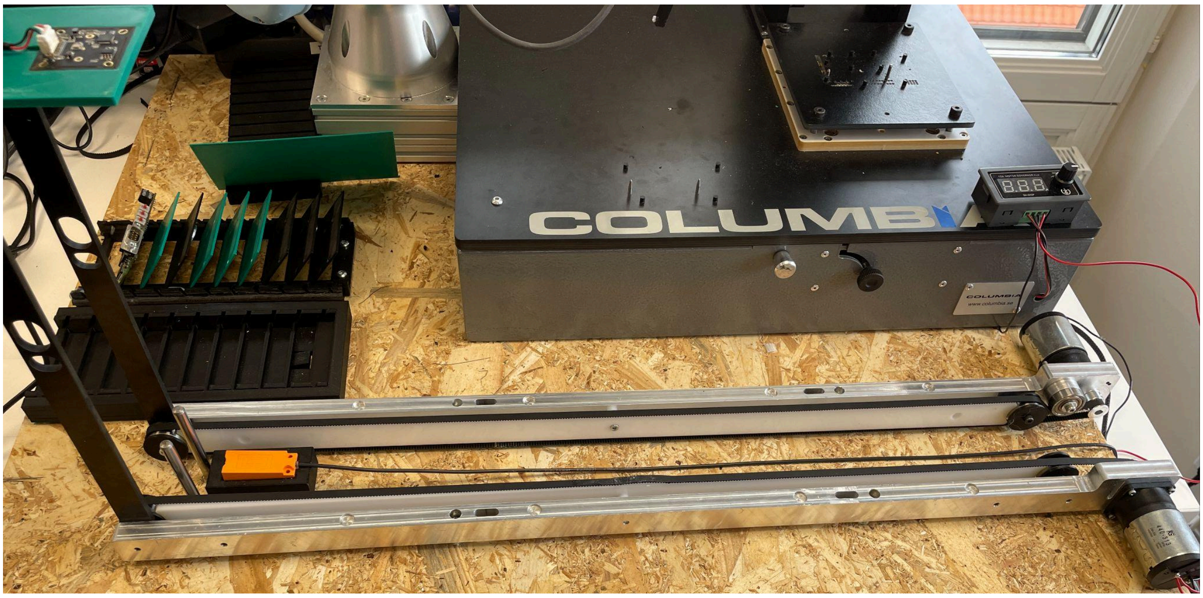


Figure 11. Final conveyor with timing belt and toothed drive wheels.

4.2.3 Grippers and Mount

The dual gripper system, shown in Figure 12, consists of two Festo electric grippers mounted on a custom 3D-printed mount. Two prototype mounts were developed during testing. The first one with a more robust design and the second one that prioritized reducing material and reducing the effective tool width of the grippers and mount. The final choice was the second version since the low forces involved in PCB handling made the mechanical reinforcement in the first version unnecessary and unwieldy.

This more compact design also improved system performance by reducing the effective tool width which reduced the rotational clearance needed and enabled tighter motion paths. As a result the robot could operate within a smaller envelope and complete cycles slightly faster.

The grippers were positioned 180 degrees opposite each other which allowed the robot to alternate between pick and place operations without returning to a central pose. This contributed to a reduction in overall cycle time and improved process flow. The dual gripper solution reduced idle time and increased throughput during test cycles.

Figure 12. Dual gripper mount with 3D-printed claws for alternating PCB pick and place.

4.2.4 Capacitive Sensor

The capacitive sensor detected the presence of PCBs as they passed over the sensing area. During testing the sensor triggered at the correct position which provided a digital input to the UR3 robot. This ensured that pick actions were only executed when a board was confirmed to be in the right place.

The wiring and integration with the robots digital input worked as intended. The sensor’s response time was fast enough to avoid delays in the sequence and no false positives were observed under normal operating conditions. A photo of the sensor placement is shown in Figure 13. The image shows the sensor positioned beneath the conveyor between the guide rails, mounted in its custom 3D-printed case which is described further in section 4.2.6.

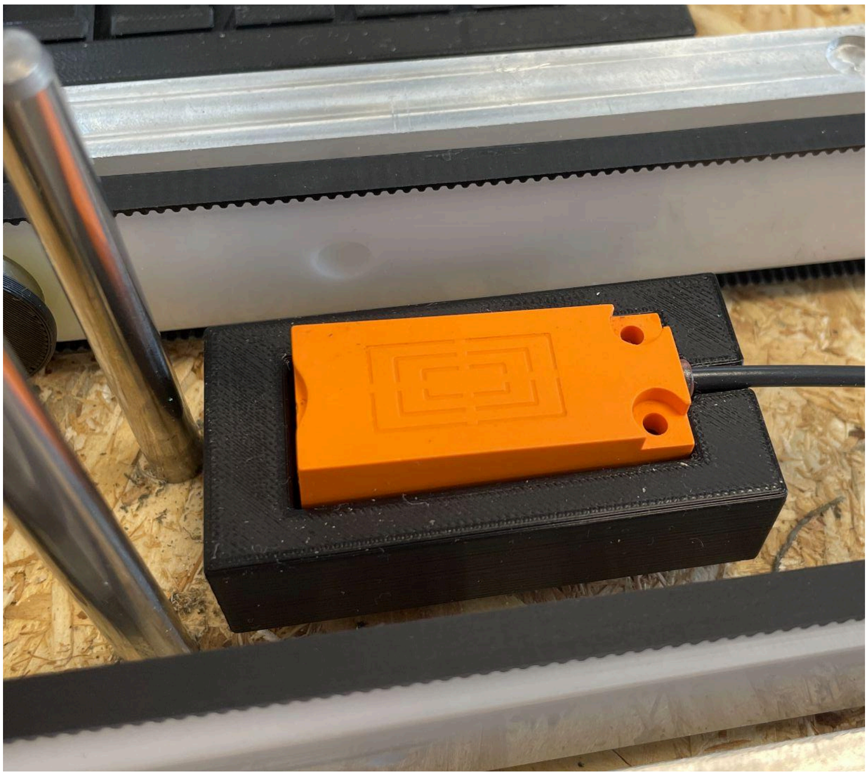


Figure 13. Capacitive sensor mounted under the conveyor in its 3D-printed housing.

4.2.5 Positioning Pins and Mechanical Adjustments

The positioning pins performed well during testing and provided a reliable solution for mechanical alignment of the PCBs. Each board consistently reached a repeatable stop point at the end of the conveyor which allowed the robot to perform gripping actions with high positional accuracy even without visual feedback. This solution proved to be both cheap and effective.

There were two instances where the pins function did not meet the needed functionality. The first instance was if an incoming PCB was very rotated on the conveyor the PCB could get stuck between the pins which would interrupt the production flow. The second was if the incoming PCB was positioned too much to one side which resulted in the grippers missing the PCB when trying to pick up.

Figure 14 shows the final placement of the pins at the end of the conveyor. Their simple yet effective function contributed to stable and repeatable system behavior throughout testing.

Figure 14. Fixed positioning pins at the conveyor end used to align PCBs prior to gripping.

4.2.6 3D Printed Components

Several components in the test station were manufactured using 3D printing to enable fast prototyping and adaptation to the mechanical and electrical subsystems. The ability to customize and 3D print parts based on the project's needs proved to be invaluable since most components for robots are very expensive, the whole system was innovated along the project duration and was built for a very compact environment.

The dual gripper mount used to hold two Festo electric grippers was designed in two different prototypes and is presented in section 4.2.3. Its geometry allowed compact spacing between the grippers while maintaining structural stability.

Two-part tray holders were printed for sorting tested PCBs. Each consisted of a fixed outer frame and a removable inner tray, making it possible to replace full trays without interrupting the system. The final tray design is shown in Figure 15.

One unexpected issue with the printed PCBs was that they did not interact with the capacitive sensor. Since the material was non-conductive, the sensor failed to register their presence. To solve this, small metal bits were glued to the bottom of each board, which restored full functionality. This workaround proved quite effective and required no adjustment of the sensor hardware or logic. The metal bit could sometimes interfere with the pins in the test fixture and needed to be glued to a different position.

Figure 15. Two-part 3D-printed tray system with removable insert for full board collection.

A camera holder was designed to mount the USB camera above the pickup zone on the conveyor. The print allowed stable attachment for better focus and viewing angle. The holder is shown in Figure 16.

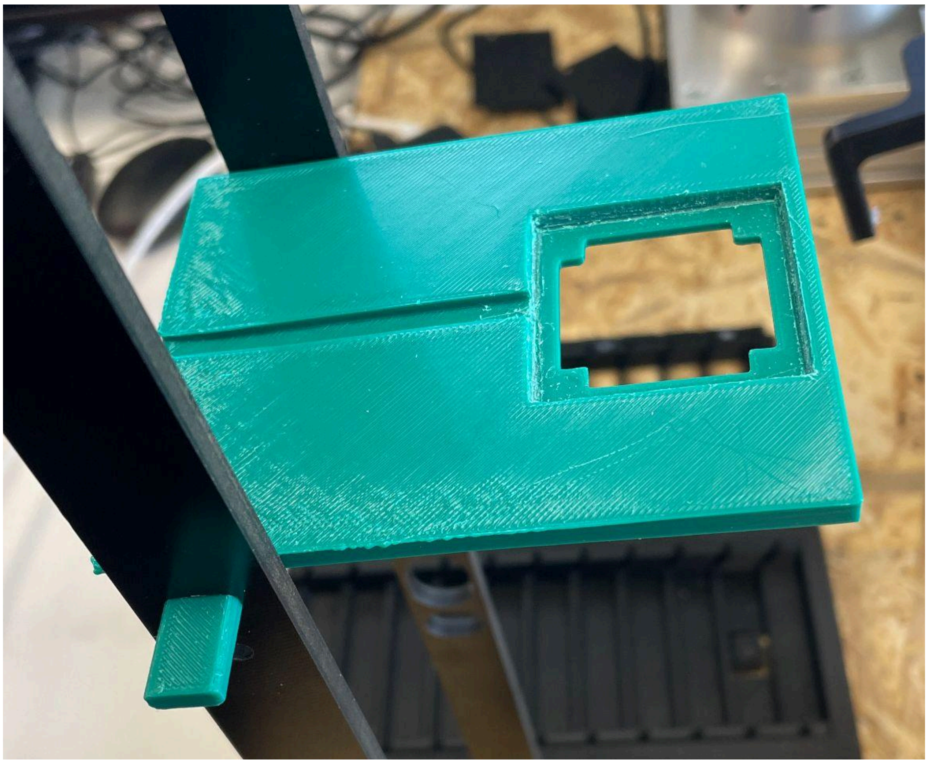


Figure 16. 3D-printed camera holder mounted above the PCB pickup area.

The claws used on both grippers were also custom printed. These were adapted to the PCB model geometry while remaining lightweight and easy to reprint. The final gripper claws are shown in Figure 17.

Figure 17. 3D-printed gripper claws adapted for PCB handling.

Lastly, a custom housing was printed for the capacitive sensor to ensure proper alignment and physical protection. The design also facilitated clean cable routing between the conveyor rails. This sensor enclosure is shown in Figure 18.

Figure 18. 3D-printed housing for capacitive sensor, mounted between the conveyor rails.

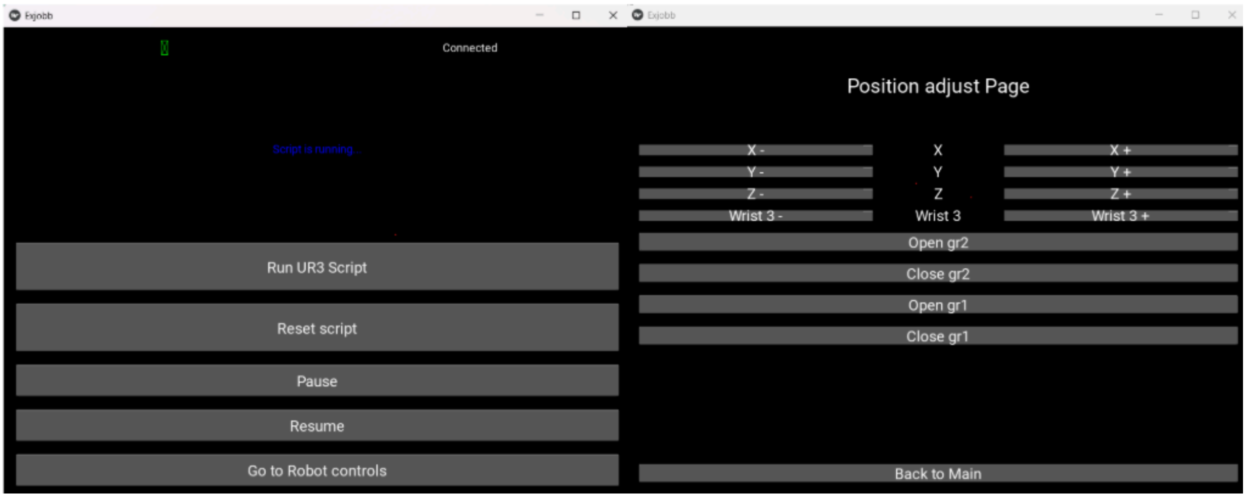
4.3 Result of Robot Programming

This section presents the outcome of the two programming approaches used to control the UR3 robot. The first one based on Python and external communication and the other using the internal PolyScope interface. Both implementations successfully executed the pick-and-place program. However their respective strengths and limitations became evident during testing. The following link leads to the Python code repository [https://svn.addq4.se:8009/svn/SVNRoboticAutomation/trunk/]

4.3.1 Python-Based Execution

The Python implementation, executed from a computer initially and later transferred to a Raspberry Pi, allowed for more modular and flexible control. The RTDE interface enabled real-time communication with the UR3. With Python libraries the functionality of the code could be customized exactly as the project needed to. The Python code also offered better opportunities for debugging and expansion which made it easier to adapt the program logic or integrate new components later in the process.

To operate and execute the program easier a graphical user interface (GUI) was developed using the Kivy framework. The interface allowed operators to start and stop test sequences, pause and play in the middle of a sequence, monitor the real-time connection status to the robot and trigger moving actions such as opening, closing the grippers or moving and rotating the robot. A screenshot of the final GUI is shown in Figure 19.



Connected

Script is running...

Figure 19. Screenshot of the custom graphical interface developed in Kivy, used for manual control and sensor monitoring.

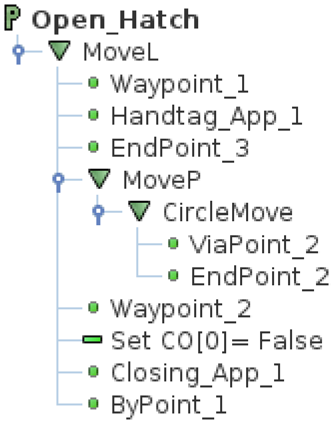
4.3.2 Polyscope-Based Execution

The PolyScope-based solution proved highly effective, successfully running multiple test cycles while managing all interactions through the robots built-in digital and configurable I/O. It communicated with the conveyor motor, capacitive sensor and hatch subprograms using only internal logic and signal control. To track the hatch position, a configurable input/output was implemented, ensuring the robot would always verify if the hatch was open before picking up a board. This functionality was structured into subprograms such as “Open Hatch” and “Close Hatch,” which were called repeatedly throughout the main program.

As shown in Figure 20, the “Open Hatch” subprogram demonstrates how reusable code blocks were used to maintain a clean and modular program structure. This approach helped improve readability and ensured that all necessary steps were coordinated correctly and kept the main code clean.

The robot completed multiple uninterrupted cycles without error. Although no vision system was used, the mechanical alignment pins described in section 3.2.5 ensured that the PCB arrived at the correct position. Due to the UR3s precision tolerance of ±0.1 mm (see section 2.1.2), some fine adjustments to the pickup coordinates were needed to ensure accurate placement on the test fixture’s alignment pegs.

Despite lacking flexibility for variations in PCB position or type, this approach was perfectly suited to the project's scope. With only one PCB type and consistent placement. The mechanical setup enabled fast and repeatable operation. The full PolyScope program is available in Appendix 2.



```
Open_Hatch
  MoveL
    Waypoint_1
    Handtag_App_1
    EndPoint_3
    MoveP
      CircleMove
        ViaPoint_2
        EndPoint_2
    Waypoint_2
    Set C0[0]= False
    Closing_App_1
    ByPoint_1
```

Figure 20. Subprogram in PolyScope used to open the hatch before PCB pickup. The logic is reused in several locations to maintain coordination and program clarity.

4.4 Result of Vision System

This section presents the outcome of the implemented vision system including calibration effectiveness and the performance of the edge detection method used for locating printed circuit boards during testing.

4.4.1 Calibration Effectiveness

The calibration of the camera was successfully done but since the first camera that was used did not work as needed the work on the vision system had to be postponed until the new camera was delivered.

After calibrating the camera, the script needed to be able to detect and separate the PCB from the conveyor belt and the surroundings. This proved to be a lot more difficult than expected. With different methods tried as described in chapter 3.4, the result was that when a PCB was placed on a solid color background or a background with less objects in the background the painted box around the PCB was stable and the code was able to give both coordinates and angle for all methods. But when placed on the conveyor the script had a hard time restricting the painted box to just the PCB which made the box flicker which in turn gave some correct positions and some wrong positions.

4.4.2 PCB Detection Performance

Two of three methods were able to detect the PCB on a solid color background. Canny edge detection provided clean and well-defined contours in most lighting conditions. The filter method was after adjusting the size filter correctly, able to detect the PCB. The background removal method did the opposite of what it was supposed to do. All objects that were not in the initial background image were made see through in the camera live feed instead of highlighted.

Since the camera is not yet able to separate the PCB from the background, the robot's coordinates and the camera's pixels have not been mapped together correctly but the script is able to return coordinates based on the camera's pixels. If correctly mapped together the system will be able to perform as intended and enable the robot to detect, pick and place PCBs with sufficient reliability on solid color backgrounds.

4.5 Result of Testing and Validation

This section presents the outcome of testing efforts, both at the subsystem level and for the complete system. The goal was to verify that the station could operate autonomously with sufficient accuracy and repeatability across multiple test cycles.

4.5.1 Subsystem Performance

Most components functioned as expected during isolated testing. The UR3 robot responded accurately to digital I/O signals and executed motion commands without drift or misalignment. The grippers actuated reliably and the dual-mount design allowed for seamless handoff between PCBs. The PWM-controlled conveyor motor maintained consistent speed regulation.

The capacitive sensor responded correctly for most objects but not to the 3D printed prototype PCB. To fix this problem metal rings were glued to the bottom of each 3D printed PCB. The first camera borrowed from someone at the company turned out to have a worse resolution and focus then needed for the purpose. To solve this problem the first step was to try and manually program the camera settings from the script but that did not suffice. Therefore another camera was purchased that fitted the requirements. The new camera produced clear images in real time.

4.5.2 Integration Testing Results

Both the PolyScope-only configuration and the Python configuration showed 100% mechanical reliability across 15 cycles, though it lacked adaptability to placement variation. Without vision input, the robot depended entirely on the fixed positioning pins to ensure alignment.

The Vision integrated configuration has not been completed because the code has problems separating the PCB from the conveyor belt.

4.5.3 Observations and Adjustments

Minor issues were observed during early integration primarily involving timing mismatches between the sensor signal and robot execution. These were resolved by adjusting wait conditions and loop delays in the Python control logic.

The system was tested using a full automation sequence recorded on video. A link to this demonstration has been made available in the following **Link** to illustrate a successful test run from detection to sorting.

5. Discussion

This chapter discusses the main findings of the project and reflects on the results in relation to the research questions and outlines limitations, failures and potential improvements.

5.1 Answer to the research question

This section answers the research questions from Chapter 1 based on how the system performed during testing. The goal is to evaluate whether the test station worked as intended and we met the goals of the project.

5.1.1 Can a collaborative robot be used to successfully automate the PCB testing process in a small lab setup?

The results from this project clearly demonstrate that a collaborative robot, such as the UR3 can be successfully used to automate PCB testing. The robot was able to consistently perform pick-and-place operations and sort the PCBs based on test outcome. The use of a collaborative robot enabled safe operation in close proximity to human operators and the size of the system remained compact. Although the prototype setup was limited in complexity and scale, the results indicate that the core functions of a test station can indeed be automated using a UR3 or a similar cobot.

5.1.2 How can a test station be automated?

The project has shown that PCB testing can be automated through multiple configurations. Two separate control approaches were implemented, one using Python in combination with RTDE (Real-Time Data Exchange) for external control and logic execution and another using the robot's internal PolyScope interface. Both methods successfully enabled autonomous operation, including PCB handling, sensor feedback and conditional sorting. The Python-based setup allowed greater flexibility and integration with additional hardware, while PolyScope offered a more direct and user friendly interface for basic sequences. These results confirm that test automation can be achieved using either internal or external robot control, depending on the system's complexity and user requirements.

5.2 Failures

One of the main shortcomings in the project was the incomplete integration of the vision system. While the Raspberry Pi camera was successfully mounted, connected and calibrated, it was not integrated in the final test sequences. The main issue was related to separation between the PCB and the conveyor belt in the setup. The background of the conveyor system, together with the material and color of the 3D-printed PCBs, made it difficult for the camera to separate them reliably.

Although the edge detection code worked well during development and performed accurately when tested on a separate table with clearer contrast, it struggled in the actual test station. Under controlled lighting and with different backgrounds, the system could detect shapes, orientations and positions as intended. However in the final mounted position above the pickup area, the camera could not consistently differentiate the PCBs from the conveyor surface.

As a result, the vision system was not included in the final integrated sequence. Instead, mechanical positioning using fixed pins was used to ensure reliable pickup.

Aside from the camera limitations, no major failures affected the project. Some hardware issues occurred. Such as the failure of one conveyor belt early in testing, but these were resolved through replacement and design updates. Most other challenges such as fitting tolerances in 3D-printed parts were addressed through normal iteration.

Another issue that emerged involved the 3D-printed PCBs used during testing. Since these were made from non-conductive plastic, they did not trigger the capacitive sensor, which relies on materials with measurable capacitance. This was not anticipated initially, as the team expected to receive multiple real PCBs for testing, which would have worked directly with the sensor. When no additional real boards could be obtained, a workaround was implemented by gluing a small metal washer to the underside of each 3D-printed board. This allowed the sensor to register their presence reliably and enabled continued testing without changes to the sensor setup.

5.3 Future Work

While the current prototype has demonstrated that PCB testing can be automated using a collaborative robot, there are several areas where future work could significantly improve performance, scalability and industrial applicability.

One potential improvement would be to integrate motorized test fixtures that can open and close automatically. In the current setup the fixture is static, which requires the robot to perform relatively slow and precise insertions. If multiple automated fixtures were installed and controlled in sequence, the robot could reduce idle time and handle a higher number of PCBs in parallel, improving throughput and cycle time.

Another possible improvement would be to replace the manual pass and fail trays with an automated conveyor that transports the tested PCBs to a packaging station or the next production step. This would remove the need for frequent manual intervention and support continuous operation making it possible to work 24/7 for this part of the production.

Replacing the simulated test output with a real working test rig is also an important next step. By integrating a fixture capable of returning actual test results, the system could be validated under more realistic conditions and used for genuine pass/fail-based sorting.

To improve vision capabilities, future systems could use a higher-contrast conveyor surface and real PCB boards instead of 3D-printed models. The current background and material choices limited edge detection accuracy. A better setup would allow full use of the vision system for alignment or identification of PCBs.

Another key step is to test the system with a wider variety of PCB sizes and shapes. As a potential solution vacuum-based grippers could replace the current electric claws to simplify adaptation between different board geometries, without having to reconfigure or reprint custom fingers.

Beyond technical improvements a critical next step is to scale up the system to more closely resemble a production environment. This includes running longer test cycles, handling higher volumes of PCBs and benchmarking actual cycle times. With a scaled system it becomes possible to make data-driven comparisons between manual workflows or existing automated stations.

Finally, expanding the system with alternative hardware components such as different robot models, sensors or conveyors would allow comparative analysis and optimization. The current setup was constrained by the equipment available through Qestit and Chalmers, but broader testing could identify more cost effective or better performing configurations for specific use cases.

6. Conclusion

The project set out to explore whether PCB testing could be automated using a collaborative robot in a lab environment. A working prototype was developed that performs full test cycles, including picking, positioning, simulated testing and sorting, without human intervention during operation.

The key result of the project is not only that automation is possible, but that it can be achieved with accessible tools and straightforward hardware. By using the UR3 robot along with simple support systems, the prototype showed that a modular and flexible test station can be constructed and operated successfully. Even though the setup used 3D printed PCBs and a non-functional fixture, the core processes were verified and shown to be reliable.

One of the main takeaways is the importance of combining basic hardware with adaptive programming. The system's performance relied on synchronized logic, digital inputs and outputs and the ability to integrate control through both PolyScope and Python.

A clear limitation was the absence of a real test fixture. As a result sorting decisions were simulated and no actual test data was used to drive robot behavior. However the system architecture allows for easy replacement of this component and real-time interaction with a functional test rig is a realistic next step, in both The Python configuration as well as the Polyscope one..

Looking forward, the most meaningful continuation would be to scale the setup and carry out extended testing under production-like conditions. This would allow for proper evaluation of cycle times, error rates and long term stability. With better visual contrast and real PCBs, the camera system could also be fully integrated. All though the mechanical version proved to be highly efficient.

In summary this thesis confirms that collaborative robots are a practical and effective solution for automated testing. The system provides a foundation that can be refined and expanded into a fully automated, industry-ready solution.

6. References

Balluff. (2024-03-04). *What is a capacitive sensor?* (Retrieved Feb, 2025, from https://www.balluff.com/en-us/blog/what-is-a-capacitive-sensor)

Bogue, R. (2022). *Robots with machine vision: a review of technologies and applications.* Robotics, 11(5), 101. (Retrieved Feb, 2025, from https://www.mdpi.com/2218-6581/11/5/101)

Control.com. (2022-09-29). *Raspberry Pi applications in industrial automation.* (Retrieved Mar, 2025, from https://control.com/technical-articles/raspberry-pi-applications-in-industrial-automation)

Electronics Tutorials. (n.d.). *Pulse width modulation (PWM).* (Retrieved Mar, 2025, from https://www.electronics-tutorials.ws/blog/pulse-width-modulation.html)

Ensenso. (n.d.). *Hand-Eye Calibration – Ensenso SDK.* (Retrieved Mar, 2025, from https://www.ensenso.com/manual/latest/guides/multi-camera-setups-and-calibrations)

HD Vision Systems. (2025-02-04). *Object detection in industrial machine vision applications.* (Retrieved Mar, 2025, from https://www.hdvisionsystems.com/en/blog/object-detection-in-industrial-machine-vision)

Instron. (2023). *CT6 Cobot Test System for Biomedical Applications.* (Retrieved Mar, 2025, from https://www.instron.com/en/products/testing-systems/automated-testing-systems)

International Federation of Robotics. (2021). *How robots work alongside humans.* (Retrieved Mar, 2025, from https://ifr.org/ifr-press-releases/news/how-robots-work-alongside-humans)

Kassow Robots. (2024-03-07). *Cobot vs. industrial robot: What's the difference?* (Retrieved Mar, 2025, from https://www.kassowrobots.com/blog/cobot-vs-industrial-robot)

Livinti, P., & Mazen, G. (2015). *PWM Control of a DC Motor Used to Drive a Conveyor Belt.* Procedia Engineering, 100, 299–304. (Retrieved Mar, 2025, from https://doi.org/10.1016/j.proeng.2015.01.371)

Medical Design Briefs. (2023). *Cobots offer advantages for biomedical testing applications.* (Retrieved Mar, 2025, from https://www.medicaldesignbriefs.com/component/content/article/46449-cobots)

Milvus. (n.d.). *What is a machine vision edge detection algorithm?* (Retrieved Mar, 2025, from https://milvus.io/ai-quick-reference/what-is-a-machine-vision-edge-detection-algorithm)

Munteanu, R., & Mazen, I. (2015). *PWM control of a DC motor used to drive a conveyor belt Procedia Engineering*, 100, 746–753. (Retrieved Mar, 2025, from https://www.researchgate.net/publication/272846417_PWM_control_of_a_DC_motor)

Parashar. A (2025-03-10). *Top 10 Python Libraries*. (Retrieved Mar, 2025, from https://www.naukri.com/code360/library/python-libraries)

OnRobot. (n.d.-a). *Gear manufacturing company Osvald Jensen decreases cycle time by 12 seconds using dual gripper.* (Retrieved Feb, 2025, from https://onrobot.com/us/news/gear-manufacturing-company-osvald-jensen-decreases-cycle)

OnRobot. (n.d.-b). *Dual gripping cuts cycle times.* (Retrieved Feb, 2025, from https://onrobot.com/en/dual-gripping-cuts-cycle-times)

OnRobot. (n.d.-c). *Dual Quick Changer – Dual Gripping, More Power for Twice the Work.* (Retrieved Feb, 2025, from https://onrobot.com/us/products/dual-quick-changer)

Predictable Designs. (2023-11-15). *Successful industrial products powered by Raspberry Pi.* (Retrieved Mar, 2025, from https://predictabledesigns.com/successful-industrial-products-powered-by-raspberry-pi)

Protolabs. (2025-02-24). *The benefits of 3D printing for rapid prototyping.* (Retrieved Mar, 2025, from https://www.protolabs.com/en-gb/resources/blog/the-benefits-of-3d-printing)

Raspberry Pi. (n.d.). *Raspberry Pi for industry.* (Retrieved Mar, 2025, from https://www.raspberrypi.com/for-industry)

RealPars. (2021-07-14). *What is a capacitive sensor?* (Retrieved Mar, 2025, from https://www.realpars.com/blog/capacitive-sensor)

Replique. (2024). *How to leverage 3D printing in robotics and automation.* (Retrieved Mar, 2025, from https://replique.io/2024/03/07/how-to-leverage-3d-printing-in-robotics-and-automation/)

Owen-Hill. A (2025-02-20). *6 Things That Are Possible With Dual Grippers (+ 3 That Aren't).* (Retrieved Mar, 2025, from https://blog.robotiq.com/6-things-that-are-possible-with-dual-grippers-3-that-arent)

Sciotex. (n.d.). *Edge detection techniques in AI machine vision.* (Retrieved Mar, 2025, from https://sciotex.com/edge-detection-techniques-in-ai-machine-vision)

Swedish Standards Institute. (2016). Industrial robots – Safety requirements – Part 6: Collaborative robots (SIS-ISO/TS 15066:2016). (Retrieved Mar, 2025, from https://www-sis-se.eu1.proxy.openathens.net/produkter/produktionsteknik/industriell)

Swedish Standards Institute. (2016). Safety of machinery – Risk assessment – Part 1: Principles (SS-EN ISO 13849-1:2016). (Retrieved Mar, 2025, from https://www.sis.se/en/produkter/environment-health-protection-safety/safety-of-machinery)

Swedish Standards Institute. (2025a). Robotics – Safety requirements for robot systems in an industrial environment – Part 1: Robots (SS-EN ISO 10218-1:2025). (Retrieved Mar, 2025, from https://www.sis.se/en/produkter/maskinsakerhet-357f77b1)

Swedish Standards Institute. (2025b). Robotics – Safety requirements for robot systems in an industrial environment – Part 2: Robot systems, robot applications and robot cells (SS-EN ISO 10218-2:2025). (Retrieved Mar, 2025, from https://www.sis.se/en/produkter/maskinsakerhet-357f77b1/advanced-manufacturing)

Tech Briefs. (2017). *Adding simple vision systems to collaborative robots.* (Retrieved Mar, 2025, from https://www.techbriefs.com/component/content/article/27076-adding-simple)

Techman Robot. (u.å.). *TM5 - 900*. (Retrieved Mar, 2025, from https://www.tm-robot.com/en/tm5-900/)

Texas Instruments. (2014). Basics of capacitive sensing and applications (Rev. A). (Retrieved Mar, 2025, from https://www.ti.com/lit/pdf/snoa927)

UltiMaker. (n.d.). *Benefits of 3D printing in automated packaging.* (Retrieved Mar, 2025, from https://ultimaker.com/learn/benefits-of-3d-printing-in-automated-packaging/)

Universal Robots. (2021). Webinar – E&T Norway. (Retrieved Feb, 2025, from https://events.universal-robots.com/scandinavia/no/online-events/norwegian/)

Universal Robots. (n.d.-a). *Cobot reduces reject rate to below 1% and increases production rate*. (Retrieved Feb, 2025, from https://www.universal-robots.com/case-stories/creating)

Universal Robots. (n.d.-b). *UR3 cobots optimize bottle cap assembly.* (Retrieved Feb, 2025, from https://www.universal-robots.com/case-stories/marka/)

Universal Robots. (n.d.-c). *Assembly | UR3 and UR5 at Albrecht Jung.* (Retrieved Feb, 2025, from https://www.universal-robots.com/case-stories/albrecht-jung-gmbh/)

Universal Robots. (n.d.-d). *PolyScope software manual.* (Retrieved Feb, 2025, from https://s3-eu-west-1.amazonaws.com/ur-support-site/44018/Software_Manual_en_Global.pdf)

Universal Robots. (n.d.-e). *Real-Time Data Exchange (RTDE) Guide.* (Retrieved Feb, 2025, from https://www.universal-robots.com/articles/ur/interface-communication)

Universal Robots. (n.d.-f). *RTDE developer portal.* (Retrieved Feb, 2025, from https://www.universal-robots.com/developer/communication-protocol/rtde/)

Universal Robots. (2016). *User Manual UR3 CB-Series SW3.3* [PDF]. (Retrieved May, 2025, fromhttps://www.universal-robots.com/download/manuals-cb-series/user/ur3/33/user-manual)

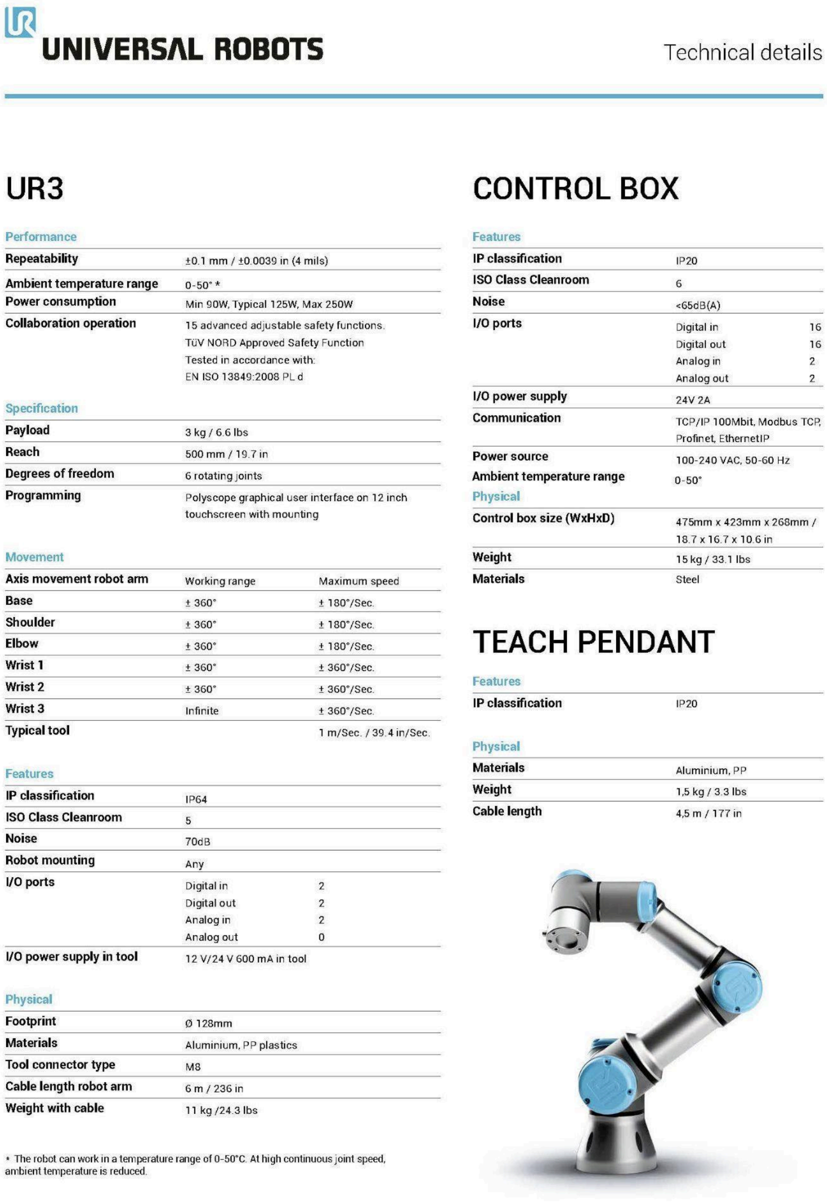
Universal Robots Forum. (n.d.). *Usefulness of UR Script.* (Retrieved Feb, 2025, from https://forum.universal-robots.com/t/usefulness-of-ur-script/25891)

Wikipedia. (n.d.). *Vision-guided robot systems.* (Retrieved Mar, 2025, from https://en.wikipedia.org/wiki/Vision-guided_robot_systems)

Hacks360. (2020-11-28). *Industrial automation with Raspberry Pi and Node-RED.* (Retrieved Mar, 2025, from https://www.youtube.com/watch?v=Tg_MPyfzhts)

7. Appendix

Appendix 1. UR3 Robot specifikation. Source: Universal Robots. (2016). UR3/CB3 User Manual – Version 3.3.3



UNIVERSAL ROBOTS

Technical details

UR3

Performance

Repeatability	±0.1 mm / ±0.0039 in (4 mils)
Ambient temperature range	0-50° *
Power consumption	Min 90W, Typical 125W, Max 250W
Collaboration operation	15 advanced adjustable safety functions. TüV NORD Approved Safety Function Tested in accordance with: EN ISO 13849:2008 PL d

Specification

Payload	3 kg / 6.6 lbs
Reach	500 mm / 19.7 in
Degrees of freedom	6 rotating joints
Programming	Polyscope graphical user interface on 12 inch touchscreen with mounting

Movement

Axis movement robot arm	Working range	Maximum speed
Base	± 360°	± 180°/Sec.
Shoulder	± 360°	± 180°/Sec.
Elbow	± 360°	± 180°/Sec.
Wrist 1	± 360°	± 360°/Sec.
Wrist 2	± 360°	± 360°/Sec.
Wrist 3	Infinite	± 360°/Sec.
Typical tool		1 m/Sec. / 39.4 in/Sec.

Features

IP classification	IP64	
ISO Class Cleanroom	5	
Noise	70dB	
Robot mounting	Any	
I/O ports	Digital in Digital out Analog in Analog out	2 2 2 0
I/O power supply in tool	12 V/24 V 600 mA in tool	

Physical

Footprint	Ø 128mm
Materials	Aluminium, PP plastics
Tool connector type	M8
Cable length robot arm	6 m / 236 in
Weight with cable	11 kg /24.3 lbs

* The robot can work in a temperature range of 0-50°C. At high continuous joint speed, ambient temperature is reduced.

CONTROL BOX

Features

IP classification	IP20	
ISO Class Cleanroom	6	
Noise	<65dB(A)	
I/O ports	Digital in Digital out Analog in Analog out	16 16 2 2
I/O power supply	24V 2A	
Communication	TCP/IP 100Mbit, Modbus TCP, Profinet, EthernetIP	
Power source	100-240 VAC, 50-60 Hz	
Ambient temperature range	0-50°	

Physical

Control box size (WxHxD)	475mm x 423mm x 268mm / 18.7 x 16.7 x 10.6 in
Weight	15 kg / 33.1 lbs
Materials	Steel

TEACH PENDANT

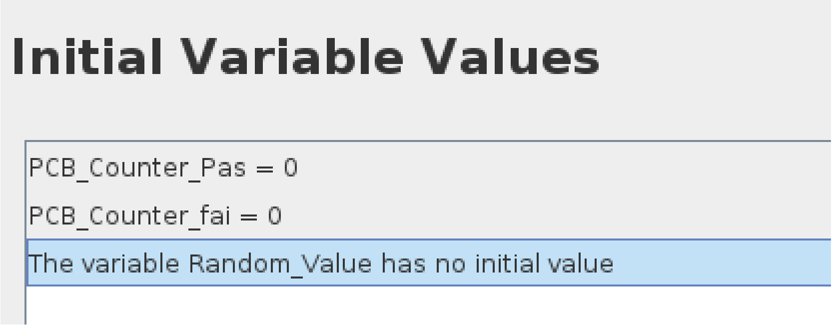
Features

IP classification	IP20

Physical

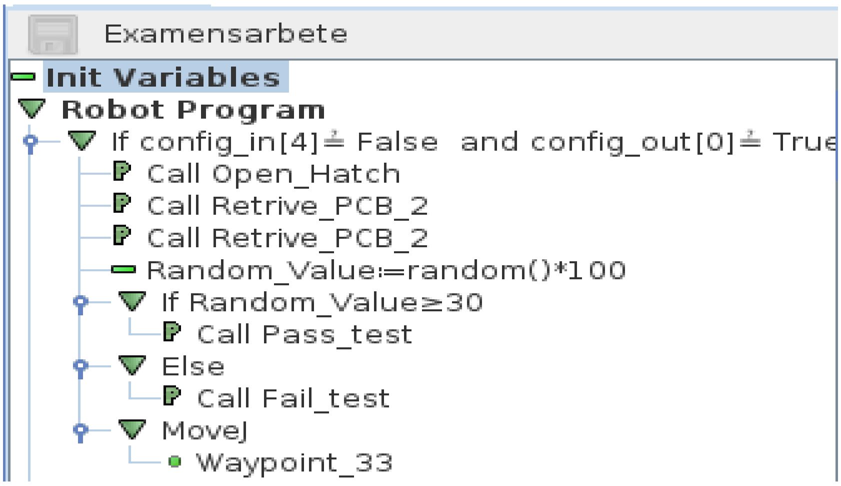
Materials	Aluminium, PP
Weight	1,5 kg / 3.3 lbs
Cable length	4,5 m / 177 in

Appendix 2. Full polyscope code

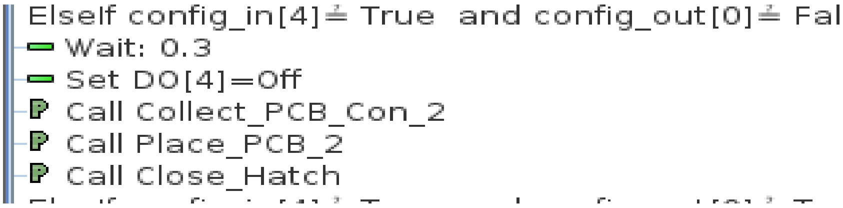


Initial Variable Values

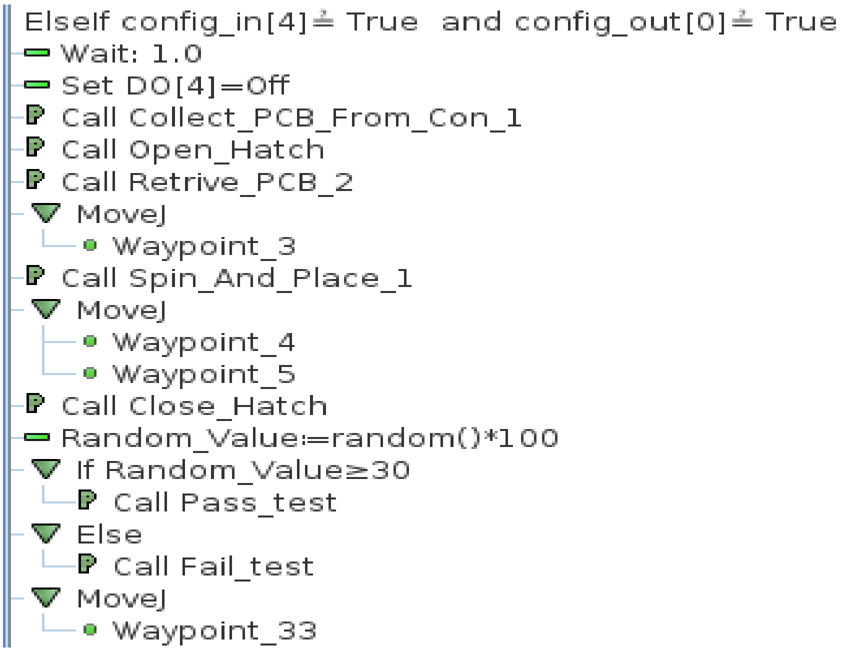
```
PCB_Counter_Pas = 0
PCB_Counter_fai = 0
The variable Random_Value has no initial value
```



```
Init Variables
Robot Program
  If config_in[4]≟ False  and config_out[0]≟ True
    Call Open_Hatch
    Call Retrive_PCB_2
    Call Retrive_PCB_2
    Random_Value≔random()*100
    If Random_Value≥30
      Call Pass_test
    Else
      Call Fail_test
    MoveJ
      Waypoint_33
```

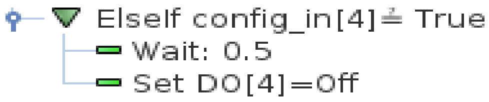


```
ElseIf config_in[4]≟ True  and config_out[0]≟ Fal
  Wait: 0.3
  Set DO[4]=Off
  Call Collect_PCB_Con_2
  Call Place_PCB_2
  Call Close_Hatch
```

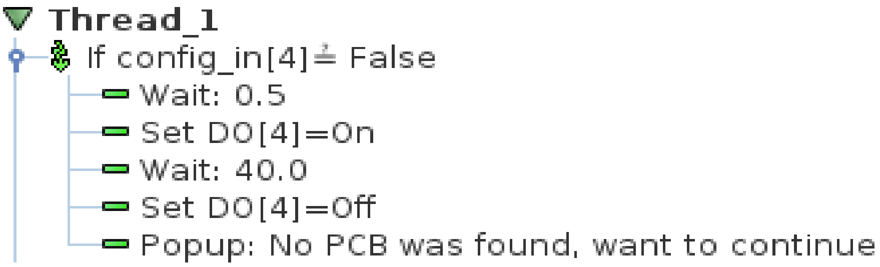


```
ElseIf config_in[4]≟ True  and config_out[0]≟ True
  Wait: 1.0
  Set DO[4]=Off
  Call Collect_PCB_From_Con_1
  Call Open_Hatch
  Call Retrive_PCB_2
  MoveJ
    Waypoint_3
  Call Spin_And_Place_1
  MoveJ
    Waypoint_4
    Waypoint_5
  Call Close_Hatch
  Random_Value≔random()*100
  If Random_Value≥30
    Call Pass_test
  Else
    Call Fail_test
  MoveJ
    Waypoint_33
```

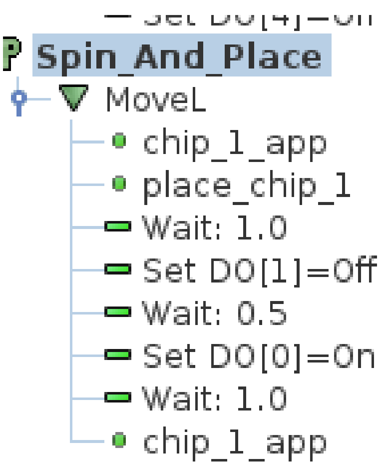
```
ElseIf config_in[4]≟ False
  MoveJ
    Homepose
```



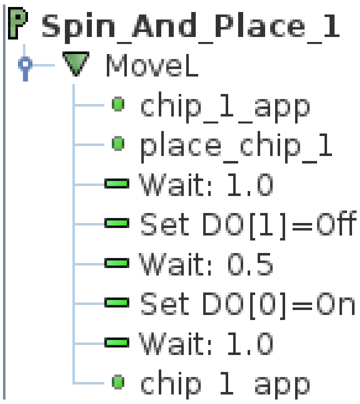
```
ElseIf config_in[4]≟ True
  Wait: 0.5
  Set DO[4]=Off
```



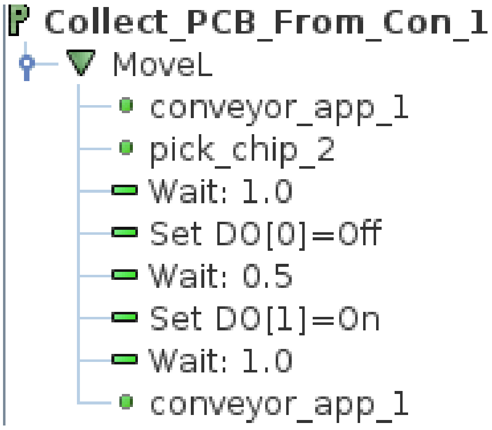
```
Thread_1
  If config_in[4]≟ False
    Wait: 0.5
    Set DO[4]=On
    Wait: 40.0
    Set DO[4]=Off
    Popup: No PCB was found, want to continue
```



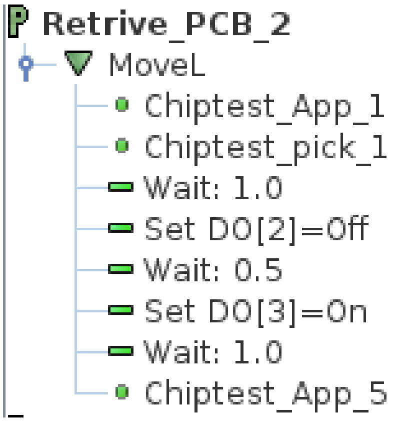
```
Spin_And_Place
  MoveL
    chip_1_app
    place_chip_1
    Wait: 1.0
    Set DO[1]=Off
    Wait: 0.5
    Set DO[0]=On
    Wait: 1.0
    chip_1_app
```



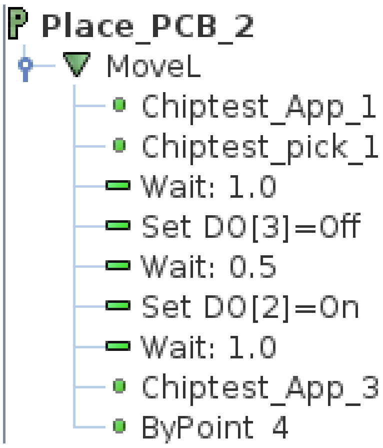
```
Spin_And_Place_1
  MoveL
    chip_1_app
    place_chip_1
    Wait: 1.0
    Set DO[1]=Off
    Wait: 0.5
    Set DO[0]=On
    Wait: 1.0
    chip 1 app
```



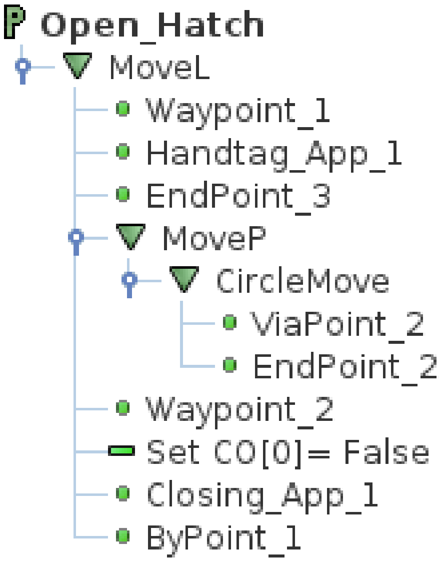
```
Collect_PCB_From_Con_1
  MoveL
    conveyor_app_1
    pick_chip_2
    Wait: 1.0
    Set DO[0]=Off
    Wait: 0.5
    Set DO[1]=On
    Wait: 1.0
    conveyor_app_1
```



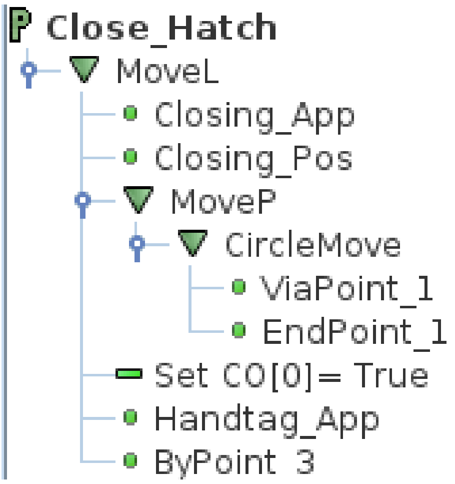
```
Retrive_PCB_2
  MoveL
    Chiptest_App_1
    Chiptest_pick_1
    Wait: 1.0
    Set DO[2]=Off
    Wait: 0.5
    Set DO[3]=On
    Wait: 1.0
    Chiptest_App_5
```



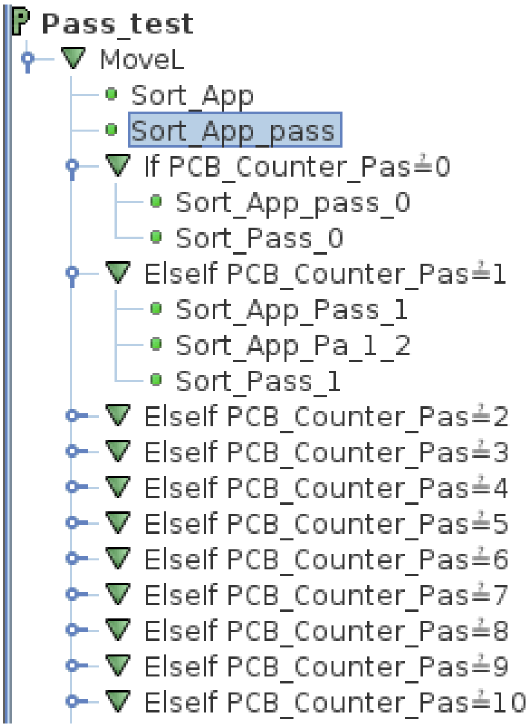
```
Place_PCB_2
  MoveL
    Chiptest_App_1
    Chiptest_pick_1
    Wait: 1.0
    Set DO[3]=Off
    Wait: 0.5
    Set DO[2]=On
    Wait: 1.0
    Chiptest_App_3
    ByPoint 4
```



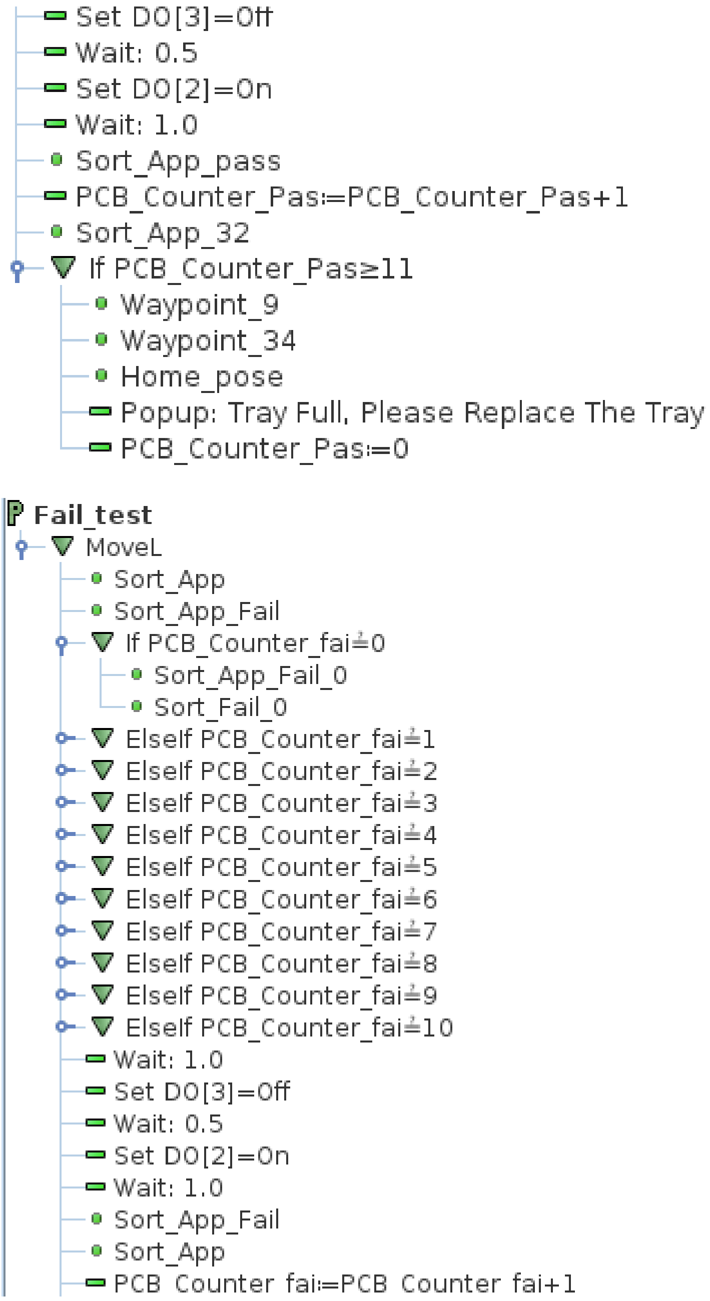
```
Open_Hatch
  MoveL
    Waypoint_1
    Handtag_App_1
    EndPoint_3
    MoveP
      CircleMove
        ViaPoint_2
        EndPoint_2
    Waypoint_2
    Set CO[0]= False
    Closing_App_1
    ByPoint_1
```



```
Close_Hatch
  MoveL
    Closing_App
    Closing_Pos
    MoveP
      CircleMove
        ViaPoint_1
        EndPoint_1
    Set CO[0]= True
    Handtag_App
    ByPoint 3
```

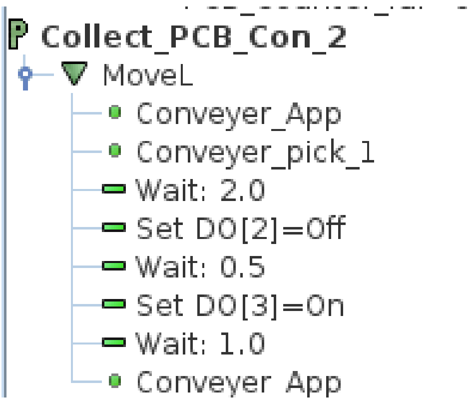


```
Pass_test
  MoveL
    Sort_App
    Sort_App_pass
    If PCB_Counter_Pas≟0
      Sort_App_pass_0
      Sort_Pass_0
    ElseIf PCB_Counter_Pas≟1
      Sort_App_Pass_1
      Sort_App_Pa_1_2
      Sort_Pass_1
    ElseIf PCB_Counter_Pas≟2
    ElseIf PCB_Counter_Pas≟3
    ElseIf PCB_Counter_Pas≟4
    ElseIf PCB_Counter_Pas≟5
    ElseIf PCB_Counter_Pas≟6
    ElseIf PCB_Counter_Pas≟7
    ElseIf PCB_Counter_Pas≟8
    ElseIf PCB_Counter_Pas≟9
    ElseIf PCB_Counter_Pas≟10
```

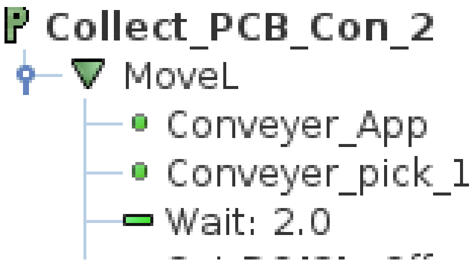


```
   Set DO[3]=Off
   Wait: 0.5
   Set DO[2]=On
   Wait: 1.0
   Sort_App_pass
   PCB_Counter_Pas≔PCB_Counter_Pas+1
   Sort_App_32
   If PCB_Counter_Pas≥11
      Waypoint_9
      Waypoint_34
      Home_pose
      Popup: Tray Full, Please Replace The Tray
      PCB_Counter_Pas≔0

Fail_test
   MoveL
      Sort_App
      Sort_App_Fail
      If PCB_Counter_fai≟0
         Sort_App_Fail_0
         Sort_Fail_0
      ElseIf PCB_Counter_fai≟1
      ElseIf PCB_Counter_fai≟2
      ElseIf PCB_Counter_fai≟3
      ElseIf PCB_Counter_fai≟4
      ElseIf PCB_Counter_fai≟5
      ElseIf PCB_Counter_fai≟6
      ElseIf PCB_Counter_fai≟7
      ElseIf PCB_Counter_fai≟8
      ElseIf PCB_Counter_fai≟9
      ElseIf PCB_Counter_fai≟10
      Wait: 1.0
      Set DO[3]=Off
      Wait: 0.5
      Set DO[2]=On
      Wait: 1.0
      Sort_App_Fail
      Sort_App
      PCB_Counter_fai≔PCB_Counter_fai+1
```



```
Collect_PCB_Con_2
  MoveL
    Conveyer_App
    Conveyer_pick_1
    Wait: 2.0
    Set DO[2]=Off
    Wait: 0.5
    Set DO[3]=On
    Wait: 1.0
    Conveyer_App
```



```
Collect_PCB_Con_2
  MoveL
    Conveyer_App
    Conveyer_pick_1
    Wait: 2.0
```

Appendix 3

Link to a video of the whole system:

https://youtu.be/S91IOP_09rE?si=_Mxy_j4toNEN_HhL

Appendix 4

Link to the repository for the code:

https://svn.addq4.se:8009/svn/SVNRoboticAutomation/trunk/

INSTITUTIONEN FÖR INDUSTRI OCH
MATERIALVETENSKAP
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2025
www.chalmers.se