Pick and Place Automation System

by

Soi Zhi Wen

A project report submitted
in partial fulfilment of the requirements for the degree of
Bachelor of Computer Science in Software Engineering

at the

Faculty of Computer Science and Information Computing Technology
NEW ERA UNIVERSITY COLLEGE

January 14, 2022

Declaration of Authorship

I, Soi Zhi Wen, declare that this thesis titled, “Pick and Place Automation System” and the work presented in it are my own except for citations and quotations, which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at New Era University College or other institutions.

Signature:

_____

Name:

Soi Zhi Wen

_____

ID No.:

1950232-BSE

_____

Date:

January 14, 2022

_____

Approval for Submission

I certify that this thesis titled, "Pick and Place Automation System" was prepared by Soi Zhi Wen has met the required standard for submission in partial fulfilment of the requirements for the degree of Bachelor of Computer Science in Software Engineering at New Era University College.

Approved by,

Signature:

_____

Supervisor:

Ts. Chng Chern Wei

_____

Date:

_____

Dedicated to my beloved family, lecturers, professors, supervisors and all my friends that went through all the peak and valleys with me during my time in FICT, NEUC.

Acknowledgements

The completion of this project could not have been possible without the participation and assistance of so many people whose names may not all be enumerated. Their contributions are sincerely appreciated and gratefully acknowledged.

I want to thank to my research supervisor, Ts. Chng Chern Wei for his invaluable advice, guidance and his enormous patience throughout the development of the research.

In addition, I would like to thank to my family, relatives, friends and others who in one way or another shared their support, either morally, financially and physically, thank you.

Pick and Place Automation System

by

Soi Zhi Wen

A project report submitted on January 14, 2022,
in partial fulfilment of the requirements for the degree of
Bachelor of Computer Science in Software Engineering

Abstract

This paper presents the design and implementation of a pick and place automation system. Robotics are important instruments for increasing production. The majority of robot adoption has occurred in the manufacturing industry, where they execute a wide range of manual jobs faster and more reliably than people. Therefore, this system was designed to perform pick and place tasks without human assistance. The system recognised objects based on shape and colour. The recognition along shape and colour was done by using a shape recognition algorithm and a L*a*b* colour space recognition algorithm. After the object was determined, the centroid of the object was identified in order to calculate the coordinate of the centroid point. Subsequently, an inverse kinematics algorithm was used to calculate the angle of the robot arms that were needed to place the end effector at desired coordinate. The robot arms then picked up the object and placed it to a location based on its shape and colour.

Contents

List of Figures

List of Tables

List of Abbreviations

API	Application Programming Interface
APT	Advanced Packaging Tool
ARM	Advanced RISC Machines
CAD	Computer-Aided Design
CAM	Computer-Aided Manufacturing
CLI	Command-Line Interface
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DC	Direct Current
DOF	Degrees of Freedom
FDM	Fused Deposition Modelling
FPS	Frames per Second
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HD	High-Definition
I/O	Input/Output
IC	Integrated Circuit
IDE	Integrated Development Environment
IK	Inverse Kinematics
IoT	Internet of Things
IR	Infrared Radiation
LAN	Local Area Network
MCU	Microcontroller Unit
MQTT	Message Queuing Telemetry Transport
OpenCV	Open Source Computer Vision Library

OS	Operating System
PC	Personal Computer
PCB	Printed Circuit Board
PID	Proportional, Integral, Derivative
RTSP	Real Time Streaming Protocol
SCARA	Selective Compliance Assembly Robot Arm
SD	Secure Digital
SoC	System on a Chip
SSH	Secure Shell
STL	Standard Template Library
USB	Universal Serial Bus
WSL	Windows Subsystem for Linux

Chapter 1

Introduction

1.1 Overview

People have been trying to replace human jobs with machines for years. Robots are machines that are faster and more efficient than humans. The term robotics is actually defined as the study, design and use of robotic systems for production. Robots are often used to perform unsafe, dangerous, highly repetitive and unpleasant tasks. They have many different functions such as material handling, assembly, arc welding, resistance welding and loading and unloading functions of machine tools, painting, spraying and so forth ("What is a pick and place robot?", 2021). Many elements of the robots were built with inspiration from nature. The construction of the robotic hand, which is the arm of the robot, is based on the human arm. The robot has the ability to manipulate objects, such as pick and place operations. It is also capable of operating on its own. The development of robotic systems technology for the electronics industry has been increasingly expanding. As one such application, service robots with machine vision capabilities have recently been developed.

The use of industrial robotic arms is characteristic of some contemporary trends in manufacturing process automation. However, today's industrial robotic arms also exhibit a monolithic mechanical structure and a closed system software architecture. They are focused on simple repetitive tasks that often do not require high accuracy ("What is a pick and place robot?", 2021).

The pick-and-place robot is a microcontroller based electromechanical system that detects an object, picks up that object from its source location and places it at the desired location. To detect objects, advanced vision systems are used to identify, grasp and move objects from one place to another ("What is a pick and place robot?", 2021).

1.2 Objectives

The goal of this project is to build a pick and place automation system. This system performs pick, classify and place operations on all sides of a compact work station with added flexibility for future change overs. To achieve the goal, we mainly focus on the following objectives:

1. **Build a dual-arm SCARA robot** that can be used to pick and place the objects from any source to destination.
2. **Develop a vision system** to perform shape and colour detection and recognition.
3. **Develop a pick and place algorithm** to control the displacement and movement of dual arm SCARA robot.

1.3 Problem Statement

Pick and place robots are used to make the process of sorting, transporting large items, and other tasks easier. Human power is typically used to transferring heavy loads, which can result in damage to operators if the operation is repeated for a long time. By using specific robots, operators will no longer need to bent and lift up heavy loads, thus preventing injuries and increasing efficiency. Operators can make mistakes, large or small, over a period of time. In the industrial world, the industry cannot afford any kind of mistakes. This is because every mistake is costly, whether it is time, money and material used.

1.4 Project Scope

This project is for the creation of a pick and place automation system that can be used in manufacturing but also used in applications such as packaging, bin picking and inspection. The system includes vision system that enable the pick and place robot to grasp, pick and place objects.

Chapter 2

Literature Review

This section introduces several research paper and related systems or techniques have been used as references to help develop the system.

2.1 Robot

The development of task-level robot systems has long been a goal of robotics research. The use of the expression task-level is attributed to Lozano-Pérez, et al. (Lozano-Pérez et al., 1989). Unlike robots that are programmed to perform specific mechanical tasks, task-level robots are designed to accept high-level goals and then determine and execute any actions required to meet those goals. They are intended to work under a variety of incidental contextual conditions, including low-frequency exceptional situations related to hardware, software, or the state of the environment (Lozano-Pérez et al., 1989). The researchers know that deciding how to grasp an object is critical in solving pick and place tasks. They pointed out that selecting a good target position is influenced by the surrounding environment. The hand position and resulting arm configuration must be collision-free and reachable. Therefore, they only use a parallel jaw gripper in the domain of polyhedral models, limiting the number of possible grasps for an object. They choose a grasping point and attempt to plan a path there, repeating the process if no path to the grasping location can be found (Lozano-Pérez et al., 1989).

Based on Shevkar et al., 2019 proposed the development and testing of a desktop SCARA with three degree of freedom (DOF) that can pick and place objects with great speed and precision. Stepper motors were chosen for the application because of their high accuracy and repeatability. On the other hand, Taylor's series is used to compute the inter-step delays for stepper motors to provide constant acceleration and deceleration, allowing the robotic

arm to work at high speed. Finally, the robot is tested for repeatability, and it is determined to be within 0.1mm, which is acceptable (Shevkar et al., 2019).

2.2 Vision

The paper by Kumar et al., 2014 describes the development of the image processing algorithm for controlling the operation of pick and place robot. For this type of task, the objects must first be detected, which is done using a feature extraction algorithm. The extracted image is then passed to the classifier to determine what type of object it is, and once this is completed, the output will contain type of the object as well as its coordinates, ready for the robot arm to perform the pick and place task. The main issue when developing this image processing system was that upon making the test subjects in compliance with the classifier parameters, resizing of the images resulted in pixel data loss. As a result, a centred image strategy was used. Finally, the objects were able to be classified, recognized, and localized with an accuracy of more than 80% (Kumar et al., 2014).

The system for shape detection recognition and classification has been studied by Poda and Qirici, 2018. The researchers outline five steps for contour identification using the OpenCV development tool, which are noise reduction, finding the intensity gradient, non-maximum suppression, hysteresis thresholding, and the contour detection algorithm (Poda & Qirici, 2018). Additionally, they came up with a concept based on the contour identification. For instance, when the approximated contour has five vertices, then a pentagon is detected, and an initial classification character 'p' is saved and sent to a hardware system for the movement of a robot arm (Poda & Qirici, 2018). According to the researchers, this strategy offers a feasible and low-cost method for robotics technologies in the future. Thus, this method was a practical and appropriate procedure for the experiment in this study.

Chapter 3

Methodology

This section is devoted to a detailed presentation of the methodology that this project is concerned with.

3.1 Hardware Modules

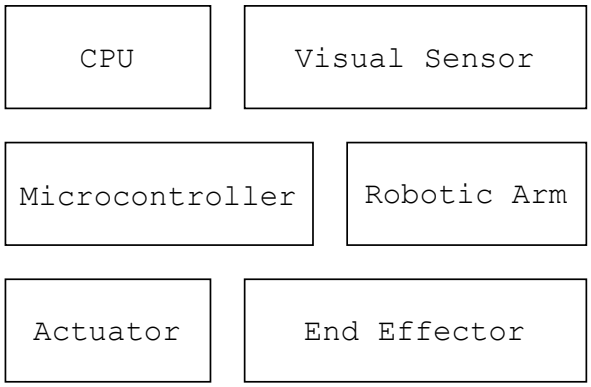


Figure 3.1: Main components of the pick and place automation system.

As depicted in Figure 3.1, there are six main components to build up the pick and place automation system. The following sections outlines all of the hardware modules used in this project along with the purpose of each one.

3.1.1 CPU

To operate the pick and place automation system, we tested it on three different CPUs.

First, we tried it on a PC with a Intel® Core™ i7-4770 Processor @ 3.40GHz with total 6 cores, and it is installed with Debian 11 (Bullseye). At the same time, it also installed with a NVIDIA GeForce GT 710. After trying to run the system in the PC, the PC able to manage

the system with Arduino CLI, OpenCV with CUDA, OpenCV without CUDA and virtual camera. Although the PC able to run OpenCV in CUDA, the processing power required to perform image processing for pick and place automation system is low. Therefore, it is enough to run OpenCV in the CPU.

When considering the portability of PC, we tried to migrate from PC to mini PC. The mini PC is installed with Intel® Celeron® Processor N3160 @ 1.60GHz with total 4 cores, but the mini PC does not has a NVIDIA GPU. Therefore, we unable to install CUDA in mini PC. Besides, the mini PC is installed with Debian 11, which same as PC that mentioned before. Hence, the performance of running the pick and place automation system in mini PC is almost the same as PC.

After trying on mini PC, it is good to try it on Raspberry Pi. We chose to run on Raspberry Pi 4 Model B, that included a high-performance processor, Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz. Meanwhile, a SD card with a data storage of 64GB is inserted into the Raspberry Pi 4 Model B. In addition, the Raspberry Pi Debian 11 OS with lite version is installed in it. When compared to PC and mini PC, the Raspberry Pi 4 Model B takes longer to process images. Another problem is that the data transfer speed in the SD card is very slow. Therefore, Raspberry Pi 4 Model B is not used in this project.

Table 3.1: Comparison between CPU.

	Arduino CLI	**OpenCV With CUDA**	**OpenCV Without CUDA**	**Virtual Camera**
PC	✓	✓	✓	✓
Mini PC	✓	–	✓	✓
Raspberry Pi	✓	–	✓	✓

Table 3.1 shows a summary of the ability to run each software in different CPUs. Finally, we chose to use mini PC to run the pick and place automation system because it is more portable than PC.

3.1.2 Visual Sensor

There are few types of visual sensors, such as thermal camera and colour camera.

A thermal camera is a device that uses infrared radiation (IR) to generate a picture, similar to how a regular camera uses visible light to make an image. Thermal cameras are sensitive to wavelengths ranging from around 1 000 nanometres to about 14 000 nanometres, rather than the visible light camera from 400 nanometre to 700 nanometre range ("Introduction to Infrared (Part 1): The Physics Behind Thermal Imaging", 2021). The most typical application of a thermal camera is for night vision. This is due to the low light intensity at night, making it harder for humans to detect and recognise objects in their environment. However, because the objects in this project are placed in a light area, a thermal camera is not required.

As an alternative, a Logitech C270 HD colour webcam is used in this project. The Logitech C270 HD camera claims HD video quality at a resolution FPS of 720p/30fps with a diagonal field of view of 55 degrees, according to the technical specifications. Therefore, the Logitech C270 HD webcam is sufficient for this project.

3.1.3 Microcontroller

A microcontroller is a tiny computer built on a single MOS integrated circuit (IC) chip. One or more CPUs, as well as memory and programmable input/output (I/O) peripherals, are all found in a microcontroller. In this section, we discuss three different types of microcontrollers, which are Arduino, ESP32 and Raspberry Pi Pico.

Arduino is an open-source electronics platform that uses simple hardware and software to make it easy to use. There is a wide range of microprocessors and controllers are used in Arduino board designs. The boards come with a number of digital and analog I/O pins that may be used to connect to expansion boards, breadboards, and other circuits. Serial communications interfaces, including Universal Serial Bus (USB) are available on the boards and are used to load applications. The microcontrollers may be programmed with the C and C++ programming languages, as well as a common API called the Arduino language.

On the other hand, ESP32 is a family of low-cost, low-power SoC microcontrollers that have built-in Wi-Fi and dual-mode Bluetooth. A variety of versions have been announced

and introduced since the original ESP32 was released. They are part of the ESP32 microcontroller family. Although the CPUs and capabilities of these chips differ, they all use the same SDK and are essentially code compatible. However, one of the drawbacks of using ESP32 is the lack of libraries and APIs.

The Raspberry Pi Pico is a low-cost microcontroller (MCU) created by Raspberry Pi that can be programmed in C and MicroPython. It was the first Raspberry Pi board to use a single microcontroller chip. It is meant for physical computing, comparable to an Arduino, rather than serving as a general-purpose computer. Although the Raspberry Pi Pico is cheap, but it is the first MCU created by Raspberry Pi, which is new and unstable.

From the three microcontrollers listed above, we decided to use Arduino board as microcontroller because of its huge community and third-party libraries and APIs. However, there are a lot of Arduino boards available in the market. The MKR WiFi 1010, which is one of the powerful microcontrollers developed by Arduino. It is powered by the popular Arm® Cortex®-M0 32-bit SAMD21 CPU and includes the ECC508 crypto-chip for security. The board is part of the MKR series, which allows to develop projects right out of the box with minimum effort by selecting from a wide range of shields, such as DYNAMIXEL Shield MKR (mentioned in Section 3.1.5). Finally, the Arduino MKR WiFi 1010 is used in this project as a microcontroller.

3.1.4 Robotic Arm

We discuss three different robotic arms in this section, including Cartesian robot, SCARA robot and delta robot.

A Cartesian robot is an industrial robot with three primary control axes that are all linear, operate in a straight line rather than rotate, and are at right angles to one another. The three sliding joints allow to move wrist up and down, in and out, and back and forth. This mechanical configuration streamlines the robot control arm solution, among other things. When functioning in three-dimensional space, it is extremely dependable and precise. It is also useful for horizontal navigation and stacking bins as a robot coordinate system.

Besides, SCARA is also a type of industrial robot. SCARAs are often faster than Cartesian robots of equivalent size. Their single pedestal mount has a tiny footprint and allows for easy and unrestricted placement. SCARAs, on the other hand, might be more costly than equivalent Cartesian systems, and the controlling software for linear interpolated motions

requires inverse kinematics. This software is normally included with the SCARA and is transparent to the end-user.

Other than Cartesian and SCARA robot, delta robot can also be one of the solutions for automatic pick and place. It is a parallel robot with three arms that are joined at the base by universal joints. The usage of parallelograms in the arms, which keeps the end effector oriented, is a major design aspect. Delta robots are commonly used in industries for picking and packing since they are rapid, with some capable of doing up to 300 picks per minute.

In this case, SCARA robot is used in this project. However, there are single-arm and dual-arm SCARA robots available in the market. The single-arm design makes it difficult for the arms to interfere with each other, but it is less precise than dual-arm design. On the other hand, the advantages of a dual-arm SCARA robot include great accuracy, rigidity, payload capabilities and reduced movement inertia. Unfortunately, the primary disadvantages of dual-arm design are its narrow workspace and several single configurations. Furthermore, the geometric characteristics of a dual-arm design have a substantial impact on its performance. As a result, the kinematic design of dual-arm SCARA robot is considerably more difficult, and the suitability and efficacy of the design technique become increasingly important.

In conclusion, the SCARA robot with dual-arm design looks more challenging. For these reasons, we chose to build a dual-arm SCARA robot in this project.

3.1.5 Actuator

An actuator is a machine component in charge of moving and directing a mechanism or system. It requires a control signal as well as an energy supply. In this section, we discuss the right actuators for the dual-arm SCARA robot. To get a solution, we compare between stepper motor and DYNAMIXEL actuator.

DC motors that move in discrete increments are known as stepper motors. Stepping controlled by a computer allows for extremely precise positioning and speed control. As a result, stepper motors are the preferred motor for a wide range of precise motion control applications. However, stepper motors have less torque at high speeds than at low speeds. Some steppers are designed to deliver greater high-speed performance, but they must be matched with the right driver to do this. Apart from that, unlike servo motors, most steppers do not have integrated position feedback, despite the fact that they may attain high accuracy.

Alternatively, there is an upgraded version of DC actuator from DYNAMIXEL that used in this project. DYNAMIXEL is a robot exclusive smart actuator with fully integrated DC motor, controller, driver, sensor, reduction gear and network in one DC servo module. The MX-12W is a type of DYNAMIXEL featuring a number of sophisticated features. A new contactless magnetic rotary encoder is built into the MX actuator, allowing for precise angular movement across a complete 360 degrees. With a resolution of 12 bits (4096) places inside 360 degrees, absolute angle measurements give fast and trustworthy information about angular position ("MX-12W", n.d.). This allows for precise angle adjustments of up to 0.088 degrees. The MX-12W also includes a novel PID control that corrects for naturally occurring defects like backlash automatically and precisely that caused by small gaps in the gears. As a result, placement is more consistent and conforms to standards.

In order to control DYNAMIXEL MX-12W, a DYNAMIXEL Shield MKR should be used. The DYNAMIXEL Shield MKR can be mounted on or below the Arduino MKR boards. At the same time, the DYNAMIXEL Shield MKR able to get real-time feedback like position, velocity, voltage, current, temperature, moving status, and additional items.

In short, the dual-arm SCARA robot is actuated with two DYNAMIXEL MX-12W, and two actuators are connected to DYNAMIXEL Shield MKR.

3.1.6 End Effector

The device at the end of a robotic arm that interacts with the environment is known as an end effector in robotics. The most popular types of end effectors are pneumatic gripper and vacuum gripper.

A pneumatic gripper is a pick and place device that operates gripper jaws with compressed air. These human-like fingers assist in gripping, holding, and releasing work components. They usually feature two or three fingers for control, with a single or double actuating cylinder. When air is delivered, the gripper jaws close on an object and securely hold it until some operation is completed; when the air direction is changed, the gripper releases the object. They are often utilised to hold a work component in automated manufacturing operations. Small things, such as circuit boards or chips, to massive ones, such as an engine block, can all be used as work pieces.

Other than that, vacuum grippers can be used to grasp non-ferrous items. As a gripping mechanism, vacuum cups, also known as suction cups, are employed. If the objects are

smooth, flat, and clean, this sort of gripper will give good handling. It just has one surface on which to grab the things. Most significantly, it is not well suited to the management of items that include holes.

In addition, it is worth mentioning that there is a soft gripper can pick a variety of irregular shapes and fragile products, making it perfect for pick and place applications in food and beverage production, manufacturing, and packaging. The gripper comes with three silicon-molded cups that may be swapped out. Food and beverage automation has suddenly gotten a whole lot easier, from eggs to fruit to bottles and cans. The soft gripper, like other grippers, integrates seamlessly with the robot. The soft gripper functions without the need for an external air supply, resulting in lower costs and less complexity, as well as a reduction in dust and noise compared to typical vacuum grippers. Therefore, it is good to study the mechanism of soft gripper as the future work.

Finally, a vacuum gripper is selected in this project because the objects that the robotic arm should pick up is flat and smooth.

3.1.7 Final Assembly

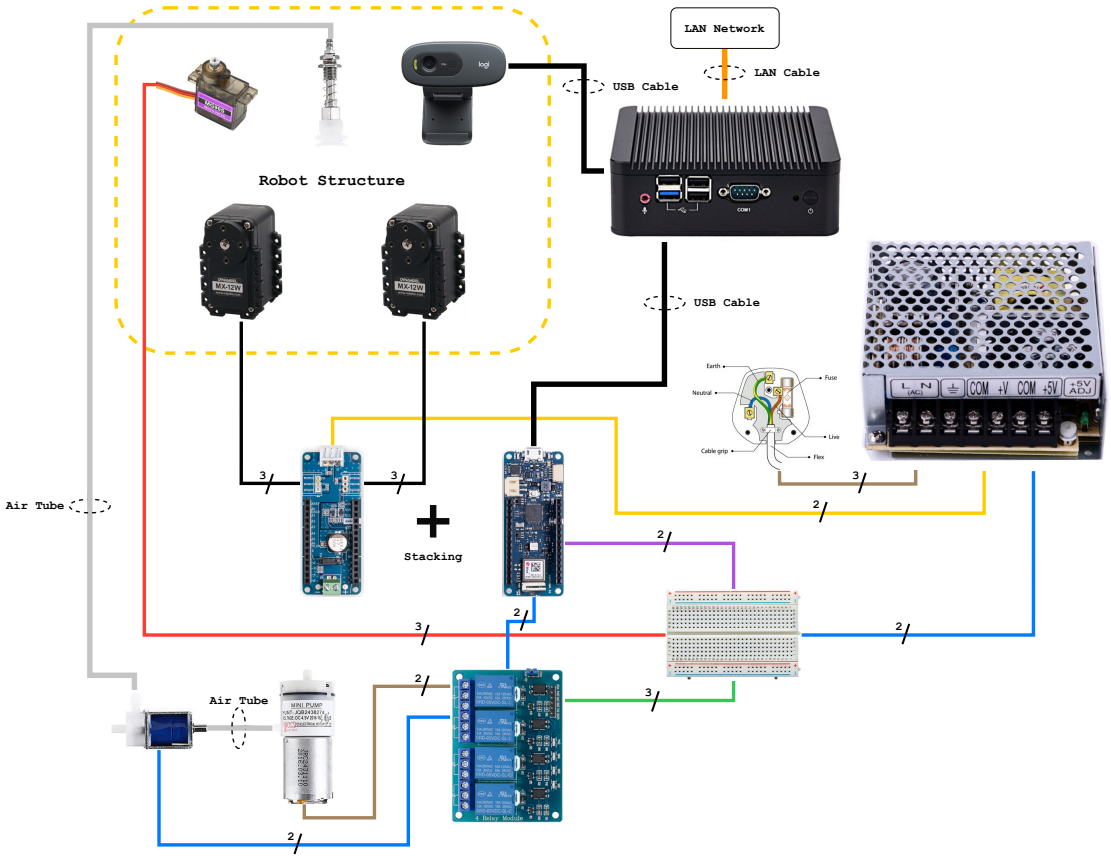


Figure 3.2: Assembly diagram of pick and place automation system.

As depicted in Figure 3.2, the mini PC is connected to a LAN network. It is used to retrieve images from the Logitech C270 webcam and upload sketch to Arduino MKR WiFi 1010 using USB cable. Besides, the DYNAMIXEL Shield MKR is stacked with microcontroller Arduino MKR WiFi 1010. The two DYNAMIXEL MX-12W are connected to the DYNAMIXEL TTL ports that supported on DYNAMIXEL Shield MKR ("DYNAMIXEL Shield MKR", n.d.). In addition, a 4 Channel 5V Relay Module is used to control the air pump and solenoid valve. In the robot structure, a MG90S metal gear micro servo is used to control the vertical mechanism of the dual-arm SCARA robot which discuss in Section 3.2.3. Finally, the dual-arm SCARA robot is powered with 5V and 12V from a 50W dual output switching power supply with an input of 240V. Figure 3.3 show the physical diagram of the system.

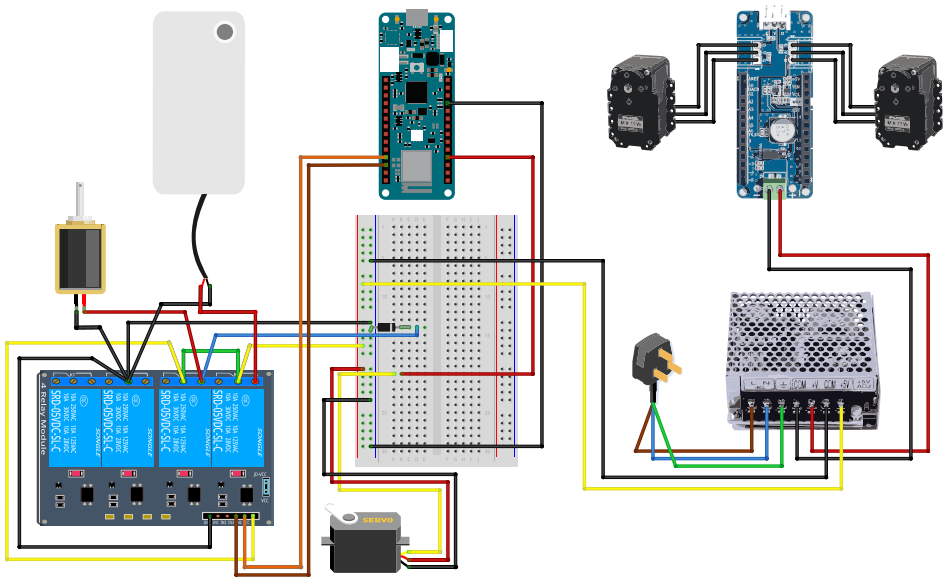


Figure 3.3: Physical diagram of pick and place automation system.

3.2 Mechanical Design

In this section, we present the mechanical design of the dual-arm SCARA robot. Figure 3.4 shows the schematic of dual-arm SCARA robot, whose dimensions are $d = 200$mm, $l_1 = 150$mm and $l_2 = 250$mm.

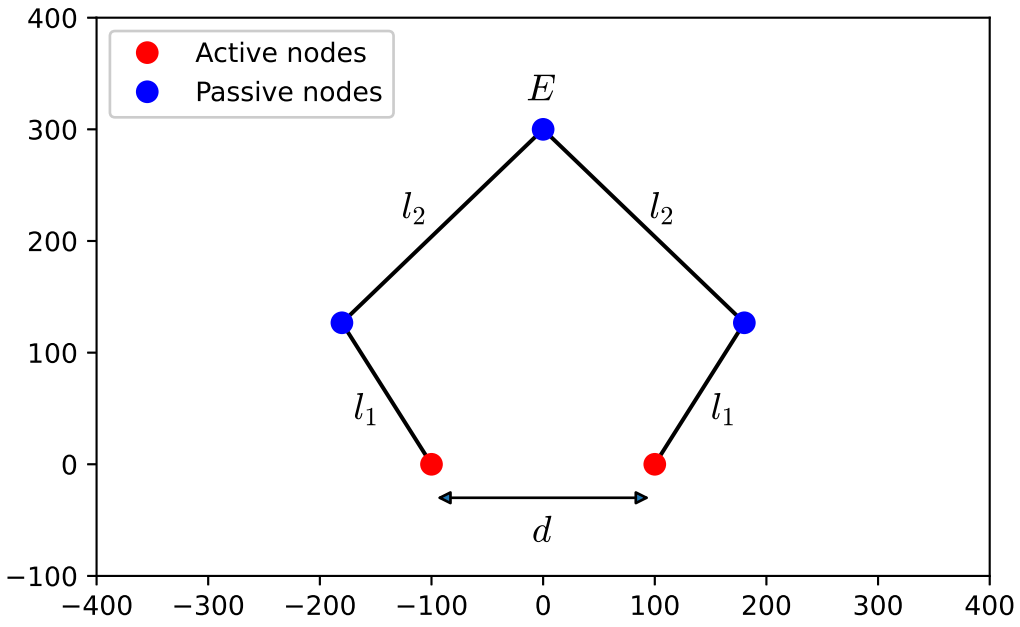


Figure 3.4: Schematic of dual-arm SCARA robot.

3.2.1 SCARA Arms Configuration

In this section, we introduce two SCARA arms configurations. Figure 3.5 shows a configuration of 2-3-2-1 (Level 2: Left l_1, Level 3: Left l_2, Level 2: Right l_2, Level 1: Right l_1). Using this configuration, the arms of the robot will enter into collision, which left l_1 will collide with right l_2 when the end effector is positioned along the x-axis and y-coordinate is 0 (schematic of dual-arm SCARA robot is shown in Figure 3.4). Hence, this configuration is deprecated.

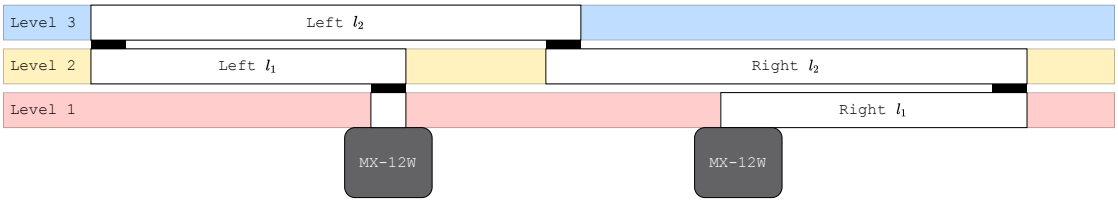


Figure 3.5: SCARA arms configuration 2-3-2-1.

On the other hand, another configuration is proposed using 1-3-2-1 as shown in Figure 3.6. By applying this configuration, left l_2 and right l_2 will not collide together. However, left l_1 and right l_1 will collide when both arms are trying to close to each other as the distance between two DYNAMIXEL MX-12Ws is 200mm. But there is no reason for both arm close together. Thus, this configuration is used in this project.

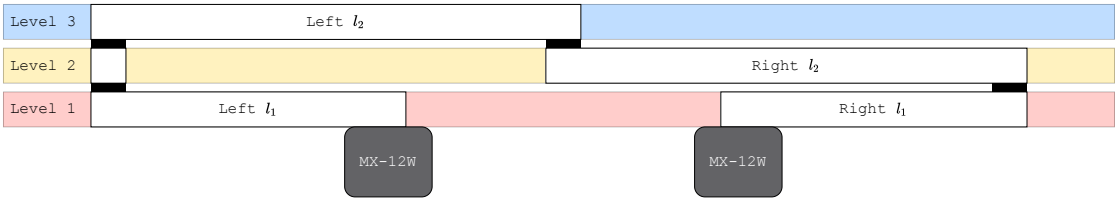


Figure 3.6: SCARA arms configuration 1-3-2-1.

3.2.2 Computer-Aided Design Software

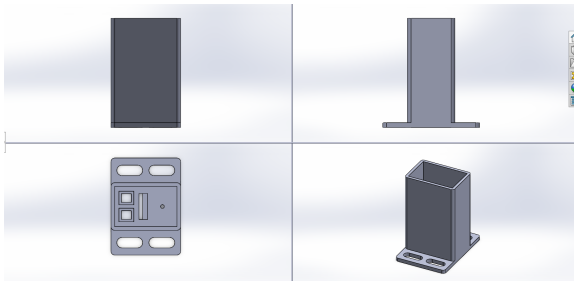
The use of computers to assist in the development, modification, analysis, or optimization of a design is known as computer-aided design (CAD). This application is used to improve the efficiency of the designer, the quality of the design, communication through documentation, and the creation of a database for production. Electronic files for printing, machining, and other manufacturing procedures are common CAD outputs.

Based on market statistics, commercial software from Autodesk and Dassault Systèmes dominate the CAD industry. Fusion 360 is a cloud-based 3D modelling, CAD, CAM, and PCB software platform for product design and manufacturing that developed by Autodesk ("Fusion 360: Cloud Powered 3D CAD/CAM Software for Product Design", 2022). Designers are able to gain access to a comprehensive set of 3D modelling tools that include parametric, freeform, direct, and surface modelling. On the other hand, Dassault Systèmes has developed another CAD software called SolidWorks. SolidWorks helps to create fast and accurate designs, including 3D models and 2D drawings of complex parts and assemblies. At the same time, it helps to eliminate errors and rework by using integrated motion and stress analysis tools.

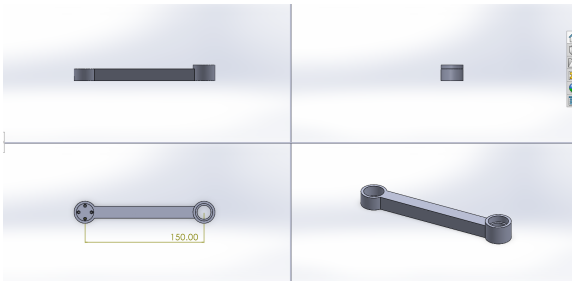
Fusion 360 and SolidWorks are two good programs, and many people use them to develop models for 3D printing. So in this project, we chose to use SolidWorks as CAD tool to design the dual-arm SCARA robot.

3.2.3 3D Modelling

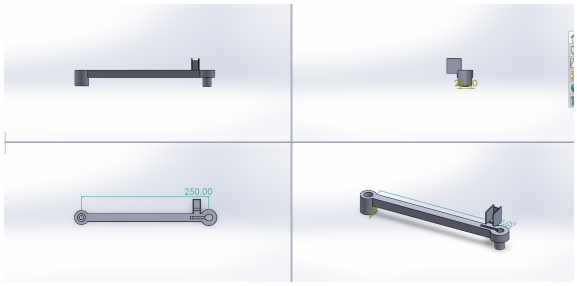
The parts of the dual-arm SCARA robot are designed based on the dimensions () using SolidWorks. Figure 3.7 shows the designed parts in workspace of SolidWorks. Every parts are designed from scratch without using design library.



(a) Actuator holder.

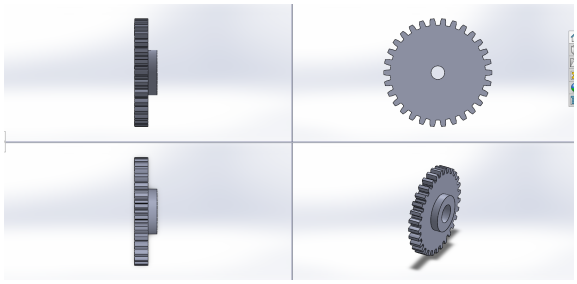


(b) Left l_1 and right l_1.

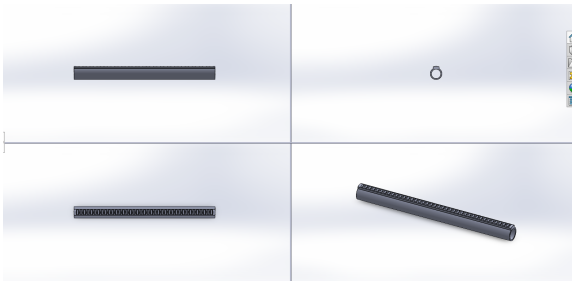


(c) Left l_2.

(d) Right l_2.



(e) Gear.



(f) Gear rod.

Figure 3.7: Parts of dual-arm SCARA robot.

The dual-arm SCARA robot is inflexible in the z-axis. One solution to this is to design a gear (Figure 3.7e) and a gear rod (Figure 3.7f). The gear meshed with the gear rod to convert rotational motion into a linear motion to transmit power. For instance, the gear is often attached to the output shaft of MG90S micro servo in a case where the gear is stationary and the gear rod moves. Additionally, the structure of the SCARA arms supports the driven side of the gear rod. As a result, the repetitive rotational motion of the gear produces a repeated

forward and backward motion of the gear rod. Figure 3.8 shows the vertical mechanism using the combination of gear and gear rod.

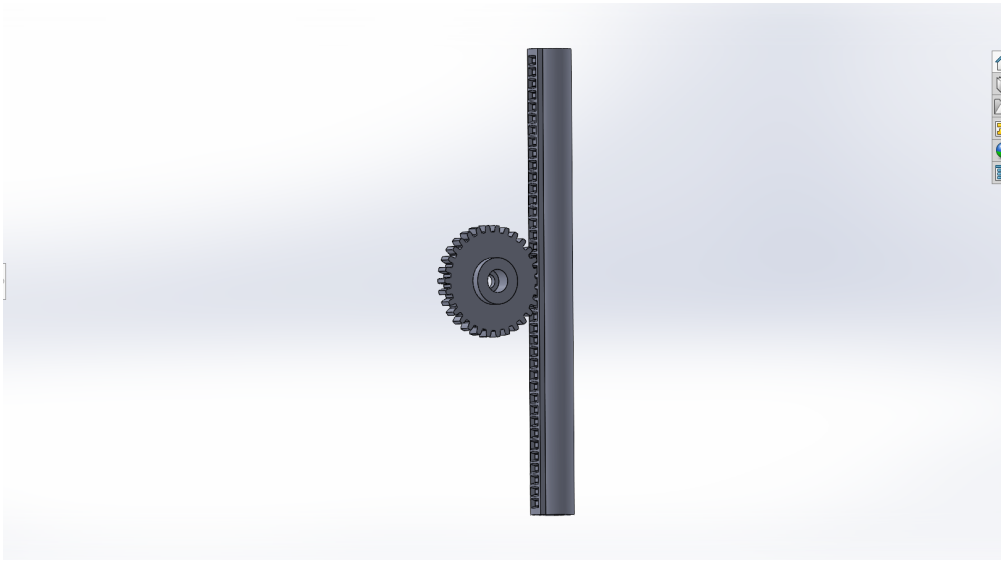


Figure 3.8: Vertical mechanism of the dual-arm SCARA robot.

3.2.4 3D Printing

The production of a three-dimensional item from a CAD model or a digital 3D model is referred to as 3D printing or additive manufacturing. One of the critical benefits of 3D printing is the capacity to create extremely complicated forms or geometries that would be generally difficult to construct manually, including empty parts or parts with inward support designs. Fused deposition modelling (FDM) uses continuous filaments of thermoplastic materials and is the most well-known 3D printing process ("What is FDM 3D Printing? - Simply Explained", 2021). Therefore, the parts of the dual-arm SCARA robot are printed with FDM approach using Creality Ender-3 V2 FDM 3D Printer. Figure 3.9 shows the parts of the dual-arm SCARA robot, which printed in yellow filament.

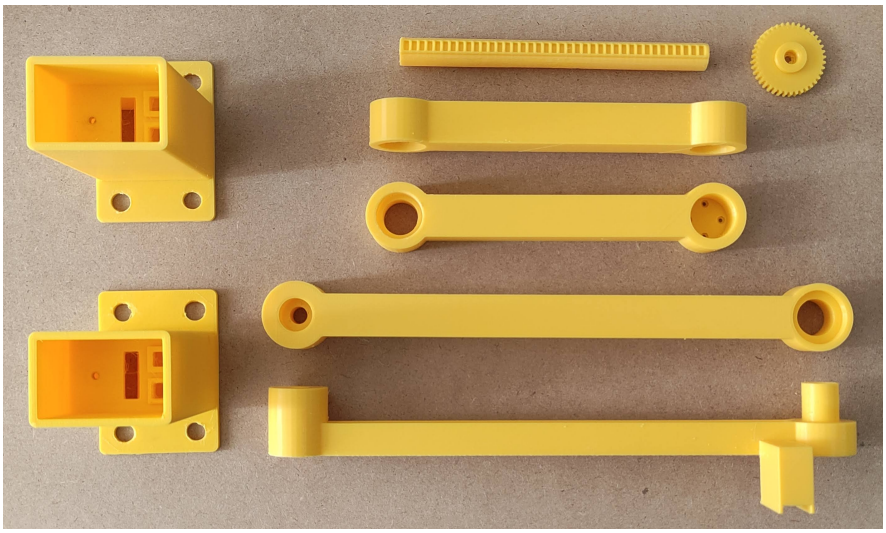


Figure 3.9: Printed parts of the dual-arm SCARA robot.

The parts are printed in yellow colour because the robot is massive and powerful machine that is programmed to do a task without regard for the environment, so it is quite harmful. When the robot is in action, there is usually a line surrounding it delimiting a danger zone where no one should be. But robot with yellow colour makes it easier to notice and yellow colour often associated to warnings and danger.

3.3 Software Development

This section describes software development process of the pick and place automation system.

3.3.1 Development Environment

A development environment is a set of procedures and tools for creating the source code for an application software product. This includes development, staging, and production servers, as well as the full infrastructure that supports the process from start to finish. The development environment automates or simplifies the procedures for developing, testing, debugging, patching, updating, and maintaining software, including long-term maintenance. In the following sections, we introduce the set up of development environment for this project.

Debian

Debian is an operating system and a distribution of Free Software. It is also one of the most ancient Linux-based OSs. Debian offers access to around 59 000 packages in online repositories ("About Debian", n.d.). Non-free software can be downloaded and installed from the Debian repository, but only free software is officially supported by Debian.

Besides, package management activities may be carried out with a variety of Debian tools, ranging from the command `dpkg` to graphical front-ends like Synaptic. Alternatively, the `apt` toolset is the preferred standard for managing packages on a Debian system. An Advanced Packaging Tool (APT) allows administrators to retrieve and install package dependencies from repositories on a Debian system. APT shares dependency information as well as packages that have been cached. Compared to more specialised APT like `apt-get` and `apt-cache`, the `apt` command is meant as an end user interface and by default allows various settings better suited for interactive usage. For instance, `git` and `ssh` can be installed using `apt` command.

Debian supports both GUI and CLI environments. In this case, a CLI environment is installed instead of GUI environment. Knowing CLI commands and utilising high-quality tools may help developers be more productive, as well as open the door to a variety of automation techniques that are considerably more practical with textual interfaces than with GUI applications. Additionally, command line software and keyboard-driven, console-based mandatory interfaces can provide easy interaction for casual users without the help from more experienced users. Many of these applications will provide simple usage information when invoked from the shell with the `--help` option, or quick access to usage help from within a captive interface with a few keystrokes. Beginners may quickly understand the fundamentals, and as they learn more about how to use it, their ability to utilise it improves.

As a user, a developer, or even in a corporate environment, there are many reasons to install Debian as operating system. The reliability and smooth upgrading operations of both packages and the complete distribution are praised by the majority of users. Debian is especially popular among software and hardware developers since it supports a broad range of architectures and devices, as well as providing a public bug tracker and other developer tools. As a result, Debian is selected as an operating system for the mini PC.

Visual Studio Code

Visual Studio Code is a lightweight yet capable source code editor developed by Microsoft that runs on the desktop and is compatible with Windows, macOS, and Linux. It contains built-in support for JavaScript, TypeScript, and Node.js, as well as a large ecosystem of extensions for additional languages, such as C++, C#, Java, Python, PHP, and Go. Support for debugging, syntax highlighting, smart code completion, snippets and code refactoring are among its features. In addition, Visual Studio Code comes with source control built-in, such as Git (see more about this in Section Git). This enables users to build repositories and push and pull requests straight from the Visual Studio Code software. Besides, it comes with a fully functional integrated terminal that is conveniently located at the root of workspace. This gives developers the ability to type and execute text-based commands in the same workspace. For these reasons, the Visual Studio Code is what we used to create programs and execute commands in this project.

Remote Development

A remote development is a server hosted externally that developers can connect remotely to do development tasks. However, Visual Studio Code comes with this feature. Developers may utilise a container, a remote computer, or the Windows Subsystem for Linux (WSL) as a full-featured programming environment using Visual Studio Code Remote Development. This prevents affecting the local system setup and keep the development environment separately. Therefore, no source code is required on the local machine.

In order to remote mini PC, we need to set up the remote development in local machine using Visual Studio Code. The Remote Development extension pack, includes Remote - SSH, Remote - Containers and Remote - WSL should be installed in Visual Studio Code. After that, Remote - SSH is used to connect to SSH hosts via SSH. SSH, also known as Secure Shell, is a network protocol that allows users, notably system administrators, to connect to a computer over an insecure network in a secure manner. Secure Shell allows for robust password and public key authentication, as well as encrypted data exchanges between two computers connected through an open network like the internet. In order to register SSH key, the script can be seen in Appendix A.1.

After finishing setting up remote development, we can clone the repository using Git (more on this in the next section). Finally, the remote development environment makes simple for new contributors to get started, and maintain a uniform environment for everyone.

Git

As highlighted in previous section, Git is a useful tool for source code management. Git allows developers to monitor changes in any group of files. It is typically used to coordinate work among programmers who are working on source code together during software development. Speed, data integrity, and support for distributed, non-linear processes are among its objectives. Every Git directory on every PC, like most other distributed version control systems and unlike most client-server systems, is a full-fledged repository with entire history and full version-tracking capabilities, independent of network connectivity or a central server.

Besides, there are free Git graphical user interface (GUI) applications available in the market. A Git GUI allows developers to view their repositories in a visual format so that they may focus only on coding. However, Git also offers several command-line commands because command-line interface is widely used for developers compare to GUI. Following is a very brief introduction to these commands that can be used often in Git:

- `git init`

  To initialize an existing directory as a Git repository.

- `git clone [url]`

  To retrieve an entire repository from a hosted location via URL.

- `git status`

  To show modified files in working directory, staged for your next commit.

- `git add [file]`

  To add a file as it looks now to your next commit (stage).

- `git commit -m ‘‘[message]’’`

  To commit your staged content as a new commit snapshot.

- `git push [alias] [branch]`

  To transmit local branch commits to the remote repository branch.

- `git pull`

  To fetch and merge any commits from the tracking remote branch.

Git is a free and open source distributed version control system that can easily manage everything from tiny to extremely large projects. As a result, Git is an essential tool For every developers.

Arduino CLI

Arduino CLI is a command line tool that includes Board and Library Managers, a sketch builder, board identification, uploader, and a slew of other utilities for using any Arduino compatible board and platform. In addition, Arduino CLI can be install on both ARM® and Intel® (x86, x86-64) architectures ("Arduino CLI (Command Line Interface) Application", n.d.). For these reasons, Arduino CLI is able to run on a Raspberry Pi or servers, so that it can be use to compile sketches on the targeted board.

Since an Arduino MKR WiFi 1010 is used in this project, we need to download the latest version of Arduino SAMD core. Other than that, the Arduino environment may be expanded by using libraries. Libraries add further functionality in sketches, such as the ability to interface with hardware or manipulate data. The following list shows the libraries used in this project:

- `DynamixelShield`

  To control DYNAMIXEL MX-12W (inherits to DYNAMIXEL2Arduino library).

- `Servo`

  To control MG90S metal gear micro servo.

To install Arduino CLI, cores and libraries, the guides can be seen in Appendix A.2.

On the other hand, a serial monitor tool is needed to transmit and receive data between mini PC and Arduino MKR WiFi 1010. Although the Arduino CLI does not include serial monitor tool, there are two common tools that we can install in Linux, which are Minicom and Picocom. Both tools are very useful, but we chose Picocom for this project. Picocom was created to be a simple modem configuration, testing, and debugging tool that could be used by any developer. With picocom, developers can monitor and communicate with Arduino

board. As a result, every character entered in picocom will be sent directly to Arduino board, that is called serial communication (more on this in Section 3.3.5).

Python

Python is a powerful programming language that can be used for web development, machine learning, data science and so on. It supports a variety of programming paradigms, including procedural, object-oriented, and functional programming. Besides, it is a dynamically typed where the variable type is determined during runtime. However, programmers are allowed to create their own types using classes in Python that commonly used in object-oriented programming.

In this case, a Python 3.10.1 is installed in mini PC. Before work on the project, it is strongly recommended to create a virtual environment. Virtual environments are an indispensable part of Python programming. They are an isolated container that contains all of the project's software dependencies. This is significant since Python and OpenCV are installed in the same directory by default. If virtual environment is used, developers can work on various projects on the same PC without having conflicting on library versions. While the environment is active, any library install using `pip` will be installed in this virtual environment. In this project we used `virtualenv` to mange virtual environments.

On the other hand, there are a bunch of packages for Python can be found in PyPI. PyPI (Python Package Index) is a software repository for the Python programming language. It helps developers in locating and installing Python-based software that has been developed and shared by the Python community. The following list shows the packages required to install for this project:

- `opencv-contrib-python-headless`

  To build shapes and colours detection and recognition algorithms.

- `pyserial`

  To perform serial communication between mini PC and Arduino MKR WiFi 1010.

- `numpy`

  To store colours and images.

- `scipy`

  To compute the distance between the current colour value and the mean of the image.

Finally, the development environment for Python is set up. We have work with virtual environments and latest version of Python. Everything is ready for the software development in this project.

3.3.2 SCARA Coverage Area Modelling

In this section, we show the theoretical workspace for the dual-arm SCARA robot. As shown in Figure 3.4, the distance between two active nodes is 200mm, and both active nodes are actuated with DYNAMIXEL MX-12W, that has a resolution of 4096 pulses per revolution ("MX-12W", n.d.). Therefore, the theoretical possible positions of the end effector have been plotted in the following figures. Figure 3.10 shows the upper coverage area of the workspace that starts from positive y-axis. On the other hand, the lower coverage area of the workspace starting from the negative y-axis is shown in Figure 3.11. And finally, Figure 3.12 shows the fully robot's theoretical workspace with excluding mechanical interferences. However, when implemented in practice, the usable workspace of the dual-arm SCARA robot is much smaller because of mechanical interferences.

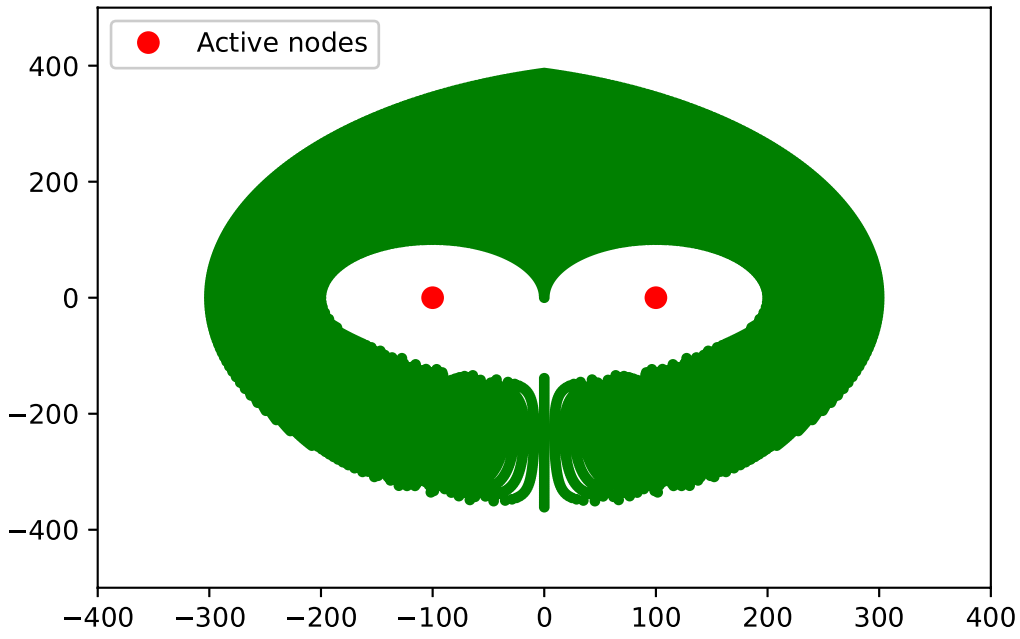


Figure 3.10: SCARA Upper Coverage Area Modelling (Possible locations at upper coverage area).

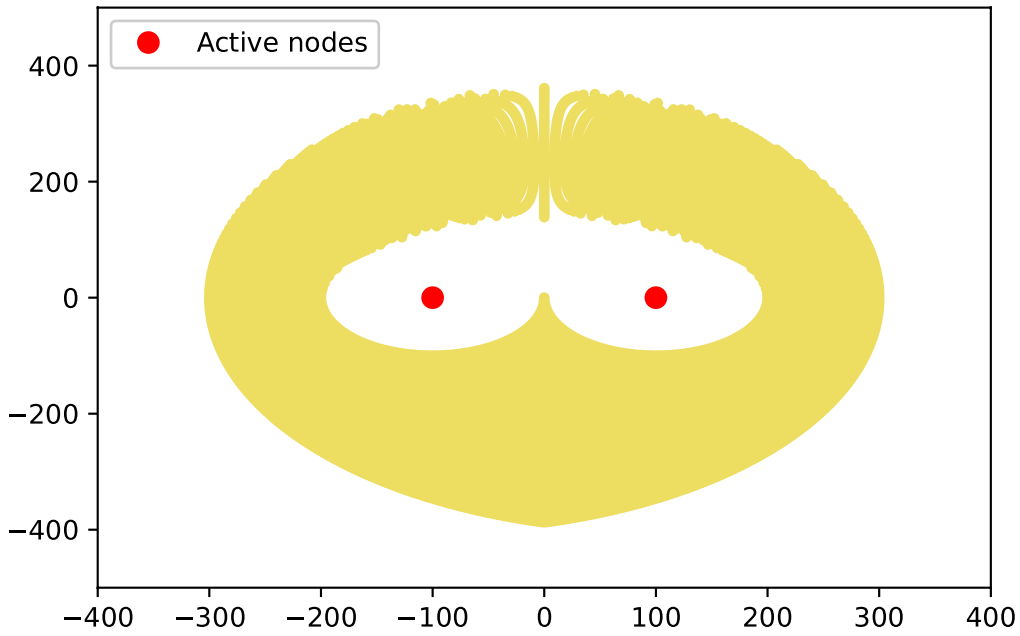


Figure 3.11: SCARA Lower Coverage Area Modelling (Possible locations at lower coverage area).

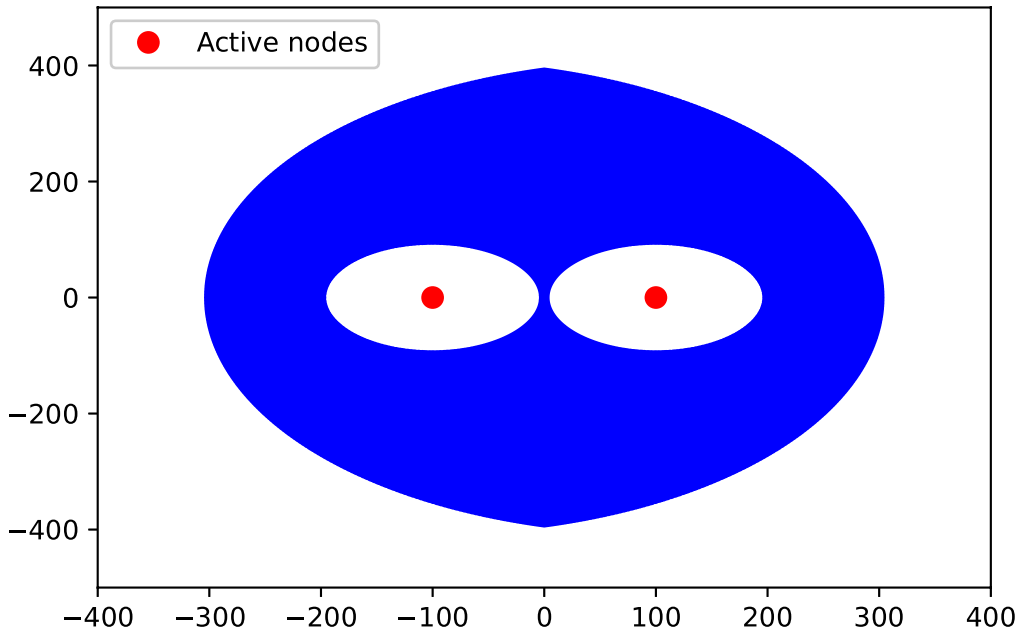


Figure 3.12: SCARA Fully Coverage Area Modelling (Possible locations at fully coverage area).

3.3.3 Inverse Kinematics

Inverse kinematics (IK) is the process of computing the angles of the active nodes by given a desired location in order to locate the end effector at the desired location. However, there is

more than one solution, and sometimes it may be difficult to solve. This section outlines the existing inverse kinematics solution and a proposed inverse kinematics used in this project.

Existing Inverse Kinematics

In this section, we discuss about the existing inverse kinematics solution. The solution to the inverse kinematics can be easily calculated using trigonometry and cosine rule (Le et al., 2013). The essential dimensions can be summarized on Figure 3.13, where $d = 200$mm, $l_1 = 150$mm and $l_2 = 250$mm. We compute θ_1 and θ_2 from the given $E\,(x, y)$.

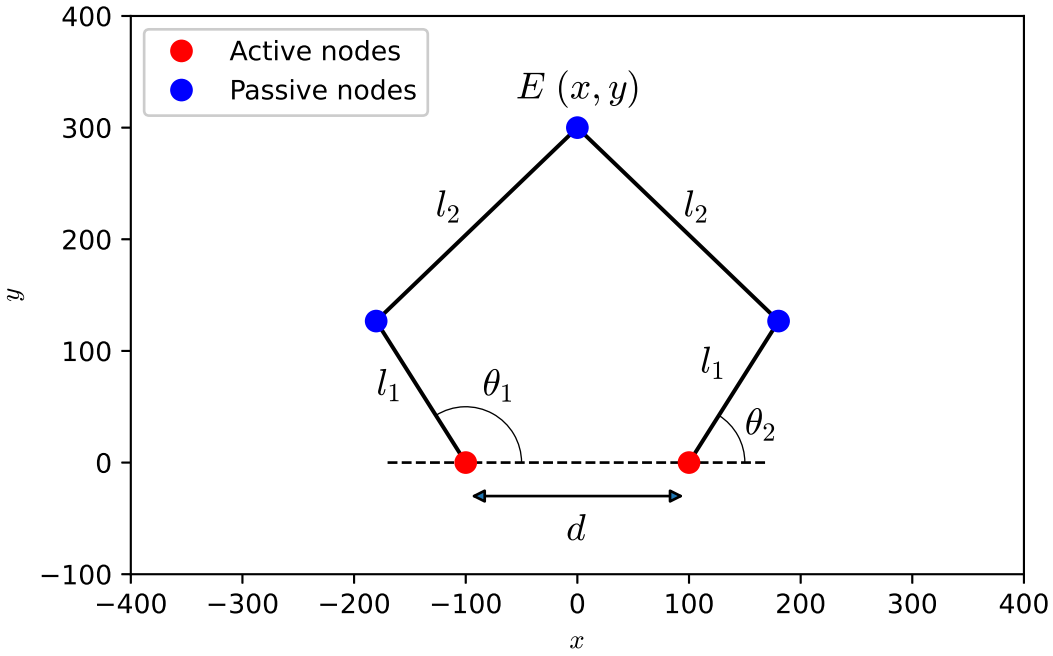


Figure 3.13: Inverse kinematics of the dual-arm SCARA robot.

$$\theta_1 = \arctan\left(\frac{y}{\frac{d}{2}+x}\right) \pm \arccos\left(\frac{l_1^2 - l_2^2 + \left(\frac{d}{2}+x\right)^2 + y^2}{2l_1\sqrt{\left(\frac{d}{2}+x\right)^2 + y^2}}\right) \tag{3.1}$$

$$\theta_2 = \pi - \arctan\left(\frac{y}{\frac{d}{2}-x}\right) \pm \arccos\left(\frac{l_1^2 - l_2^2 + \left(\frac{d}{2}-x\right)^2 + y^2}{2l_1\sqrt{\left(\frac{d}{2}-x\right)^2 + y^2}}\right) \tag{3.2}$$

The angles of the dual-arm SCARA robot can be configured using four different modes (Le et al., 2013). These modes are summarized as follows.

$$Let \quad m_1 = \frac{y}{\frac{d}{2}+x} \quad , \quad n_1 = \frac{l_1^2 - l_2^2 + \left(\frac{d}{2}+x\right)^2 + y^2}{2l_1\sqrt{\left(\frac{d}{2}+x\right)^2 + y^2}} \tag{3.3}$$

$$Let \quad m_2 = \frac{y}{\frac{d}{2}-x} \quad , \quad n_2 = \frac{l_1^2 - l_2^2 + \left(\frac{d}{2}-x\right)^2 + y^2}{2l_1\sqrt{\left(\frac{d}{2}-x\right)^2 + y^2}} \tag{3.4}$$

1. Mode "$+-$" : $\theta_1 = \arctan(m_1) + \arccos(n_1)$ and $\theta_2 = \pi - \arctan(m_2) - \arccos(n_2)$
2. Mode "$-+$" : $\theta_1 = \arctan(m_1) - \arccos(n_1)$ and $\theta_2 = \pi - \arctan(m_2) + \arccos(n_2)$
3. Mode "$--$" : $\theta_1 = \arctan(m_1) - \arccos(n_1)$ and $\theta_2 = \pi - \arctan(m_2) - \arccos(n_2)$
4. Mode "$++$" : $\theta_1 = \arctan(m_1) + \arccos(n_1)$ and $\theta_2 = \pi - \arctan(m_2) + \arccos(n_2)$

Although the calculation is easy to understand, it is difficult to satisfy the behaviour of the dual-arm SCARA robot we want. Therefore, we proposed an inverse kinematics in Section Proposed Inverse Kinematics.

Proposed Inverse Kinematics

The proposed inverse kinematics is presented in this section. This method is based on using the intersection of circles to calculate the angles of active nodes. It is also easy to determine whether the desired location is reachable using this approach.

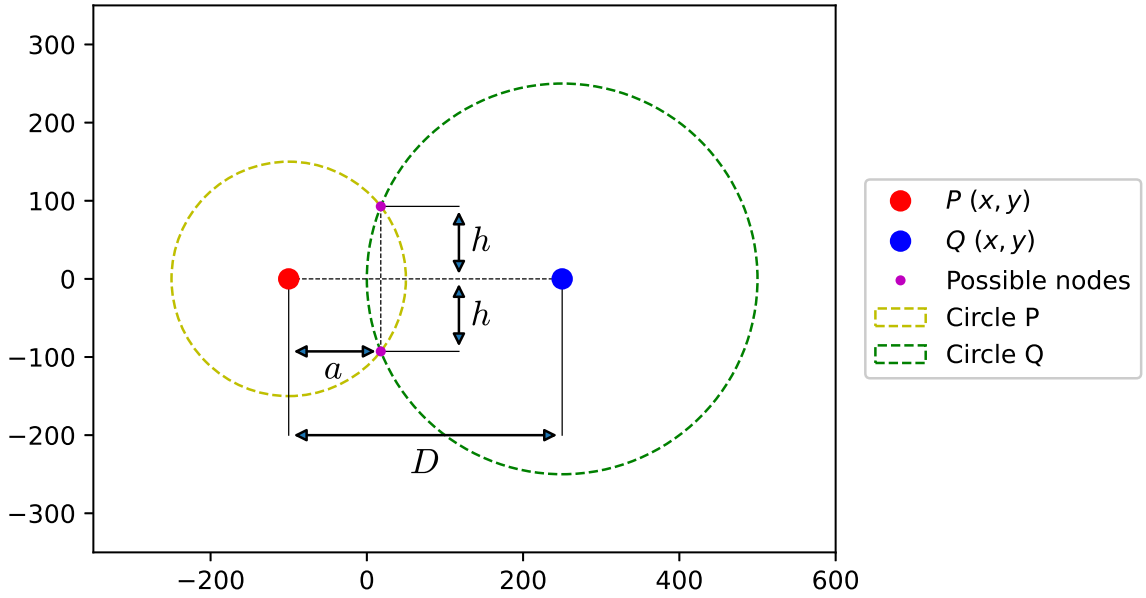


Figure 3.14: Inverse kinematics using intersection of circles.

Figure 3.14 shows the inverse kinematics using intersection of circles. Circle P and Circle Q intersect at two points, where the intersection points are the possible solutions to locate the passive nodes. The calculation for intersection of circles is summarized as follows.

$$D = \sqrt{(Q_x - P_x)^2 + (Q_y - P_y)^2} \tag{3.5}$$

$$a = \frac{P_r^2 - Q_r^2 + D^2}{2D} \tag{3.6}$$

$$h = \sqrt{P_r^2 - a^2} \tag{3.7}$$

$$x_1 = \frac{P_x + a\,(Q_x - P_x) + h\,(Q_y - P_y)}{D} \tag{3.8}$$

$$y_1 = \frac{P_y + a\left(Q_y - P_y\right) - h\left(Q_x - P_x\right)}{D} \tag{3.9}$$

$$x_2 = \frac{P_x + a\left(Q_x - P_x\right) - h\left(Q_y - P_y\right)}{D} \tag{3.10}$$

$$y_2 = \frac{P_y + a\left(Q_y - P_y\right) + h\left(Q_x - P_x\right)}{D} \tag{3.11}$$

From the calculation above, (x_1, y_1) and (x_2, y_2) are the coordinates for both possible nodes. As a result, two lines can be drawn from active node $[P\left(x, y\right)]$ to each possible node. At the same time, the angle between each line and x-axis can be calculated using the formula below.

$$\theta = \arctan\left(\frac{y - P_y}{x - P_x}\right) \tag{3.12}$$

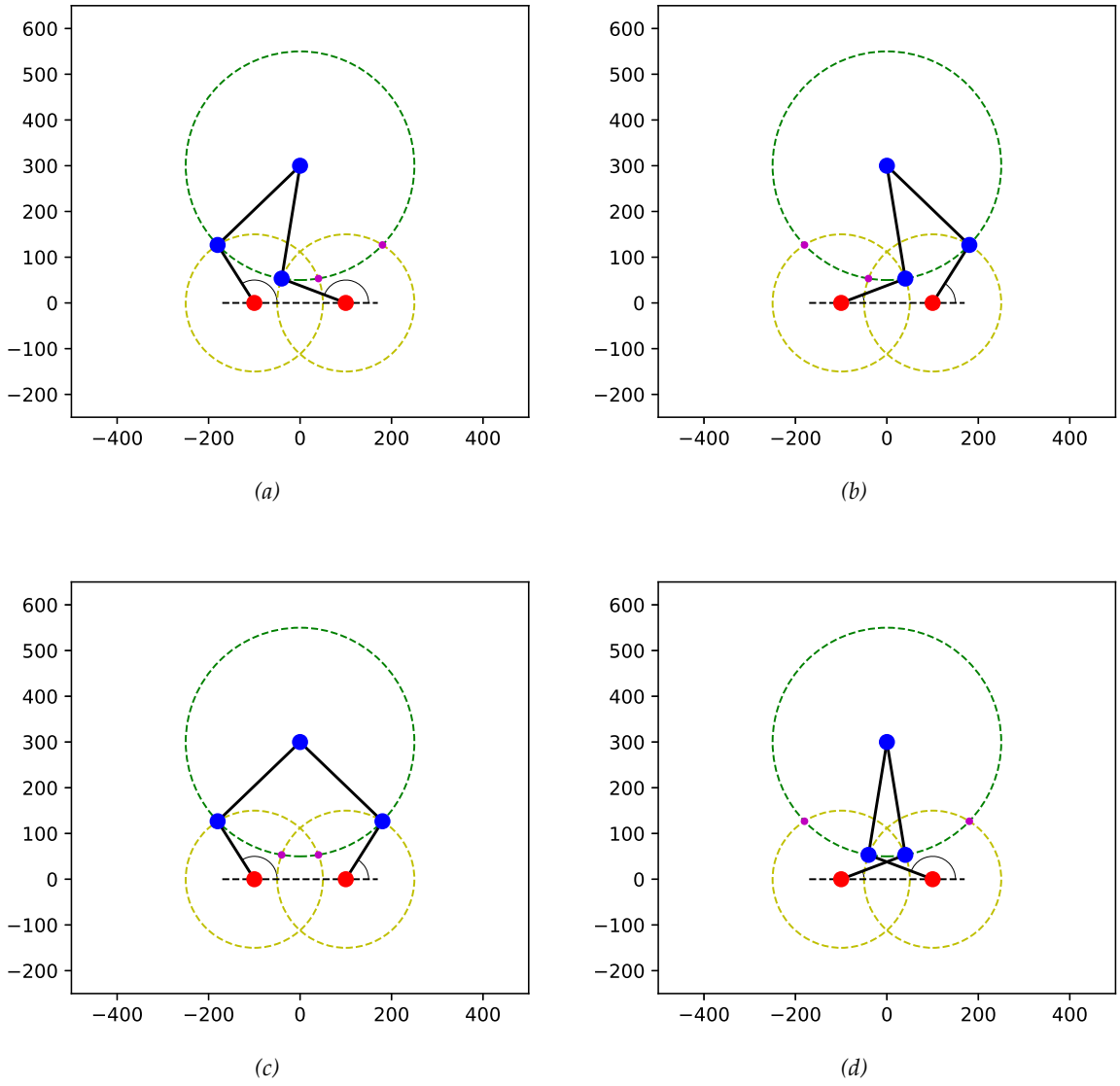


Figure 3.15: Four possible solutions to reach desired position.

As depicted in Figure 3.15, there are four possible solutions to reach the desired position (labels are defined in Figure 3.13). However, we need to choose one of four solutions that has the shortest distance from the current angle to the target angle. This can be done by getting the minimum difference between current angle and all the possible target angles. As a result, the smallest pair is selected. On the other hand, Figure 3.15d shows both l_1 intersect each other. In practice, both l_1 are built at the same level (seen in Figure 3.6), so this will cause left l_1 and right l_1 collide and stick together. Therefore, in order to eliminate the solution in Figure 3.15d, we need to verify that x-coordinate of passive node for left l_1 should not pass through x-coordinate of passive node for right l_1.

In conclusion, the proposed inverse kinematics can be implemented in three main steps. First, the intersection points between two circles are determined. Second, the angles between active node and each possible node are calculated. Lastly, the collided l_1 should be eliminated.

3.3.4 Shape and Colour Detection and Recognition

In this project, the vision system able to detect and recognise three different shapes (triangle, square and circle) and two colours (black and red). The solution to this problem is to implement the algorithm using OpenCV that originally developed by Intel.

OpenCV

OpenCV is a free software library for computer vision and machine learning. OpenCV was created to offer a standard infrastructure for computer vision applications and to let commercial solutions integrate machine perception more quickly. Besides, more than 2500 optimised algorithms are included in the library, which contains a comprehensive mix of both traditional and cutting-edge computer vision and machine learning techniques ("About", 2020). These algorithms can be used to detect and recognise faces, determine objects, track camera movements, classify human actions in videos, track moving objects, follow eye movements, recognise scenery, and so on ("About", 2020). The library is widely utilised by businesses, research groups, and government agencies.

Additionally, OpenCV supports Windows, Linux, Android, and macOS and offers C++, Python, Java, and MATLAB interfaces. Other than that, CUDA and OpenCL interfaces with full functionality are currently being developed. Over 500 algorithms exist, with roughly ten times as many functions that compose or support them. OpenCV is developed in C++ and provides a templated interface that integrates with STL (Standard Template Library) containers elegantly. To build shape and colour detection and recognition algorithms, OpenCV is used, which we coded it in Python without using CUDA.

Virtual Camera

Before implementing the algorithms, we need to set up a virtual camera for OpenCV. One of the advantages of creating virtual camera is that it can be created from a custom video

source. Consider being able to get video from an RTSP source, process it using OpenCV like adding some cool filters, and then distributing the output to the rest of the system, much like a standard USB webcam. At the same time, the processed video might be used in any other software, such as web browser, Zoom, Microsoft Teams, Google Meet, Skype, and so on.

To do this, the video is sourced from a Logitech C270 HD camera connected through USB, which means that the Video For Linux (v4l2) subsystem will be in charge. Then, a virtual device is created by the module Video For Linux Loopback (v4l2loopback). The v4l2loopback is the crucial component that provides virtual video devices that typical applications would treat as if they were real devices, but the video stream will originate from another application or service rather than from some piece of hardware. Because v4l2loopback is a kernel module, we achieve this by registering it with the appropriate parameters inside the kernel. In this case, the command `modprobe v4l2loopback` is used to set the identifier of the virtual camera as `/dev/video6`, and name it as `OpenCV Camera`.

After that, the image data is retrieved frame by frame using OpenCV, and it is applied with the filters and algorithms that we want. Once we have achieved the desired output, the frame is written to the created virtual camera. Finally, we able to view the output device virtually using Linux software. In this case, Motion is used to output the result and it can be accessed via localhost with port 8080 by default.

Algorithms

In this section, we introduce the algorithms of shape and colour detection and recognition in OpenCV. After getting a frame, the first stage is to eliminate high-frequency material from the image, such as noise and edges. As a result, the edges are slightly blurred in this procedure. However, there are four different types of blurring algorithms available in OpenCV. In this case, a Gaussian kernel is used. This is because when it comes to eliminating Gaussian noise from a image, Gaussian blurring is quite successful. In addition, it can be done with the `cv.GaussianBlur()` function. Therefore, a 7×7 Gaussian filter is applied to the image.

Then, the frame is converted from RGB (Red Green Blue) into grayscale. In many circumstances, analysing a RGB image will not provide more information than analysing a

grayscale image, but grayscale image analysis is around three times faster. Therefore, grayscale colour space is more useful for intermediate processing. This can be done by converting the blurred image to grayscale image using the function, `cv2.cvtColor()`.

One step before finding the contours, Canny edge detection is applied to the converted grayscale image with a value of 50 for both minimum and maximum. The Canny Edge Detection technique is a well-known edge detection method. John F. Canny created it in 1986. It is a method for extracting relevant structural information from a variety of visual objects while drastically reducing the quantity of data that has to be processed ("Canny Edge Detection", n.d.). Besides, it has been used in a lot of computer vision systems. And finally, the Canny edge detection can be applied using `cv2.Canny()` function.

In order to detect and recognise shape, contour detection algorithm is implemented to the Canny image. Contours are just a curve that connects all of the continuous points along the border that have the same colour or intensity. The contours are a powerful tool for object detection and recognition as well as shape analysis. For more accurate result, it is recommended to use use a binary image, as we applied Canny filter before looking for contours. This is because finding contours in OpenCV is similar to finding a white object against a black background. Hence, to find out the shapes in the image, the `cv2.findContours()` method is used.

After detecting all contours in the image, the contours are drew on the detected shapes in the image. In this project, there are four triangles, four squares and four circles should be detected and recognised. To be able to recognise the shapes, they are labelled based on the number of vertices according to the approximated contour. For instance, if the approximated contour has three vertices, the detected shape should be a triangle. Therefore, the shape is labelled with `triangle`.

On the other hand, to recognise the colour of the detected shape, the blur image is converted to L*a*b* colour space. Then, calculate the Euclidean distance between each known colour (black and red) and the average colour, and then return the name of the colour with the smallest Euclidean distance. Theoretically, the length of a line segment connecting two locations in Euclidean space is the Euclidean distance between them. It is also referred to as the Pythagorean distance since it can be determined from the Cartesian coordinates of the locations using the Pythagorean theorem. For these reasons, we used L*a*b* colour space because of the Euclidean distance between L*a*b* colours has more significant meaning

compare to HSV or RGB.

Finally, Figure 3.16 summarises the process of shape and colour detection and recognition.

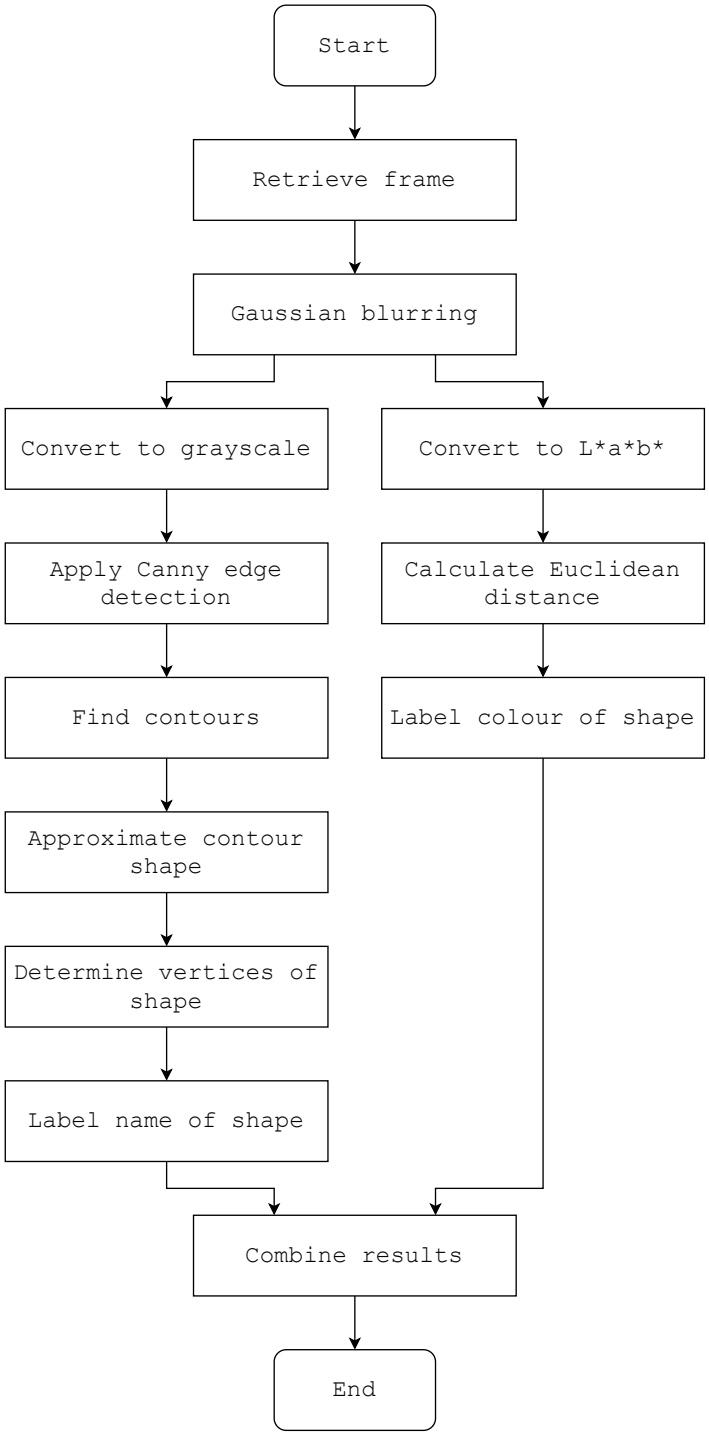


Figure 3.16: Flowchart for shape and colour detection and recognition.

3.3.5 Serial Communication

The connection between Arduino and Python can be easily done via serial communication. Both Arduino and Python contain a module that allow access for the serial port, which are Serial and pySerial respectively.

In order to achieve high speed communication, the transmitted data should be in byte array, which only accepts values from 0 to 255. Therefore, if a value is out of range, it should be manipulated before transmitting the data. For instance, the goal position of DYNAMIXEL MX-12W can be set from -28 672 to 28 672 (multi-turn mode), in this case we set a value of 1023. Therefore, the value should be modulus and divided by 255. The following Python code shows the manipulation of the value 1023:

```
angle = 1023
angle_1 = int(abs(angle) % 255)
angle_2 = int(abs(angle) / 255)
angle_3 = 2 if angle < 0 else 1
arr = [angle_1, angle_2, angle_3]
```

After that, a byte array, `[3, 4, 1]`, is sent to Arduino MKR WiFi 1010. The value `1` in the array indicated that 1023 is a positive value. However, if the value is -1023, then the byte array should be `[3, 4, 2]`.

At the same time, the values in the byte array should be manipulated after receiving it. Taking the above example, Arduino MKR WiFi 1010 is received a byte array of `[3, 4, 1]`. As a result, the values in the array should be manipulated before use. The following C++ code shows the manipulation of the byte array:

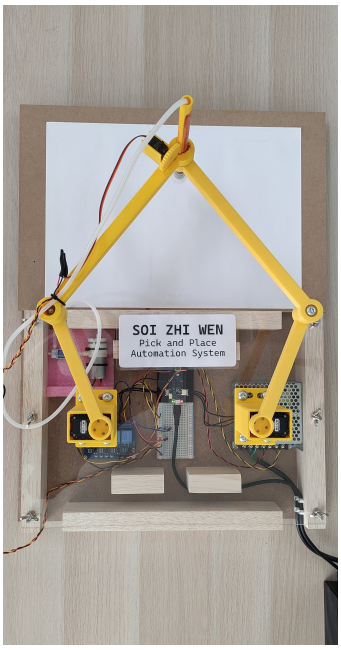
```
int sign = 1;
if (int(arr[2]) == 2)
    sign = -1;
int angle = (int(arr[0]) + (255 * int(arr[1]))) * sign;
```

Thus, the manipulated value should be 1023, which is same as the value in Python before being manipulated.

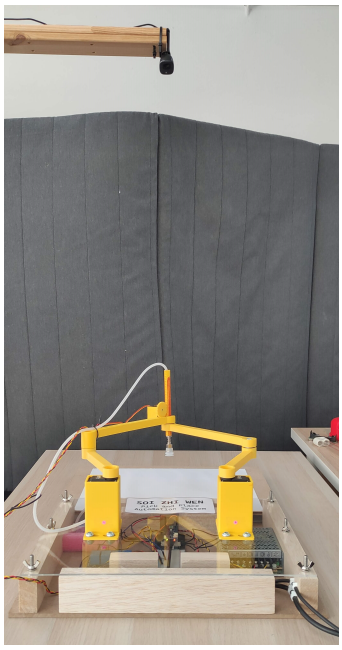
Chapter 4

Results and Analysis

The hardware and software of the system are tested to achieve the objectives of this project. This section shows the results acquired by testing the system and analyses the results to study them. The complete system consists of the dual-arm SCARA robot and vision system. Figure 4.1a shows the prototype of the dual-arm SCARA robot, whereas Figure 4.1b shows the camera mounted above the dual-arm SCARA robot.



(a) Prototype of the dual-arm SCARA robot.



(b) Camera mounted above the SCARA robot.

Figure 4.1: The complete system.

Several aspects of the project must be evaluated in order to determine the success of the system. The accuracy and ability of the system to identify objects based on the two criteria, which are shape and colour, as well as the correctness of the inverse kinematics algorithm,

are all sections of the system that must be evaluated. Because we are primarily concerned with sorting and usability, the speed, torque and power consumption of the system are not critical criteria for the system.

To evaluate the accurateness of shape recognition, if the detected object is labelled with the correct shape, the shape recognition test is considered a success. On the other hand, if the correct colour is labelled to the detected shape, then the colour recognition test is successful. Both shape and colour recognition tests are repeated for 20 times each. Table 4.1 and Table 4.2 show the results of the tests.

Table 4.1: Result of shape recognition test.

Shape	**Attempts**	**Success**	**Failure**	**Success Rate (%)**
Triangle	20	20	0	100
Square	20	19	1	95
Circle	20	20	0	100
Overall	60	59	1	98.33

Table 4.2: Result of colour recognition test.

Colour	**Attempts**	**Success**	**Failure**	**Success Rate (%)**
Black	20	20	0	100
Red	20	20	0	100
Overall	40	40	0	100

For shape recognition test, we have a total of 60 attempts, which 59 are successful, and 1 is considered failures. The algorithm for one of the square failures mistook it for a circle. We found that the failed attempt occur when shadows are cast on the workspace. However, the overall rate of success for shape recognition test has a 98.33%, which is acceptable. On the other hand, the overall success rate for colour recognition test is 100% and it can be successful because black and red have a big colour difference, so it is easy to identify between them. Figure 4.2 shows the result of correctly labelling each of the shapes and colours after implementing the detection and recognition algorithms. Since the white colour paper is placed underneath the objects, the output is very satisfactory.

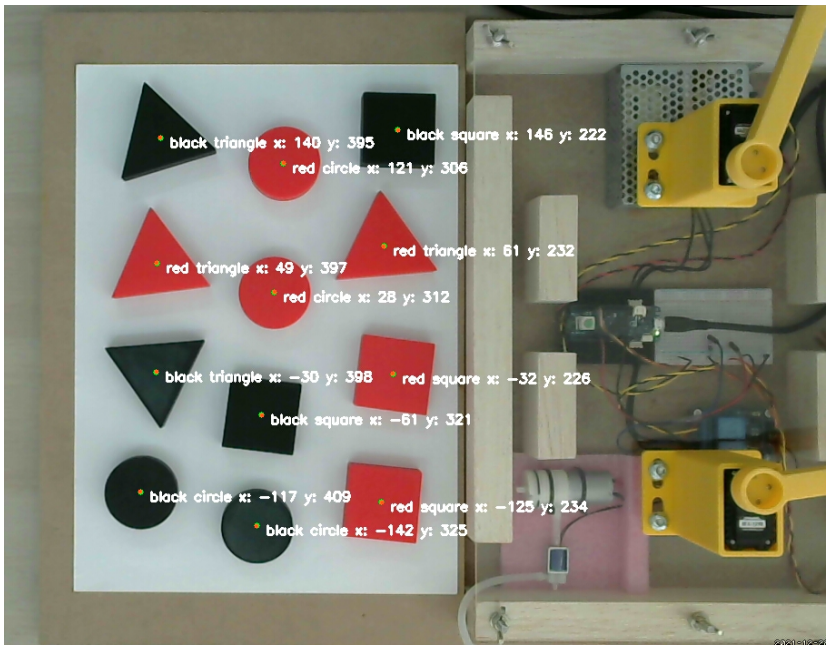


Figure 4.2: Result of shape and colour detection and recognition.

To determine the accuracy of the inverse kinematics algorithm, test was carried out to ensure that the dual-arm SCARA robot is placed correctly to pick up the target object. There are 5 x-coordinates and 3 y-coordinates were chosen as input for inverse kinematics algorithm. By measuring the distance between the end effector and the centroid of the object, an error was computed. The test was done three times for each coordinate. The result is shown in Table 4.3.

Table 4.3: Result of inverse kinematics algorithm test.

x **(cm)**	y **(cm)**	**Test 1 (cm)**	**Test 2 (cm)**	**Test 3 (cm)**	**Test Average (cm)**
-16	6	0.5	0.3	0.7	0.50
-16	12	0.9	0.7	0.9	0.83
-16	18	1.2	0.9	1.1	1.07
-8	6	0.8	0.5	0.5	0.60
-8	12	0.6	0.4	0.8	0.60
-8	18	0.9	0.7	0.7	0.77
0	6	0.2	0.4	0.3	0.30
0	12	0.5	0.6	0.5	0.53
0	18	0.8	0.3	0.7	0.60
8	6	0.6	0.7	0.6	0.63
8	12	0.7	0.7	0.7	0.70
8	18	0.8	1.1	0.6	0.83
16	6	0.2	0.5	0.5	0.40
16	12	0.5	0.7	0.9	0.70
16	18	1.0	1.2	0.9	1.03
Test Average (cm)		0.68	0.65	0.69	0.67

The average inaccuracy of the test was 0.67cm, indicating that the distance between the end effector and the centroid of object was 0.67cm on average. The highest inaccuracy was 1.2cm. The mechanical design of the SCARA arms is the cause of inaccuracy. This is because the end effector is heavy at the end of the SCARA arms. Therefore, the SCARA arms are bent when the end effector is far away from the origin, (0, 0). Besides, the two largest errors occurred at (-16, 18) and (16, 18). However, the reason mentioned earlier may be responsible for this inaccuracy.

Chapter 5

Conclusion

In this project, a dual-arm SCARA robot to perform pick and place tasks was presented. It is slightly compliant in the x and y direction but rigid in the z direction. Besides, an inverse kinematics was developed to be compatible with the dual-arm SCARA robot. The correctness and effectiveness of the solution was shown by the dual-arm SCARA robot simulation diagram. Finally, the vision system was implemented in the pick and place automation system in order to perform shape and colour detection and recognition. Once the shape and colour are classified, the system outputs the coordinate of the classified shape to be ready for the dual-arm SCARA robot to execute the pick and place task.

5.1 Future Work

As future improvements to the current system, it could be suggested to develop and test different end effectors on this dual-arm SCARA robot as the dual-arm SCARA robot is designed to install different types of end effectors, such as pneumatic gripper and soft gripper that mentioned in Section 3.1.6. Besides, by attaching an extruder to the end effector, the dual-arm SCARA robot can perform 3D printing tasks using a given STL file. On the other hand, the Arduino MKR WiFi 1010 can be accessed to a WiFi network, and connected to MQTT (Message Queuing Telemetry Transport), so that it can send/receive data to/from another WiFi compatible device, such as mobile phone. As a result, the dual-arm SCARA robot can be transformed into an IoT (Internet of Things) device, and the mobile phone can communicates with the Arduino MKR WiFi 1010 to control the dual-arm SCARA robot over MQTT.

References

About. (2020). *Opencv*. Retrieved January 10, 2022, from https://opencv.org/about/

About debian. (n.d.). *Debian*. Retrieved January 10, 2022, from https://www.debian.org/intro/about

Arduino cli (command line interface) application. (n.d.). *Arduino pro*. Retrieved January 10, 2022, from https://www.arduino.cc/pro/cli

Canny edge detection. (n.d.). *Opencv*. Retrieved January 11, 2022, from https://docs.opencv.org/3.4/da/d22/tutorial_py_canny.html

Dynamixel shield mkr. (n.d.). Retrieved January 8, 2022, from https://emanual.robotis.com/docs/en/parts/interface/mkr_shield/

Fusion 360: Cloud powered 3d cad/cam software for product design. (2022). *Autodesk*. Retrieved January 9, 2022, from https://asean.autodesk.com/products/fusion-360/overview

Introduction to infrared (part 1): The physics behind thermal imaging. (2021). *Opgal*. Retrieved January 9, 2022, from https://www.opgal.com/blog/thermal-cameras/the-physics-behind-thermal-imaging

Kumar, R., Lal, S., Kumar, S., & Chand, P. (2014). Object detection and recognition for a pick and place robot. *Asia-Pacific world congress on computer science and engineering*, 1–7. https://doi.org/10.1109/apwccse.2014.7053853

Le, T. D., Kang, H.-J., & Doan, Q. V. (2013). A method for optimal kinematic design of five-bar planar parallel manipulators. *2013 International Conference on Control, Automation and Information Sciences (ICCAIS)*, 7–11. https://doi.org/10.1109/iccais.2013.6720521

Lozano-Pérez, T., Jones, J. L., Mazer, E., & O'Donnell, P. A. (1989). Task-level planning of pick-and-place robot motions. *Computer*, *22*, 21–29.

Mx-12w. (n.d.). Retrieved January 8, 2022, from https://emanual.robotis.com/docs/en/dxl/mx/mx-12w/

Poda, X., & Qirici, O. (2018). Shape detection and classification using opencv and arduino uno. *RTA-CSIT*, *2280*, 128–36.

Shevkar, P., Bankar, S., Vanjare, A., Shinde, P., Redekar, V., & Chidrewar, S. (2019). A desktop scara robot using stepper motors. *International Research Journal of Engineering and Technology (IRJET)*, *06*.

What is a pick and place robot? (2021). *6 river systems*. Retrieved November 18, 2021, from https://6river.com/what-is-a-pick-and-place-robot/

What is fdm 3d printing? - simply explained. (2021). *All3dp*. Retrieved January 8, 2022, from https://all3dp.com/2/fused-deposition-modeling-fdm-3d-printing-simply-explained/

Appendix A

Shell Script

A.1 Development Environment Setup

```
### Step 1 : Basic Setup Linux
cat > /etc/apt/sources.list << EOF
deb http://mirror.0x.sg/debian/ bullseye main non-free contrib
deb-src http://mirror.0x.sg/debian/ bullseye main non-free contrib
deb http://security.debian.org/debian-security bullseye-security main contrib non-free
deb-src http://security.debian.org/debian-security bullseye-security main contrib non-free
deb http://mirror.0x.sg/debian/ bullseye-updates main contrib non-free
deb-src http://mirror.0x.sg/debian/ bullseye-updates main contrib non-free
deb http://ftp.debian.org/debian bullseye-backports main contrib non-free
deb-src http://ftp.debian.org/debian bullseye-backports main contrib non-free
EOF
apt -y update && apt -y dist-upgrade
apt -y install wl

### Step 2.A : Setup Wi-Fi Connectivity
apt-get install linux-image-$(uname -r|sed 's,[^-]*-[^-]*-,,') linux-headers-$(uname -r|sed 's,[^-]*-[^-]*-,,') broadcom-sta-dkms
apt-get install -f
dpkg-reconfigure broadcom-sta-dkms
find /lib/modules/$(uname -r)/updates
modprobe -r b44 b43 b43legacy ssb brcmsmac bcma
modprobe wl
wpa_passphrase tuvbo-2.4GHz tuvbowin > /root/wifi.info
wpa_supplicant -i wlp3s0 -c /root/wifi.info
dhclient wlp3s0b1
cat > /etc/network/interfaces << EOF
source /etc/network/interfaces.d/*
auto lo
iface lo inet loopback
auto wlp3s0
iface wlp3s0 inet static
    wpa-ssid tuvbo-2.4GHz
```

```
32      wpa-psk ad47fcf178c32e83977e3c8e787d168e33d15143009832a2c9e4a2b233ee1084
33      address 192.168.0.69/24
34      gateway 192.168.0.1
35      dns-nameservers 1.1.1.1
36  EOF
37
38  ### Step 2.B : Setup Wi-Fi Connectivity
39  cat > /etc/network/interfaces << EOF
40  source /etc/network/interfaces.d/*
41  auto lo
42  iface lo inet loopback
43  auto enp1s0
44  iface enp1s0 inet static
45      address 192.168.0.69/24
46      gateway 192.168.0.1
47      dns-nameservers 1.1.1.1
48  EOF
49
50  ### Step 3 : Enable Remote Development with VSCode
51  apt -y install ssh git wget curl
52
53  ### Step 4 : Enable Windows-Developer Access it without password
54
55  ### Step 4.1 : Generate Developer Key in Windows Machine
56  # As NORMAL user in developer's Windows, open PowerShell
57  ssh-keygen.exe -q -t rsa -N '' -f ~/.ssh/id_rsa <<<y 2>&1 >/dev/null
58  cat ~/.ssh/id_rsa.pub
59
60  ### Step 4.2 : Register Developer Key in Linux Machine
61  nano ./.ssh/authorized_keys
62  chmod 600 ./.ssh/authorized_keys
63
64  ### Step 5 : Prepare Soruce Code in Linux Machine
65  ### As NORMAL user get the code
66  su - engineer
67  ssh-keygen
68  cat ./.ssh/id_rsa.pub
69  # copy the key into this link https://git.toastunix.com/user/settings
70  if [[ ! -d ~/papas/ ]]; then
71      echo "~/papas/ does not exists on your filesystem."
72      cd ~
73      git clone ssh://git@git.toastunix.com:2202/tigaky/papas.git papas
74  else
75      cd ~/papas
76      git pull
77  fi
78  git config --global user.email "zhiwen39@gmail.com"
79  git config --global user.name "soizhiwen"
```

A.2 Arduino CLI Setup

```
1  ################################################################################
2  ### Arduino CLI
3  ################################################################################
4
5  ### Step 1 : Install Arduino-CLI
6  # REFER : https://arduino.github.io/arduino-cli/installation/
7  apt -y install curl wget
8  curl -fsSL
   ↪  https://raw.githubusercontent.com/arduino/arduino-cli/master/install.sh |
   ↪  BINDIR=/usr/bin sh
9  arduino-cli version
10
11 ### Step 2 : Prepare Board Library
12 arduino-cli core install arduino:samd
13 arduino-cli core list
14
15 ################################################################################
16 ### Enable Serial Debug and Remote Sync
17 ################################################################################
18
19 ### Serial Debuging
20 # Install
21 apt -y install picocom
22 # Enter Serial Com
23 picocom -b 9600 -r -l /dev/ttyACM0
24 # Quit Serial Com
25 # To exit picocom, use CTRL-A followed by CTRL-X.
26
27 ################################################################################
28 ### Other Libraries
29 ################################################################################
30
31 ### Install Related Library
32 arduino-cli lib update-index
33 arduino-cli lib install DynamixelShield Servo
34
35 ################################################################################
36 ### Detect Hardware
37 ################################################################################
38
39 apt -y install hwinfo
40
41 ### Senario 1.A : Plug in Arduino MKR WiFI 1010 as NORMAL start up mode
42 hwinfo | grep Arduino
43 #   Model: "Arduino SA Arduino MKR WiFi 1010"
44 #   Vendor: usb 0x2341 "Arduino SA"
45 #   Device: usb 0x8054 "Arduino MKR WiFi 1010"
46
```

```
### Senario 1.B : Plug in Arduino MKR WiFI 1010 as BOOTOADER start up mode
hwinfo | grep Arduino
#   Model: "Arduino SA Arduino MKR WiFi 1010"
#   Vendor: usb 0x2341 "Arduino SA"
#   Device: usb 0x0054 "Arduino MKR WiFi 1010"

################################################################################
### Timestamp for PICOCOM
################################################################################

apt -y install screen moreutils
picocom -b 9600 -r -l /dev/ttyACM0 | ts
```

A.3 Python Setup

```
################################################################################
### PYTHON ### PYTHON ### PYTHON ### PYTHON ### PYTHON ### PYTHON ### PYTHON
################################################################################

### As ROOT : Install Programming Language - Python
su -
apt -y install build-essential zlib1g-dev libncurses5-dev libgdbm-dev \
    libnss3-dev libssl-dev libreadline-dev libffi-dev libsqlite3-dev wget
    ↪ libbz2-dev
cd ~
wget https://www.python.org/ftp/python/3.10.1/Python-3.10.1.tgz
tar -xf Python-3.10.1.tgz
cd Python-3.10.1
./configure --enable-optimizations
make -j 4
make altinstall
cd ~
rm -rf Python-3.10.1
rm -rf Python-3.10.1.tgz

### As ROOT : Check it.
# python3 --version
python3 --version
python3.10 --version
python3.10 -m pip install --upgrade pip
python3.10 -m pip install virtualenv

### As NORMAL user : Prepare a Virtual Environment for Python
which virtualenv
which python3.10
virtualenv -p /usr/local/bin/python3.10 venv
# Activate VIRTUALENV
source /usr/local/bin/venv/bin/activate
```

```
python3.10 -m pip install --upgrade pip
python3.10 -m pip install opencv-contrib-python-headless pyserial numpy scipy
python3.10 -c "import cv2; print(cv2.__version__)"
```

A.4 Virtual Camera Setup

```
### Step 1 : Setup Motion
apt -y install motion
mkdir -p /var/log/motion/
chmod 777 /var/log/motion/
mkdir -p /var/lib/motion/
chmod 777 /var/lib/motion/

sed -i 's/^width 640/width 1280/g' /etc/motion/motion.conf
sed -i 's/^height 480/height 720/g' /etc/motion/motion.conf
sed -i 's/^framerate 15/framerate 30/g' /etc/motion/motion.conf
sed -i 's/^stream_localhost on/stream_localhost off/g'
 ↪  /etc/motion/motion.conf
sed -i 's/^webcontrol_localhost on/webcontrol_localhost off/g'
 ↪  /etc/motion/motion.conf
sed -i 's/^movie_output on/movie_output off/g' /etc/motion/motion.conf
# sed -i 's/^daemon off/daemon on/g' /etc/motion/motion.conf

nano /etc/motion/motion.conf
### Add stream_quality and stream_maxrate
stream_quality 100
stream_maxrate 100
### Change /dev/video0 to /dev/video6
videodevice /dev/video6

systemctl start motion
systemctl restart motion
systemctl stop motion

### Step 2 : Setup v4l2loopback
apt -y install v4l2loopback-dkms v4l2loopback-utils \
    gstreamer1.0-tools gstreamer1.0-plugins-good libopencv-dev \
    build-essential

modprobe v4l2loopback \
        devices=1 exclusive_caps=1 video_nr=6 \
        card_label="OpenCV Camera"

v4l2-ctl --list-devices -d6
```

Appendix B

Costing

Table B.1 shows the project cost with a total of RM5331.50 for 30 items.

Table B.1: The costs used in this project.

Item	Quantity	Unit Cost (MYR)	Total (MYR)
Computer and Accessories			
Mini PC	1	731.00	731.00
Raspberry Pi 4 Model B	1	349.00	349.00
Logitech C270 HD Webcam	1	83.99	83.99
USB 3.0 Extension	1	18.50	18.50
Electronics			
DYNAMIXEL MX-12W	3	333.74	1001.22
Arduino MKR WiFi 1010	2	179.00	358.00
OpenCM 485 Expansion Board	1	209.94	209.94
MEAN WELL RD-50A Switching Power Supply	1	150.00	150.00
DYNAMIXEL Shield MKR	2	68.27	136.54
Vacuum Gripper	2	51.42	102.84
OpenCM9.04-C	1	100.78	100.78
Limit Switch	9	4.53	40.77
MG90S Micro Servo	2	14.00	28.00

Continued on next page

Table B.1: The costs used in this project. (Continued)

Item	**Quantity**	**Unit Cost (MYR)**	**Total (MYR)**
Jumper Wire	8	2.50	20.00
4 Channel 5V Relay Module	1	13.90	13.90
Mechanics			
Art Friend Block Board	1	121.30	121.30
Bearing Type-1	2	39.58	79.16
Bolts & Nuts Type-1	36	1.09	39.24
Bearing Type-2	6	5.15	30.90
Bearing Type-3	6	2.97	17.82
Bolts & Nuts Type-2	10	1.50	15.00
Washer Type-1	2	4.80	9.60
Washer Type-2	100	0.09	9.00
Washer Type-3	10	0.05	0.50
Tools			
Creality Ender-3 V2 FDM 3D Printer With CR Touch	1	1267.00	1267.00
3D Printer 1.75mm PLA Filament	2	52.00	104.00
Soldering Iron	1	74.00	74.00
Creality Carborundum Glass Platform	1	54.00	54.00
M8 Drill Bit	1	15.30	15.30
Others			
Import Duty	1	150.20	150.20
Total Cost (MYR)			**5331.50**

Figure B.1 illustrates the percentage cost of each category.

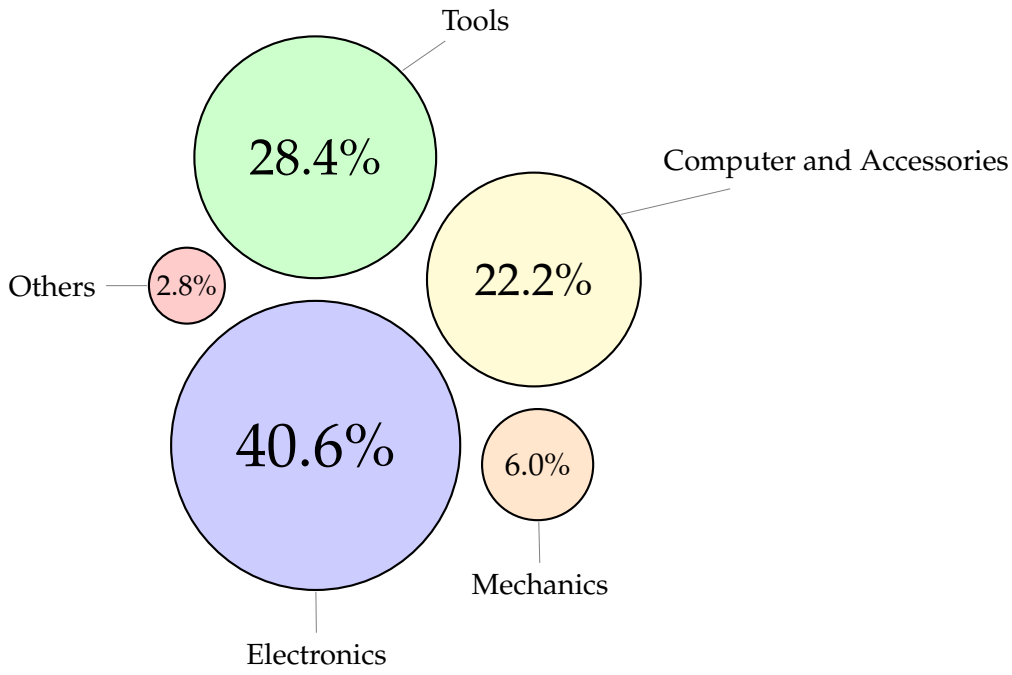


Figure B.1: Percentage cost of each category.