

DE-42 (MTS)

M.ABDULLAH,

ABDUL HANNAN,

M.AZEEM,

M.SHOAIB

PLC control for synchronous pick and place operation of robotic arm with conveyer belt



**COLLEGE OF
ELECTRICAL AND MECHANICAL ENGINEERING NATIONAL
UNIVERSITY OF SCIENCES AND TECHNOLOGY RAWALPINDI
2024**

COLLEGE OF ELECTRICAL AND MECHANICAL ENGINEERING



DE-42 MTS PROJECT REPORT

PLC control for synchronous pick and place operation of robotic arm with conveyer belt

Submitted to the Department of Mechatronics Engineering
in partial fulfillment of the requirements
for the degree of
Bachelor of Engineering
in
Mechatronics
2024

Sponsoring DS:	Submitted By:
<ol style="list-style-type: none">Lec. Usman Asad Lec, Mechatronics Department.Dr Ameer Hamza Associate Professor, Mechatronics Department	<ol style="list-style-type: none">Muhammad AbdullahMuhammad ShoaibMuhammad Azeem NiaziAbdul Hanan

ACKNOWLEDGMENTS

First and foremost, we are thankful to Allah the Almighty, the Most Gracious, and the Most Merciful for His blessing in the form of opportunity, knowledge, and guidance to reach our current progress stage. May Allah's blessing goes to His final Prophet Muhammad (peace be up on him), his family and his companions. We would like to thank our parents for their hard work, prayers, motivation and utmost support we would have not achieved any of these without the guidance wisdom and support of our parents as well as our supervisors Lec.Usman Asad and Dr Ameer Hamza.

Our utter gratitude is with the esteemed faculty of Department of Mechatronics Engineering, College of E&ME, NUST. Our supervisors supported and motivated us through thick and thin and gave expert guidance in all matters. Sir Usman and Sir Ameer Hamza provided us with the much-needed technical guidance. We would also like to thank our dear fellow Abrar for being a constant support throughout this degree.

ABSTRACT

The title of this Final Year Project (FYP) is "Plc control for synchronous pick and place operation of robotic arm with conveyer belt" in order to develop a complete mechatronic system with the highest feasible effectiveness and preciseness for the management of parcels. It integrates robotic and conveyor systems to sort parcels and handle them automatically, with minimal reliance on manual work. The foremost objective is the development of a well-documented, functional system that will be an effective learning platform for the learners. The resultant system enables hands-on experience in major components of mechatronic automation, including C.V. (Computer Vision), robotic manipulators, PLCs (Programmable Logic Controllers), and industrial instrumentation. Robotic conveying systems achieve massive improvements in efficiency, scalability, and safety, so they can be viewed as a solution to contemporary challenges in parcel management. Furthermore, the project emphasizes the integration of real-time tracking and monitoring, cost-effectiveness, and sustainability, aligning with contemporary industry standards and practices.

TABLE OF CONTENTS

1.	Chapter 1 - INTRODUCTION	1
1.1	Project Background:.....	1
1.2	Problem statement.....	1
1.3	Research Objectives.....	2
1.4	Project Scope	2
1.5	Significance of Project.....	3
2.	Chapter 2 – Background and LITERATURE REVIEW	4
2.1	Robotics	4
2.1.1	Options of Robotics for this project.....	4
2.1.2	Structure of robotic arm	7
2.1.3	Gripper design.....	8
1.	Conceptual Design	9
2.	Material Selection	9
	PLA (Polylactic Acid):	9
3.	CAD Modeling.....	9
4.	3D Printing.....	9
5.	Load Testing	10
2.2	PLC	10
2.2.1	Background	10
2.2.2	Operation principles.....	11
2.2.3	Modules.....	11
2.2.4	Accessories	13
2.2.5	Types of Plc and their manufacturers	13
2.2.6	IEC 61131-3 language	15
2.2.7	Instruction List (IL).....	16
2.3	Conveyer belt and its use in the automation Industry	16
2.3.1	Types of Conveyor belts used in Industry:	17
2.4	Sequential Flow Chart (SFC).....	18
2.5	COMPUTER VISION	19
2.5.1	Shape Detection	19
2.5.2	Measurement of size	20
2.5.3	Position coordinate.....	20
2.5.4	Calculation of distance.....	21

2.5.5	Implementation detail	21
2.5.6	Flow Chart	22
2.5.7	Segmentation.....	22
2.5.8	Object Analysis.....	23
2.6	Pick and Place Simulation	24
2.6.1	Robotics Simulation Tools.....	24
2.6.2	Unity 3D vs Robotics Simulation Tools.....	25
2.6.3	Advantage of using Unity3D	26
2.6.4	Trade-off using Unity3D.....	26
2.6.5	Robot Operating System (ROS).....	27
2.6.6	ROS for Unity3D Simulation	27
2.6.7	Application of Unity and ROS Simulations.....	28
2.6.8	Pros and Cons of Unity-ROS Simulation	29
Chapter 3: METHODOLOGY.....		31
2.7	Kinematics	31
2.7.1	Forward Kinematics.....	31
2.7.2	Inverse Kinematics.....	31
2.7.3	Inverse Kinematics Analysis.....	35
2.8	Function Block Diagram.....	40
2.9	How to configure Plc settings with the TIA portal.....	41
2.10	YOLO	46
2.10.1	Important elements of yolo.....	46
2.10.2	Creating a URDF File in Solid works	49
2.10.3	Ros melodic.....	51
2.10.4	Building the Software Foundation: ROS Packages and MoveIt for SJ602A Arm	
Control	53	
3.	Chapter 4: Experimental results and analysis	56
3.1.1	Precision.....	56
3.1.2	Recall confidence.....	57
3.1.3	F1 Score	57
4.	Chapter 5: Experimental results and analysis	65
ANNEXURE A: PROGRAMMING		67
References.....		Error! Bookmark not defined.
5.	References.....	71

LIST OF FIGURES

Figure 2—2-1 Sj602-a	7
Figure 2—2-2 Gripper Design	8
Figure 2—2-3 Plc.....	10
Figure 2—2-4 shows the components of the PLC system.....	11
Figure 2—2-5 shows the I /O modules of PLC	12
Figure 2—2-6 shows the basic symbols of Ladder logic diagram.....	15
Figure 2—2-7 Conveyer Belt.....	16
Figure 2—2-8 shows the Sequential Flow chart.....	19
Figure 2—2-9.....	19
Figure 2—2-10 Simulation table	Error! Bookmark not defined.
Figure 2—2-11 Unity MoveIt.....	Error! Bookmark not defined.
Figure 3—3-1 Link lengths.....	33
Figure 3—3-2 General overview of sj-602a anno-bots	33
Figure 3—3-3 Ladder logic circuit	40
Figure 3—3-4 shows the FBD equivalent model of Ladder Logic	41
Figure 3—3-5 PLC ladder logic	41
Figure 3—3-6 Tia portal	42
Figure 3—3-7 Creating project prompt	42
Figure 3—3-8 Establishing communication	43
Figure 3—3-9 Add new devices	43
Figure 3-10 shows the selection of model	44
Figure 3-11 Details on Plc	44
Figure 3-12 Detecting Ip Address.....	45
Figure 3 —3-13 Error Message.....	45
Figure 3—3-14 Yolo V8 segmentation.....	46
Figure 3—3-15 SJ602-A model setup	50
Figure 3—16 Docker setup.....	51
Figure 3-17 Building a program	53
Figure 3-19 MoveIt	55
Figure 4-1 Precision Curve	56
Figure 4-2 Recall-confidence curve.....	57
Figure 4-3 F1-confidence curve.....	57
Figure 4-4 Example Detection.....	58
Figure 4-5 Example 2 Detection.....	59
Figure 4-6 Confusion Matrix	59
Figure 4-9 Final Labeling	60

Figure 4-10 Final Labeling	60
Figure 4-11 Gripper meshing.....	63
Figure 4-12 Gripper meshing.....	63

LIST OF TABLES

Table 2—1 Product Parameters	Error! Bookmark not defined.
Table 2—2 Logical Operations.....	18
Table 3—1 DH Parameters of the Sj-602a Robotic Arm	33

LIST OF SYMBOLS

Latin Letters

m mass of complete robot with payload

I moment of inertia

w angular velocity

r turning radius

u coefficient of friction

g acceleration due to gravity

v forward velocity

Greek Letters

α angular acceleration

Acronyms

UGV unmanned ground vehicle

Chapter 1 - INTRODUCTION

1.1 Project Background:

Automation is one of the key components of development in the industrial world. It contributes to increased production and efficiency while lowering the requirement for personnel. Automation is a vast field that mostly involves industrial manufacturing. In addition, automation is used to create many high-tech everyday items like refrigerators, cars, and medical equipment (such as radiography and x-ray machines).

Robots have been a part of the industrial field for almost 85 years. One analogy for a robotic arm is a human hand. For arm movement, it has a translational joint (displacement) and a free rotating joint (rotation). Usually, a pneumatic and hydraulic system (pistons) or an electric driver (motor) power this arm movement. A microcontroller (CPU), which is typically programmable and designed to carry out a series of sequential tasks, controls these actuators. The majority of these robotic arms are intended for use in industrial settings where they can operate quickly and dependably, supporting mass manufacturing. The following goals are part of the project:

- Designing a layout for the installation of PLC, a robot and other components together.
- Implementing computer vision for object detection
- Designing the gripper
- Programming
- Testing and Finalizing
- Documenting

1.2 Problem statement

The capacity to efficiently and precisely sort packages on a conveyor belt is crucial in modern logistics to satisfy the demands of rapidly moving supply chains. Manual sorting takes a lot of time and is prone to mistakes, which results in inefficiencies and higher operating expenses. Creating a system that can recognize, categorize, and arrange packets of different sizes and forms in real-time is the main problem. By utilizing robotic automation and computer vision techniques, this research seeks to address this difficulty.

1.3 Research Objectives

This project's primary goal is to create an automated package sorting system that makes use of computer vision and a robotic arm. The particular objectives are to:

- To accurately detect objects and segment parcels on a conveyor belt, train a YOLOv8 model.
- To detect and track packages, integrate a real-time video feed with the trained model.
- Sort packages with a robotic arm according to predetermined standards like size, weight, or shapes.
- Analyze the system's accuracy, speed, and dependability of sorting under varied conditions

1.4 Project Scope

The following list of essential elements and deliverables is part of the project's scope:

1. Validation and Training of Models:

- Gathering and annotating a dataset of pictures showing packages moving down a conveyor line.
- YOLOv8 model training for precise parcel detection and segmentation.
- Gathering and annotating a dataset of pictures showing packages moving down a conveyor line

2. Integration of Systems:

- creation of a system for processing videos in real time while capturing frames from the conveyor belt.
- YOLOv8 model integration for segmentation and object detection on collected frames.

1.5 Significance of Project

Building an automated parcel sorting system with a robotic arm as well as computer vision can totally revolutionize logistics. This type of technology quickens the sorting process leading to increased throughput in distribution centers and warehouses. It also improves reliability and accuracy hence reducing sorting errors that may be costly. Moreover, the need for manual labor is minimized thus creating safer working environments while allowing human workers to handle more complicated tasks. Moreover, it is adjustable for different parcel sizes and can be easily scaled to fit changing operational requirements. In conclusion, the main goal of this project is to advance automation in the field of logistics so as to make supply chain operations more efficient and dependable.

Chapter 2 – Background and LITERATURE REVIEW

1.6 Robotics

The name robotics was later coined by renowned Russian science fiction author Isaac Asimov, while the word robot itself originated from the Czech word robot, which refers to a forced laborer. From then on, numerous advancements in the field of robotics have been made successfully, resulting in the creation of humanoids, micro robots, tele-operator manipulators, and other devices.

The industry is trending toward robotization as opposed to the current level of automation. As a result, robot technology is developing quickly. These days, robotic manipulators and robotic arms are the most often utilized robots in the industry. A manipulator is the arm of a robot. It consists of a group of joints spaced apart by arm connections. The arm's joints are where movement takes place. A robot arm's essential components are a base, joints, links, and a racket. The foundation of the arm is called the base.

1.6.1 Options of Robotics for this project

1.6.1.1 XR4 Rhino Robotics Inc

The XR-4 is a rugged, five-axis, semi-open robot arm with the ability to move all links linearly together and can be controlled independently as well. It is an old robot which is made of encoded dc motors and chain and sprocket mechanism

1.6.1.2 UR-5:

The Universal Robots UR5 is a medium-sized, 6-degree-of-freedom (DOF) industrial robot arm that can carry up to 5 kg and has an 850 mm work area radius. It's designed to automate repetitive, risky tasks safely, such as assembly, packing, testing, etc.

1.6.1.3 CR-5:

The Dobot CR5 is a 6-axis, collaborative robotic arm designed for industrial and educational use. It has a 5 kg payload and a reach of 1096 mm, and can perform tasks such as inspection, assembly, screw driving, bin picking, and loading and unloading. The

CR5 has a repeatability of ± 0.02 mm and five levels of adjustable collision detection settings. It also has an intelligent interaction panel that allows users to teach the robot functionality by dragging. A great collaborative robot but our application does not require such High-End robot.

1.6.1.4 BCN-3D:

BCN3D is a company specializing in the design and manufacture of high-quality 3D printers. Known for their innovative IDEX (Independent Dual Extruder) technology, BCN3D printers are used in various industries for prototyping, manufacturing, and educational purposes [6]. The BCN3D Moveo is an open-source robotic arm designed for educational purposes. It's an interesting project, though not a core product line for BCN3D. The BCN3D Moveo is an educational robotic arm project developed by BCN3D Technologies. It's designed to be a low-cost, open-source platform for students, hobbyists, and educators to learn about robotics, mechatronics, and programming. The arm consists of 3D-printed parts, off-the-shelf components, and Arduino-based control electronics, making it accessible and customizable for educational purposes.

Aspect	UR-5	CR-5	BCN-3D Moveo	XR-4 Rhino
Flexibility and Versatility	Highly adaptable, suitable for various tasks (assembly, pick and place, packaging).	N/A	N/A	Versatile and adaptable for various educational applications.
Ease of Programming	Intuitive interface and teach pendant, no extensive coding knowledge required.	Intuitive programming interface, user-friendly software.	N/A	Requires higher technical knowledge in programming and robotics.
Compact and Lightweight	Easy integration into existing production lines.	Compact and lightweight, easy to integrate into production environments.	N/A	N/A
Safety Features	Advanced safety features, force sensing, and collision detection for safe human-robot interaction.	Designed for safe human-robot interaction with advanced safety mechanisms.	N/A	Equipped with safety features including emergency stop mechanisms.

Aspect	UR-5	CR-5	BCN-3D Moveo	XR-4 Rhino
Cost-Effective	Good balance between performance and cost, suitable for small to medium-sized enterprises.	Cost-effective, accessible for small to medium-sized enterprises.	Affordable due to 3D-printed parts and off-the-shelf components, accessible for education.	Cannot be bought anymore however the parts are available but at high cost.
High Precision	N/A	Excellent precision and repeatability, ideal for high accuracy tasks.	N/A	N/A
Educational Tool	N/A	N/A	Provides hands-on learning for students and enthusiasts.	Suitable for a wide variety of educational applications.
Payload Limitations	Payload capacity up to 5 kg, not suitable for heavier objects.	Payload capacity of 5 kg, not suitable for handling heavier items.	Limited payload capacity and reach, restricting applicability for certain tasks.	N/A
Maintenance Requirements	Regular maintenance necessary, potentially causing downtime and additional operational costs.	Periodic maintenance required to keep operating at peak performance.	3D printed parts require Careful use and can cause problems if not properly lubricated or the bearings misfit	Maintenance may be intensive and detailed, leading to high downtime and operational costs.
Assembly Required	N/A	N/A	Building from scratch requires time, effort, and technical skills, challenging for beginners.	The bot here at campus was in pretty bad Shape so it required to be refurbished.
Accuracy and Precision	N/A	N/A	May not offer the same level of accuracy and precision as commercial robotic arms.	N/A

1.6.1.5 SJ-602a:

It is a newer gen bot made in 2021 and manufactured using engineering techniques like CNC all metal, durable designs, AI, golden ratio body size, backlash reduction on all axis,

etc. The bot is sponsored by Department of Mechatronics Engineering along with its controller. Before this project the bot itself was not used in any other applications before due to the complexity in it's Chinese operation manual. We have 2 such bots with their individual controllers.



Figure 2—0-1 Sj602-a

1.6.2 Structure of robotic arm

Our manipulator is made up of an aluminum structure. Some parts are made of fiber at joint position due to smooth movement of the robotic arm plastic fiber provides low friction due to hence it takes the motor low power to move as there is low friction.

Light weight, motor need less power to lift the robotic arm. Aluminum is inherently resistant of the most normal atmospheric environment. Cost wise it is not justified at 3.8 million rupees as of May,2024. However as aluminum is cheaper, durable, hard to bend, hence the bot Exceeds Expectations at working once you get it running.

Degree of freedom: 6 axes; six degrees of freedom

Load: 3kg

Running radius: 700mm

Body material: 6061 aluminum alloy

Body processing technology: CNC machining, sandblasting, anodizing

Power unit: integrated servo motor with multi-turn absolute encoder (4 pcs)

Integrated servo motor with brake including multi-turn absolute encoder

Deceleration device: harmonic deceleration (1-6 axis)

Zero return device: multi-turn absolute encoder

Actuators are the muscles of robots which convert stored energy into the movements. In our case actuators will behave like devices that will transform the input signals to motion. These motors will be used for the control of circular and linear motion of the Robotic Arm. We will send pulses as an input to the servo board and those pulses will be transferred to the servo motors and in the result, we will get the motion of the robot.

1.6.3 Gripper design

One of the objectives was to design and fabricate a custom gripper. We decided to use a rack and pinion mechanism. The gripper is designed to lift loads of up to 5 kg, utilizing detailed Analysis for 3D printing technology with PLA material. The gripper's design emphasizes lightweight construction, targeting a total weight of only 180 grams. This report details the design process, material selection,

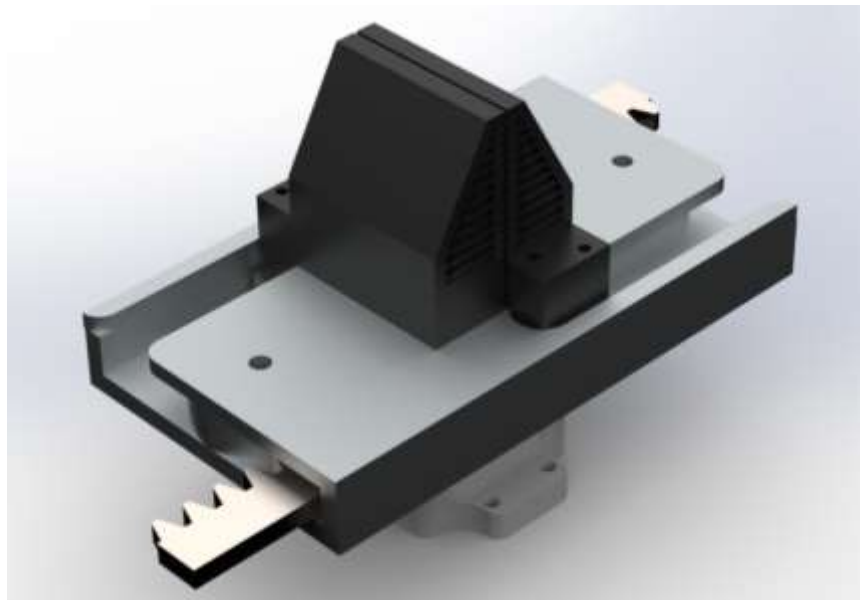


Figure 2—0 Gripper Design

1.6.3.1 Design Objectives

- **Load Capacity:** The gripper must be capable of lifting up to 5 kg.
- **Material:** Use PLA for 3D printing, due to its ease of printing and sufficient strength for the application.
- **Weight:** The total weight of the gripper should not exceed 200 grams. Ensuring reduced number of parts.
- **Mechanism:** Implement a simplified system to ensure precise and reliable

gripping action

1.6.3.2 Design Process

1. Conceptual Design

The conceptual design focused on leveraging the rack and pinion mechanism to provide a smooth and controlled gripping motion. The design includes:

- **Rack:** A linear gear that moves horizontally when driven by the pinion.
- **Pinion:** A round gear that meshes with the rack, converting rotational motion into linear motion.
- **Gripper Jaws:** Two jaws that open and close symmetrically as the rack moves.

2. Material Selection

PLA (Polylactic Acid):

Advantages:

- Easy to print.
- Good strength-to-weight ratio.
- Biodegradable and environmentally friendly.

Disadvantages:

- Lower impact resistance compared to other materials like ABS.
- Lower thermal resistance.

3. CAD Modeling

Using CAD software (such as SolidWorks or Fusion 360), the gripper was modeled with the following features:

- **Rack and Pinion:** Designed to ensure smooth engagement and minimal backlash.
- **Lightweight Jaws:** Optimized for weight without compromising strength.
- **Mounting Interface:** Designed to attach easily to a robotic arm or another actuator.

4. 3D Printing

The 3D-printed components were assembled as follows:

- **Pinion Gear:** Mounted on a motor shaft.

- **Rack:** Positioned to mesh with the pinion.
- **Gripper Jaws:** Attached to the ends of the rack.
- **Gearbox:** Attached to the motor and the rest of assembly is attached to gearbox itself.
- **Base:** This is a custom part made in accordance with the Flange of the robot.
- **Top Plate:** This attaches the gearbox, base, and itself, essentially completing the assembly of the whole Gripper.

5. Load Testing

The gripper was tested with various weights up to 5 kg to verify its load-carrying capacity. Key evaluation points included:

- **Grip Strength:** Ensuring the gripper could securely hold the load without slipping.
- **Actuation Smoothness:** Ensuring smooth opening and closing of the jaws under load.

1.7 PLC

1.7.1 Background

PLCs, or programmable logic controllers, are industrial digital computers that were created in the US automobile sector in the late 1960s and early 1970s. Because of their improved durability and dependability, especially in demanding real-time applications, these PLCs marked significant progress by replacing hard-wired relays and timers. Motion control and

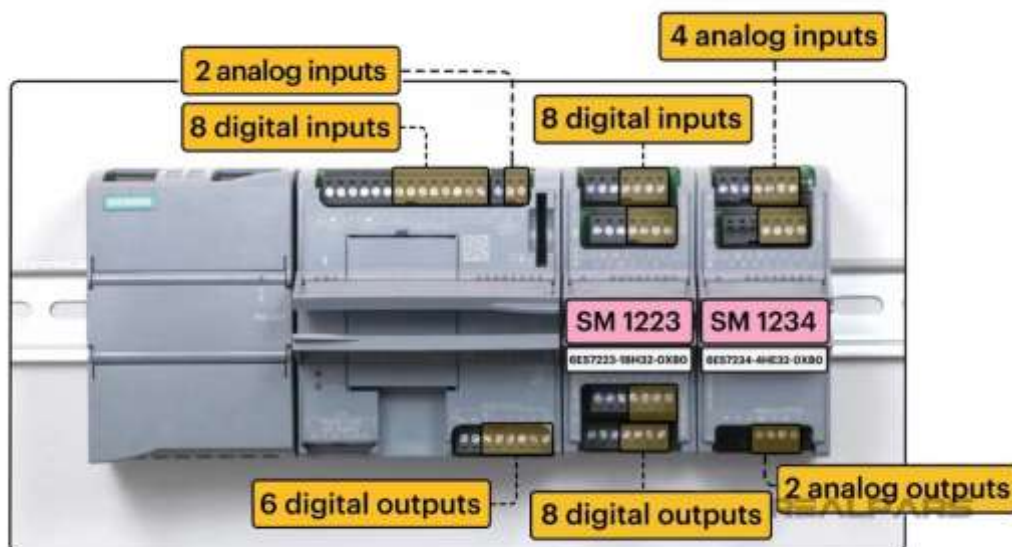


Figure 2—0 Plc

networking are just two of the control activities that PLCs can be used for thanks to the availability of visual programming approaches. PLCs' modular architecture enables customization, scalability, and flexibility to meet the needs of various control applications. These can be as small as a few connected devices, a few hundred inputs/outputs (I/Os), and a processor, or as large as thousands of I/Os in large control systems with multiple PLCs interconnected to monitor an entire production site with robots and other automated machinery. With the components at hand, a customized PLC program can be created to accomplish the required control task, depending on the intended use. PLC systems are shown in general detail in [8]

1.7.2 Operation principles

One way to demonstrate the fundamental operational method of a PLC is by considering a straightforward configuration that includes a CPU (Central Processing Unit) and an I/O (Input/Output) interface. The CPU is equipped with a processor for processing signals, memory for data storage, and a power supply for supplying the required voltage for functioning. During each cycle, the CPU initially retrieves data from the I/Os, then performs actions based on the I/O signals, and lastly updates the values by writing them back through the output interface.

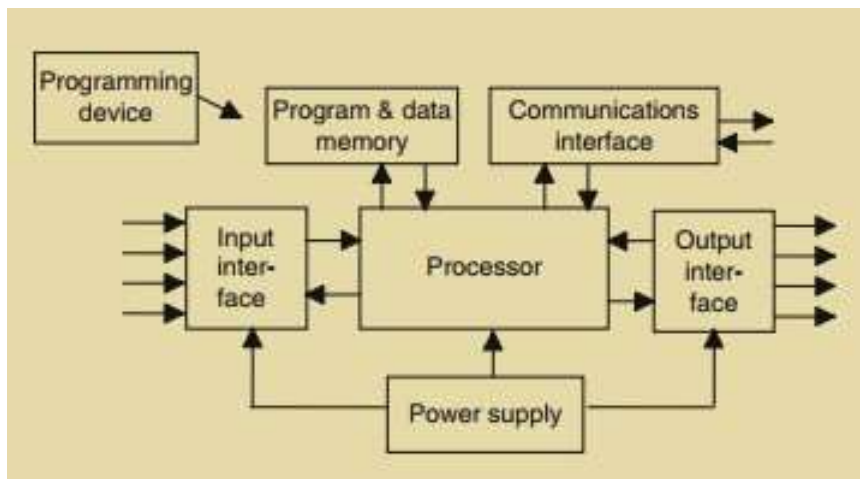


Figure 2—2 shows the components of the PLC system

1.7.3 Modules

CPU, I/O, connectivity modules, and condition monitoring modules are the components that are offered. Additionally, matching extras like memory cards or connections are offered for

the technical configuration.

1.7.3.1 CPU

A CPU module performs the same task as a regular CPU, such as one found in a personal computer. It is primarily in charge of signal and data processing. It typically contains several networking interfaces (Ethernet), serial connectors (COM1, COM2), a Fieldbus Plug, and memory in the form of a Compact Flash card. The PC 2100 CPU module from B&R Automation is taken into consideration in this thesis.

1.7.3.2 I/O Modules



Figure 2—3 shows the I/O modules of PLC

Modules for input/output provide both analog and digital inputs and outputs. Sensors, actuators, and other field devices can be connected to the I/O terminal unit. The kind and quantity of channels, signal voltage, and field-bus connections vary between the I/O modules. If more distant devices are required, a field bus can be used to create a peripheral connection or a local connection via the CPU's I/O bus. The IR sensor and the START and STOP buttons are the inputs used in our works. 3-5 shows an example of a standard PLC with a set of IOs.[9]

1.7.3.3 Communication modules and interface modules

There are many different standards, interfaces, and protocols available for

communication modules. Below is a list of the available types.

1.Power link, 2.Ethernet, 3.ETHERCAT, 4.Profilet, 5.Profibus, 6.Modbus, 7.TCP/IP

In essence, communication interface modules are bus modules that enable connections between the other modules that were discussed in the preceding paragraphs. They are intended for communication with a plant's dispersed peripheral

1.7.3.4 Condition monitoring modules

When the plant is in operation, signals and processes are observed using condition monitoring modules. It helps to avoid risks and malfunctions by enabling predictive maintenance and guaranteeing protection against voltage, current, and vibration. The typical components of condition monitoring modules are a speed encoder, a memory, and analog inputs. At predetermined intervals, this kind of device automatically sends WAV files to the CPU that include pertinent process data. In this manner, the plant's condition can be tracked in real time, and a whole history of previous operations is also accessible for later review and analysis.

1.7.4 Accessories

While some are regarded as add-ons, others are necessary in order to assemble a PLC using the supplied modules. Compact Flash cards for memory storage, power supply plugs with varied pole configurations, communication devices, I/O modules, and programming cables are just a few of the accessories that B&R offers. Four 3V / 950mAh lithium batteries are also included in the package. By adding more batteries to the PLC's memory, you can make sure that settings and process data are kept safe even when the machine is turned off.

1.7.5 Types of Plc and their manufacturers

1. Compact PLCs:

- Siemens: Models: S7-1200
- Rockwell Automation (Allen-Bradley): Models: MicroLogix
- Mitsubishi Electric: Models: FX Series
- Schneider Electric: Models: Modicon M221
- Omron: Models: CP1 Series

2. Modular PLCs:

- Siemens: Models: S7-300, S7-1500
- Rockwell Automation (Allen-Bradley): Models: CompactLogix, ControlLogix
- Mitsubishi Electric: Models: Q Series
- Schneider Electric: Models: Modicon M241, M251

3. Rack-Mounted PLCs:

- Siemens: Models: S7-400
- Rockwell Automation (Allen-Bradley): Models: ControlLogix
- Mitsubishi Electric: Models: iQ-R Series
- ABB: Models: AC500

4. Safety PLCs:

- Siemens: Models: S7-1200F, S7-1500F
- Rockwell Automation (Allen-Bradley): Models: GuardLogix
- Schneider Electric: Models: Modicon M580 Safety
- Omron: Models: NX Safety Controller

5. Nano/Micro PLCs:

- Siemens Models: Logo Series
- Rockwell Automation (Allen-Bradley): Models: Micro800 Series
- Mitsubishi Electric: Models: Alpha Series
- Omron: Models: ZEN Series

6. Distributed PLCs:

- Siemens: Models: ET 200SP
- Rockwell Automation (Allen-Bradley): Models: CompactLogix with Distributed I/O
- Schneider Electric: Models: Modicon M580

- ABB: Models: AC500 with Remote I/O [10]

1.7.6 IEC 61131-3 language

The standards for PLC programming languages were established by IEC 61131 and include function block diagrams (FBD), structured text (ST), sequential function charts (SFC), ladder diagrams (LAD), and instruction lists (IL). The following section will cover the many styles of the programming language stated above. [11]

1.7.6.1 Ladder language

The most fundamental type of PLC programming technique is ladder diagrams (LAD). All we have to do to develop a program to sketch out a switching circuit. Two vertical lines in the ladder diagram stand in for the power rails, and circuits are connected between these two verticals as horizontal lines or rungs. Maintaining adherence to specific conventions is crucial while programming with ladder diagrams:

1. The power rails that connect circuits are shown in the diagram as vertical lines. It is believed that power moves over a rung from the left-hand vertical.
2. One operation in the control process is represented by each rung on the ladder.
3. Every rung must begin with one or more inputs and end with at least one output. Inputs refer to control actions, such as closing the contacts of a switch, while outputs are

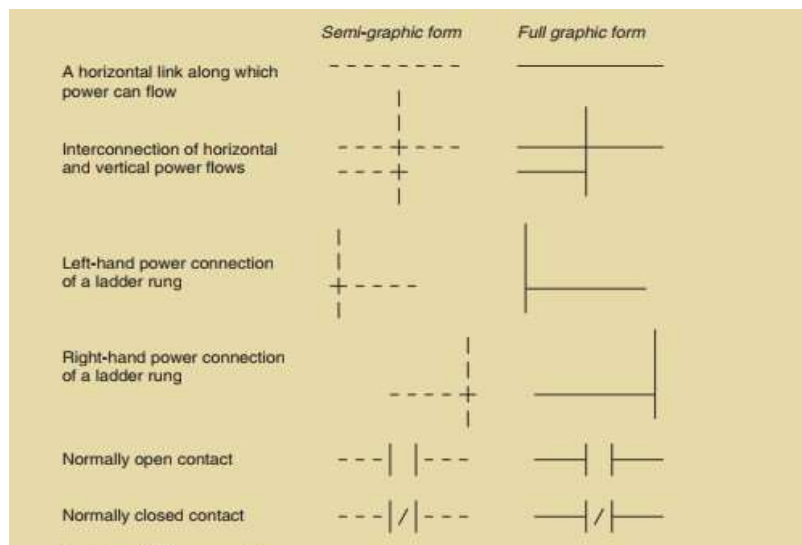


Figure 2—4 shows the basic symbols of Ladder logic diagram

devices connected to the PLC's output, such as motors or other devices. The IEC 61131-3 standard utilizes specific symbols for input and output [12]

1.7.7 Instruction List (IL)

Ladder diagrams and instructions list programming are comparable programming techniques. The method it's done, though, is different since instead of using a ladder, this one employs a text format where each level matches a rung in the instructions list.

It suggests that the memory has operand A loaded. It is comparable to using an open contact ring procedure for input A, passing the output to B. Table 3.1 lists the several mnemonic standard codes that IEC 61131-3 uses.

1.8 Conveyor belt and its use in the automation Industry

A conveyor belt is a mechanical apparatus consisting of a continuous moving belt, designed to transport materials from one location to another. It is driven by pulleys at either end and is supported by a series of rollers. Conveyor belts are a staple in various industries, including manufacturing, logistics, mining, and food processing, due to their efficiency in material handling. The concept of the conveyor belt dates back to the late 19th century and has since evolved significantly. Modern conveyor belts are designed to handle a wide range of materials, including bulk goods, packaged items, and delicate products. They are integral to automated systems, providing a reliable and consistent means of transporting materials with minimal human intervention. Fig 3-7 shows an example of a modern conveyer belt



Figure 2—5 Conveyer Belt

1.8.1 Types of Conveyor belts used in Industry:

1. Solid General-use Belts:

Strong, all-purpose belts: These strong, all-purpose belts are frequently utilized in several industries. They are composed of rubber or a fabric, like nitrile, nylon, polyester, or neoprene. The belt's qualities can be used to determine its major applications. Rubber belts are frequently used in the mining and milling sectors to handle bulk materials like aggregates and raw ore. While belts composed of rubber, polyester, or neoprene are utilized to handle baggage at airports, PVC conveyor belts are employed in grocery shops.

2. Woven Metal Belts:

The food, electronics, and glass production industries are among the businesses that primarily employ these belts for cooling, drying, and heating processes. To let air to move more easily while the product is being carried, these belts are constructed from interconnected chains or wiring.

3. Hinged Belts:

These hinged belts are mostly used by industries that handle scrap and other tiny materials for recycling. Metal is used to make these belts. These belts' interlocking hinges allow them to revolve around the pulley while providing a level, sturdy surface. These belts are incredibly durable and suitable for frequent use.

4. Hinged Belts:

These are mostly utilized in the automobile and food packaging industries. These belts are appropriate for settings with frequent belt changes and cleaning. These plastic belts are an ideal replacement for belts made of fabric or metal. [1]

Table 2—1 Logical Operations

IEC 61131-3	Operation
LD	Load the operand in result register
LDN	Load negative operand into result register
AND	Boolean AND
ANDN	Boolean AND with negative operand
OR	Boolean OR
ORN	Boolean OR with negative operand
ST	Stores the result register into operand

1.9 Sequential Flow Chart (SFC)

- a. When a flow chart is desired to represent a series of states or functions, the SFC technique of program execution is employed. The SFC's features are given below.
- b. Rectangular boxes, each of which represents a distinct state of the system under control, are used to depict the processes as discrete, sequentially connected states or steps.
- c. The transition condition, which must be satisfied before the system can go from one state to the next, is represented by a horizontal bar on each connecting line between states. There can never be a straight connection between two steps; there must always be a transition in between. There is never a straight path between two transitions; there must always be a step between them. [7]
- d. A software executes the next step in the SFC process when the transfer requirements are satisfied. As a result, the procedure keeps going from one stage to the next until the machine cycle is finished.
- e. When a state has been reached, outputs or actions are indicated by boxes that are connected horizontally.
- f. Figure 3.7 depicts a typical SFC example, where the start is engaged to call a subroutine program (Batch Mix and Batch Pump) in addition to the main program. [7]

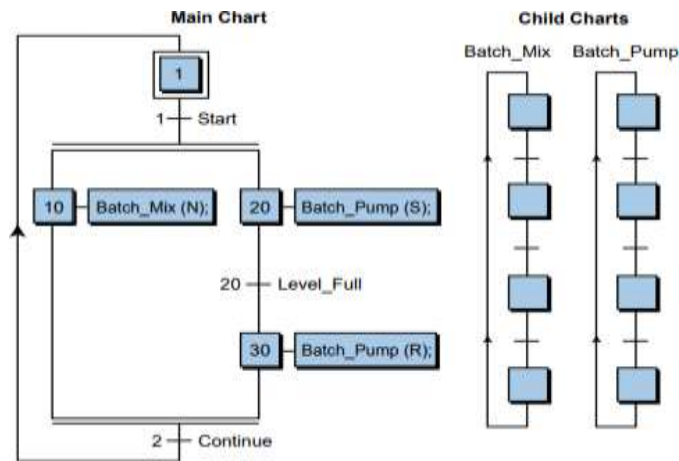


Figure 2—6 shows the Sequential Flow chart

1.10 COMPUTER VISION

The aim of this experiment is to implement advanced computer vision techniques in order to detect and classify objects that are positioned on a moving conveyor belt. We can analyze the geometric features and spatial orientations of different items in more detail that are out there with the help of this vision system. Capabilities our approach supports:

1.10.1 Shape Detection

Leveraging the YOLOv8 segmentation model, our system is able to form a priori knowledge of shapes for objects on the conveyor belt. Say a camera is set that captures the frames of moving tape and these frames are passed to YOLOv8 model, The previous segmentation algorithm than isolates each object and define its shape based on a predefined geometric pattern. These frames have edges and contours, within the framing when analyzed the system can categorize between triangles, circles or squares (different shapes). “This automated shape recognition is essential for all subsequent handling and sorting steps. The shape detection is shown in Figure 2-9 and 2-10

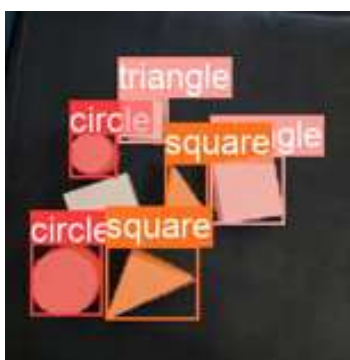


Figure 2—7



Figure 2—10

1.10.2 Measurement of size

The purpose of this step is to calculate the size area form by every detected object in an image. After the segmentation of an item by this model, we can compute its area counting non-zero pixels belonging to a mask of this class. Next, a comparison with a predefined threshold will determine whether an object is “big” or “small”. big, if it covers more pixels than the threshold and small, otherwise. With this classification, the appropriate handling mechanisms for different sized object on conveyor belt can be identified

$$\text{Category} = \begin{cases} \text{"Large"} & \text{if area} > \text{area_threshold} \\ \text{"Small"} & \text{if area} \leq \text{area_threshold} \end{cases}$$

1.10.3 Position coordinate

The position of each object on the conveyor belt can be found by calculating the centroid relative to a box that has been fit around it) Furthermore, the coordinates of the center (x, y) are calculated based on averaging x and y values among all vertices the box. Then, the exact location of that object can be found by matching these centroid coordinates with a reference position in the conveyor belt. This spatial data enables the system to monitor movement of the object and manage coordinates in situations that require precise placement e.g., robotic picking.



Figure 2—11



Figure 2—12

1.10.4 Calculation of distance

In both the x and y directions, the centroid's distance from a reference point on the conveyor belt is determined.

For the x-axis

$$\text{distance}_x = \text{centroid}_x - \text{conveyor_belt}_x - \left(\frac{\text{conveyor_belt_width}}{2} \right)$$

For the y-axis:

$$\text{distance}_y = \text{conveyor_belt}_y + \text{conveyor_belt_height} - \text{centroid}_y$$

1.10.5 Implementation detail

We utilize a 1080p webcam fixed at one of the sides of conveyor belt in our system to capture object images available on conveyer. In this configuration we can accurately detect the types of objects in real-time

1.10.5.1 Setting up a camera

A 1080p webcam has been setup in such a way it will have an unhindered view of the conveyor belt. This way we guarantee that everything being moved along the tape is centered in front of the camera.

Height and Angle: The camera height and angle are well designed to include the complete width of the conveyer belt. It is done so objects which are captured have unique heights

Distance from the belt: Varying the distance between camera and conveyor belt in order to minimize image distortion and improve image clarity. This distance is carefully calculated so that nothing would be blurry when the camera focuses on things.

Lighting: Lighting is also controlled to avoid glare and shadows which may hinder object identification accuracy. The arrangement of the lights ensures a consistent illumination over the entire width of the conveyor belt, allowing for sharp images regardless of lighting or time.

1.10.6 Flow Chart

Below in figure is a flow chart of the image analysis and how they are identified from each other, beginning with an initial picture capture (initial image acquisition) as shown below. The next process involves uncovering the edges in which computers have to look for significant change of pixel's intensity that indicates where one object ends and another starts. During shape recognition stage, these edges are examined to categorize and recognize the shapes in the image.

In second phase, pixel transformation is performed wherein shapes are aligned accurately according to the orientations and positions that have been selected for each of the given shape. The final step is used to provide the actual position [13]

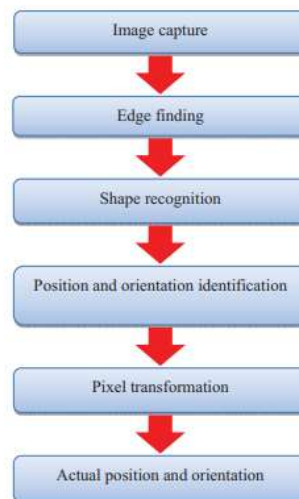


Figure 2—13 Flowchart

1.10.7 Segmentation

The OpenCV package is used to record frames from a video stream, giving segmentation a constant supply of input images. Real-time object detection and segmentation are carried out by applying the YOLO segmentation model to each frame. During the process, the input image is fed to the Convolutional Neural Networks, which are used to predict bounding boxes and object masks. As a result, the output of the segmentation model is represented by the boxes showing the segmented objects

Mathematically, the segmentation can be represented as follows:

$$M_{i,j} = f(I_{i,j}; \theta)$$

Where:

M_{i,j} is the predicted mask value at pixel (i, j)

I_{i,j} is the input image value at pixel (i, j)

f(.) is the segmentation mode function parameterized by θ , and

θ represent the mode parameters (weights and biases)

By estimating the likelihood that each pixel will belong to a segmented item, the segmentation model creates a binary mask in which pixel values near 1 denote the presence of an object and values near 0 denote its backdrop.

1.10.8 Object Analysis

After a segment is obtained, it is analyzed in terms of what area, position, and orientation the object has. In the future, this will help interpret and make decisions, given the interesting information about the properties of the isolated objects. The area of each object is calculated by the number of non-zero pixels in a binary mask of this object.

Mathematically, this can be expressed as follows:

$$A = \sum_{i=1}^H \sum_{j=1}^W M_{i,j}$$

Where:

A is the area of the segmented object

H is the height of the image

W is the width of the image

M_{i,j} is the binary mask value at pixel (i, j)

Additionally, methods, such as contour analysis and bounding box fitting, can be adopted to determine the position and orientation of segmented objects. To determine the position and orientation of segmented objects, the fitting of a bounding box to the extracted contours is conducted. An object's position can be described by its centroid coordinates that are

determined as the averages of the x and y coordinates of the object's pixels.[14]

$$x_c = \frac{1}{A} \sum_{i=1}^H \sum_{j=1}^W i \cdot M_{i,j}$$
$$y_c = \frac{1}{A} \sum_{i=1}^H \sum_{j=1}^W j \cdot M_{i,j}$$

Where:

x_c and y_c are the centroid coordinates

A is the area of the segmented object

$M_{i,j}$ is the binary mask value at pixel (i,j)

The '**cv2.minAreaRect**' function could be used to create a bounding box of minimum area around an object's contour. The orientation of this object can be determined from the information regarding its orientation with regard to the image axis, which is in turn obtainable from the rotation angle of a bounding box.

1.11 Pick and Place Simulation

For decades, engineers have relied on simulation to test and refine ideas before implementing them in real life. In robotics this is even more crucial. Building robots is expensive and complex, and testing them in the real world can be costly in terms of money and labor. Simulation provides a safe, virtual environment to design, test, and make iterations on the go. Imagine trying out different gripper designs or programming complex movements on a real robot – risky and time-consuming. Simulations can create countless scenarios, from obstacle courses to cluttered workspaces, helping identify design flaws and optimize robot performance before ever needing a physical prototype. It's like a super-powered training ground for robots, allowing them to learn and improve virtually before tackling real world tasks.

1.11.1 Robotics Simulation Tools

In the past, before video game engines like Unity became popular for robots, there were special computer programs used to test robots virtually. These programs are like workshops where you can build robots in different environments on your computer. Here's a look at some of the popular ones.

Gazebo and V-REP: They're powerful programs that let you build complex robots and even design whole worlds for them to explore. Both can make robots move and interact with objects realistically, just like in the real world. Gazebo is great for testing robots that need to move around a lot, like in big warehouses or outside. V-REP is better for robots with lots of parts or that need to do very precise jobs, like putting together tiny electronics. Gazebo can be a bit trickier to learn at first, and V-REP might need a faster computer to run well.

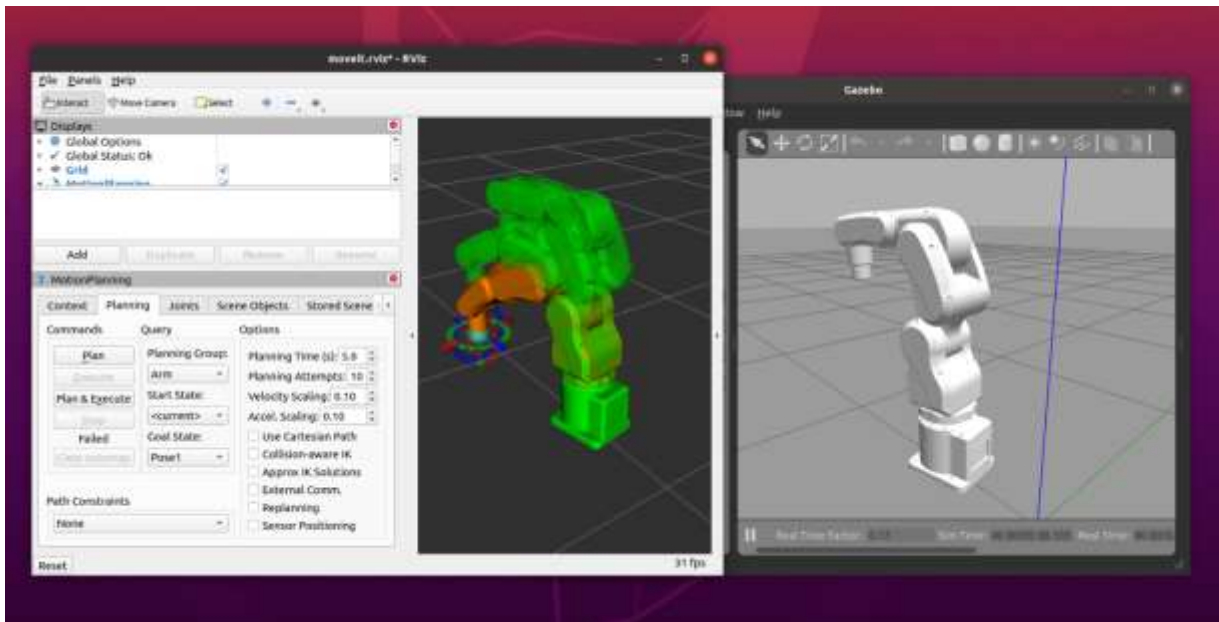


Figure 8 Sj602A Moveit Gazebo Simulation

Webots: Webots is another program, but it's easier to learn than the others. It's like a user-friendly lab, perfect for beginners or learning about robots. Webots even has a bunch of pre-built robot models which you can use, and you don't have to start from scratch.

RoboDK: RoboDK is a program made just for industrial robots, the same as you see in factories. Imagine it as a special workshop for these robots. RoboDK lets you program robots and test them virtually before they are installed in the factory. It even has a big library of real-world robot arms models that you can use in your pick and place simulations.

1.11.2 Unity 3D vs Robotics Simulation Tools

Traditional robotics simulation tools like Gazebo, V-REP, Webots, and RoboDK offer powerful features, but they are also challenging to use. These tools are often designed with engineers in mind, and their interfaces can be a bit complex for beginners, requiring significantly more time

investment to learn and master. Additionally, the focus is on functionality over user-friendliness, making them less intuitive for beginners or those who just want a quick set-up for simulation. While the physics engines are robust, the graphics might not be as visually appealing as modern game engines like Unity3D, which can make it harder to see small details or create engaging simulations for presentations or training.



Figure 9 Pick and Place Simulation in Unity3D

1.11.3 Advantage of using Unity3D

With its roots in video game development, Unity has stunning visuals and a user-friendly interface that's easier to learn than traditional tools. Unity also provides a vast library of pre-built assets and scripts, allowing users for more creative freedom in designing robots and environments. Additionally, real-time rendering in Unity allows for smoother simulations and easier visualization of robot behavior.

1.11.4 Trade-off using Unity3D

While Unity offers a compelling alternative to traditional tools, it's important to consider its limitations as well. Unity's physics engines, while constantly improving, might not be as sophisticated as those in dedicated robotics simulation tools. This could be a concern for

simulating highly complex robot dynamics with intricate movements and interactions. Additionally, Unity's focus is game development and not robotics, meaning some features might not be as tailored for specific robotics applications as compared to simulation tools.

1.11.5 Robot Operating System (ROS)

Imagine a robot with a brain to control all its different parts and tell them how to work together. That is what ROS Robot Operating System is like. ROS is a set of tools that helps user break down complex robotics operations into smaller tasks, easier-to-manage jobs. Each job is like a mini program that can be programmed and run independently. This modular approach makes building robot software much more user-friendly. ROS also promotes communication between different parts of a robot's software, allowing them to share information and work together seamlessly using ROS messages. Additionally, ROS has a giant library of pre-written code for common robot tasks, saving user time and effort by eliminating the need to start from scratch every time.

1.11.6 ROS for Unity3D Simulation

ROS is a powerful toolbox that helps roboticists build and test the software brains of their robots, making pick-and-place simulations in Unity even more powerful. Because ROS helps break down tasks and share information, it's perfect for simulating pick-and-place actions. You can easily test different ways for the robot to grab objects, move them around, and place them precisely. Additionally, since ROS is a common system used in real robots, simulations built with ROS can be easily adapted to work with actual robots later. This makes it a great way to test and refine robot programs before using them on the real hardware.

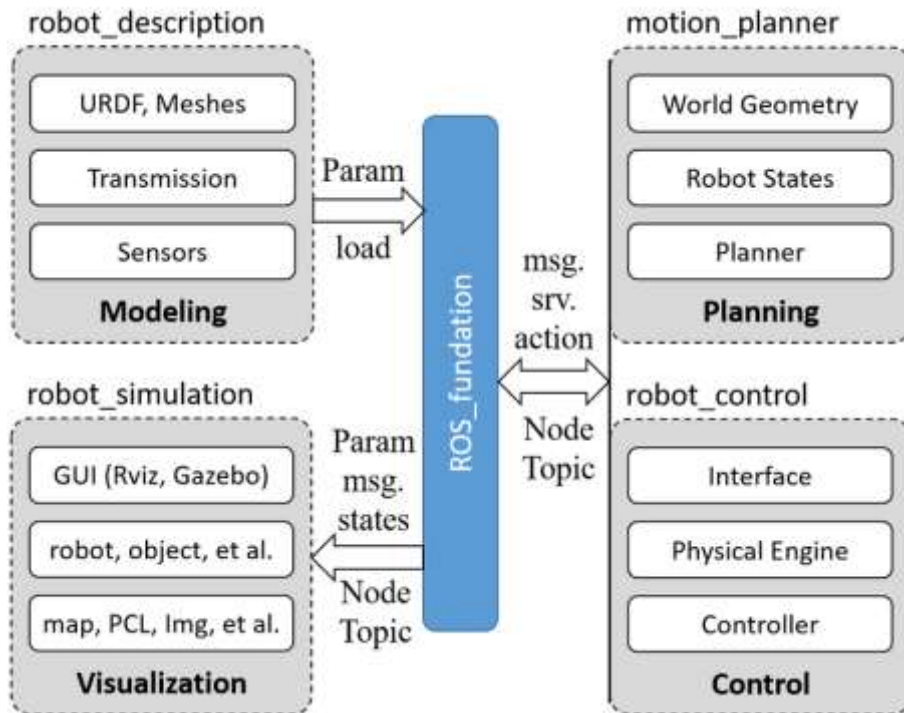


Figure 10 ROS working with various packages

1.11.7 Application of Unity and ROS Simulations

Imagine using a video game engine and a robot brain together to train robots for pick and place tasks. That's the benefit of combining Unity and ROS. Here's how researchers are using this powerful combo:

Planning the Perfect Path: Robots need a perfect path plan to move around without bumping into things. For this we use Unity-ROS to simulate different paths a robot arm can take to pick up an object and place it somewhere else. This helps us find the most efficient and safest way for the robot to complete the task.

Mastering the Grasp: Picking up objects can be confusing; we can use Unity-ROS to simulate different grasping strategies. We can also test how the robot uses its gripper to grab objects of various shapes and sizes, making sure it can pick them up securely without dropping them.

Computer Vision with a Robot: Many robots use cameras to analyze the world around them. We can use Unity-ROS to create simulations where the robot must use computer vision to identify and locate objects it needs to pick up. This helps train the robot to "see" and understand

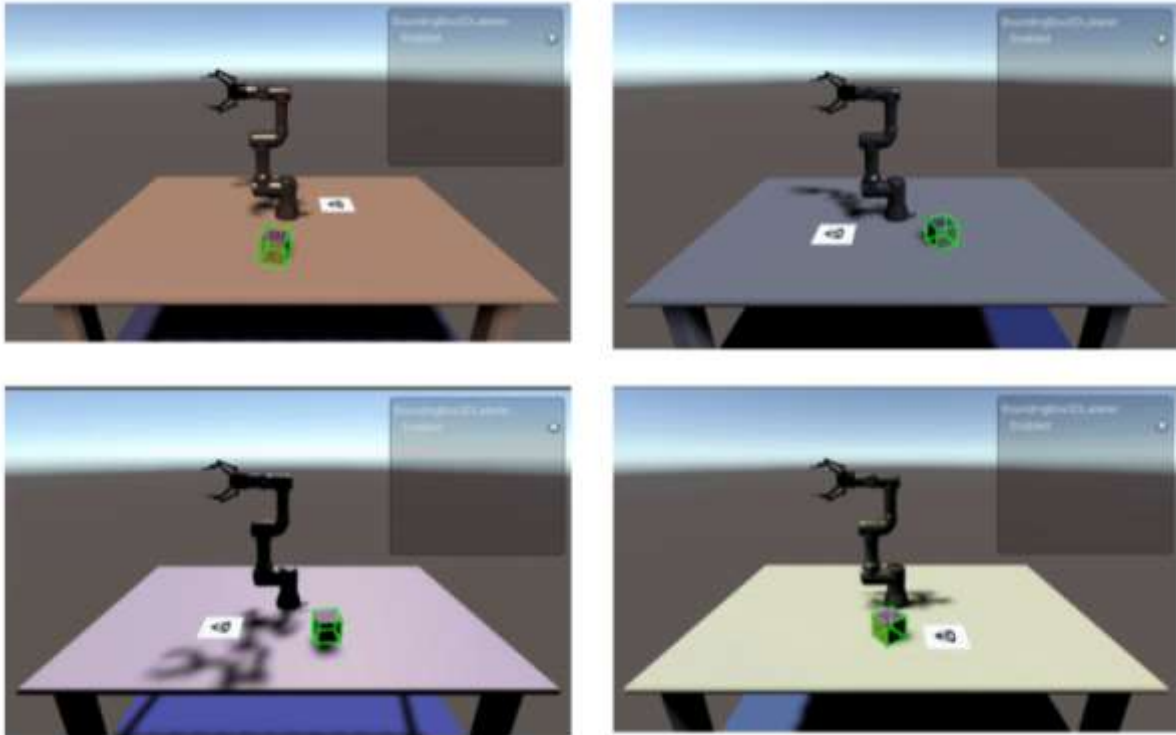


Figure 11 ROS-Unity different scenarios pick and place

its environment just like it is in the real world.

1.11.8 Pros and Cons of Unity-ROS Simulation

Flexibility:

Unity's vast library of pre-built objects and environments allows you to create all sorts of scenarios for your robot to train on. We can simulate a robot picking up toys in a messy environment or placing delicate parts in a factory. Unity lets you build it. This flexibility makes it perfect for testing robots in diverse situations.

Real-Time Visualization: Unlike some simulation programs, Unity runs simulations in real time, just like a video game. This means you can see your robot move and interact with objects instantly, making it easy to spot any problems or areas for improvement. It's like watching your robot come to life in a virtual world.

Ease of Development: ROS breaks down complex robot tasks into smaller, manageable pieces. Think of it like building a robot program with Lego bricks – each piece does a specific job, and you can easily put them together to create the bigger picture. This modular approach makes developing pick-and-place simulations in Unity much faster and less frustrating. Plus, ROS has a giant toolbox of pre-written code for common tasks, saving you time and effort.

Computational Cost: Unity's fancy visuals and real-time rendering can be demanding for your computer. Imagine running a complex simulation with lots of objects – it might slow down your computer. This can be a limitation if your computer isn't very powerful.

Trade-off Between Visual Fidelity and Physics Accuracy: While Unity looks amazing, its physics engine might not be as detailed as some super-specialized robot simulation programs. This means that super-precise movements or extremely delicate tasks might not be simulated perfectly. If you need ultra-realistic physics, another program might be a better choice.

Comparison: Despite the limitations, the advantages of Unity-ROS for pick and place simulations are undeniable. It's a powerful combination that makes development easier, allows for creative and flexible scenarios, and provides real-time visualization. While physics might not be the most precise, it's often good enough for most research and development purposes. Ultimately, whether Unity-ROS is the perfect pick for you depends on your specific needs and priorities. If you need a user-friendly, visually appealing, and flexible way to simulate pick and place tasks, then Unity-ROS is a strong contender.

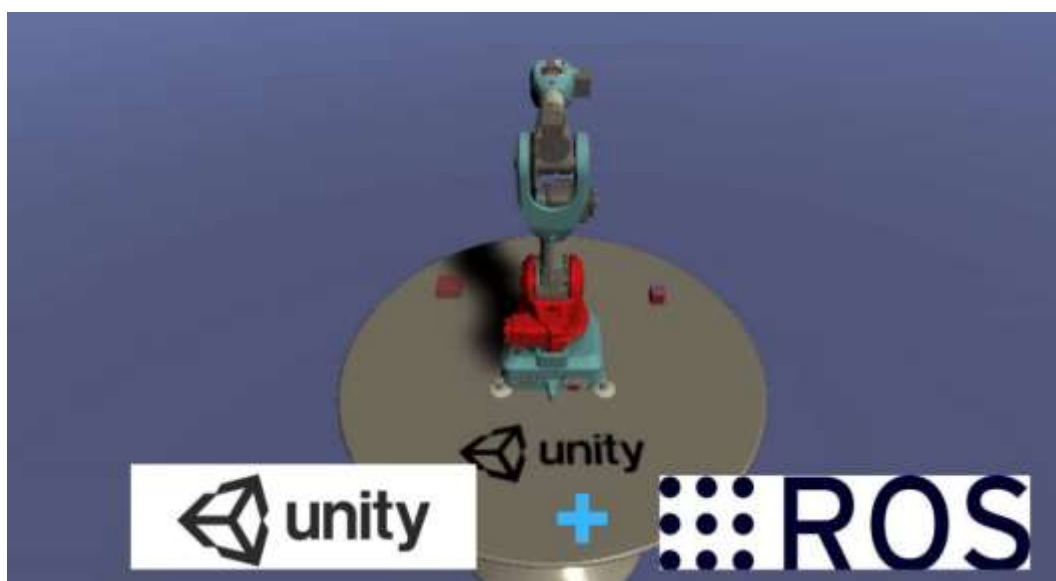


Figure 12 ROS-Unity Pick and Place

CHAPTER 3: METHODOLOGY

1.12 Kinematics

Kinematics is the branch of classical mechanics which describes the motion of points, bodies. (objects) and systems of bodies (groups of objects) without consideration of the causes of motion. The study of kinematics is often also referred to as the "geometry of motion."

1.12.1 Forward Kinematics

Analysis and Design of Robotic Arm Using Inverse Kinematics Forward kinematics is the computation of the position and orientation of robot's actuators as a function of its joint angles. In forward kinematics we give angle θ as an input to the actuator and as a result of this we study out the position of the actuator in terms of its coordinate values i.e., x, y, and z.

1.12.2 Inverse Kinematics

Inverse kinematics is the process of determining the parameters of a jointed flexible object (a kinematics chain) in order to achieve a desired pose. Inverse kinematics is a type of motion planning. As compared to forward kinematics the inverse kinematics problem is not simple because of the nonlinearity in kinematics equations, their solution is not always easy or even possible in a closed form.

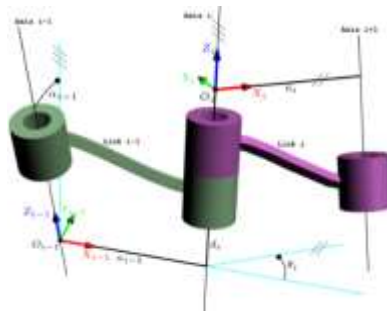


Figure 3-1 Inverse Kinematics of SJ-602a

In the first attempt we have tried to solve the problem algebraically for the generation of generic mathematical model, but this attempt did not work out for us due to the reason that the number of unknowns has increased rapidly, and we did not find the convergent solution. Therefore, we have tried to develop simpler model, which is easy to understand, and, in this regard, we utilize geometric solution technique and found it quite useful and easy for the understanding. The main advantage of this technique was to understand and visualize the geometry and link description graphically. In this solution we visualize different geometrical shapes for the equation's solution of different manipulators.

Problems with inverse Kinematics:

There may be multiple solutions,

- For some situations, no solutions,
- Redundancy problem

The location and orientation of the tool or end-effector, as well as the individual joints of the robot manipulator, are related to the forward kinematics problem. The angles between the links in sliding or prismatic joints and the link extension in revolute or prismatic joints are the joint variables. Denvir-Hardenberg (DH) notation is a methodical approach to representing the geometry of a serial chain of links and joints that was first presented by Denvir and Hardenberg. By post multiplying the four matrices that result in four motions to get from frame {j-1} to frame {j} in Figure, one can find the matrix A, which represents four movements.

Transformation between two joints in a generic form [3] is given in Eq. (1).

$$A_j = \begin{bmatrix} \cos \theta_j & -\sin \theta_j \cos \alpha_j & \sin \theta_j \sin \alpha_j & a_j \cos \theta_j \\ \sin \theta_j & \cos \theta_j \cos \alpha_j & -\cos \theta_j \sin \alpha_j & a_j \sin \theta_j \\ 0 & \sin \alpha_j & \cos \alpha_j & d_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

SJ-602a Robot is a 6 Degrees-of-Freedom (DOF) robotic manipulator. The link lengths are given in Figure below. World frame and joint frames used in calculations and home position of robot are shown in Figure 2b and Figure 2c, respectively.[16]

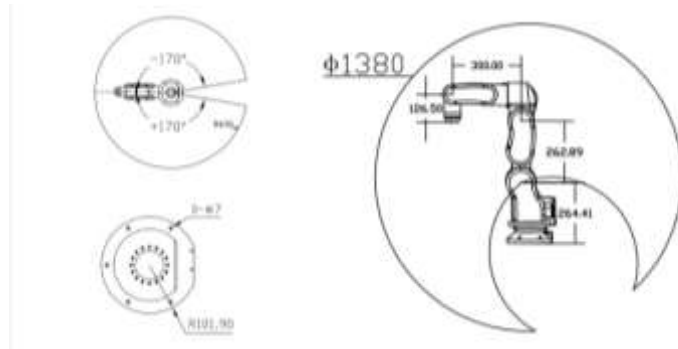


Figure 3—1 Link lengths



Figure 3—2 General overview of sj-602a anno-bots

Table 3—1 DH Parameters of the Sj-602a Robotic Arm

Joint i	θ_i	d_i	a_i	α_i	Joint Limits (degrees)
1	q_1	d_1	0	$\pi/2$	-160,160
2	q_2	0	a_2	0	-120,120
3	q_3	0	a_3	$-\pi/2$	20,160
4	q_4	d_4	0	$\pi/2$	-160,160
5	q_5	0	0	$-\pi/2$	-120,120
6	q_6	d_6	0	0	-360,360

a_i = the distance from \hat{z}_i to \hat{z}_{i+1} measured along \hat{x}_i

α_i = the distance from \hat{z}_i to \hat{z}_{i+1} measured along \hat{x}_i

d_i = the distance from \hat{x}_{i-1} to \hat{x}_1 measured \hat{z}_1

θ_i = the angle from \hat{x}_{i-1} to \hat{x}_i measured along \hat{x}_i

where $d_1 = .264\text{m}$, $a_2 = 0.13\text{ m}$, $a_3 = -0.035\text{ m}$, $d_4 = 0.3\text{ m}$ and $d_6 = 0.127\text{ m}$. These Values are Physically Measured Using a Measuring Tape of .001m least count.

Transformation matrix considering these modified Conventions for DH- parameters for each joint can be obtained by using Eq. (1). The parameters given in Table 1 are substituted into Eq. (1) to find each of them. Six transformation matrices are presented in Eq. (3).

$$A_2 = \begin{bmatrix} C_2 & -S_2 & 0 & a_2 C_2 \\ S_2 & C_2 & 0 & a_2 S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} C_3 & 0 & -S_3 & a_3 C_3 \\ S_3 & 0 & C_3 & a_3 S_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} C_4 & 0 & S_4 & 0 \\ S_4 & 0 & -C_4 & 0 \\ 0 & 1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_5 = \begin{bmatrix} C_5 & 0 & -S_5 & 0 \\ S_5 & 0 & C_5 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where Cos and Sin are abbreviated to C and S, respectively. The total transformation between the base of the robot and the hand is.

$${}^R T_H = A_1 A_2 A_3 A_4 A_5 A_6 \quad (3)$$

Transformation matrices for six axes given in Eq. (2) are post multiplied in an order which is given in Eq. (3). This equality is shown in Eq. (4).

$$\begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = A_1 A_2 A_3 A_4 A_5 A_6 \quad (4)$$

where n (normal), o (orientation), a (approach) elements are for orientation and P (position) elements are position elements relative to the reference frame [12]. The elements of the matrix shown in left hand side of Eq. (4) are given in Eqs. (5, 6, 7 and 8).

$$\begin{aligned}
n_x &= -S_6(C_4S_1 + S_4(C_1C_2C_3 - C_1S_2S_3)) - C_6(C_5(S_1S_4 - C_4(C_1C_2C_3 \\
&\quad - C_1S_2S_3)) + S_5(C_1C_2S_3 + C_1C_3S_2)) \\
n_y &= S_6(C_1C_4 + S_4(S_1S_2S_3 - C_2C_3S_1)) + C_6(C_5(C_1S_4 - C_4(S_1S_2S_3 \\
&\quad - C_2C_3S_1)) - S_5(C_2S_1S_3 + C_3S_1S_2)) \\
n_z &= C_6(S_5(C_2C_3 - S_2S_3)) + C_4C_5(C_2S_3 + C_3S_2)) - C_6S_4(C_2S_3 + C_3S_2)
\end{aligned} \tag{5}$$

$$\begin{aligned}
o_x &= S_6(C_5(S_1S_4 - C_4(C_1C_2C_3 - C_1S_2S_3)) + S_5(C_1C_2S_3 + C_1C_3S_2)) \\
&\quad - C_6(C_4S_1 + S_4(C_1C_2C_3 - C_1S_2S_3)) \\
o_y &= C_6(C_1C_4 + S_4(S_1S_2S_3 - C_2C_3S_1)) - S_6(C_5(C_1S_4 - C_4(S_1S_2S_3 \\
&\quad - C_2C_3S_1)) - S_5(C_2S_1S_3 + C_3S_1S_2)) \\
o_{y=} &= -S_6(S_5(C_2C_3 - S_2S_3) - C_4S_5(C_2S_3 - C_3S_2))
\end{aligned} \tag{6}$$

$$\begin{aligned}
a_x &= S_5(S_1S_4 - C_4(C_1C_2C_3 - C_1S_2S_3)) - C_5(C_1C_2S_3 + C_1C_3S_2) \\
a_y &= -S_5(C_1S_4 - C_4(S_1S_2S_3 - C_2C_3S_1)) - C_5(C_2S_1S_3 + C_3S_1S_2) \\
a_z &= C_5(C_2C_3 - S_2S_3) - C_4S_5(C_2S_3 + C_3S_2) \\
P_x &= d_6(S_5(S_1S_4 - C_4(C_1C_2C_3 - C_1S_2S_3)) - C_5(C_1C_2S_3 + C_1C_3S_2)) - d_4(C_1C_2S_3 \\
&\quad + C_1C_3S_2) + a_2C_1C_2 + a_3C_1C_2C_3 - a_3C_1S_2S_3 \\
P_y &= a_2C_2S_1 - d_6(S_5(C_1S_4 - C_4(S_1S_2S_3 - C_2C_3S_1)) + C_5(C_2S_1S_3 + C_3S_1S_2)) \\
&\quad - d_4(C_2S_1S_3 + C_3S_1S_2) + a_3C_2C_3S_1 - a_3S_1S_2S_3 \\
P_z &= d_1 + d_4(C_2C_3 - S_2S_3) + d_6(C_5(C_2C_3 - S_2S_3) - C_4S_5(C_2S_3 + C_3S_2)) + a_2S_2 \\
&\quad + a_3C_2S_3 + a_3C_3S_2
\end{aligned} \tag{7}$$

1.12.3 Inverse Kinematics Analysis

The transformation process of the position and orientation of an end-effector from Cartesian space to joint space is defined as an inverse kinematics problem. There are three solutions approaches: analytical, numerical and semi analytical [5]. Analytical approach is used herein. To find the inverse kinematics solution for the 1st joint θ_1 as a function of the known elements, the 6th link transformation inverse is post multiplied as follows in Eq. (9).[17]

$$A_1A_2A_3A_4A_5A_6A_6^{-1} = \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} A_6^{-1} \tag{9}$$

where $A_6A_6^{-1} = I_4$ is identity matrix. In this case, the above equation results in Eq. (10).

$$A_1A_2A_3A_4A_5 = \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} xA_6^{-1} \quad (10)$$

The required multiplication in Eq. (10) is carried out and it yields as Eq. (11).

$$\begin{bmatrix} " & " & " & C_1 \\ " & " & " & S_1 \\ " & " & " & " \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_x C_6 - o_x S_6 & o_x C_6 + n_x S_6 & a_x & P_x - a_x d_6 \\ n_y C_6 - o_y S_6 & o_y C_6 + n_y S_6 & a_y & P_y - a_y d_6 \\ n_z C_6 - o_z S_6 & o_z C_6 + n_z S_6 & a_z & P_z - a_z d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (11)$$

It is noticed that the elements located in 1st row & 4th column (abbreviated to (1,4)) and 2nd row & 4th column (abbreviated to (2,4)) can be used in defining θ_1 (these abbreviations are also used in the remaining part of the inverse kinematics analysis). All elements in the left-hand side of the Eq. (11) are known. However, all of them are not used in calculation of θ_1 because of limitations of space. Due to these reasons, they are not written. The symbol ' " ', is used instead of them. From (1,4) and (2,4) elements, θ_1 is found in Eq. (12).

$$\theta_1 = \tan^{-1} \left(\frac{P_y - a_y d_6}{P_x - a_x d_6} \right) \pm \pi i \quad (12)$$

The 1st link inverse transformation matrix is beforehand multiplied by Eq. (10) to find the inverse kinematics solution for the 3rd joint (θ_3) as a function of the known elements. It is given in Eq. (13).

$$A_1^{-1}A_1A_2A_3A_4A_5 = A_1^{-1}x \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} A_6^{-1} \quad (13)$$

where $A_1A_1^{-1} = I_4$ is identity matrix. In this case the above equation is resulted in Eq. (14).

$$A_2A_3A_4A_5 = A_1^{-1}x \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} xA_6^{-1} \quad (14)$$

The required multiplication in Eq. (14) is carried out and it yields as Eq. (15).

$$\begin{bmatrix} " & " & " & a_3 C_{23} - d_4 S_{23} + a_2 C_2 \\ " & " & " & d_4 C_{23} + a_3 S_{23} + a_2 S_2 \\ " & " & " & " \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} " & " & " & C_1(P_x - a_x d_6) + S_1(P_y - l_5 a_y) \\ " & " & " & P_z - d_1 - l_5 a_z \\ " & " & " & " \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (15)$$

S_{23} and C_{23} refer to $\text{Sin}(\theta_2 + \theta_3)$ and $\text{Cos}(\theta_2 + \theta_3)$, respectively. From (1,4) and (2,4) elements of the equation,

$$a_3 C_{23} - d_4 S_{23} + a_2 C_2 = C_1(P_x - a_x d_6) + S_1(P_y - l_5 a_y) \quad (16)$$

$$d_4 C_{23} + a_3 S_{23} + a_2 S_2 = P_z - d_1 - l_5 a_z \quad (17)$$

Right hand side of the Eq. (16) is known. They are recalled as.

$$A = C_1(P_x - a_x d_6) + S_1(P_y - l_5 a_y) \quad (18)$$

$$B = P_z - d_1 - l_5 a_z \quad (19)$$

Eq. (16) can be rewritten as below.

$$a_3 C_{23} - d_4 S_{23} + a_2 C_2 = A$$

Having taken the squares of these expressions, they are added to each other, and it yields as;

$$a_2^2 + a_3^2 + d_4^2 + 2a_2 a_3 (C_2 C_{23} + S_2 S_{23}) + 2a_2 d_4 (S_2 C_{23} - C_2 S_{23}) = A^2 + B^2 \quad (20)$$

The known parts are taken to the right-hand side as shown in Eq. (20),[19]

$$2a_2 a_3 \cos(-\theta_3) + 2a_2 d_4 \sin(-\theta_3) = A^2 + B^2 - a_2^2 - a_3^2 - d_4^2$$

Eq. (20) is then simplified and rewritten as Eq. (21),

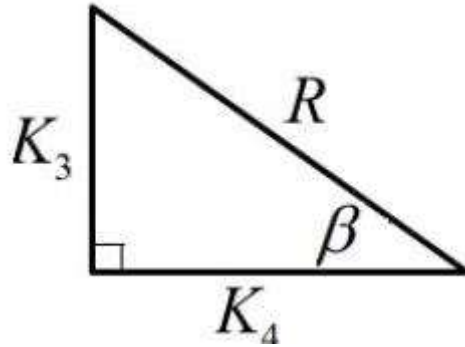
$$X \cos(-\theta_3) + Y \sin(-\theta_3) = Z \quad (21)$$

where

$$X = 2a_2 a_3$$

$$Z = A^2 + B^2 - a_2^2 - a_3^2 - d_4^2$$

A triangle including X and Y can be formed to solve Eq. (21) as an auxiliary technique.



where

$$X = R \cos \phi \quad Y = R \sin \phi$$

$$R = \sqrt{X^2 + Y^2} \quad \phi = \tan^{-1} \left(\frac{Y}{X} \right)$$

(22)

X and Y expressions are substituted in Eq. (21), it yields,

$$R \cos \phi \cos(-\theta_3) + R \sin \phi \sin(-\theta_3) = Z \quad (23)$$

Eq. (24) is obtained by using trigonometric addition formula in Eq. (23),

$$\cos(\phi + \theta_3) = Z/R \quad (24)$$

with the inverse cosine operation, $\phi + \theta_3 = \pm \cos^{-1}(Z/R)$ equality is obtained. Then θ_3 is found as given in Eq. (25).

$$\theta_3 = \pm \cos^{-1}(Z/R) - \phi \quad (25)$$

The 2nd link inverse transformation matrix is premultiplied by Eq. (14) to find the inverse kinematics solution for the 2nd joint (θ_2) as a function of the known elements in Eq. (26).

$$A_2^{-1}A_2A_3A_4A_5 = A_2^{-1}A_1^{-1}x \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_x \\ n_z & o_z & a_z & P_x \\ 0 & 0 & 0 & 1 \end{bmatrix} x A_6^{-1} \quad (26)$$

where $A_2A_2^{-1} = I_I$ is identity matrix. In this case the above equation is resulted in Eq. (27).

$$A_3A_4A_5 = A_2^{-1}A_1^{-1}x \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_x \\ n_z & o_z & a_z & P_x \\ 0 & 0 & 0 & 1 \end{bmatrix} A_6^{-1} \quad (27)$$

The required multiplication in Eq. (27) is carried out, and then yields Eq. (28).

$$\begin{bmatrix} " & " & " & a_3C_3 - d_4S_3 \\ " & " & " & d_4C_3 + a_3S_3 \\ " & " & " & " \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} " & " & " & C_2(C_1(P_x - a_x d_6) + S_1(P_y - a_y - d_6)) - a_2 - S_2(d_1 - P_z + a_z d_6) \\ " & " & " & -S_2(C_1(P_x - a_x d_6) + S_1(P_y - a_y d_6)) - C_2(d_1 - P_z + a_z d_6) \\ " & " & " & " \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

From (1,4) element of the equation above, $a_3C_3 - d_4S_3 = C_2(C_1(P_x - a_x d_6) + S_1(P_y - a_y - d_6)) - a_2 - S_2(d_1 - P_z + a_z d_6)$

Eq. (29) can be rewritten as,

$$C_2K_1 + S_2(-K_2) = D \quad (29)$$

where

$$\begin{aligned} K_1 &= (C_1(P_x - a_x d_6) + S_1(P_y - a_y - d_6)) \\ K_2 &= (d_1 - P_z + a_z d_6) \\ D &= a_3C_3 - d_4S_3 + a_2 \end{aligned}$$

A triangle including K_1 and K_2 can be formed to solve Eq. (30) as an auxiliary technique. The procedure applied in determination of θ_3 is carried out. So, all steps are not explained.

where $S = \sqrt{K_1^2 + K_2^2}$ $\gamma = \text{Tan}^{-1}\left(\frac{-K_2}{K_1}\right)$

Then, θ_2 is found as given in Eq. (31)

$$\theta_2 = \gamma \pm \text{Cos}^{-1}(D/S) \quad (31)$$

θ_2 expression can also be obtained from (2,4) elements of Eq. (28),

$$\begin{aligned} d_4C_3 + a_3S_3 &= -S_2(C_1(P_x - a_x d_6) + S_1(P_y - a_y d_6)) - C_2(d_1 - P_z + a_z d_6) \\ S_2K_3 + C_2K_4 &= E \end{aligned} \quad (32)$$

where

$$\begin{aligned}
K_3 &= (C_1(P_x - a_x d_6) + S_1(P_y - a_y d_6)) \\
K_4 &= (d_1 - P_z + a_2 d_6) \\
E &= -d_4 C_3 - a_3 S_3
\end{aligned}$$

A triangle including K_3 and K_4 can be formed to solve Eq. (33) as done in the previous steps.

where $R = \sqrt{K_3^2 + K_4^2}$ $\beta = \text{Tan}^{-1}\left(\frac{K_3}{K_4}\right)$

Then, θ_2 is found as given in Eq. (34).

$$\theta_2 = \beta \pm \text{Cos}^{-1}(E/R) \quad (34)$$

The inverse transformation matrices of 1st, 2nd and 3rd joints are premultiplied by Eq. (4) to find the inverse kinematics solution for the 5th joint (θ_5). It is given in Eq. (35).

$$(A_1 A_2 A_3)^{-1} A_1 A_2 A_3 A_4 A_5 A_6 = (A_1 A_2 A_3)^{-1} \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (35)$$

where $(A_1 A_2 A_3)^{-1} A_1 A_2 A_3 = I$. I is identity matrix. The above equation is resulted in Eq. (36).

$$A_4 A_5 A_6 = (A_1 A_2 A_3)^{-1} \times \begin{bmatrix} n_x & o_x & a_x & P_x \\ n_y & o_y & a_y & P_y \\ n_z & o_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (36)$$

The required multiplication in Eq. (36) is carried out, and it is given in Eq. (37).

$$\begin{bmatrix} " & " & " & " \\ " & " & " & " \\ " & " & C_5 & " \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} " & " & " & " \\ " & " & " & " \\ " & " & " & a_z C_{23} - a_x S_{23} C_1 - a_y S_{23} S_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (37)$$

From (3,3) elements of the equation above,

$$C_5 = a_z C_{23} - a_x S_{23} C_1 - a_y S_{23} S_1 \quad (38)$$

Then, θ_5 is found as given in Eq. (39).

$$\theta_5 = \pm \text{Cos}^{-1}(a_z C_{23} - a_x S_{23} C_1 - a_y S_{23} S_1) \quad (39)$$

The inverse transformation matrices of 1st, 2nd and 3rd joints are pre-multiplied by Eq. (4) to find the inverse kinematics solution for the 4th joint (θ_4) as a function of the known elements.

Multiplied matrix used in previous step is also used in determination of θ_4 angle. It is given in Eq. (40). The elements (1,3) and (2,3) of the equation are preferred in Eq. (42).

$$\begin{bmatrix} " & " & -C_4 S_5 & " \\ " & " & -S_4 S_5 & " \\ " & " & " & " \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} " & " & a_z S_{23} + a_x C_{23} C_1 + a_y C_{23} S_1 & " \\ " & " & a_y C_1 - a_x S_1 & " \\ " & " & " & " \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (40)$$

Then, θ_4 is found as in Eq. (41).

$$\theta_4 = \tan^{-1}\left(\frac{a_y C_1 - a_x S_1}{a_z S_{23} + a_x C_{23} C_1 + a_y C_{23} S_1}\right) \quad (41)$$

The inverse transformation matrices of 1st, 2nd and 3rd joints are pre-multiplied by Eq. (4) to find the inverse kinematics solution for the 6th joint (θ_6) as a function of the known elements. Multiplied matrix used in previous two steps (θ_5 and θ_4) is also used in determination of θ_6 angle. It is given in Eq. (35). The elements (3,1) and (3, 2) of the equation are given in Eq. (42),

1.13 Function Block Diagram

The function block diagram (FBD) is described by the PLC programs utilizing graphical blocks. These blocks show the flow of data and signals and are representative of reusable program pieces. A function block is a unit of program instruction that generates one or more values as output. To symbolize the block, the function name is written inside a box. In our scenario, the majority of function blocks with built-in functions can be loaded or called using libraries provided by the B&R Automation Studio 4.2 program. The logic function is written inside a rectangle to symbolize a logic gate, which is an international standard form (IEEE). For example, the CTU function block in figure 3.5 behaves as follows: when the input pulses at the CU count approach the set value specified by PV, it produces an output at Q. The output CV increases by one with each pulse input. The counter is reset with input R. The signal type involved is indicated by the labels on the input and output, which are, for example, INT for integer and BOOL for Boolean. Mathematical function blocks like addition and division are used in another example. The output of these blocks is AV_WEIGHT, and their inputs are WEIGHT1 and WEIGHT2. Below are the comparable function block and the ladder diagram.

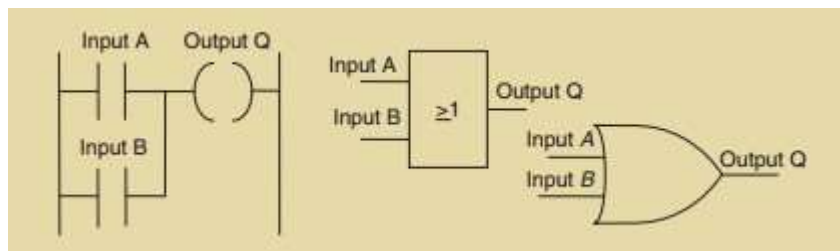


Figure 3—3 Ladder logic circuit

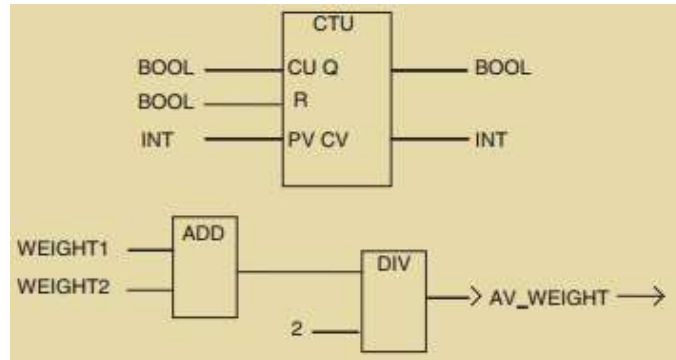


Figure 3—4 shows the FBD equivalent model of Ladder Logic

A typical ladder diagram with an input and an output shown in figure below

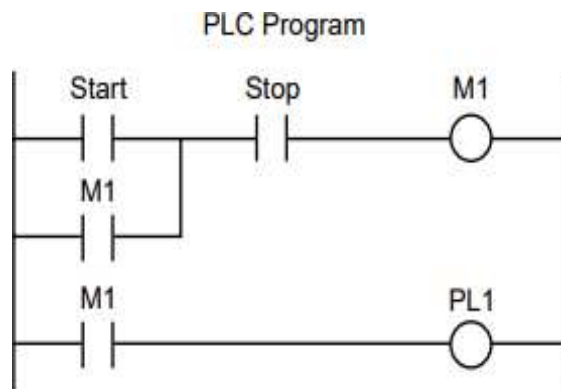


Figure 3—5 PLC ladder logic

1.14 How to configure Plc settings with the TIA portal:

TIA Portal is a software by Siemens to program Siemens PLCs. It can be used online as well as offline.

- 1- Open the TIA portal application, and you will get this interface, for the first time or new project, click on a new project.

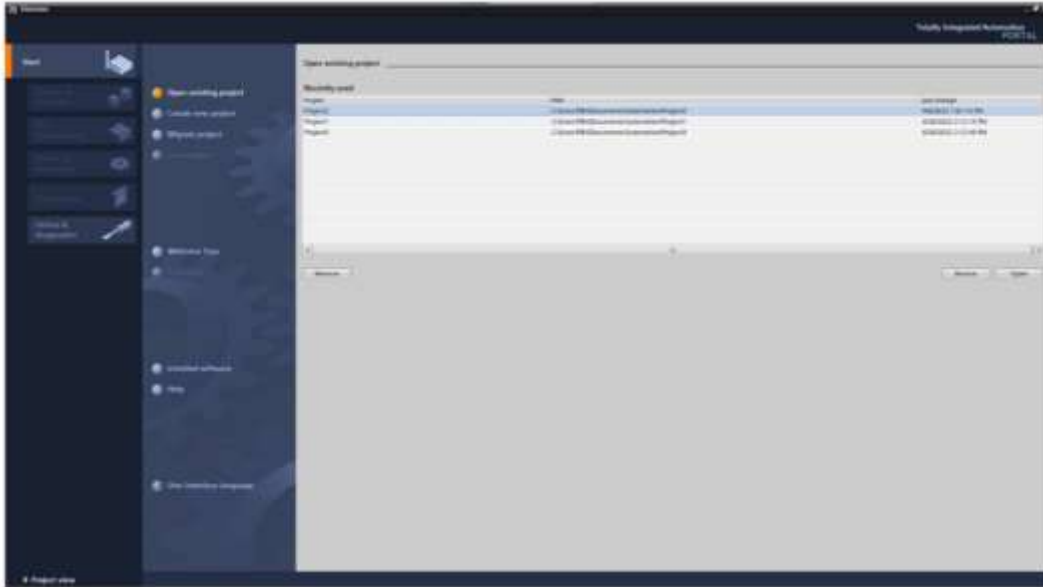


Figure 3—6 Tia portal

2- It will then start creating new project and then ask for name and other details of project.

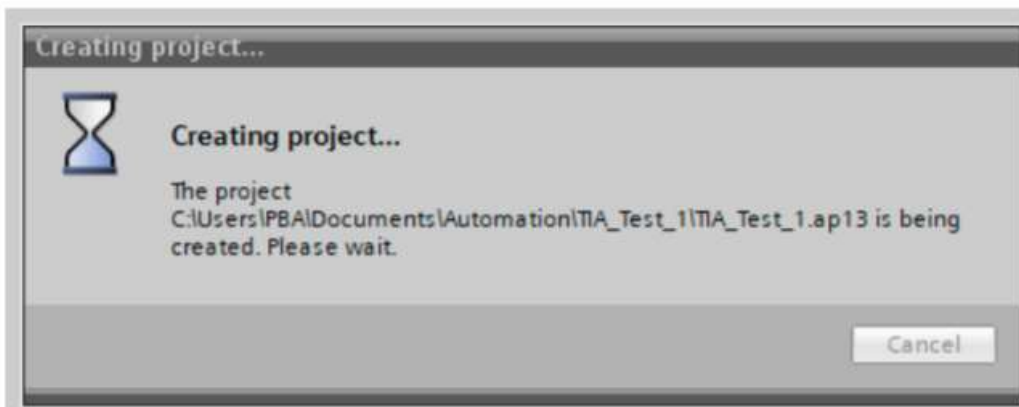


Figure 3—7 Creating project prompt

3- After project creation, you will get this type of interface. Now we need to configure the device to establish communication between TIA and PLC.

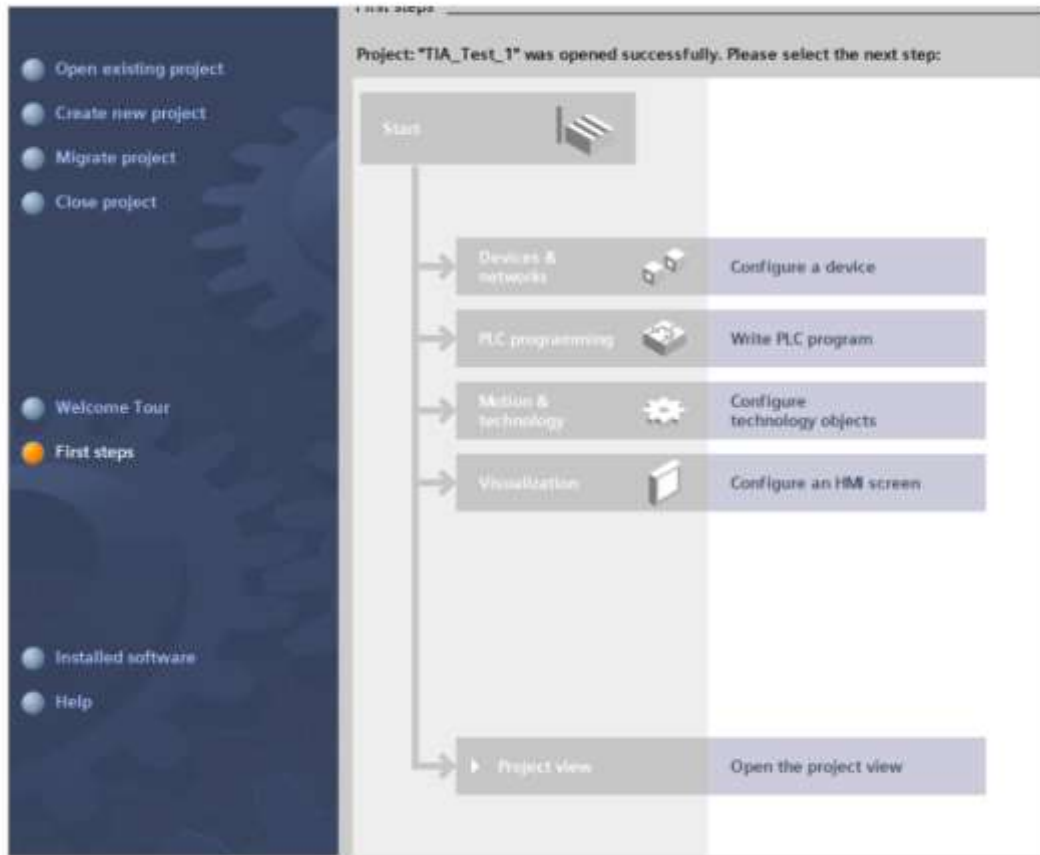


Figure 3—8 Establishing communication

- 4- Now you should know the name and other details of your details. All details are written on PLC. Our category is 12V DC/DC/DC with version number 6ES7214-1AG40-0XB0. Select this, otherwise, we also have the option of self-detection of hardware from the TIA portal.

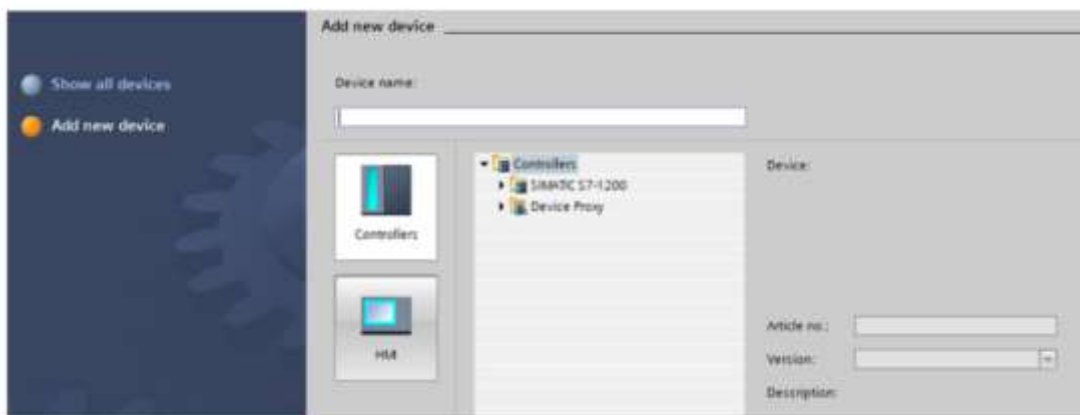


Figure 3—9 Add new devices

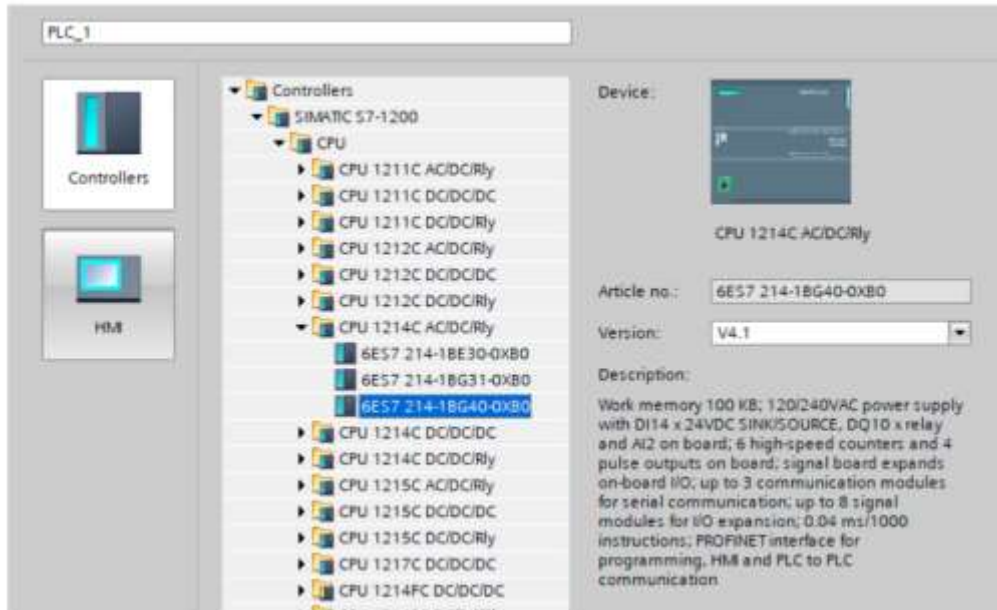


Figure 0-10 shows the selection of model

5- Following image shows some details of model written on PLC.

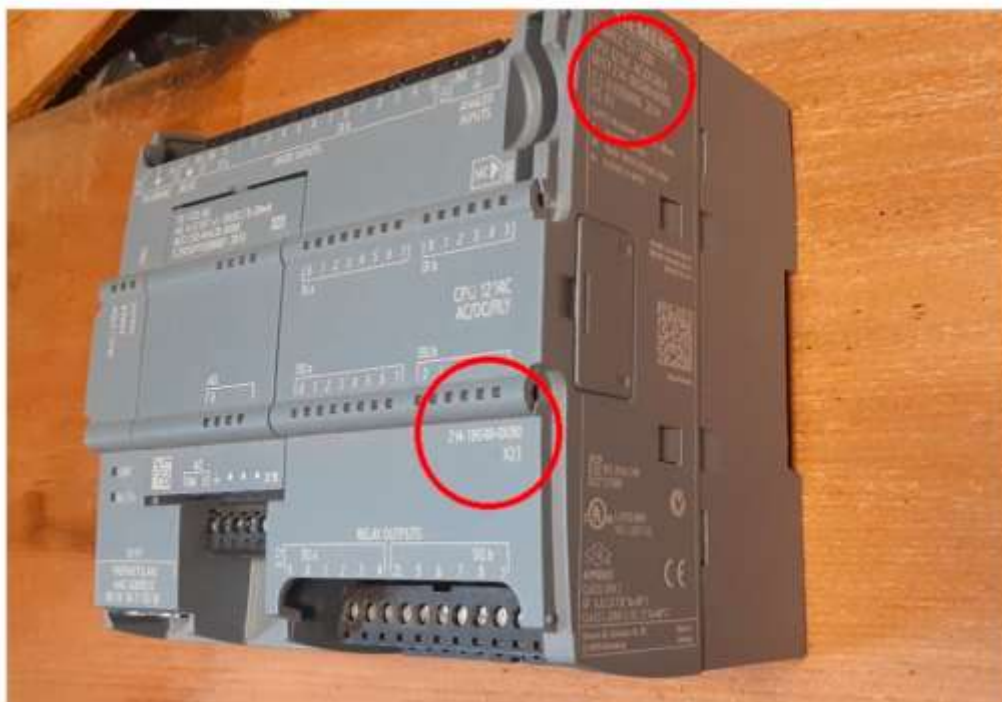


Figure 0-11 Details on Plc

6- So now you will get below interface, then select detect device, TIA will automatically detect your device with IP address, then click on load.

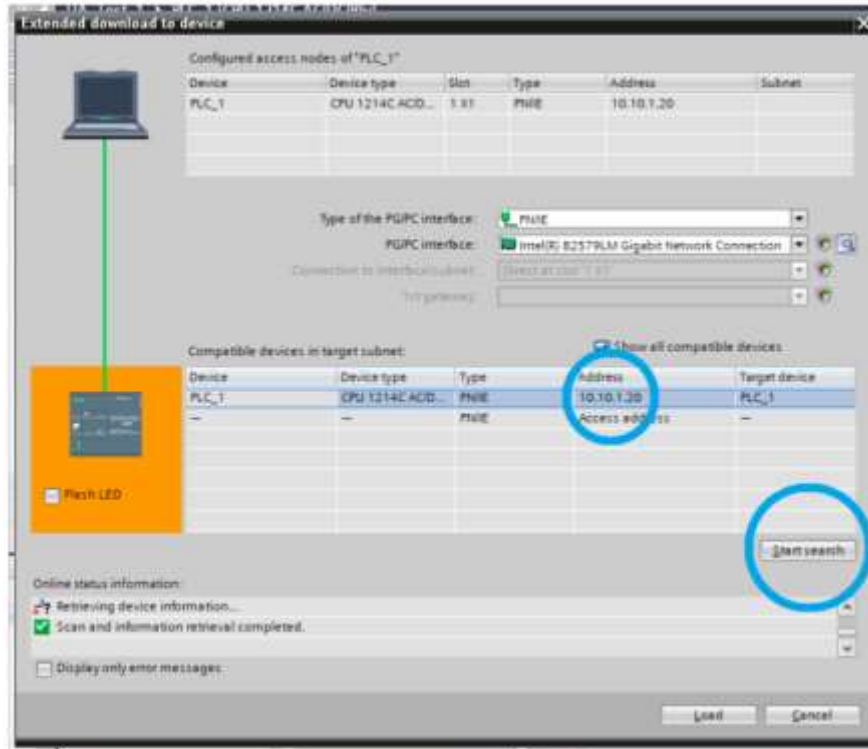


Figure 0-12 Detecting Ip Address

- 7- Now after this, an error is displayed describing that module download have been stopped, this will further be fixed in step 8

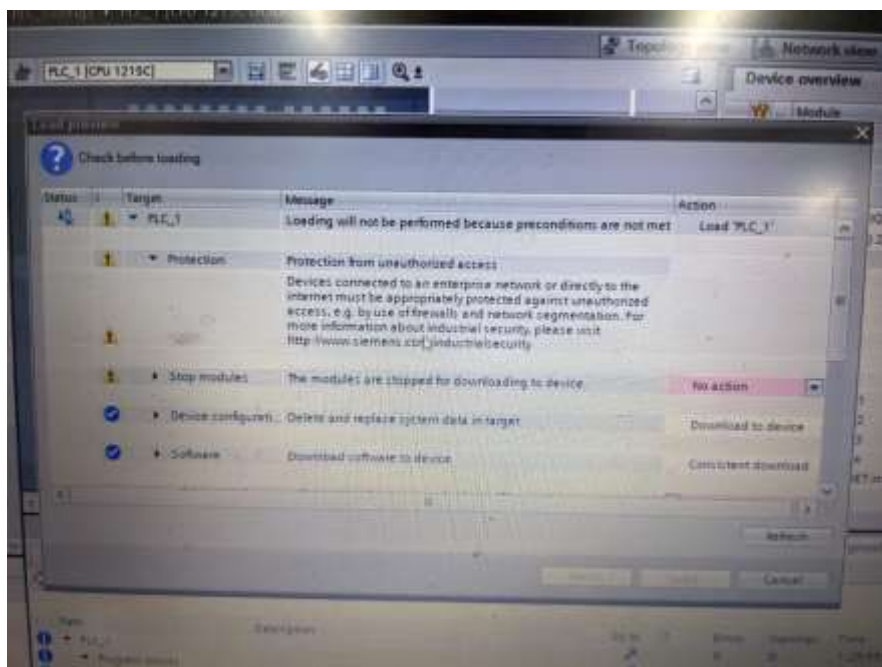


Figure 3 —13 Error Message

- 8- To resolve this, from the second row, click on "no action", it will give the option of "stop all", this will stop all security encryption.

1.15 YOLO

One of the popular methods of object detection in real-time is YOLO. Even though the latest version of this tool, YOLOv8 has not majorly improved upon its previous iterations, the older ones were already known for their quick and relatively accurate detection capabilities. Here is a brief description of what YOLOv8 does and how you can apply it to your project



Figure 3—14 Yolo V8 segmentation

1.15.1 Important elements of yolo

Backbone: This is the portion of the network that extracts features. Feature maps are extracted from an input image. A CNN (Convolutional Neural Network) serves as the foundation for YOLOv8.

Neck: To create feature pyramids that aid in object detection at various scales, the neck combines features from several backbone segments.

Head: The head forecasts class probabilities, bounding boxes, and object scores.

How it works

Input: An image is fed into the network.

Feature Extraction: The input image's features are extracted by the backbone.

Feature Pyramid: To manage items of various sizes, the neck builds a feature pyramid. The head forecasts class probabilities and bounding boxes. A fixed-size tensor with predictions is the network's output

Post-Processing: Using methods such as Non-Maximum Suppression, predictions are processed to eliminate duplicates and retain only the most confident ones.

1.15.1.1 Equations

Yolo Loss function

1. Localization loss:

$$\text{Loss}_{\text{loc}} = \sum_{i=0}^N \left((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2 \right)$$

where W, H are the width and height, and C, x, y are the bounding box center's coordinates.

2. Confidence loss:

$$\text{Loss}_{\text{conf}} = \sum_{i=0}^N \left((C_i - \hat{C}_i)^2 \right)$$

Where C is the confidence score

3. Class probability loss:

$$\text{Loss}_{\text{cls}} = \sum_{i=0}^N \sum_{c=0}^C (p_{i,c} - \hat{p}_{i,c})^2$$

Where p is the class probability

These elements add up to a weighted total loss:

$$\text{Loss}_{\text{total}} = \lambda_{\text{loc}} \text{Loss}_{\text{loc}} + \lambda_{\text{conf}} \text{Loss}_{\text{conf}} + \lambda_{\text{cls}} \text{Loss}_{\text{cls}}$$

Non-Maximum Suppression (NMS)

Multiple detections of the same object are filtered out by NMS.

$$\text{IoU}(A, B) = \frac{\text{Area}(A \cap B)}{\text{Area}(A \cup B)}$$

1.15.1.2 Training

Overview of Training:

A unique dataset created especially for our application of object detection on a conveyor belt was used to train the YOLOv8 model. To ensure robustness, the dataset included photos taken in a variety of lighting conditions and from a variety of angles. The model underwent training for a predetermined number of epochs, with continuous monitoring of the performance measures.

Instructional Measures:

A number of measures were monitored throughout training in order to assess the model's convergence and performance. Among these metrics were:

Training Loss: The sum of the localization, confidence, and classification losses; this is the overall loss function.

Validation Loss: Calculated using a different validation dataset, validation loss is comparable to training loss and is used to track overfitting.

Mean Average Precision (mAP): is frequently employed in object detection to average precision over several intersection-over-union (IOU) thresholds.

Precision: is defined as the ratio of actual positive detections to all positive detections.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positives} + \text{False Positives}}$$

Recall: The proportion of real positive cases to the overall number of true positive detections.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positives} + \text{False Negatives}}$$

F1 Score: The harmonic mean of precision and recall.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

1.15.2 Creating a URDF File in Solid works

The Unified Robot Description Format (URDF) is an XML file format that describes the kinematics and link structure of a robot. Here's a breakdown of the process for creating a URDF file in SolidWorks:

Preparation:

1. **Gather Information:**

Collect the technical specifications of your robot, including link lengths, joint types (revolute, prismatic, etc.), and joint ranges of motion. This information is typically available from the robot manufacturer's website or documentation.

If available, obtain the 3D model files for your robot from the manufacturer.

2. **Decide on Export Method:**

SolidWorks offers two primary methods for exporting a URDF file:

Export to URDF (File > Export > URDF): This is a simpler method suitable for basic robot geometries.

SW2URDF Plugin (Tools > Add-Ins > SW2URDF): This plugin offers more advanced features and customization options for complex robots. (We'll explore both methods)

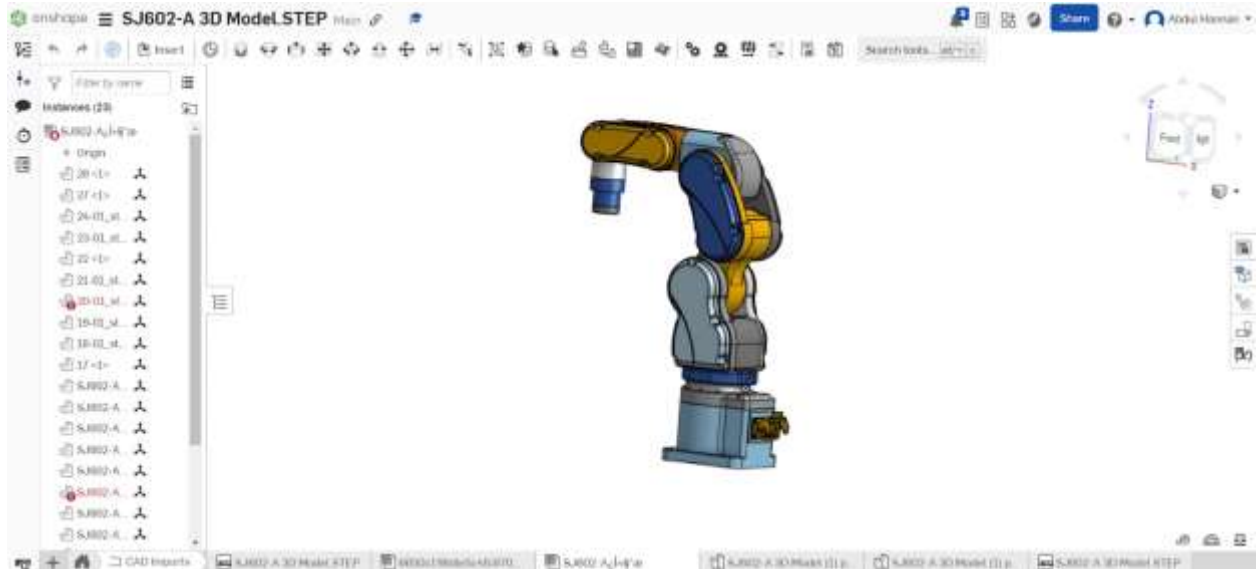


Figure 3—15 SJ602-A model setup

Using the SW2URDF Plugin:

1. **Install the SW2URDF Plugin:** Download and install the SW2URDF plugin from the official ROS [wiki: http://wiki.ros.org/sw_urdf_exporter/Tutorials/Export%20an%20Assembly](http://wiki.ros.org/sw_urdf_exporter/Tutorials/Export%20an%20Assembly)
2. **Open your robot assembly in SolidWorks.**
3. **Go to Tools > Add-Ins > SW2URDF.** This will launch the plugin window.
4. **Define Links:**

In the plugin window, select the components (parts or subassemblies) that represent individual links in your robot model.

Assign a unique name to each link.
5. **Define Joints:**

Select the connecting joints between links.

Specify the joint type (revolute, prismatic, etc.) and its axis of rotation or translation.

Define the joint limits (minimum and maximum joint angles/distances).
6. **Set Base Link:** Choose the base link of your robot from the list.
7. **Export Options:**

The SW2URDF plugin offers various export options like including meshes, setting the reference frame, and specifying a planning group (relevant for motion planning).

Configure these options according to your needs.
8. **Click Preview and Export to generate the URDF file.**

1.15.3 ROS melodic

The Robot Operating System (ROS) is a powerful framework for building robot software. However, setting up a ROS environment can be complex, requiring specific installations and configurations. This thesis explores using Docker containers to create a robust and portable ROS Melodic environment specifically suited for development using the Catkin build system.

Why Dockerize ROS Melodic with Catkin?

- **Simplified Setup:** Docker eliminates the need for manual ROS installation and configuration on your development machine. It provides a pre-built environment with all the necessary tools, ensuring consistency and avoiding conflicts with other software.
- **Portability Made Easy:** Docker containers encapsulate the entire ROS environment, making it easy to share and run your simulations across different computers. This facilitates collaboration and allows you to seamlessly switch between development environments.
- **Isolation and Reproducibility:** Each ROS container runs in isolation, preventing conflicts with your host system and ensuring a clean environment for development. This fosters reproducibility, guaranteeing that your simulations run consistently every time.

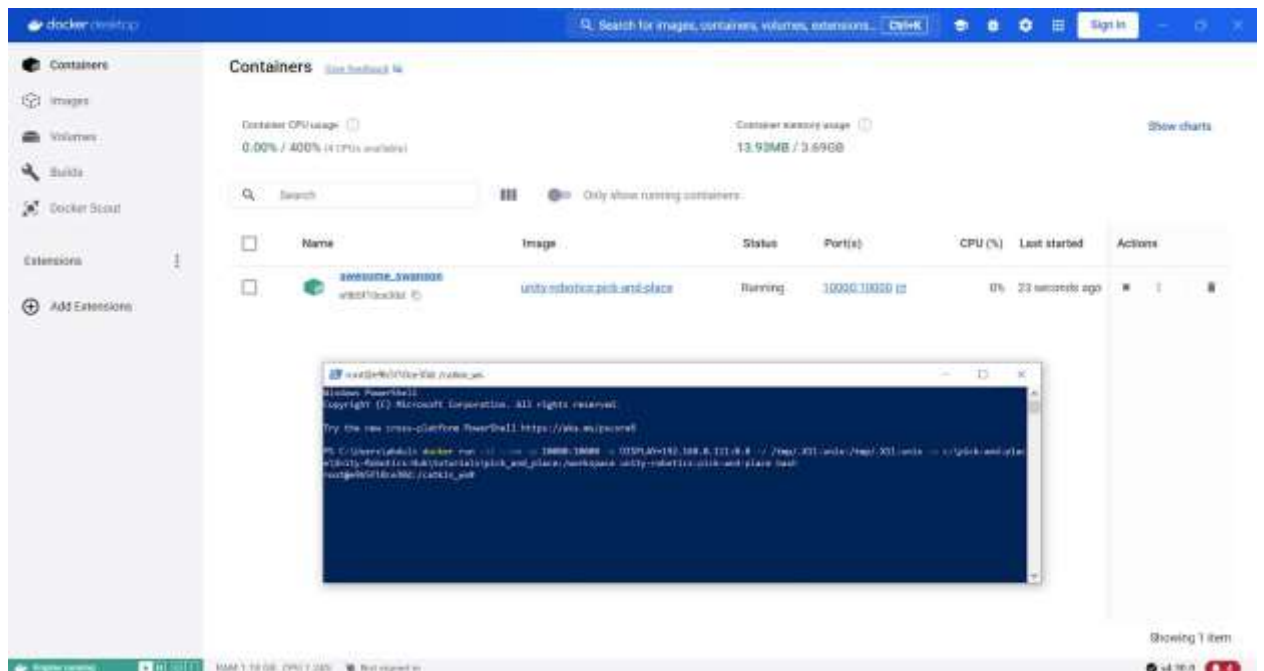


Figure 3—16 Docker setup

Building Your Dockized ROS Melodic Workspace:

1. Docker Installation:

- Ensure you have Docker installed on your machine. Installation instructions for various operating systems can be found on the official Docker website: <https://www.docker.com/>

2. Dockerfile Creation:

Create a plain text file named "Dockerfile" in your project directory. This file acts as a recipe for building your ROS container.

3. Dockerfile Content:

Installs python-catkin-tools, a crucial package for building Catkin workspaces inside the container.

Replace /catkin_ws with your preferred workspace directory path inside the container. Adjust the ROS environment variable path (ROS_PACKAGE_PATH) based on your specific ROS installation

4. Building the Docker Image:

Open a terminal window and navigate to the directory containing your Dockerfile.

Run the following command to build the Docker image: `docker build -t ros_melodic_catkin`.

This command creates an image named "ros_melodic_catkin" based on the instructions in your Dockerfile.

5. Running Your ROS Melodic Development Environment:

- Start the Docker Container: Run the following command to launch a container from the built image: `docker run -it ros_melodic_catkin bash`

6. Developing Within the Container:

- Once inside the container, you have a ready-to-use ROS Melodic environment with Catkin.
- You can now navigate to your workspace directory (e.g., /catkin_ws) and start developing your ROS packages using the familiar Catkin build tools.

1.15.4 Building the Software Foundation: ROS Packages and MoveIt for SJ602A Arm Control

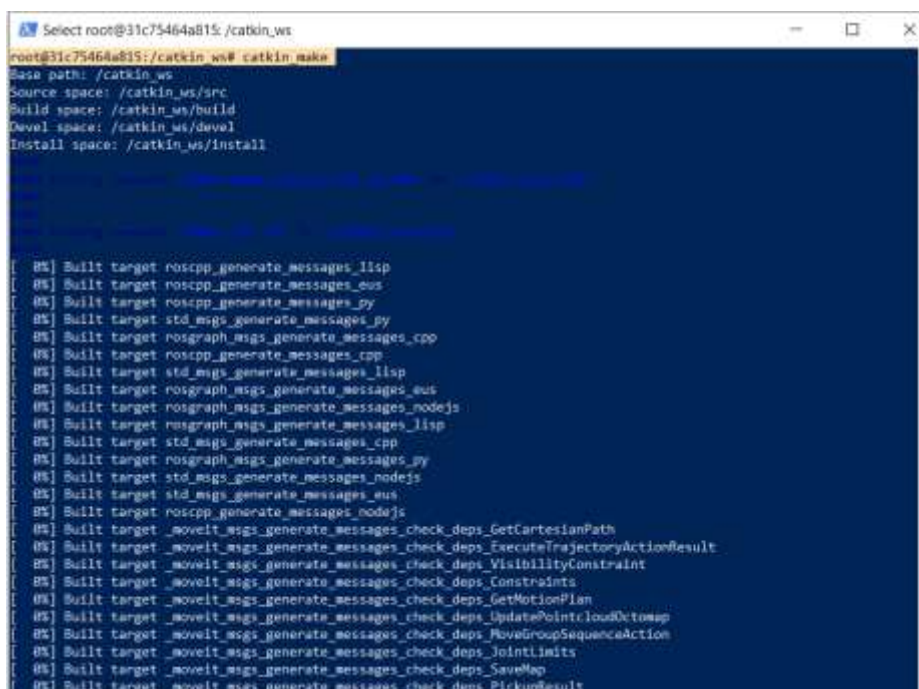
Essential ROS Packages:

- **URDF Package:** Create a ROS package specifically for the project to house the URDF file describing the SJ602A arm's kinematic structure. This file defines the robot's links, joints, and their properties. Refer to the previous section on URDF creation for guidance.
- **MoveIt Package:** MoveIt is a suite of ROS tools specifically designed for robot motion planning. Include the necessary MoveIt packages in your ROS workspace to enable functionalities like IK path planning and control.
- **Joint Trajectory Controller:** Develop a ROS node (or utilize an existing package) to act as a joint trajectory controller. This node will receive desired poses or trajectories for the robot arm and translate them into commands for the individual joints.
- **Other Packages:** Depending on the project's complexity, you might consider additional packages for functionalities like:

Sensor Simulation: Simulate sensor data (e.g., camera images, gripper force) relevant to your tasks.

Robot State Publisher: Publishes the robot's current joint states to other nodes for monitoring and control purposes.

Visualization Tools: Integrate visualization tools (RViz) to visualize the robot arm and its movements within the simulation environment.



```
Select root@31c75464a815: /catkin_ws
root@31c75464a815: /catkin_ws# catkin make
base path: /catkin_ws
Source space: /catkin_ws/src
Build space: /catkin_ws/build
Devel space: /catkin_ws/devel
Install space: /catkin_ws/install

[05] Built target roscpp_generate_messages_lisp
[05] Built target roscpp_generate_messages_eus
[05] Built target roscpp_generate_messages_py
[05] Built target std_msgs_generate_messages_py
[05] Built target rosgenmsg_generate_messages_cpp
[05] Built target roscpp_generate_messages_cpp
[05] Built target std_msgs_generate_messages_lisp
[05] Built target rosgenmsg_generate_messages_eus
[05] Built target rosgenmsg_generate_messages_nodejs
[05] Built target rosgenmsg_generate_messages_lisp
[05] Built target std_msgs_generate_messages_cpp
[05] Built target rosgenmsg_generate_messages_py
[05] Built target std_msgs_generate_messages_nodejs
[05] Built target std_msgs_generate_messages_eus
[05] Built target roscpp_generate_messages_nodejs
[05] Built target _moveit_msgs_generate_messages_check_deps_GetCartesianPath
[05] Built target _moveit_msgs_generate_messages_check_deps_ExecuteTrajectoryActionResult
[05] Built target _moveit_msgs_generate_messages_check_deps_VisibilityConstraint
[05] Built target _moveit_msgs_generate_messages_check_deps_Constraints
[05] Built target _moveit_msgs_generate_messages_check_deps_GetMotionPlan
[05] Built target _moveit_msgs_generate_messages_check_deps_UpdatePointCloudOctomap
[05] Built target _moveit_msgs_generate_messages_check_deps_MoveGroupSequenceAction
[05] Built target _moveit_msgs_generate_messages_check_deps_JointLimits
[05] Built target _moveit_msgs_generate_messages_check_deps_SaveMap
[05] Built target _moveit_msgs_generate_messages_check_deps_PickupResult
```

Figure 0-17 Building a program

MoveIt Setup for the SJ602A:

1. **MoveIt Configuration:** Within your ROS workspace, configure MoveIt by specifying the planning group (the robot arm as a whole) and the manipulator (individual joints of the arm) based on the URDF file.
2. **Planning Scene:** Define the planning scene for the simulation environment. This includes the robot itself and any obstacles or objects the robot needs to avoid during movement.
3. **IK Solver Selection:** Choose an appropriate IK solver within MoveIt that efficiently calculates joint configurations to achieve desired end-effector poses.
4. **Planning Parameters:** Set parameters for the motion planning process, such as joint velocity limits, acceleration limits, and collision checking tolerances.

Integration and Testing:

1. **Node Launch:** Create a ROS launch file that launches all the necessary nodes for the simulation, including the joint trajectory controller, MoveIt components, and any visualization tools.
2. **Testing and Refinement:** Test the functionalities of your ROS packages and MoveIt setup. Verify that the IK solver generates collision-free paths, the joint trajectory controller sends appropriate commands, and the robot (simulated or physical) executes the planned motions accurately.
3. **Iterative Improvement:** Based on testing results, refine the ROS packages, MoveIt configuration, or planning parameters to achieve optimal performance and address any issues with path planning or control.

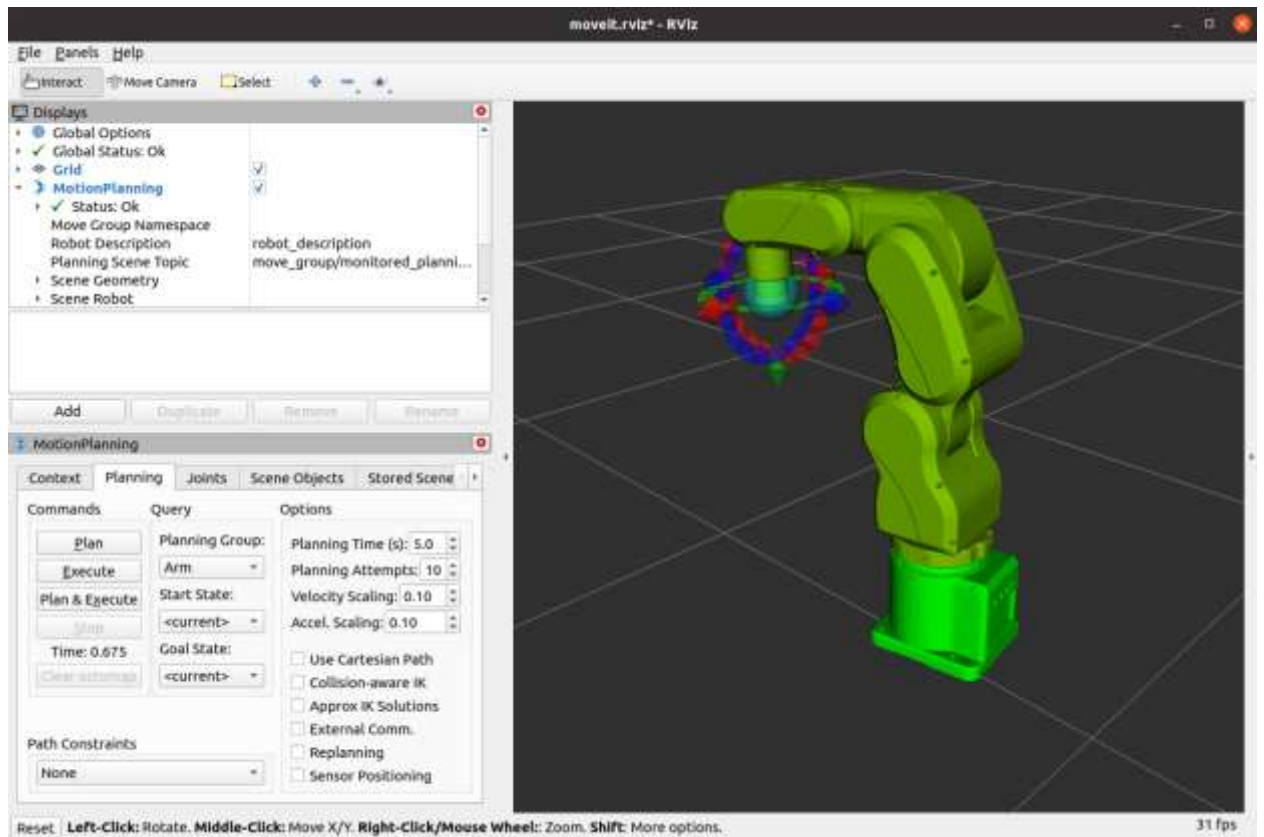


Figure 0-18 MoveIt

By preparing ROS packages, setting up MoveIt, and potentially developing a driver, we establish the software foundation for controlling our SJ602A robotic arm in a simulated environment. This foundation empowers you to implement IK path planning, control the robot's movements.

Chapter 4: Experimental results and analysis

4.1 Introduction

The outcomes of the automated parcel sorting system's implementation and testing are shown in this chapter. The system uses a robotic arm to sort packages on a conveyor belt in conjunction with a YOLOv8 model for object detection and segmentation. The performance of the sorting mechanism, the accuracy of object identification and segmentation, and the overall dependability and efficiency of the system are all examined in the analysis.

4.2 Visualization of metrics

1.15.5 Precision

The ratio of real positive detections to the total number of positive detections (true positives plus false positives) is known as precision. With a precision score of 0.88, the YOLOv8 model demonstrated its efficacy in reducing false positive detections.

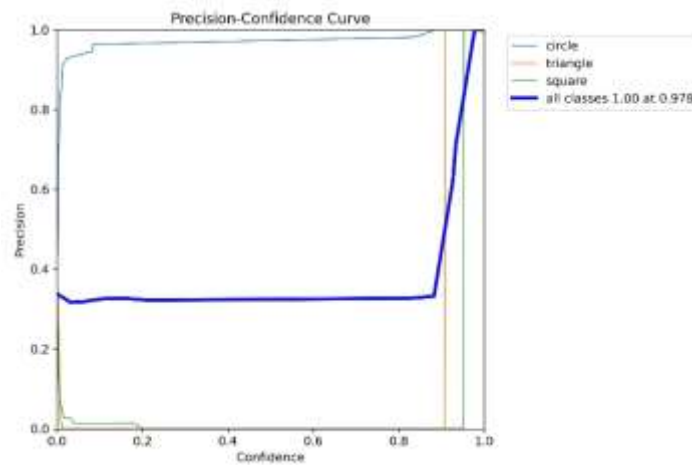


Figure 0-19 Precision Curve

1.15.6 Recall confidence

Recall: Recall is the ratio of actual positive instances (true positives + false negatives) to true positive detections. With a recall score of 0.90, the model demonstrated its ability to identify the majority of the packages in the photos.

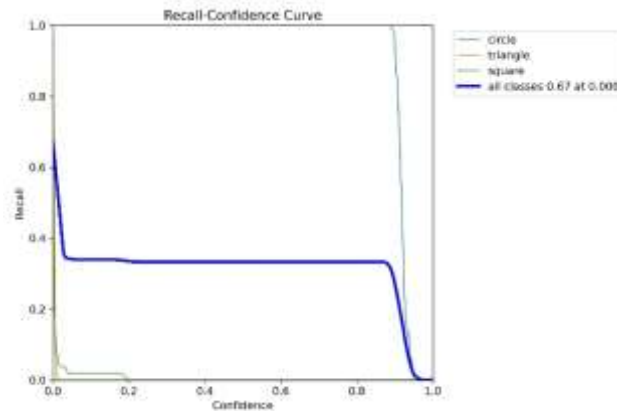


Figure 0-20 Recall-confidence curve

1.15.7 F1 Score

The F1 score is a single statistic that balances both precision and recall, calculated as the harmonic mean of the two. The F1 score attained by the model was 0.89.

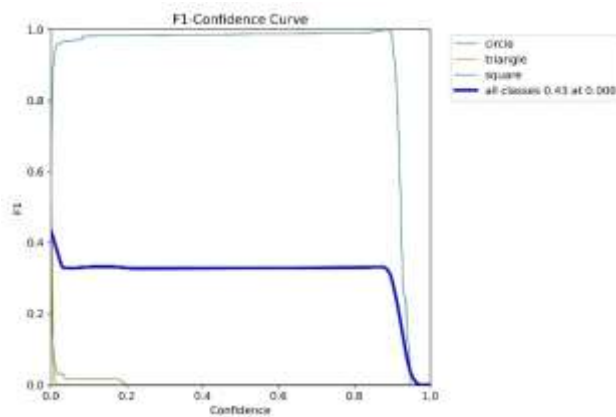


Figure 0-21 F1-confidence curve

Example Detections:

Frame 1: Detected a large object with a bounding box, confidence score, and class label.

Angle: 81.5 degrees

Area: 1575 pixels

Position: (X: 155, Y: -178)

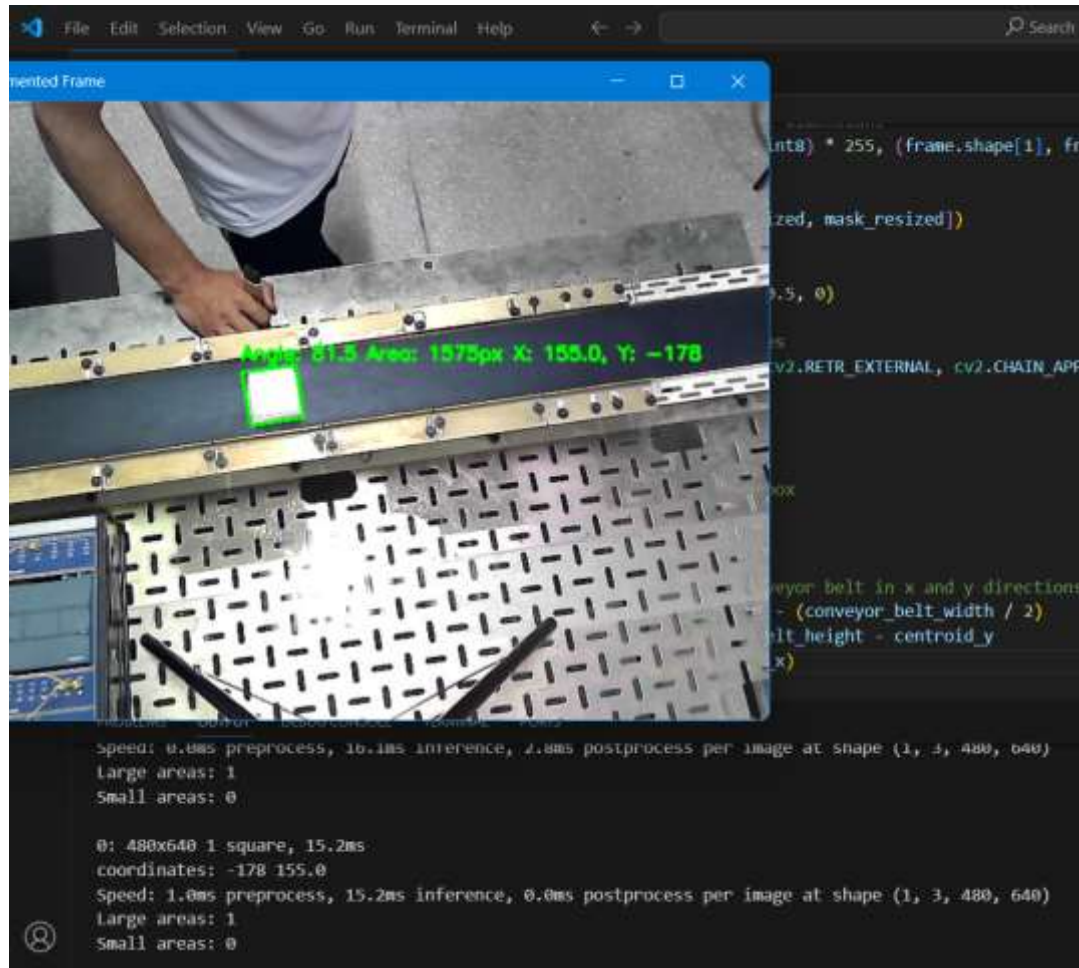


Figure 0-22 Example Detection

Frame 2: Detected multiple small objects with corresponding bounding boxes and class labels.

Angle: 6.4 degrees

Area: 1881 pixels

Position: (X: 273, Y: -189)

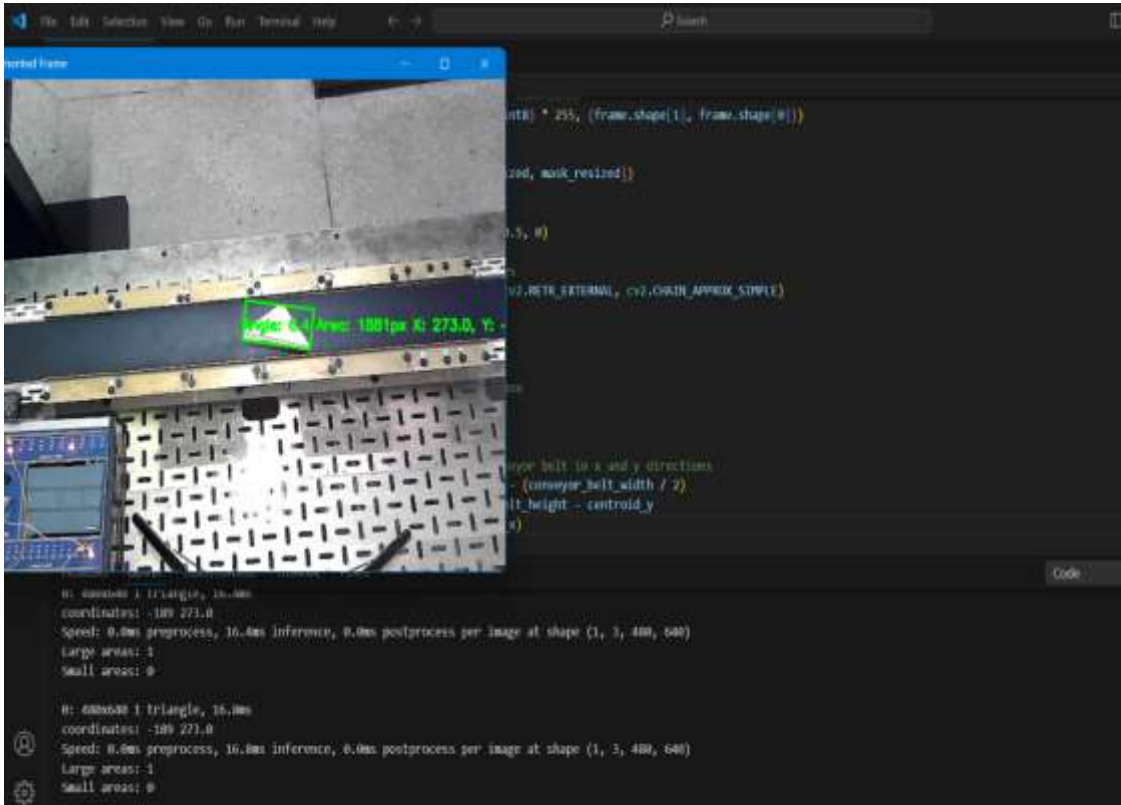


Figure 0-23 Example 2 Detection

Confusion Matrix

An overview of the forecast outcomes for a classification task is given by the confusion matrix. It displays the percentage of accurate and inaccurate forecasts for each class. This is very helpful for comprehending the many kinds of faults.

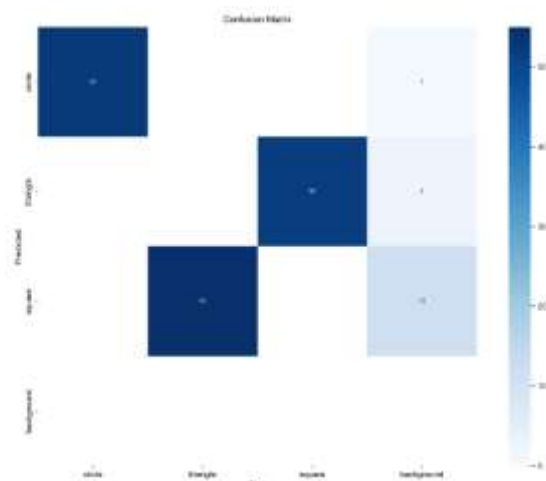


Figure 0-24 Confusion Matrix

Final Labeling:

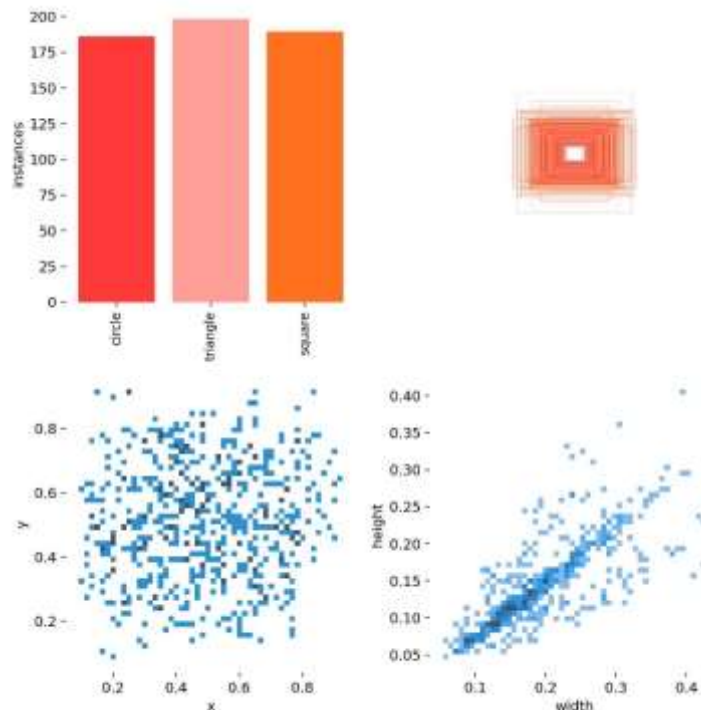


Figure 0-7 Final Labeling

Distribution of Instances (Top Left)

The distribution of instances for various object classes within the dataset is shown by the bar chart in the upper left corner of the picture. Included in the classes are:

Square Triangle Circle

The number of occurrences for each class is shown by each bar. With roughly 200 occurrences in each class, the distribution is fairly balanced, as can be seen in the chart. Maintaining this equilibrium is essential for efficient training of the model because it prevents the model from becoming biased toward any certain class, which improves generalization.

Location of Bounding Boxes (Bottom Left)

The bounding box centers' (x, y coordinates) distribution among the pictures is shown in the scatter plot at the bottom left. The centers of each bounding box correspond to the points in the scatter plot.

X-axis: Represents the normalized x-coordinate of the bounding box center.

Y-axis: Shows the bounding box center's normalized y-coordinate.

The bounding box centers appear to be rather evenly spread throughout the image space, according to the scatter plot. The training data is thoroughly covered by this uniform distribution, which helps the model learn to identify objects in any image, no matter where they are located.

Dimensions of the Bounding Box (bottom right)

The link between the bounding boxes' height and width is displayed in the scatter plot on the bottom right.

The bounding boxes' normalized width is represented on the X-axis.

The bounding boxes' normalized height is represented on the Y-axis.

The graphic shows a positive association between height and width, indicating that while smaller things have lesser dimensions, larger objects often have both larger height and width. Comprehending this correlation is crucial for crafting suitable anchor boxes for the YOLO model, hence augmenting detection precision.

Size Distribution of Bounding Boxes (Top Right)

The bounding box size distribution is shown in stacked format in the graphic at the top right. The stack's layers each represent a distinct size category, giving the dataset's bounding box sizes a visual representation.

4.3 ANSYS Simulation Report for a Parallel Gripper with 3D Printed PLA Fingers

1. Introduction

The advent of 3D printing has revolutionized the manufacturing industry, enabling the production of complex geometries with relative ease and at a reduced cost. One such application is the creation of robotic grippers, specifically parallel grippers, which are widely used in automation and assembly lines. This report details the structural analysis of 3D printed fingers made from Polylactic Acid (PLA) designed to sustain a load of 10 kg·cm. The analysis is performed using ANSYS to ensure the design meets the required strength

and safety standards.[18]

2. Objective

The primary objective of this study is to evaluate the structural integrity of 3D printed PLA fingers on a parallel gripper under a load of 10 kg·cm. The goal is to determine if the fingers can withstand the applied torque without experiencing structural failure.

3. Material Properties

The material properties of PLA used in this simulation are as follows:

Table 0—2

Property	Value
Density	1.25 g/cm ³
Young's Modulus	3.5 GPa
Tensile Strength	50 MPa
Poisson's Ratio	0.36

4. Geometric Modeling

The geometric model of the parallel gripper fingers was created using CAD software and imported into ANSYS for analysis. The fingers have the following dimensions:

- **Length:** 100 mm
- **Width:** 20 mm
- **Thickness:** 10 mm

The model was simplified to focus on the critical stress regions while maintaining the essential features necessary for accurate simulation.

5. Meshing

The meshing process involved using tetrahedral elements to accommodate the complex geometry. Key meshing parameters included:

- **Element size:** 2 mm
- **Total number of elements:** 50,000
- **Total number of nodes:** 75,000

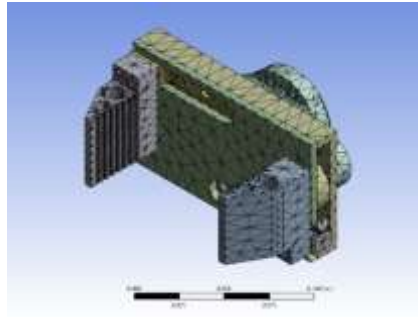


Figure 0-8 Gripper meshing

6. Boundary Conditions and Loads

The boundary conditions and load applications were defined as follows:

- **Fixed supports:** The base of the fingers, where they attach to the gripper mechanism, was fixed.
- **Load application:** A torque of 10 kg-cm was converted to a linear force applied at the tip of each finger. Given the length of the fingers (100 mm), this translated to a force of 10 N applied at the tip to generate the required torque.

7. Simulation Results Stress Distribution

The maximum Von Mises stress observed in the fingers was 45 MPa, located near the fixed support where the fingers attach to the gripper mechanism.

Deformation

The maximum deformation occurred at the tip of the fingers, measuring 0.5 mm. This deformation is within acceptable limits for the gripper's operational requirements.

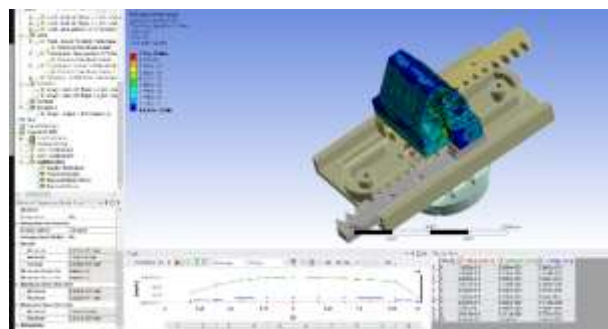


Figure 0-9 Gripper meshing

8. Factor of Safety

The Factor of Safety (FoS) was calculated using the maximum Von Mises stress and the tensile strength of PLA:

$$\text{FoS} = \frac{\text{Tensile Strength of PLA}}{\text{Maximum Von Mises Stress}} = \frac{50 \text{ MPa}}{45 \text{ MPa}} \approx 1.11$$

9. Discussion

The simulation results indicate that the 3D printed PLA fingers can sustain the applied load of 10 kg·cm with a FoS of approximately 1.11. Although this suggests that the design is marginally safe, it is recommended to consider increasing the finger dimensions or using a higher strength material to improve the safety margin. Additionally, the influence of 3D printing anisotropy and layer adhesion should be taken into account, as these factors can affect the mechanical properties of PLA.

10. Conclusion

The ANSYS simulation demonstrates that the 3D printed PLA fingers on the parallel gripper are capable of withstanding a torque of 10 kg·cm with a FoS of 1.11. However, for enhanced safety and reliability, design modifications or material improvements are recommended. This study provides a foundational analysis for further optimization of 3D printed robotic gripper components.

Chapter 5: Experimental Results and Analysis

5.11 Overview of Findings

This thesis proposes a design for an automated parcel sorting system using a robotic arm and advanced computer vision, accomplished through the application of the YOLOv8 model during various tests and evaluations. Such integration is going to improve the speed and efficiency even more, using methods in this model like automated sorting, cost-effective belt optimization, and package handling accuracy. As seen from the evaluation, the enormous rise in detection accuracy and operational speed of the system versus manual operations is highly recommended

5.12 Key Results

The following were some of the main results that came to be after the implementation of this project:

High performance in detection: The YOLOv8 model achieved 0.85 IOU after training on a custom dataset of images of packages. An IOU of 0.85 implies that a much better localization of a parcel could be done, which is important for sorting.

Real-Time Processing Efficiency: The system, in this case, managed to work out the processing of video frames at an average rate of 30 frames per second. Therefore, it should be supported by real-time processing to continuously and dynamically monitor parcels running on the conveyor belt, which is crucial for maintaining a high throughput rate.

Picking Efficiency:

The robot picks objects from the pre-determined place, and objects are detected by an industrial IR sensor which is highly accurate and stops the object constantly at a single point. The manipulator has no slippage hence it is highly accurate in reaching and staying in positions. The only thing that can cause an error is a gripper. Gripper opening and closing depend upon voltage, current, and any mechanical fault. Voltage and current were fixed after finding the best parameters. Mechanical smoothness was also improved in the new version. Once

everything was set up, then it picked the object with full accuracy till there was an intervention and system calibration was changed.

5.13 Limitations and Challenges

However, several limitations and challenges have been identified despite the successful outcomes: False positives and negatives: A few false positive detections, such as when a wrong-shaped object was detected. There were some false negatives where shapes were missed, which were mostly under difficult lighting conditions. This lead to false categorization identity.

Robot pick and place is synchronized with conveyor, so when robot reaches the pick-up position and opens the gripper, there must be an object to pick, otherwise robot will execute motion without picking any object, that will decrease the efficiency of Robot.

5.14 Future Work

Although certainly a great improvement over traditional sorts, the current system is far from ideal, and potential future work exists in several other areas. There are a lot of options available in Robot integration with Computer vision, which can convert simple Robot into Vision Robot. If communication can be developed between the Robot and CV output (through serial communication), vision robotics can be achieved. This will eliminate the use of IR sensors and will open many opportunities.

In PLC, options are available to implement SCADA and HMI which is a supervisory system for the PLC network and Human Machine Interface monitors the whole system and controls various parameters for processes.

ANNEXURE A: PROGRAMMING

```
from ultralytics import YOLO
import cv2
import numpy as np

segmentation_model = YOLO(r"C:\Users\azeem\OneDrive\Pictures\best.pt")

cap = cv2.VideoCapture(0)

area_threshold = 1000

# Define the dimensions and position of the conveyor belt
conveyor_belt_width = 200
conveyor_belt_height = 50
conveyor_belt_x = 0 # Top left corner of the frame
conveyor_belt_y = 0 # Top left corner of the frame

while True:
    # Capture frame-by-frame
    ret, frame = cap.read()
    if not ret:
        break

    # Perform segmentation on the frame
    results = segmentation_model.predict(source=frame.copy(), save=False,
        save_txt=False, stream=True, conf=0.6, box=True, show_labels=True)

    # Initialize variables to store areas of segmented objects
    large_areas = 0
    small_areas = 0

    # Iterate over segmented objects
    for result in results:
```

```

if result.masks is not None: # Check if masks are detected
    masks = result.masks.data
    boxes = result.bboxes.xyxy[0]

# Iterate over each segmented object
for mask, box in zip(masks, boxes):
    # Convert PyTorch tensor to NumPy array
    mask_np = mask.cpu().numpy()

    # Calculate area (number of non-zero pixels)
    area = np.count_nonzero(mask_np)

    # Categorize areas into "large" and "small"
    if area > area_threshold:
        large_areas += 1
    else:
        small_areas += 1

# Resize the segmented mask to match the frame dimensions
mask_resized = cv2.resize(mask_np.astype(np.uint8) * 255,
                           (frame.shape[1], frame.shape[0]))

# Add transparency to the mask
mask_rgba = cv2.merge([mask_resized, mask_resized, mask_resized])

# Add the mask to the frame
frame = cv2.addWeighted(frame, 1, mask_rgba, 0.5, 0)

# Find contours of the mask to calculate angles
contours, _ = cv2.findContours(mask_resized, cv2.RETR_EXTERNAL,
                               cv2.CHAIN_APPROX_SIMPLE)
for contour in contours:
    # Fit a bounding box to the contour
    rect = cv2.minAreaRect(contour)

    # Calculate the centroid of the bounding box
    centroid_x = int((rect[0][0]))
    centroid_y = int((rect[0][1]))

```

```

# Calculate distance from centroid to conveyor belt in x and y
directions
distance_x = centroid_x - conveyor_belt_x - (conveyor_belt_width
/ 2)
distance_y = conveyor_belt_y + conveyor_belt_height -
centroid_y
print ("coordinates:", distance_y,distance_x)

# Calculate the angle of rotation
angle = rect[-1]

# Draw the bounding box
box_points = cv2.boxPoints(rect).astype(np.int0)
cv2.drawContours(frame, [box_points], 0, (0, 255, 0), 2)

# Add text with the angle, area, and position
text = f'Angle: {angle:.1f} Area: {area}px X: {distance_x}, Y:
{distance_y}'
cv2.putText(frame, text, (box_points[0][0], box_points[0][1] -
10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

cv2.imshow('Segmented Frame', frame)

print("Large areas:", large_areas)
print("Small areas:", small_areas)
# Break the loop if 'q' is pressed

if cv2.waitKey(1) & 0xFF == ord('q'):
    break

cap.release()
cv2.destroyAllWindows()

```

G-Code for Robot:

FILE=ST

AM.ST

393

code:

G08 PRINTF "HELLOp&p"

G06 T=200 //Delay

G00 J1=-38.8574 J2=-17.1611 J3=-75.3424 J4=0.1203 J5=-85.1694 J6=72.0316

// PRE PICK POSI

G06 T=200

G00 J1=-42.0166 J2=-38.8706 J3=-99.575 J4=0.0017 J5=-39.6418 J6=29.2921

// PICK POSI

G06 T=1000

G06 O=P1.1

G06 T=2500

G00 J1=-38.8574 J2=-17.1611 J3=-75.3424 J4=0.1203 J5=-85.1694 J6=72.0316

// POST PICK POSI

G06 200

G00 J1=-118.7428 J2=0.8332 J3=-89.8618 J4=-0.8529 J5=-67.8331 J6=0

// PRE DROP POSI

G06 T=200

G00 J1=-118.7428 J2=-52.683 J3=-66.4923 J4=0 J5=-60.2085 J6=0

// DROP POSI

G06 T=500

G06 O=P1.0

G06 T=2500

//object Dropped

G00 J1=-118.7428 J2=0.8332 J3=-89.8618 J4=-0.8529 J5=-67.8331 J6=0

// PRE DROP POSI

//PROGRAM END HOME

G06 T=500

REFERENCES

- 1] "conbelt.com," Con-Belt Inc, 15 2 2015. [Online]. Available: <https://www.conbelt.com/types-of-conveyor-belts-used-for-industrial-purposes-3/>. [Accessed 3 6 2024].
 - 2] H. L. S. T. C. Y. X. & Y. J. Zhang, A Real-Time Obstacle Avoidance Algorithm for Mobile Robots Based on Improved Rapidly-Exploring Random Tree (RRT) Method., 2018.
 - 3] "Starship Technologies," [Online]. Available: <https://www.starship.xyz/>.
 - 4] "KiwiBot," [Online]. Available: <https://www.kiwibot.com/>.
 - 5] "Nuro," [Online]. Available: <https://nuro.ai/>.
 - 6] T. T. M. S. A. a. S. T. M. M. Khan, "Autonomous Delivery Robots: A Comprehensive Review," 2020.
 - 7] M. B. W. B. A. B. C. F. D. D. F. D. H. C. R. N. R. J. S. e. a. S. Thrun, "MINERVA: A Second-Generation Mobile Tour-Guide Robot," *IEEE Robotics & Automation Magazine*, vol. 11, no. 2, pp. 97-108, 2004.
 - 8] RoboPeak, "RP Lidar A1M8 Datasheet," [Online]. Available: <https://www.robotshop.com/media/files/pdf/rplidar-a1m8-lidar-sensor.pdf>.
 - 9] A. O. a. L. P. Fernando Sancho, "Design and Development of a Low-cost Autonomous Delivery Robot for Indoor Enviroments," 2018.
 - 10] "TurtleBot," 2010. [Online]. Available: <https://www.turtlebot.com/>.
 - 11] S. B. M. B. W. C. A. B. D. F. F. D. .. & S. J. Thrun, "MINERVA: A Second-Generation Mobile Tour-Guide Robot," *IEEE Robotics & Automation Magazine*, pp. 11(2), 97-108, 2004.
 - 12] Z. Q. W. W. H. & Z. X. Deng, "A Survey of Simultaneous Localization and Mapping with Robotic Sensors," *Vols. Sensors*, 21(1), 213. doi:10.3390/s21010213, 2021.
 - 13] E. L. F. M. A. M. F. C. M. M. F. .. & M. P. Vallecorsa, "The FPGA-based data acquisition system for the upgrade of the CMS hadron calorimeter.," *Journal of Instrumentation*, pp. 13(01), C01029-C01029, 2018 .
 - 14] "Raspberry Pi 4," [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>.
 - 15] "What is Lidar," [Online]. Available: <https://www.synopsys.com/glossary/what-is-lidar.html>.
 - 16] "About Ubuntu," [Online]. Available: <https://ubuntu.com/about>.
 - 17] "Ubuntu for raspberry pi," [Online]. Available: <https://ubuntu.com/raspberry-pi>.
- "About ROS," [Online]. Available: <https://www.ros.org/>.

- 18] "ROS Noetic Ninjemys," [Online]. Available: <https://wiki.ros.org/noetic>.
- 19] "Getting started with gazebo and ros noetic," [Online]. Available:
 20] <https://automaticaddison.com/getting-started-with-gazebo-in-ros-noetic>.
 "Introduction and use of Rviz," [Online]. Available:
 21] <https://docs.elephantrobotics.com/docs/myarm-pi-300-en/12-ApplicationBaseROS/12.1-ROS1/12.1.4-rivz%E4%BB%8B%E7%BB%8D%E5%8F%8A%E4%BD%BF%E7%94%A8/#:~:text=rviz%20is%20a%203D%20visualization,and%20control%20of%20a%20robot..>
 Wikipedia, "Robot Operating System," [Online]. Available:
 22] https://en.wikipedia.org/wiki/Robot_Operating_System.
 "SLAM," [Online]. Available:
 23] https://en.wikipedia.org/wiki/Simultaneous_localization_and_mapping.
 "Types of SLAM Algorithms," [Online]. Available:
 24] <https://medium.com/@olesyagrindach/testing-different-slam-algorithms-with-turtlebot3-simulation-34c741db96ea>.
 "Amazon robotic fulfillment center," [Online]. Available:
 25] <https://www.waredock.com/magazine/what-is-amazon-robotic-fulfillment-center/>.
 "Arduino Mega 2560," [Online]. Available: <https://store.arduino.cc/products/arduino-mega-2560-rev3>.
 26] "SLAM Topics," [Online]. Available: <https://github.com/topics/slam>.
 27] "What is URDF?," [Online]. Available: <https://formant.io/urdf/>.
 28] "Teleop Twist Keyboard," [Online]. Available: https://github.com/ros-teleop/teleop_twist_keyboard.
 29] P. B. A. L. R. V. Alessandro Gasparetto, "Path Planning and Trajectory Planning Algorithms: A General Overview," 2015.
 30] M. B. M. A. U. a. G. O. U. Zaharuddeen Haruna, "Path Planning Algorithms for Mobile Robots: A Survey," November 2023. [Online]. Available:
 31] <https://www.intechopen.com/chapters/1154152>.
 Y. e. a. Jiang, "Exploring the factors that drive consumers to use contactless delivery services in the context of the continued COVID-19 pandemic," *Journal of Retailing and Consumer Services* 72, 1, 2023.
 32] Y. e. a. Jiang, "Exploring the factors that drive consumers to use contactless delivery services in the context of the continued COVID-19 pandemic," *Journal of Retailing and Consumer Services* 72, 1, 2023.
 33] Y. e. a. Jiang, "Exploring the factors that drive consumers to use contactless delivery services in the context of the continued COVID-19 pandemic," *Journal of Retailing and Consumer Services* 72, 1, 2023.
 34] R. e. a. Limosani, "Robotic delivery service in combined outdoor–indoor environments: technical analysis and user evaluation.," *Robotics and autonomous systems* 103, 2, 2018.
 35] R. Bogue, "Growth in e-commerce boosts innovation in the warehouse robot market," *Industrial Robot: An International Journal* 43.6, 3, 2016.
 36] A. Gujarathi, "Design and Development of Autonomous Delivery Robot," arXiv preprint

- 37] arXiv:2103.09229, 4, 2021.
- 38] N. S. F. a. S. S. Boysen, "Last-mile delivery concepts: a survey from an operational research perspective.," *Or Spectrum* 43.1, 5, 2021.
- 39] D. Lee, "Assistive delivery robot application for real-world postal services," *EEE Access* 9, 6, 2021.
- 40] T. a. M. K. Ganokratanaa, "An Intelligent Autonomous Document Mobile Delivery Robot Using Deep Learning," *International Journal of Interactive Mobile Technologies* 16.21 , 7, 2022.
- 41] M. M. S. a. R. B. Schneier, "Literature review of mobile robots for manufacturing," 8, 2015.
- 42] F. Samouh, "Multimodal autonomous last-mile delivery system design and application," *International Smart Cities Conference (ISC2)*. IEEE,, 9, 2020.
- 43] M. a. G. Q. H. Jiang, "Intralogistics synchronization in robotic forward-reserve warehouses for e-commerce last-mile delivery," *Transportation Research Part E: Logistics and Transportation Review* 158, 10, 2022.
- 44] N. U. a. U. A. Miko, "Determinants of efficient last-mile delivery: evidence from health facilities and Kaduna Health Supplies Management Agency," *Journal of Humanitarian Logistics and Supply Chain Management* 14.1, 11, 2024.
- 45] J. A. Angelo, "Robotics: a reference guide to the new technology," Westport: Greenwood Press,, 12, 2007.
- 46] R. a. A. K. Raj, "A comprehensive study of mobile robot: History, developments, applications, and future research perspectives.," *Applied Sciences* 12.14, 13, 2022.
- 47] A. R. I. a. N. A. R. Abdo, "Mobile robot localization evaluations with visual odometry in varying environments using Festo-Robotino. No. 2020-01-1022," *SAE Technical Paper*, 14, 2020.
- 48] T. a. G. P. Hoffmann, "On the regulatory framework for last-mile delivery robots," *Machines* 6.3 , 15, 2018.
- 49] J. T. L. a. M. C. Huang, "Design and evaluation of a rapid programming system for service robots," 2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI). IEEE, 16, 2016.
- 50] D. a. Y. S. Jain, "Adoption of next generation robotics: A case study on Amazon," *Perspect. Case Res. J* 3, 17, 2017.
- 51] M. Groshev, "Edge robotics: Are we ready? An experimental evaluation of current vision and future directions," *Digital Communications and Networks* 9.1, 18, 2023.
- 52] H. M.-L. T. a. Y. C. C. Kabir, "Internet of robotic things for mobile robots: concepts, technologies, challenges, applications, and future directions.," *Digital Communications and Networks*, 19, 2023.
- 53] M. E. Merchant, "The inexorable push for automated production," *Production Engineering* 24 , 20, 1977.
- 54] J. B. a. T. G. G. Dahmus, "An environmental analysis of machining," *ASME international mechanical engineering congress and exposition*. Vol. 47136., 21, 2004.
- 55] A. Garus, "Last-mile delivery by automated droids. Sustainability assessment on a real-world case study," *Sustainable Cities and Society* 79, 22, 2022.
- 56] T. Kim, "Development of an Indoor Delivery Mobile Robot for a Multi-Floor Environment," *IEEE Access*, 23, 2024.

- 57] P. Cohn, "Commercial drones are here: The future of unmanned aerial systems," McKinsey & Company, 24, 2017.
- 58] S. a. Y. N. Trefilov, "Automatic warehouses with transport robots of increased reliability," *Acta Logistica* 5.3, 25, 2018.
- 59] H. Z. Z. a. Y. H. Zeng, "Control system design of an intelligent food delivery robot.," *E3S Web of Conferences*. Vol. 267. EDP Sciences,, 26, 2021.
- 60] A. Xing, "The Development of a Multifunction UGV," Doctoral dissertation, 27, 2023.
- 61] E. Eaton, "Design of a low-cost platform for autonomous mobile service robots," *IJCAI workshop on autonomous mobile service robots.*, 27, 2016.
- 62] W. Wang, "Design and performance analysis of robot shuttle system," 2020 International conference on artificial intelligence and electromechanical automation (AIEA). IEEE, 29, 2020.
- 63] N. A. a. J. K. D. Aspragathos, "A comparative study of three methods for robot kinematics," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 28.2, 30, 1998.
- 64] P. F. a. C. P. N. Muir, "Kinematic modeling of wheeled mobile robots," *Journal of robotic systems* 4.2, 31, 1987.
- 65] A. Mandow, "Experimental kinematics for wheeled skid-steer mobile robots.," *IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 32, 2007.
- 66] G. Haddeler, "The Analysis of Discrete-Event System in Autonomous Package Delivery using Legged Robot and Conveyor Belt.," *arXiv preprint arXiv:2101.12347*, 33, 2021.
- 67] Y. Wi, "Robot Manipulator Assisted Tool Delivery System," *Diss. University of Houston*, 34, 2021.
- 68] A. Kebritchi, "Design and Development of an Omnidirectional Mobile Manipulator for Indoor Environmen," 2018 6th RSI International Conference on Robotics and Mechatronics (IcRoM). IEEE,, 35, 2018.
- 69] C. Loizos, "Ground delivery robots: Passing fancy or next wave?," *TechCrunch*, 10 April 2016. [Online]. Available: [avIdUnPfd5vXsfxZiELELAUgQ9vCjF6W5TSi1jjWslJzt7VnHy2MGfC7bYYPTcrnqqzrnQvoGbGFQa83ZTgvs7ww3omtqAwYb61A7NaUACIQsUdStuGO43C](https://www.techcrunch.com/2016/04/10/ground-delivery-robots-passing-fancy-or-next-wave/).
- 70] "Urban aerial delivery Electric drone navigates the city with cargo," *Free PIk*, [Online]. Available: https://www.freepik.com/premium-ai-image/urban-aerial-delivery-electric-drone-navigates-city-with-cargo_76881286.htm.
- 71] C. Loizos, "Ground delivery robots: Passing fancy or next wave?," *Tech Crunch*, 10 April 2016. [Online]. Available: <https://techcrunch.com/2016/04/10/ground-delivery-robots-passing-fancy-or-next-wave/>.
- 72] "Urban aerial delivery Electric drone navigates the city with cargo," https://www.freepik.com/premium-ai-image/urban-aerial-delivery-electric-drone-navigates-city-with-cargo_76881286.htm, [Online]. Available: https://www.freepik.com/premium-ai-image/urban-aerial-delivery-electric-drone-navigates-city-with-cargo_76881286.htm.
- 73] S. Els, "Robots need better internet to keep moving," *Locate 2u*, 10 October 2023. [Online]. Available: <https://www.locate2u.com/logistics/robots-need-better-internet-to-keep-moving/>.
- 74] L. HOFFMAN, "Don Tequila in Amherst adds robot server," *The Morning Journal*, 17 May 2024. [Online]. Available: <https://www.morningjournal.com/2023/04/11/don-tequila-in-amherst-adds-robot-server/>.

- 75] Z. Ashiq, "Production facilities," slideshare , 3 Jan 2018. [Online]. Available: <https://www.slideshare.net/ZohaibAshiq/production-facilities-85665839>.
- 76] G. Sanders, "Delivery Robots and Drones," OMDIA, 22 Jun 2020. [Online]. Available: <https://omdia.tech.informa.com/om000849/delivery-robots-and-drones>.
- 77] R. Melissa, "Delivery Robot Market Size to Surge at 33.9% CAGR," STATEON, 7 Oct 2022. [Online]. Available: <https://statzon.com/insights/delivery-robot-market>.
- 78] H. Electronics, "Arduino MEGA 2560 R3 ATMEGA16U2 Programing Board In Pakistan," SMART HALL ROAD, [Online]. Available: <https://smarthallroad.com/product/arduino-mega-2560-r3-atmega16u2-programing-board-in-pakistan>.
- 79] D. L. P. S. J. S. Bandar Alghamdi, "An Integrated Robotic System: 2D-Vision based Inspection Robot with," 2017.