AUTOMATION OF PLASTIC WASTE SORTING THROUGH ROBOTIC TECHNOLOGY

CHONG YOONG KIAT

UNIVERSITI TUNKU ABDUL RAHMAN

AUTOMATION OF PLASTIC WASTE SORTING THROUGH ROBOTIC TECHNOLOGY

CHONG YOONG KIAT

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Mechanical Engineering with Honours

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

September 2025

DECLARATION

Name		Chong Yoong Kiat
ID No.	:	21UEB04922
Date	:	22th September 2025

COPYRIGHT STATEMENT

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Dr. Liang Meng Suan, my supervisor, for his invaluable advice, continuous guidance and encouragement throughout the course of this project. His patience and constructive feedback have been instrumental in ensuring the successful completion of this work.

My sincere appreciation also goes to Dr. Ting Chen Hunt, my co-supervisor, for his generous support and assistance. In particular, I am grateful to him for lending the Delta X robot, which served as a critical component of this project. His technical insights and suggestions greatly contributed to the development of the system.

I would also like to extend my thanks to the Lee Kong Chian Faculty of Engineering and Science (LKC FES), Universiti Tunku Abdul Rahman, for providing access to laboratory facilities. The opportunity to use a dedicated lab room for setting up and testing the prototype allowed me to carry out the fabrication and experimentation effectively.

Finally, I would like to acknowledge my family and friends for their constant encouragement, understanding and moral support during the entire duration of this project.

ABSTRACT

This project presents the design, development, fabrication and evaluation of an automated waste sorting system integrating computer vision, robotic actuation and electronic control. The primary objective was to automate the classification and segregation of recyclable waste to improve accuracy and efficiency compared to manual sorting. The methodology involved fabricating a conveyor belt system, designing a slider mechanism to extend the Delta X robotic arm's reach and equipping the robot with a vacuum gripper for pick-and-place operations. A YOLOv8 deep learning model, trained on a custom dataset of waste images, was integrated with the ByteTrack algorithm to provide real-time object detection and tracking. An ESP32 microcontroller and a Python-based GUI coordinated the conveyor, slider, robot arm and vision subsystems for seamless operation. Experimental testing demonstrated high detection accuracies of 100% for aluminium, 96% for plastics and 94% for paper. Pick-and-place success rates were 92% for aluminium, 98% for paper and 48% for plastics, the latter being affected by transparency, irregular surfaces and limitations of the IR sensor. The overall throughput achieved was 8 - 15 items per minute, with reliable continuous operation over 20 minutes, though positional drift of the robot arm and slider was observed due to the lack of feedback mechanisms. These results indicate that the prototype successfully met its objectives, demonstrating the feasibility of low-cost AI-enabled robotic sorting.

Keywords: waste sorting; YOLOv8; machine vision; robotic arm; automation; object detection

Subject Area: Robotics

TABLE OF CONTENTS

DECLARATION **i**
ACKNOWLEDGEMENTS **iii**
ABSTRACT **iv**
TABLE OF CONTENTS **v**
LIST OF TABLES **vii**
LIST OF FIGURES **ix**
LIST OF SYMBOLS / ABBREVIATIONS **xi**
LIST OF APPENDICES **xii**

CHAPTER

1 **INTRODUCTION** **1**
1.1 General Introduction 1
1.2 Importance of the Study 2
1.3 Problem Statement 2
1.4 Aim and Objectives 3
1.5 Scope and Limitation of the Study 4
1.6 Contribution of the Study 4
1.7 Outline of the Report 5

2 **LITERATURE REVIEW** **6**
2.1 Introduction 6
2.2 Robot 6
2.2.1 Robot Arm Type 6
2.2.2 Gripper Types 10
2.3 Motor 14
2.4 Object Detection Model 17
2.5 Object Tracking Algorithms 20
2.6 Summary 23

3 **METHODOLOGY AND WORK PLAN** **24**
3.1 Introduction 24

3.2 System Overview 24
3.3 Fabrication of the Conveyor System 25
3.3.1 Conceptual Design of Conveyor System 26
3.3.2 Fabrication of Conveyor Belt 27
3.4 Fabrication of the Slider 28
3.4.1 Conceptual Design of Slider 28
3.4.2 Fabrication of Slider 29
3.5 Machine Vision 30
3.5.1 Dataset Collection 30
3.5.2 Vision Training 31
3.5.3 Object Tracking Algorithm 32
3.6 System Assembly and Integration 33
3.6.1 Assembly of Mechanical Components 33
3.6.2 Wiring and Electronics Setup 34
3.6.3 System Coding and Control Flow 35
3.7 Work Plan 36
3.7.1 Gantt Chart 36
3.7.2 Milestones and Deliverables 37
4 RESULTS AND DISCUSSION 39
4.1 Introduction 39
4.2 Performance of Fabricated Parts 39
4.2.1 Conveyor Belt Performance 39
4.2.2 Slider Mechanism Performance 40
4.2.3 Delta X Robot Arm Performance 40
4.3 Performance of Machine Vision 43
4.4 System Testing and Evaluation 46
4.4.1 Sorting Accuracy 46
4.4.2 Picking Speed 48
4.4.3 Reliability Testing 48
4.5 Discussion of Findings 49
5 CONCLUSIONS AND RECOMMENDATIONS 50
5.1 Conclusions 50
5.2 Recommendations for future work 51
REFERENCES 52

APPENDICES **55**

LIST OF TABLES

Table 2.1:	Scoring Matrix for Selection of Robot Arm	10
Table 2.2:	Scoring Matrix for Selection of Gripper	13
Table 2.3:	Scoring Matrix for Selection of Motor for Conveyor System	16
Table 2.4:	Scoring Matrix for Selection of Motor for Slider	16
Table 2.5:	Scoring Matrix for Selection of Object Detection Model	19
Table 2.6:	Scoring Matrix for Selection of Object Tracking Algorithms	22
Table 3.1:	Electrical Components of Conveyor System	26
Table 3.2:	Electrical Components of Slider	29
Table 3.3:	Waste Material Types in Dataset Images	31
Table 4.1:	Gripping Performance Test of Vacuum Gripper	41
Table A.1:	Material Properties Used in Simulation	55

LIST OF FIGURES

Figure 1.1: Landfill at Teluk Mengkudu, Perak 1

Figure 1.2: Manual Waste Sorting Process 2

Figure 2.1: Picture of Delta Robot (DeltaX Robot, 2020). 7

Figure 2.2: Picture of SCARA Robot 8

Figure 2.3: Picture of Cartesian Robot 8

Figure 2.4: Picture of Articulated Robot 9

Figure 2.5: Picture of Vacuum Grippers 11

Figure 2.6: Picture of Mechanical Grippers 12

Figure 2.7: Picture of Adhesive Grippers 12

Figure 2.8: Picture of Electrostatic Grippers 13

Figure 2.9: Picture of 775 DC Motor 14

Figure 2.10: Picture of JGB37-545 Gear Motor 15

Figure 2.11: Picture of NEMA 23 Stepper Motor 15

Figure 2.12: Mechanism of YOLO 18

Figure 3.1: Schematic Diagram of Waste Sorting Machine 25

Figure 3.2: Picture of Conveyor Belt 28

Figure 3.3: Picture of Slider 30

Figure 3.4: Assembly of Conveyor Belt and Frame of Delta X Robot Arm 33

Figure 3.5: Picture of IR Sensor 34

Figure 3.6: Picture of Dropping Boxes 34

Figure 3.7: Placement of Power Supply, Motor Controller and ESP32 35

Figure 3.8: Graphical User Interface 36

Figure 3.9: Gantt Chart for FYP 1 37

Figure 3.10: Gantt Chart for FYP 2 37

Figure 4.1: Picture of Thin and Thick Aluminium Cans 42

Figure 4.2: Training and Validation Loss Curve 43

Figure 4.3: Evaluation Metrics Curve 44

Figure 4.4: Normalized Confusion Matrix 45

Figure 4.5: Real-time GUI Feed with Assigned Label 47

Figure A.1: Displacement Distribution of the Motor Bracket 55

Figure A.2: Von Mises Stress Distribution of the Motor Bracket 56

Figure A.3: Slider Shafts with Different Loading Scenarios 57

Figure A.4: Displacement Distribution of the Slider Shaft A 58

Figure A.5: Von Mises Stress Distribution of the Slider Shaft A 58

Figure A.6: Displacement Distribution of the Slider Shaft B 59

Figure A.7: Von Mises Stress Distribution of the Slider Shaft B 59

LIST OF SYMBOLS / ABBREVIATIONS

CNN	Convolutional Neural Network
CVAT	Computer Vision Annotation Tool
DOF	Degrees Of Freedom
FYP	Final Year Project
IoU	Intersection Over Union
KCF	Kernelized Correlation Filter
mAP	Mean Average Precision
NMS	Non-Maximum Suppression
R-CNN	Region-based Convolutional Neural Network
SCARA	Selective Compliance Assembly Robot Arm
SORT	Simple Online and Realtime Tracking
SSD	Single-Shot Detector
SMPS	Switched Mode Power Supply
SVM	Support Vector Machine
YOLO	You-Only-Look-Once

LIST OF APPENDICES

Appendix A: Static Simulation of Motor Bracket 55

Appendix B: Static Simulation of Slider Shaft 57

Appendix C: Full Code 60

CHAPTER 1

INTRODUCTION

1.1 General Introduction

In Malaysia, daily municipal solid waste generation has increased rapidly from 25,000 tonnes in 2013 to approximately 39,000 tonnes in 2024. The rapid rise in waste production is driven primarily by population growth, lifestyle changes and rapid urbanization (International Trade Administration, 2024). The import of waste, which weight to over 68,000 tonnes in 2022, has only worsened this situation (World Bank. org, 2022). This leads to clogged drains, urban flooding and escalating greenhouse gas emissions from anaerobic decomposition in overflowing landfills (Chuah et al., 2023).

Plastic, paper and aluminium, materials that people use almost every day, contribute 21.9 %, 15.3 % and 6 % of total municipal waste respectively (Zainal, 2024). Although these wastes are highly recyclable, Zainal (2024) reported that Malaysia's national recycling rate stood at just 35.38 % in 2023, with only 24 % of plastic waste produced in 2019 being recycled. The remainder is sent to landfills or mismanaged disposal sites. Figure 1.1 shows the landfill at Teluk Mengkudu, Perak. International Trade Administration (2024) warns that if waste generation continues at this alarming rate, Malaysia will exhaust its landfill capacity by 2050.

Figure 1.1: Landfill at Teluk Mengkudu, Perak

To mitigate the issue, the government has introduced measures such as the RM0.20 plastic bag in 2017, the "Roadmap Towards Zero Single Use

Plastics 2018–2030" and widespread public awareness campaigns. However, mixed collection systems and reliance on manual segregation at Materials Recovery Facilities mean that waste often arrives commingled. Manual sorting is slow, inefficient, prone to contamination and exposes workers to physical and health risks (Dodampegama et al., 2024). Hence, implementation of automated waste sorting system that combine machine vision for waste identification with robot arms for waste separation can improve throughput, reduce contamination and enhance worker safety.

1.2 Importance of the Study

Waste sorting is the critical link between waste collection and remanufacturing. If collected waste is not accurately separated, downstream processes suffer from contamination, degraded material quality and increased remanufacturing costs. Figure 1.2 shows the waste-sorting process performed by human labor. In Malaysia, poor waste sorting capabilities result in significant quantities of valuable recyclables being lost to landfills. By integrating machine vision technologies with robotic arms, the identification and separation of waste materials such as plastics, paper and aluminium can be automated. This automation not only improves sorting accuracy and consistency but also reduces reliance on manual labour, minimizing workers' exposure to unhygienic and hazardous environments while lowering operational manpower costs.

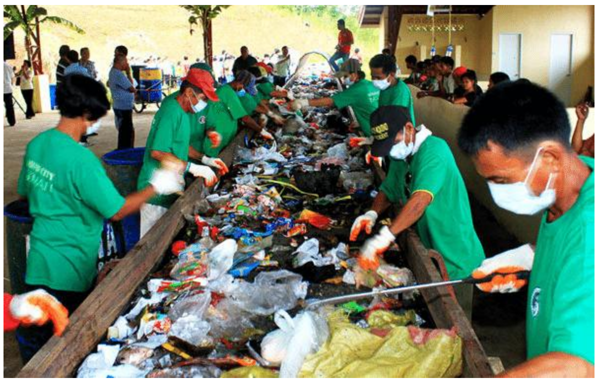


Figure 1.2: Manual Waste Sorting Process

1.3 Problem Statement

In Malaysia, the current waste sorting process at Materials Recovery Facilities depends almost entirely on manual labour, resulting in low efficiencies and inconsistent separation of recyclables. Workers in the recycling sector earn an average of MYR 38,081 per year (approximately MYR 18 per hour), despite performing physically demanding and hazardous tasks in cramped, unsanitary conditions. On average, human pickers can sort only 20 - 40 waste per minute and sorting accuracy declines rapidly as fatigue sets in, necessitating time consuming validation and re-sorting process. Moreover, prolonged manual handling without adequate training or protective equipment exposes workers to elevated risks of musculoskeletal injuries and cuts from sharp or contaminated materials.

Implementation of machine-vision and robotic systems can automate the waste sorting process. Advanced waste sorting solutions are already available in other countries. For instance, AMP Robotics has offered AI-powered waste sortation services, which increase sorting efficiency and reduce human involvement. However, high purchase fees, taxes, delivery fees and maintenance fees remain barriers. Furthermore, these imported systems are often optimized for waste compositions in Europe or North America, making them less effective at processing Malaysia’s unique mix of plastics, paper and aluminium. Hence, an automated sorting system that integrates machine vision with a robotic arm on a conveyor system that designed locally can tackle such issues, effectively reduce the system implementation costs.

1.4 Aim and Objectives

This project aims to design and manufacture an automated waste sorting system that capable of identifying and segregating different types of wastes on a conveyor belt. The types of waste are plastic (Plastic Bottle), paper (Beverage Carton) and aluminium (Aluminium Can). The following objectives are to be completed to achieve the aim:

i. To develop a robotic system equipped with computer vision for accurate detection and classification of various waste types.

ii. To automate the sorting process in order to ensure consistent and accurate waste segregation.

iii. To evaluate the system’s performance by measuring sorting accuracy, processing speed and reliability.

1.5 Scope and Limitation of the Study

This project's scope encompasses the design and manufacture of an automated waste sorting system for recycling facilities in Malaysia. Deep learning models will be trained to identify and classify plastics, paper and aluminium waste based on visual features such as size, shape and colour. An object tracking algorithm will then track and assign a specific ID to each identified waste. A conveyor belt will transport waste and its speed will be adjusted dynamically according to the number of wastes in the working range to maximize picking efficiency. A linear slider will extend the robot arm's horizontal reach across the full width of the conveyor, ensuring complete coverage and increased picking speed. Finally, a sensor that installed on the gripper will confirm successful grasp of each piece of waste.

There are a few of limitation for the project. First, the deep learning algorithm is trained on a dataset limited to three materials, which are plastic, paper and aluminium can. This may restrict its ability to generalize to other common waste types such as glass or rubber. Second, this study employs a small-scale neural network architecture for rapid prototyping. Thus, scaling up to a production grade model will require substantially more computational resources and may incur longer inference times, potentially affecting real time performance. Third, the current prototype has been designed on a small scale, suitable for demonstration purposes. This limited scale may not effectively handle large volumes of waste or operate efficiently in industrial settings where higher throughput is required.

1.6 Contribution of the Study

This study contributes to the advancement of low-cost automated waste sorting by integrating deep-learning based computer vision with robotic and mechanical actuation into a functional prototype. It demonstrates the feasibility of using YOLOv8 combined with the ByteTrack tracking algorithm for real-time detection and classification of common waste types, achieving high levels of accuracy in distinguishing aluminium, plastic and paper. The development of a conveyor, slider and robot arm system using commercially available components, highlights a practical and scalable approach suitable for small-scale waste management applications. Furthermore, the project provides experimental data and analysis on the challenges of handling irregular and transparent waste items, contributing to the broader understanding of limitations in current automated sorting technologies. Overall, the findings of this study serve as a reference for future research in developing sustainable, affordable and intelligent waste sorting solutions.

1.7 Outline of the Report

This report is organized into five main chapters. Chapter 1 introduces the background, problem statement, objectives, scope, contributions and overall structure of the project. Chapter 2 presents a literature review on related topics, including robotic arms, gripper types, motors, object detection models and object tracking algorithms. Chapter 3 explains the methodology and work plan, covering system overview, fabrication of the conveyor and slider, development of the machine vision system, integration of hardware and electronics and coding workflow. Chapter 4 discusses the results and findings, including the performance of the fabricated parts, evaluation of machine vision, system testing and comparative analysis with related works. Finally, Chapter 5 provides conclusions, challenges encountered and recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter presents a comprehensive review of existing technologies and research relevant to the development of an automated plastic waste sorting system. Various types of robotic arms, including Delta, SCARA, Cartesian and Articulated arms, equipped gripper with different gripping mechanisms, are studied and compared to determine the most suitable component for the system. Additionally, different types of motors are evaluated for driving both the conveyor belt and the slider system.

On the software side, this chapter examines machine vision, focusing on deep learning models such as YOLO, SSD and RCNN for real time waste identification and object tracking algorithms such as including KCF, SORT, Deep SORT and ByteTrack for identified waste tracking. The goal of this literature review is to provide the justification for the design choices made in this project.

2.2 Robot

Robots are programmable, actuated mechanisms with varying degrees of autonomy that perform tasks in response to internal commands and external stimuli. According to the international standard ISO 8373, an industrial robot is defined as “a programmed actuated mechanism with a degree of autonomy to perform locomotion, manipulation or positioning” (ISO, 2021). These systems typically exhibit multiple degrees of freedom, allowing precise, repeatable motions that surpass human endurance and accuracy.

In this project, a robot arm equipped with gripper is mounted above a conveyor to automate the pick and place routine of wastes.

2.2.1 Robot Arm Type

In industrial automation, several types of robotic arms are available, each with its own unique mechanical structure, range of motion and application suitability. Among the most widely used types are Delta, SCARA, Cartesian and

Articulated robot arms (Techman Robot, 2023). For lightweight waste sorting applications, where high speed and precision are essential but heavy payload capacity is not, a careful comparison of these arm types is necessary to determine the most effective solution.

Delta robot arms, also known as parallel robots, are specially designed for high speed pick and place tasks. They feature three lightweight arms connected to a common base, with actuators mounted at the top. This design allows for minimal moving mass and extremely fast acceleration and deceleration. As a result, Delta robots, such as the Delta X1 from DeltaX Robotics, can achieve max speed of 700 mm/s, with typical payload capacity of up to 0.5 kg, which is more than sufficient for lightweight materials such as plastic bottles (DeltaX Robot, 2020). Due to their high-speed capabilities, Delta robots are commonly used in food packaging, pharmaceutical sorting and other industries that demand rapid material handling (Robots Done Right, 2025). Their workspace is typically dome-shaped, which can be limiting in range, but can be overcome with the integration of a linear slider, as proposed in this project. Figure 2.1 shows the picture of Delta X1 from DeltaX Robotics.

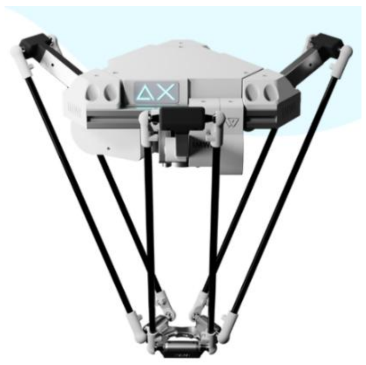


Figure 2.1: Picture of Delta Robot (DeltaX Robot, 2020).

Selective Compliance Articulated Robot Arm (SCARA) are another fast and accurate option used in industrial applications. They operate mainly in the X-Y plane with limited vertical movement, making them ideal for pick-and-place tasks (Standard Bots, 2025). As shown in Figure 2.2, SCARA robots typically have four degrees of freedom (DOF): three rotational movements and one vertical movement. (Flexi Bowl, 2014). However, their range of motion is more constrained compared to Delta robots and their speed is generally slightly lower due to their heavier mechanical arms (Robots Done Right, 2022). While suitable for moderately fast sorting tasks, SCARA robots may not meet the highest speed requirements of waste sorting operations where materials are rapidly moving on a conveyor.

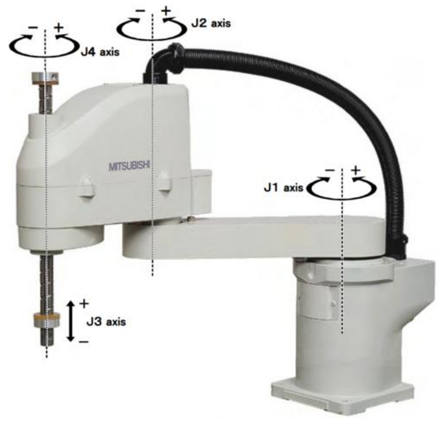


Figure 2.2: Picture of SCARA Robot

Cartesian robots, also known as gantry robots, use three linear actuators aligned with the X, Y and Z axes to provide precise motion in a rectangular workspace as shown in Figure 2.3. These systems can handle very large payloads and are relatively easy to program due to their orthogonal structure (Standard Bots, 2025). They are excellent for tasks that require heavy lifting or large working areas. However, Cartesian robots are generally slower than Delta and SCARA robots and their large moving masses make them less suitable for rapid pick-and-place operations with lightweight objects.

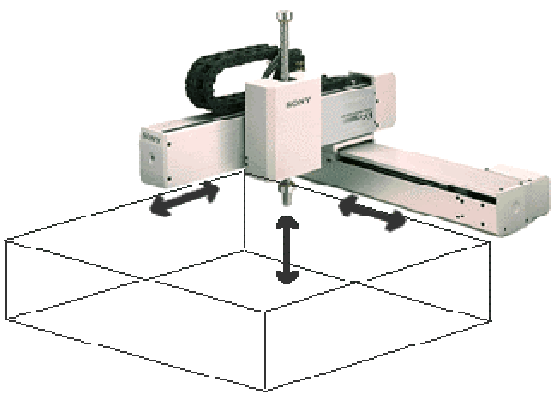


Figure 2.3: Picture of Cartesian Robot

Articulated robots, which resemble a human arm with multiple rotary joints (typically 5–7 degrees of freedom), offer the most flexible movement. They are widely used in complex tasks such as welding, painting and part manipulation that require versatile orientation (Balluff, n.d.). While these robots can handle high payloads and have broad reach, they are typically slower in operation due to their heavier joints and require more complex programming and safety measures. For high-speed, lightweight sorting tasks like those involved in this project, articulated arms are generally overengineered and less cost-effective compared to Delta or SCARA robot arm. Figure 2.4 shows an example of an articulated robot.

Figure 2.4: Picture of Articulated Robot

To determine the most suitable robot arm type for this lightweight waste sorting system, a scoring matrix is used based on four key criteria, which are speed, precision, payload suitability and ease of integration. Each arm type is rated on a scale from 1 (poor) to 5 (excellent). Table 2.1 shows the scoring matrix for selection of robot arm.

Table 2.1: Scoring Matrix for Selection of Robot Arm

Criteria	Delta	SCARA	Cartesian	Articulated
Speed	5	4	3	3
Precision	4	5	4	3
Suitability for lightweight tasks	5	4	3	3
Simplicity & integration	4	3	3	1
Total Score	**18**	**16**	**13**	**10**

Based on the scoring matrix, the Delta robot arm achieves the highest total score and is therefore selected for this project. Its lightweight construction and exceptional speed make it the most suitable choice for accurately sorting moving plastic waste items on a conveyor system. Furthermore, the addition of a linear slider allows the Delta arm to overcome its workspace limitations, enabling it to handle a wider area without compromising speed. This configuration ensures high efficiency in picking, sorting and placing lightweight recyclable materials such as plastic, paper and aluminium waste.

2.2.2 Gripper Types

Grippers serve as the end effector of a robotic arm, allowing robot to hold, grasp, manipulate, or transport objects (Dorna Robotics, 2023). In automated waste sorting system, where throughput and versatility are paramount, the choice of gripper technology directly influences both the speed of pick-and-place operations and the ability to handle items of varying shape, size and surface properties. Consequently, selecting an appropriate gripper, whether based on suction, mechanical fingers, adhesion, or electroadhesion, is critical to achieving reliable, efficient sorting performance.

Vacuum grippers lift objects by creating negative pressure against the object's surface. It comprises a vacuum source such as an electric vacuum pump or a venturi system that connects to a gripping interface equipped with vacuum suction cups (Granta, 2023). They are widely used for picking and placing, palletizing and depalletizing and loading and unloading (Standard Bot, 2025). Vacuum suction cups can be made of flexible materials so they partly conform to curved or uneven surfaces giving them good performance on rounded bottles and cans. However, porous materials like loose paper can leak air and are harder to pick by suction. Overall, vacuum grippers excel at rapid handling of smooth or partially irregular waste, combining high speed with broad applicability. Figure 2.5 displays a vacuum gripper gripping a paper box with a flat surface.

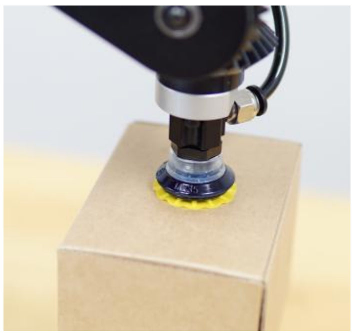


Figure 2.5: Picture of Vacuum Grippers

Mechanical grippers typically use fingers or jaws to grip objects. These grippers can be very fast and reliable. For example, two-finger grippers, such as the example shown in Figure 2.6, are known to operate in high-speed assembly lines (Dorna Robotics, 2023). Such grippers can achieve rapid open/close cycles when handling repeatable, well-defined objects. Multi-finger grippers can conform around irregular shapes. For instance, underactuated finger grippers in bin-picking tasks adjust their fingers to grasp assorted parts (Dorna Robotics, 2023). This adaptability lets them handle items of varying geometry. However, mechanical grippers must often align fingers around the object, which can slow the cycle compared to a single suction action. Mechanical claws may also risk dropping deformable items like thin paper or plastic bag that slip from their grip. In summary, mechanical grippers offer fast cycles and moderate adaptability to irregular waste.

Figure 2.6: Picture of Mechanical Grippers

Adhesive grippers, as shown in Figure 2.7, use sticky or gecko-inspired pads to attach to objects. They can attach to flat or rough surfaces without leaving residue. Since the adhesive relies on shear force instead of normal force or friction, it can handle fragile items gently without applying too much pressure (Stanford University, 2015). This allows exceptionally good handling of irregular or flexible objects. On the downside, adhesives typically require full contact and time to engage or disengage, making their cycle slower. Moreover, adhesive surfaces can lose their effectiveness when dirty or dusty, which is almost certain to occur in a waste handling environment. Overall, adhesive grippers would have moderate picking speed but high adaptability to diverse object shapes

Figure 2.7: Picture of Adhesive Grippers

Electrostatic grippers, as illustrated in Figure 2.8, apply a high-voltage field to attract and hold objects via electroadhesion (Ackerman, 2014). They can rapidly attach or release by switching voltage on/off, enabling quick cycles that are comparable to other high-speed grippers. Electrostatic grippers excel at lightweight, flat objects but they struggle with bulky 3D items or conductive metals. Moreover, curved bottles or crumpled items offer less contact area, leading to reduce in electrostatic gripping force. In summary, electrostatic grippers offer a fast pick rate but limited suitability for 3D irregular shapes.

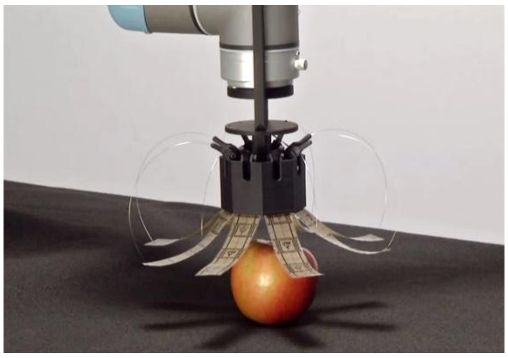


Figure 2.8: Picture of Electrostatic Grippers

To identify the most suitable gripper for this waste sorting system, a scoring matrix was developed using two critical criteria, which are picking speed and the ability to handle irregularly shaped objects. Each gripper type was evaluated on a scale from 1 (poor) to 5 (excellent). Table 2.2 shows the scoring matrix for selection of gripper.

Table 2.2: Scoring Matrix for Selection of Gripper

Criteria	**Vacuum**	**Mechanical**	**Adhesive**	**Electrostatic**
Picking Speed	5	4	3	4
Irregular Object Handling	4	3	5	2
Total Score	9	7	8	6

Vacuum grippers achieve the highest combined performance, reflecting both high cycle rate and broad object compatibility. Other types each have critical drawbacks. For instance, mechanical claws are fast but less adaptable, adhesives gripper grasp many shapes but cycle more slowly and electrostatic gripper grips thin, flat materials only. Thus, a vacuum gripper is the best choice for gripper of the waste sorting system.

2.3 Motor

Motors serve as the driving force of a conveyor system and slider of this waste sorting system. Selecting the right motors for both the conveyor belt and the slider is critical to ensure the automated sorting system meets its performance and reliability targets. For the conveyor, a motor must deliver high rotational speed, sufficient torque to move loaded belts and sustained operation without overheating. For the slider, priorities shift toward precise, repeatable motion, adequate holding torque and moderate travel speed. Three different type of motor, including 775 DC motor, JGB37-545 Gear Motor and NEMA 23 stepper motor will be discussed and evaluated to determine the most suitable motor for both conveyor belt and the slider.

775 DC motor, as shown in Figure 2.9, is a high-performance brushed electric motor renowned for its compact yet powerful design. The motor is optimized for mid to high power applications delivering no-load speeds of approximately 4,100 RPM at 12 V and up to 8,400 RPM at 24 V and stall torque is rated near 0.79 Nm (SM Tech, 2019). Widely used in robotic and hobbyist applications, 775 DC motor offers an excellent speed to size ratio but lacks precise position control, requiring additional encoders or feedback for accurate synchronization with sorting task.

Figure 2.9: Picture of 775 DC Motor

JGB37-545 Gear Motor, as shown in Figure 2.10, is a compact gear motor that combines a 12 V brushed DC motor with an integrated gearbox. It provides a wide range of gear reduction ratios, typically yielding speeds between 6 RPM and 1,000 RPM depending on the version, with torque values up to above 35 kgcm at lower speeds. This makes it highly adaptable for medium-load conveyor systems that require steady, continuous operation. JGB37-545 offers significantly higher torque output, improved durability and smoother performance, making it a practical option for real conveyor applications where both strength and reliability are needed.

Figure 2.10: Picture of JGB37-545 Gear Motor

NEMA 23 stepper motor, as shown in Figure 2.11, provides 1.8° step resolution and a holding torque of approximately 0.6 Nm, delivering precise, repeatable movements without feedback sensors, which is critical for the slider. While its maximum unloaded speed is relatively modest, the use of micro stepping drivers can tailor torque-speed curves to match application needs, ensuring smooth operation under load. NEMA 23's robust construction and constant holding torque make it an excellent choice for positional accuracy. However, its higher cost and the need for more complex drive electronics add to system complexity and increase power demands.

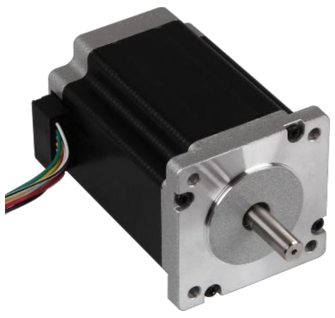


Figure 2.11: Picture of NEMA 23 Stepper Motor

To determine the most suitable motor for the conveyor system and slider, two scoring matrices are developed using two different set of criteria. For the conveyor system, the criteria are speed, torque and continuous duty capability; for the slider, the criteria are positional control, holding torque and travel speed. Each motor was scored from 1 (poor) to 5 (excellent). Table 2.3

and table 2.4 show the scoring matrix for selection of motor for conveyor system and slider.

Table 2.3: Scoring Matrix for Selection of Motor for Conveyor System

Criteria	775 DC Motor	JGB37-545 Gear Motor	NEMA 23 Stepper Motor
Speed	5	4	3
Torque	4	5	4
Continuous Duty Capability	3	5	4
Total Score	**12**	**14**	**11**

The JGB37-545 gear motor scored the highest overall for torque and continuous-duty capability while maintaining a decent rotational speed, resulting in a total score of 14. This reflects its suitability for the conveyor of the waste sorting system. Consequently, the JGB37-545 gear motor is recommended for the conveyor, balancing steady torque output, sufficient speed and reliable continuous operation.

Table 2.4: Scoring Matrix for Selection of Motor for Slider

Criteria	775 DC Motor	JGB37-545 Gear Motor	NEMA 23 Stepper Motor
Positional Control	2	2	5
Holding Torque	4	4	4
Travel Speed	5	4	3
Total Score	**11**	**10**	**12**

NEMA 23 stepper motor achieved top marks in precision control and high holding torque and moderate speed, resulting in total score of 12, which is well above the threshold for reliable, repeatable slider movement. Thus, NEMA 23 stepper motor is recommended for the slider, offering the high accuracy and holding power for consistent pick-and-place operations.

2.4 Object Detection Model

In industrial automation, different object detection model categories offer diverse trade-offs between accuracy, speed and resource requirements. These include single stage and two stage detectors, each with specific strengths and weaknesses, influencing their suitability for various applications. Single stage detectors, such as YOLO and SSD, process the entire image in a single pass. These models are known for their speed and efficiency, making them suitable for real-time applications where quick processing is crucial. They typically achieve a lower accuracy compared to two stage detectors (SharkYun, 2024). On the other hand, two stage detectors like R-CNN employ a separate region-proposal step to boost precision at the expense of increased computational overhead. Selecting the optimal model for conveyor-belt waste sorting thus requires evaluating detection accuracy, speed, hardware cost and ease of implementation.

You-Only-Look-Once (YOLO) is a real-time object detection model widely used across various applications due to its high speed and accuracy. YOLO processes an entire image in a single forward pass of a deep convolutional network, partitioning it into an S×S grid where each cell predicts B bounding boxes along with confidence scores and class probabilities. The confidence score reflects both the probability that an object exists within the box and the predicted intersection over union (IoU) with ground truth. During training, each object is assigned to the bounding box predictor with the highest IoU, encouraging specialization among predictors for different sizes or aspect ratios. At inference time, non-maximum suppression (NMS) prunes overlapping boxes by retaining only the highest-confidence detection per object, reducing false positives and ensuring crisp localization (Joseph et al., 2016). Figure 2.12 summarizes the core steps of how YOLO turns a single image into object detections in real time.

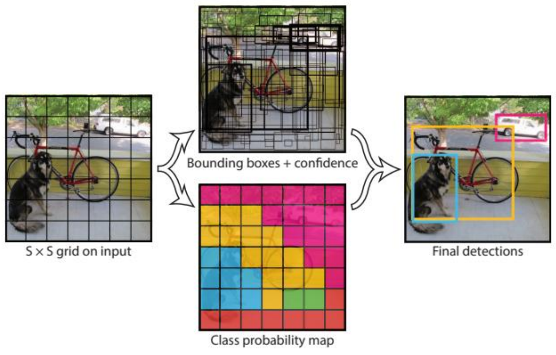


Figure 2.12: Mechanism of YOLO

YOLO is commonly applied in industrial automation for tasks like robotic pick-and-place, quality inspection and waste sorting. It is lightweight, easy to deploy on embedded systems and capable of detecting multiple objects per frame. YOLO still has some drawbacks, including reduced accuracy in detecting small or overlapping objects and reliance on predefined anchor boxes. However, in the newer version, like YOLOv8, many of these issues have been addressed through improvements such as anchor-free detection, better backbone architecture and refined training strategies (docs.ultralytics.com, n.d.).

Single Shot Detector (SSD) is a single stage object detection algorithm that performs object localization and classification in a single forward pass of the neural network, making it significantly faster than two-stage detectors like R-CNN. SSD divides the input image into a grid and generates default bounding boxes of different aspect ratios and scales at each grid location. During inference, SSD predicts both the presence of objects and their class scores for each bounding box. This architecture enables real-time detection with relatively high accuracy (Liu et al., 2016). One of its main advantages is its ability to balance speed and accuracy, especially on medium to large objects. However, it has limitations in detecting small objects, as it relies on lower resolution feature maps for some detections. SSD also depends on predefined anchor boxes, which can reduce flexibility in complex scenarios. Despite these limitations, SSD remains a popular choice for real time applications where inference speed is critical

Regions with Convolutional Neural Network features (R-CNN) is a pioneering two stage detection framework that significantly improved detection accuracy compared to earlier methods. The R-CNN architecture first generates region proposals using an algorithm like selective search, which identifies candidate regions in the image that may contain objects. These proposals are then passed through a convolutional neural network to extract features, which are subsequently classified using a set of Support Vector Machine (SVM) classifiers and bounding boxes are refined using linear regressors. R-CNN is known for its high accuracy and was among the first models to effectively apply deep learning to object detection. It is particularly useful in medical imaging, autonomous driving and surveillance applications where detection precision is crucial. However, R-CNN has notable drawbacks, including slow inference time, high computational cost and complex training, as it requires multiple separate training steps for the CNN, SVMs and bounding box regressors. These limitations led to the development of more efficient successors like Fast R-CNN and Faster R-CNN. Despite this, R-CNN remains foundational in the evolution of object detection models

To determine the most suitable object detection model for this system, a scoring matrix is used based on two key criteria, which are accuracy, speed, Hardware cost and ease of integration. Each object detection model is rated on a scale from 1 (poor) to 5 (excellent). Table 2.5 shows the scoring matrix for selection of object detection model.

Table 2.5: Scoring Matrix for Selection of Object Detection Model

Criteria	**YOLOv8**	**SSD**	**R-CNN**
Accuracy	3	4	5
Speed	4	4	2
Hardware Cost	4	5	2
Ease of Integration	5	2	3
Total Score	**16**	**15**	**12**

Based on the scoring matrix, YOLOv8 achieves the highest total score by achieving an optimal balance across all evaluation criteria. While R-CNN leads in accuracy, its slower inference speed, higher hardware demands and more complex integration requirements make it less suited for a real time waste sorting application. SSD offers strong speed and low hardware cost but falls short in integration ease and marginally in detection accuracy. In contrast, YOLOv8 delivers moderate accuracy with real-time performance, requires only moderate computing resources and integrates seamlessly into existing robotic and vision frameworks. This combination of speed, affordability and low implementation overhead makes YOLOv8 the most practical choice for the waste sorting system.

2.5 Object Tracking Algorithms

Object tracking in a waste sorting system involves linking YOLOv8 detections across frames to maintain consistent object identities for robotic pick-up. Several classical and modern trackers can will be discussed in this part, including KCF, SORT, Deep SORT and ByteTrack.

Kernelized Correlation Filter (KCF) is a single object tracker that uses fast Fourier-domain correlation to predict motion. KCF is extremely fast and it can be implemented in a few lines of code. It generally attains high localization accuracy under stable conditions, but it struggles with abrupt appearance changes or occlusions and does not handle scale variation well (KALRA, 2023). Since it tracks one object per model, KCF must be re-initialized for each new waste item. This complicates multi-object scenarios. In practice, KCF can be attached to YOLO bounding boxes as initial regions of interest, but coordinating many KCF trackers in parallel adds complexity. Overall, KCF’s tracking accuracy is only moderate for a complex scene, but its speed is excellent. It has low computational cost and mature implementations. However, it has virtually no re-identification capability. This means that if a waste item is occluded or leaves and re-enters the scene, KCF will generally lose it.

Simple Online and Realtime Tracking (SORT) is a multi-object tracking framework that builds on YOLO-like detections by applying a Kalman filter for motion prediction and the Hungarian algorithm for bounding box association. (Sanyam, 2022). SORT is deliberately minimalistic, as it uses only the box coordinates and a constant velocity motion model. This makes SORT extremely fast and lightweight. Due to SORT relies entirely on Kalman-filter predictions and IoU-based matching, SORT produces few false positives but suffers frequent ID switches if objects intersect or occlude. Integration with YOLOv8 is straightforward, the tracker simply takes each frame’s detections and outputs persistent track IDs. The computational cost is negligible and SORT scales to many objects in real time. Its main weakness is that it does not handle re-identification. This means that when a waste item is fully occluded or briefly missed, SORT may drop it as a track.

Deep SORT extends SORT by adding a learned appearance model for data association. In practice, Deep SORT extracts a deep “re-ID” feature vector from each detection crop and uses both motion and appearance similarity in the association. This substantially improves ID consistency, as Deep SORT can correctly re-link an object after it reappears and it handles partial occlusion better than pure SORT (Sanyam, 2022). Its tracking accuracy is higher in crowded or complex scenes. However, the extra CNN feature extraction makes Deep SORT heavier. It requires a GPU or powerful CPU and it runs slower than SORT. Deep SORT remains real-time on modern hardware, but its throughput is limited by the embedding network. In terms of computational cost, Deep SORT is significantly higher than SORT because of the neural network. On the positive side, Deep SORT is a mature, well-tested method and offers robustness to occlusion and re-appearance that KCF and SORT lack.

ByteTrack is a recent multi-object tracker that improves on SORT by using both high-confidence and low-confidence detections in its association strategy. Instead of discarding low-confidence detections, as SORT does, ByteTrack assigns them as candidates when IoU matching fails, greatly reducing ID switches and track fragmentation. This makes ByteTrack especially strong in real-world waste sorting, where occlusion, overlapping items and partial visibility are common. ByteTrack is lightweight, runs in real time and integrates seamlessly with YOLOv8. Compared to Deep SORT, it achieves similar or better tracking accuracy while avoiding the computational overhead of an additional embedding network. (Zhang et al., 2021)

To determine the most suitable object tracking algorithms for this system, a scoring matrix is used based on five key criteria, which are accuracy, speed, ease of integration, computational cost and scalability. Each algorithm is rated on a scale from 1 (poor) to 5 (excellent). Table 2.6 shows the scoring matrix for selection of object tracking model.

Table 2.6: Scoring Matrix for Selection of Object Tracking Algorithms

Criteria	**KCF**	**SORT**	**DEEP SORT**	**ByteTrack**
Tracking Accuracy	3	4	5	5
Speed	5	5	3	5
Ease of Integration	3	5	4	5
Computational Cost	5	5	2	4
Resistance to Occlusion	2	3	4	4
Scalability	4	5	3	5
Total Score	**22**	**27**	21	**28**

Based on the evaluation, ByteTrack emerges as the most suitable tracking algorithm for this application. Although Deep SORT offers the highest accuracy, its greater computational demands and slightly slower speed make it less ideal for a high-throughput, resource constrained environment. KCF excels in speed and scalability but falls behind in both tracking precision and resistance to occlusion. By contrast, ByteTrack achieves the best balance of accuracy, real-time speed, low computational cost and robustness against occlusion. Its seamless integration with YOLOv8 makes it ideal for maintaining consistent object identities on a moving conveyor, ensuring reliable robotic pick up in multi object waste sorting scenarios.

2.6 Summary

The literature review for this automated waste sorting system provides insights across mechanical, actuation and machine vision to identify the most effective components. Delta robot is recommended as the optimal choice due to their exceptional cycle times and compact work envelopes. Among end effectors, vacuum grippers provide the best balance of gentle handling and high throughput for lightweight recyclables, outperforming mechanical claws, adhesives and electrostatic solutions in versatility and cycle speed. For motor of conveyor, the JGB37-545 gear motor was selected for the conveyor because of its reliable torque output, continuous duty capability and suitability for medium-load applications., while the NEMA 23 stepper motor offers the precision and holding torque required for the slider’s lateral traverse.

For the machine vision, YOLOv8 stands out as the most practical object detection model, delivering real time inference with strong mean average precision suitable for the varied shapes and sizes of plastics, paper and aluminium. To maintain object identities across fast moving frames, the ByteTrack was chosen, as it combines real-time performance with high robustness to occlusion and identity switching. Its seamless integration with YOLOv8 ensures stable tracking performance under realistic waste sorting conditions.

In conclusion, these components, including Delta robot, vacuum gripper, JGB37-545 gear motor, NEMA 23 stepper, YOLOv8 and ByteTrack, form a high efficiency solution for automated waste sorting system that maximizes throughput, accuracy and system reliability.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

This chapter presents the methodology carried out for the development of the automated waste sorting system, covering the design, fabrication and integration phases. A systematic approach was followed to ensure that each subsystem, including mechanical, electrical and machine vision, was developed in a structured and coordinated manner. A work plan, including a Gantt chart and defined milestones, is presented to illustrate how the project was managed and executed within the allocated 14-week duration.

3.2 System Overview

The automated waste sorting system is designed to integrate mechanical, electrical and machine vision for real-time waste detection and sorting. The workflow begins with the conveyor belt, which continuously transports waste items under the overhead camera. The camera captures real-time video footage and streams it to a laptop, where the YOLOv8 model processes each frame to identify and classify waste objects.

Once identified, the system calculates the coordinates of each waste item on the conveyor. Then, the waste coordinates are transmitted to the Delta X robot, which is mounted on a slider mechanism to extend its working range across the full conveyor width. The robot executes pick-and-place operations based on the received coordinates, controlled via Python programming and G-code instructions. The slider and conveyor subsystems are controlled separately using an ESP32 microcontroller. The conveyor speed dynamically adjusts according to the waste load in order to ensure efficient sorting.

During pickup, a digital infrared (IR) sensor attached to the vacuum gripper detects physical contact between the gripper and the waste item. If contact is confirmed, the system signals the gripper to lift the item and place it into the assigned dropping area. This feedback mechanism reduces failed pickups and ensures reliable handling of waste items with varying heights.

A Graphical User Interface (GUI) provides an interactive platform for system monitoring and control. The GUI displays real-time camera footage, allows the user to start, stop or exit the system and enables assignment of different waste categories to designated dropping areas. The GUI also records the number of picked items.

Overall, the system integrates machine vision (YOLOv8 with ByteTrack), robotic actuation (Delta X robot with vacuum gripper), mechanical actuation (conveyor belt and slider), sensing (IR sensor) and electronic control (ESP32 microcontroller) into a fully functional automated waste sorting machine. Figure 3.1 shows Schematic Diagram of Waste Sorting Machine.

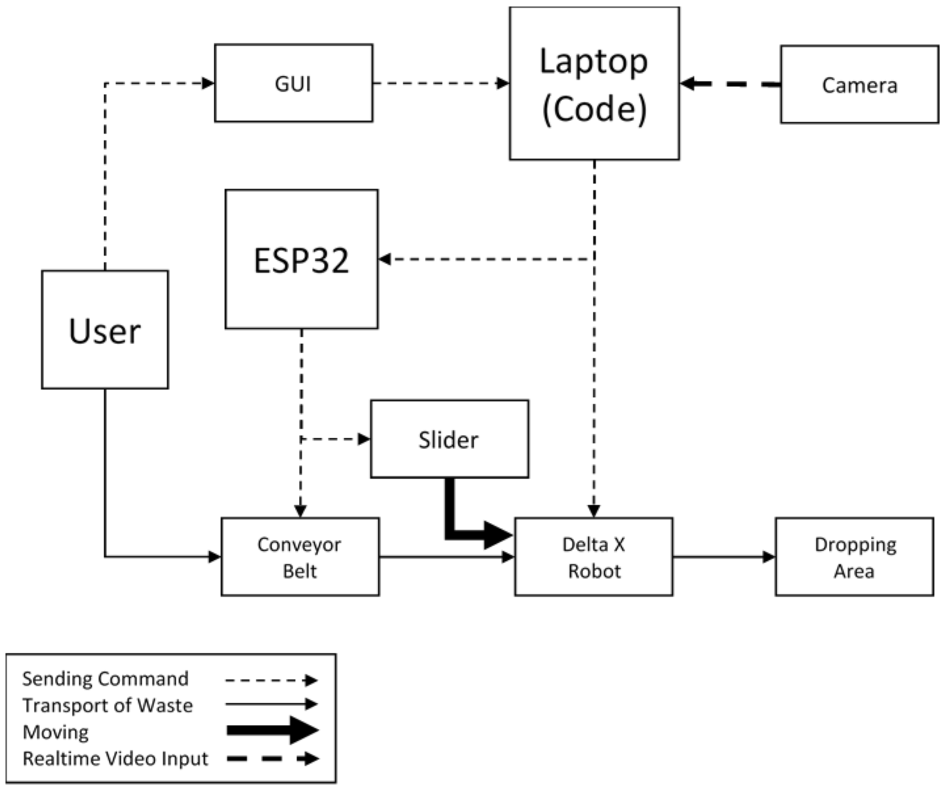


Figure 3.1: Schematic Diagram of Waste Sorting Machine

3.3 Fabrication of the Conveyor System

The conveyor system is designed with a working surface of 0.3m × 1m width, suitable for handling continuous stream of various waste materials. The conceptual design and fabrication process of conveyor system will be discussed in this section.

3.3.1 Conceptual Design of Conveyor System

For this conveyor system, the drive mechanism utilizes a JGB37-545 DC gear motor (200 RPM) connected to the head roller via a pulley system with a 1:5 speed reduction ratio. The motor is controlled by a BTS7960 motor driver, which is interfaced with an ESP32 microcontroller. This setup allows for adjustable control of motor speed and direction, ensuring synchronization of the conveyor movement with other subsystems. Power is supplied by a 12V, 10A switched mode power supply (SMPS).

The conveyor frame is constructed using aluminium profile extrusions (2020 and 2040 types), providing high rigidity, modularity and ease of assembly. A belt tightening mechanism is incorporated to maintain proper tension, reducing slippage and ensuring reliable operation. The motor bracket is designed to allow for custom fitting. To validate the design, static simulation was conducted to verify that the frame stiffness and bracket strength were sufficient to withstand the expected motor torque. The detailed simulation parameters and results are provided in the Appendix A.

The rollers are fabricated from PVC pipes mounted over aluminium cylindrical shafts, while the conveyor deck surface is formed from three layers of plastic corrugated board, offering a lightweight yet durable surface for smooth material transport.

Table 3.1: Electrical Components of Conveyor System

Component	**Name**	**Quantity**	**Picture**
Motor	JGB37-545 gear motor	1	
Motor Controller	BTS7960 Motor Driver	1	

Microprocessor	ESP32	1	
Power Supply	Switched Mode Power Supply	1	

3.3.2 Fabrication of Conveyor Belt

The fabrication of the conveyor system was carried out using machines available at the UTAR workshop, such as the turning machine and drilling machine. The conveyor system was fabricated using aluminium profiles (2040 and 2020) as the main frame structure. The profiles were first measured and marked to the desired lengths and cut using a horizontal bandsaw. After cutting, bolt holes were drilled at the appropriate locations to mount the pillow bearings.

The rollers were manufactured by preparing aluminium shafts through turning process to achieve the required diameter and smooth surface finish. These shafts were then press-fitted into PVC pipes to form lightweight rollers. The conveyor surface was made from corrugated plastic boards, which were measured and cut to the required dimensions using a saw. The motor bracket, designed specifically to fit the selected gear motor, was fabricated using 3D printing. After all mechanical parts were fabricated, the components were assembled using T-nuts, bolts, L-brackets and washers to form a conveyor belt.

A sprocket with 80 teeth was mounted on the head roller, while a smaller sprocket with 16 teeth was mounted on the gear motor shaft. Both sprockets were linked using a GT2 timing belt, providing a reliable 1:5 reduction ratio for smooth torque transfer and speed control of the conveyor. The completed assembly resulted in a fully functional conveyor belt as shown in Figure 3.2.

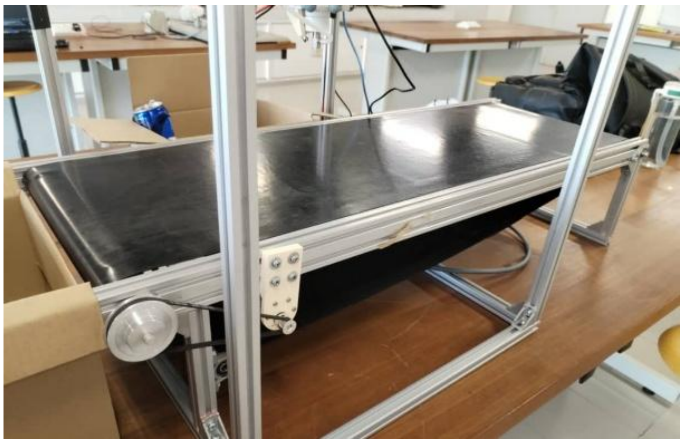


Figure 3.2: Picture of Conveyor Belt

3.4 Fabrication of the Slider

The slider subsystem provides 240 mm of lateral mobility for the robot arm, allowing it to traverse the full width of the conveyor. The conceptual design and fabrication of slider will be discussed in this section.

3.4.1 Conceptual Design of Slider

The slider was designed using two SUS304 stainless steel shafts with a diameter of 10 mm, providing sufficient rigidity and wear resistance for repeated sliding motion. The shafts were supported by four SK10 linear shaft holders, which were mounted onto the frame of the existing Delta X robotic arm.

Linear motion was achieved using SC10UU linear ball bearings installed on the shafts. The Delta X robotic arm was directly mounted onto these linear bearings, enabling smooth horizontal travel along the slider’s span. This design minimized backlash while maintaining a compact profile.

For actuation, the system employed a NEMA 23 stepper motor coupled with a GT2 timing belt and pulley system, with an idler sprocket installed at the opposite end of the slider to maintain belt tension. The timing belt was fixed to the moving plate that carried the robotic arm, ensuring synchronized and repeatable motion along the slider’s 240 mm travel range. The motor is controlled by a TMC2209 motor driver, which is interfaced with an ESP32 microcontroller. Power is supplied by a 12V, 10A switched mode power supply (SMPS).

A static simulation was conducted to evaluate the deflection of the shafts under the expected load of the robotic arm. The maximum deflection was required to be within 1 mm to ensure precise pick-and-place performance. The simulation results confirmed that the shafts met this requirement. The detailed parameters and result are documented in Appendix B.

Table 3.2: Electrical Components of Slider

Component	**Name**	**Quantity**	**Picture**
Motor	NEMA 23 Stepper Motor	1	
Motor Controller	TMC2209 Motor Driver	1	
Microprocessor	ESP32	1	
Power Supply	Switched Mode Power Supply	1	

3.4.2 Fabrication of Slider

The fabrication of the slider was mainly an assembly process involving pre-purchased mechanical components. First, four SK10 linear shaft supports were bolted onto the main machine frame to provide rigid support points. Next, two SUS304 steel shafts, which are 10 mm in diameter and 500 mm in length, each equipped with SC10UU linear bearings, were mounted into the SK10 supports. The Delta X robotic arm was then directly installed onto on a moving plate and the plate was mounted onto the SC10UU bearings, enabling smooth linear motion along the shafts.

For actuation, a NEMA 23 stepper motor was installed at one end of the frame and an idler sprocket at the opposite end. A timing belt was fixed to the moving plate and looped through the motor pulley and idler sprocket to provide synchronized, backlash-free motion. Proper belt tensioning was ensured during assembly to minimize slippage and maintain positional accuracy. The completed assembly resulted in a fully functional slider as shown in Figure 3.3.

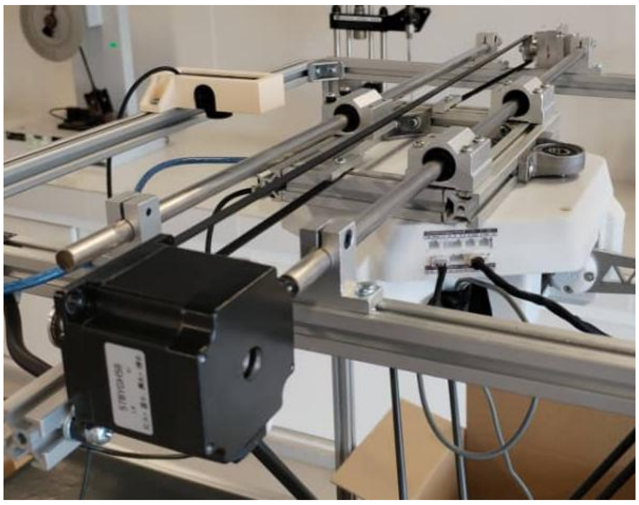


Figure 3.3: Picture of Slider

3.5 Machine Vision

This section describes the design and implementation of the machine vision subsystem, which enables real-time detection, classification and tracking of waste items on the conveyor. YOLOv8, deep learning model used in the project, provides high-accuracy, millisecond-scale inference, while ByteTrack, an object tracking algorithm maintains object identities across frames, ensuring reliable pick-and-place coordination

3.5.1 Dataset Collection

The deep learning model employed in this project is YOLOv8. The initial step involved collecting a comprehensive dataset for training purposes. This dataset was sourced from various environments, including the actual scenes at the plastic sorting facility and online platforms such as Roboflow. The dataset encompassed different types of waste materials, including plastic, paper and aluminium. All images in the dataset maintain a resolution of 640 × 360 pixels. Table 3.1 presents the types of waste materials along with the estimated number of labels.

Table 3.3: Waste Material Types in Dataset Images

Class	Waste Material Type	Estimated Number of Labels
00	Plastic bottle (Plastic)	100
01	Beverage Carton (Paper)	100
02	Aluminium Can (Aluminium)	100

Following the collection, image annotation was conducted. Each waste item in the dataset was annotated according to its type using the Computer Vision Annotation Tool (CVAT) platform, which offers a suite of tools and algorithms for annotating images. Post-annotation, the dataset was exported in the YOLO format, wherein each image is accompanied by a text file detailing the detected objects' classes and coordinates. Subsequently, the dataset was split into training, validation and testing subsets in accordance with the YOLO format to facilitate deep learning model training.

3.5.2 Vision Training

Upon completion of the annotation process, the YOLOv8 deep learning model was trained using the annotated dataset. The training process incorporated libraries such as OpenCV, which provided a real-time optimized computer vision library and Ultralytics YOLO, the latest advancement in the YOLO series known for its enhanced performance and efficiency. To augment the diversity of the training dataset, data augmentation techniques like flipping and scaling were applied.

During training, the model's performance was evaluated based on precision, recall and mean average precision (mAP) metrics. These metrics were computed by comparing the training results with the validation dataset, providing insights into the model's accuracy in identifying and detecting various types of waste materials.

Hyperparameter tuning was performed during training, adjusting parameters such as epochs, batch size and learning rate to enhance the model's performance. At the conclusion of the training process, two versions of the deep learning model were selected, which were the first with the highest mean average precision, indicating optimal accuracy in object detection and the second from the final training epoch. Both models underwent real-time application testing to determine the most suitable model for deployment.

3.5.3 Object Tracking Algorithm

While the YOLOv8 model was proficient in object detection, it may occasionally missed detections or lose track of objects across frames, particularly when objects were in motion on the conveyor. To mitigate this, the ByteTrack algorithm was integrated to enhance multi-object tracking performance.

ByteTrack built on the principles of SORT by combining high-confidence and low-confidence detections during data association. Unlike SORT, which discards low-confidence detections, ByteTrack used them as secondary candidates when matching tracks. This strategy significantly reduced ID switches and track fragmentation, which were common in real-world waste sorting scenarios where partial occlusion or overlapping items occur.

Similar to SORT, ByteTrack employed a Kalman Filter to predict object positions between frames, while the Hungarian algorithm was used to assign new detections to existing tracks efficiently. By leveraging both motion prediction and confidence-based association, ByteTrack maintained robust and consistent tracking of waste items on the conveyor.

To ensure reliable memory management, tracks were cleared if an object left the frame and remained undetected beyond a predefined time window, thus optimizing computational resources. This approach allowed the vision system to maintain stable object identities for the robotic arm, ensuring precise pick-and-place coordination.

3.6 System Assembly and Integration

The final stage of the methodology involved assembling all subsystems including mechanical, electrical and software, into a fully functional automated waste sorting machine. This ensured that the conveyor, slider, robotic arm, sensors and vision system worked seamlessly together.

3.6.1 Assembly of Mechanical Components

The conveyor belt was rigidly fixed to the frame of the Delta X robot arm, as shown in Figure 3.4, to ensure a stable relative position between the transport system and the robotic workspace. This alignment was critical for accurate coordination between object detection and robotic pick-and-place operations.

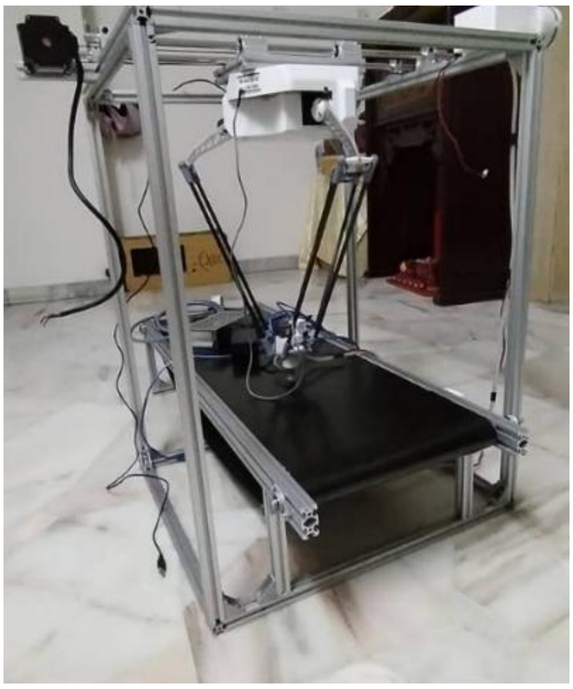


Figure 3.4: Assembly of Conveyor Belt and Frame of Delta X Robot Arm

The camera was securely mounted onto the robot arm frame using a custom 3D-printed bracket, ensuring precise positioning and minimizing vibration during operation. Similarly, the IR sensor near the suction nozzle, as shown in Figure 3.5, was fixed in place using a 3D-printed holder, which provided reliable stability and ensured consistent contact detection with waste items.

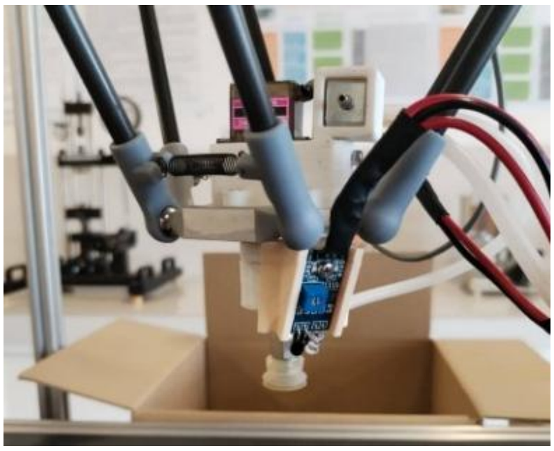


Figure 3.5: Picture of IR Sensor

The dropping boxes, as shown in Figure 3.6, were positioned adjacent to the conveyor system to receive sorted waste items, with their placement optimized for efficient reach of the robotic arm and slider system.

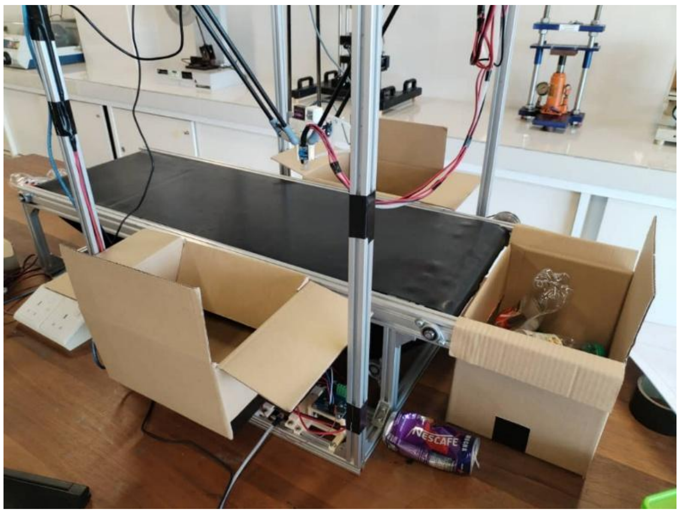


Figure 3.6: Picture of Dropping Boxes

3.6.2 Wiring and Electronics Setup

All electrical components were first tested individually to verify their functionality before integration. After successful testing, the motor drivers and ESP32 microcontroller were mounted onto the robot arm frame using custom 3D-printed holders, ensuring secure placement and accessibility for wiring. Figure 3.7 shows the physical placement of the power supply, motor controller and microcontroller (ESP32).

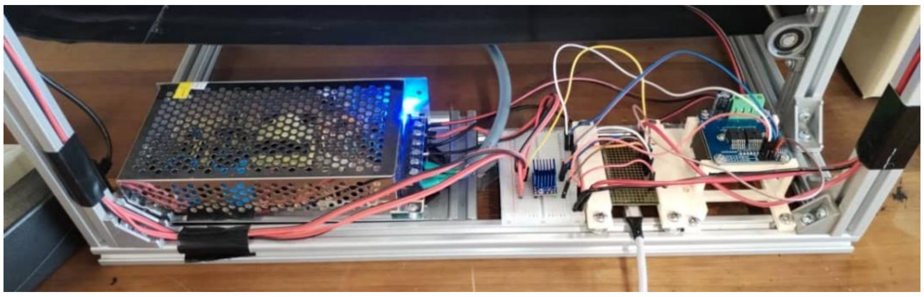


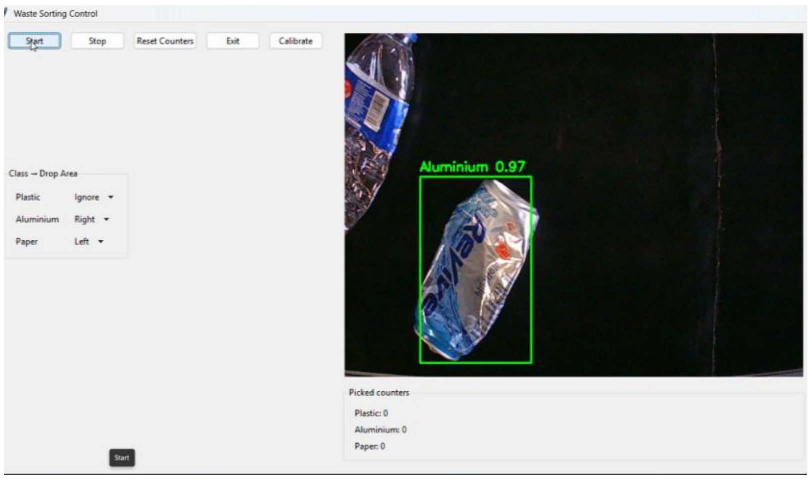
Figure 3.7: Placement of Power Supply, Motor Controller and ESP32

The wiring process was carried out using jumper wires with heat-shrink tubing applied at connections to improve insulation and durability. Special attention was given to the routing of cables for moving components, such as the slider and robotic arm, to prevent entanglement or obstruction during operation.

3.6.3 System Coding and Control Flow

The control system was developed incrementally, with functions implemented and tested individually to ensure stability and correctness before full integration. Initially, the system was tested on a stationary conveyor, where YOLOv8 was used to detect waste items and the Delta X robot arm performed basic pick-and-place operations without considering conveyor motion. Once the stationary tests were successful, the moving conveyor was introduced, and the vision system tracked objects on the belt, sending corresponding coordinates to the robotic arm for dynamic pick-and-place tasks.

To accommodate waste items of different heights, variable Z-axis picking was implemented, allowing the arm to adjust its vertical position during grasping, with feedback from the IR sensor confirming successful contact with the items. A graphical user interface (GUI) was also integrated to enable real-time monitoring and control, including start/stop operations, waste category assignment to designated drop boxes and live camera feed visualization. Figure 3.8 shows the Graphical User Interface (GUI) implemented for the automated waste sorting system.



Class → Drop Area

Plastic	Ignore
Aluminium	Right
Paper	Left

Picked counters

Plastic: 0
Aluminium: 0
Paper: 0

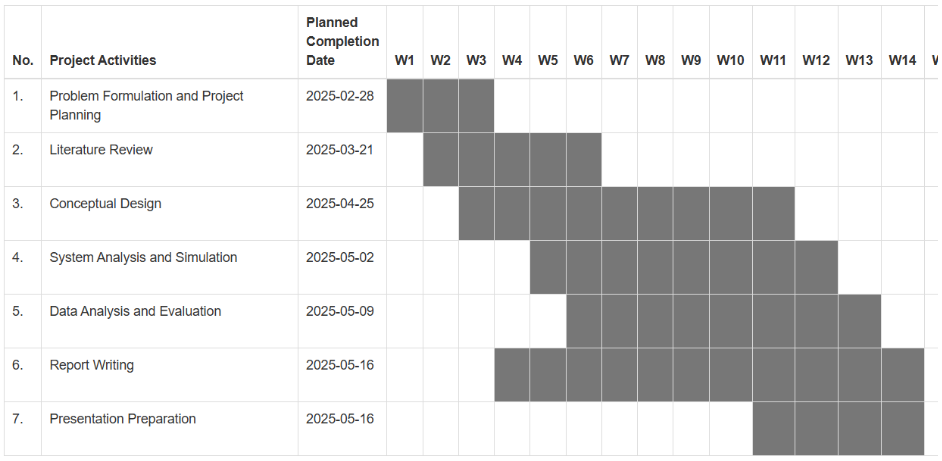
Figure 3.8: Graphical User Interface

The final integrated code coordinated the conveyor via the ESP32, the slider and the Delta X robot arm, synchronizing inputs from the vision system and IR sensor. This modular development approach ensured that each subsystem was individually verified before being incorporated into the complete automated waste sorting workflow. The full source code is provided in Appendix C.

3.7 Work Plan

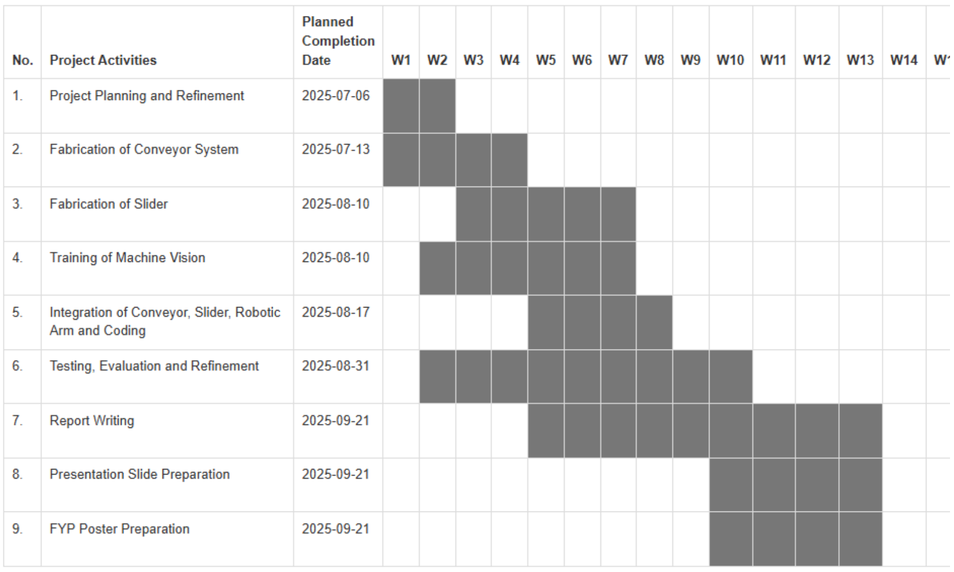
3.7.1 Gantt Chart

A Gantt chart will be utilized to plan and manage the project's timeline over a 14-week period. This visual tool will outline key tasks, their durations and dependencies, ensuring efficient coordination and timely completion. The primary tasks include the fabrication of the conveyor and slider systems, machine vision model training, system integration, testing and evaluation, report writing, presentation preparation and system improvement. Figure 3.9 and 3.10 show the Gantt Chart for FYP1 and FYP2



No.	Project Activities	Planned Completion Date	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14	V
1.	Problem Formulation and Project Planning	2025-02-28	▨	▨	▨												
2.	Literature Review	2025-03-21		▨	▨	▨	▨	▨									
3.	Conceptual Design	2025-04-25			▨	▨	▨	▨	▨	▨	▨	▨	▨				
4.	System Analysis and Simulation	2025-05-02					▨	▨	▨	▨	▨	▨	▨	▨			
5.	Data Analysis and Evaluation	2025-05-09						▨	▨	▨	▨	▨	▨	▨	▨		
6.	Report Writing	2025-05-16				▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	
7.	Presentation Preparation	2025-05-16											▨	▨	▨	▨	

Figure 3.9: Gantt Chart for FYP 1



No.	Project Activities	Planned Completion Date	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14	W
1.	Project Planning and Refinement	2025-07-06	▨	▨													
2.	Fabrication of Conveyor System	2025-07-13	▨	▨	▨	▨											
3.	Fabrication of Slider	2025-08-10			▨	▨	▨	▨	▨								
4.	Training of Machine Vision	2025-08-10		▨	▨	▨	▨	▨	▨								
5.	Integration of Conveyor, Slider, Robotic Arm and Coding	2025-08-17					▨	▨	▨	▨							
6.	Testing, Evaluation and Refinement	2025-08-31		▨	▨	▨	▨	▨	▨	▨	▨	▨					
7.	Report Writing	2025-09-21					▨	▨	▨	▨	▨	▨	▨	▨	▨		
8.	Presentation Slide Preparation	2025-09-21										▨	▨	▨	▨		
9.	FYP Poster Preparation	2025-09-21										▨	▨	▨	▨		

Figure 3.10: Gantt Chart for FYP 2

3.7.2 Milestones and Deliverables

The project was carried out in a structured manner, with progress tracked in two-week intervals to ensure timely completion of tasks.

In the early phase (Week 2), the conveyor system was upgraded by integrating a new motor and verifying its functionality. At the same time, the 3D model of the slider was refined and procurement of necessary materials began. A dataset collection process was also initiated to support the training of the YOLOv8 machine vision model.

By Week 4, the conveyor system was fully fabricated and tested. Integration of the conveyor with the robotic arm (without the slider) was successfully completed and initial control code was developed and tested. The materials required for the slider were also purchased during this stage.

At Week 6, all mechanical subsystems, including the conveyor, slider and robotic arm frame, were completed. Individual control codes for each subsystem were written and verified to ensure functionality before integration.

The mid-phase milestone (Week 8) focused on integrating all components into a single system. Functional testing was performed and additional features such as IR switch were incorporated to enhance reliability. At this stage, preparation of the final report and project poster was also started.

By Week 10, the ESP32 microcontroller was wired to the machine and full integration of all subsystems was achieved. Testing and evaluation of the code and functionality were carried out, confirming that the mechanism for waste identification performed satisfactorily. Documentation tasks, including the report, slides and poster, were also actively developed.

In the final stages (Week 12), full testing and evaluation of the complete system were conducted. Minor code upgrades were implemented to improve performance and administrative tasks such as preparing the reimbursement form were completed. Work on the final report and presentation slides continued to progress toward completion.

Overall, the systematic achievement of milestones ensured that the project proceeded according to schedule. Each phase delivered tangible outcomes, including mechanical fabrication, subsystem integration, vision model development and system-level testing, which collectively contributed to the final functional prototype of the automated waste sorting machine.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

This chapter presents the results obtained from the development, testing and evaluation of the automated waste sorting system. The findings are organized according to the performance of individual subsystems, including the conveyor belt, slider mechanism, robotic arm with vacuum gripper and the machine vision system. System-level integration results, covering coordination between the mechanical, electrical and vision components, are also discussed.

In addition to reporting quantitative performance parameters such as conveyor speed, slider travel range, robotic arm accuracy and detection precision of the vision model, qualitative observations are included to assess the reliability and practicality of the system. The results are further compared with findings from related studies and similar projects to highlight improvements and limitations.

Finally, this chapter provides a discussion of the system's overall performance, emphasizing key challenges encountered, trade-offs made during development and potential areas for improvement.

4.2 Performance of Fabricated Parts

4.2.1 Conveyor Belt Performance

The fabricated conveyor belt provided a working area of 960 mm × 300 mm, which was sufficient to transport multiple waste items simultaneously. To evaluate its performance, two main aspects, which are conveyor speed and load capacity, were tested

The conveyor normally runs in high speed mode. The low speed mode was automatically triggered by the controller when a higher number of waste items were detected on the belt, allowing more time for the vision system and robotic arm to complete pick and place operations. For speed testing, the time taken for the belt to move a waste item across a 700 mm travel distance was recorded under high speed and low speed mode. In high speed mode, the conveyor required 13.4 seconds to travel 700 mm, corresponding to an average speed of 52 mm/s. In low speed mode, the travel time increased to 24.1 seconds, giving an average speed of 29 mm/s."

For load capacity testing, incremental weights of 200 g were gradually placed on the belt. Slippage was first observed at a load of approximately 5.2 kg. This capacity is considered acceptable, as the system is designed primarily to handle lightweight recyclable waste such as plastic bottles, paper and aluminium cans, which typically weigh less than 500 g each.

4.2.2 Slider Mechanism Performance

The slider mechanism was implemented to extend the effective working range of the Delta X robotic arm, allowing it to access a wider area of the conveyor. The slider provided an additional 120 mm travel distance in both the left and right directions (total of 240 mm), effectively increasing the robot's pick and place coverage.

Performance testing was carried out by measuring the time required for the slider to move across its full 120 mm span. The travel time was recorded as 1.41 seconds, demonstrating a fast and responsive motion suitable for real-time waste sorting operations.

To evaluate repeatability, the slider was commanded to perform continuous left - right - left movements for 100 consecutive cycles. After completion, the measured positional deviation was found to be within 2 mm of the original reference point. This error margin is considered acceptable for the application, as the robotic arm can tolerate slight positional variation without significant impact on picking accuracy.

Overall, the slider mechanism exhibited reliable, smooth and repeatable operation, ensuring effective horizontal extension of the robotic arm's workspace with minimal loss in positional accuracy.

4.2.3 Delta X Robot Arm Performance

The Delta X robotic arm served as the primary mechanism for waste pick-and-place tasks. Its horizontal working range was measured to be approximately 240 mm in both the X and Y directions, which was sufficient to cover the conveyor width when combined with the slider's extended motion.

A gripping performance test was carried out to evaluate the effectiveness of the vacuum gripper on different waste materials. The test procedure started with commanding the robot to pick up a waste item, then move it repeatedly to the left and right in five cycles, simulating potential disturbances during operation. The objective was to determine whether the gripper could maintain its hold on the item under dynamic conditions.

The results of the test are summarized in Table 4.1, categorizing materials into those that passed, meaning they remained securely gripped and those that failed, meaning they slipped or detached during the motion. Failures were mainly observed with plastics and aluminium cans. For plastics, deformation occurred when the gripper applied suction, since thin or flexible surfaces tend to bend, reducing the sealing area and causing air leakage. Additionally, many plastic items had uneven or curved surfaces, which made it difficult for the gripper to achieve full contact. Aluminium cans faced a similar issue. Their cylindrical shape and tendency to roll reduced stability during lateral shaking, making them more prone to slipping. In contrast, paper-based materials typically presented flat and porous surfaces, enabling more reliable suction and resulting in a higher success rate.

Table 4.1: Gripping Performance Test of Vacuum Gripper

Material	Passed	Failed
Plastic		
Aluminium		

Paper		

Another gripping performance test was carried out to evaluate the effectiveness of the vacuum gripper on waste materials of different heights. It is important to note that the machine vision system did not have the capability to determine the height of waste items. To address this, an IR sensor was equipped on the vacuum gripper to detect when the gripper made contact with an item. This sensor provided real-time feedback to the system, ensuring that the gripper could adjust its vertical position accurately. During testing, this setup proved effective, allowing the system to successfully pick both thick and thin aluminium cans, as shown in Figure 4.1.

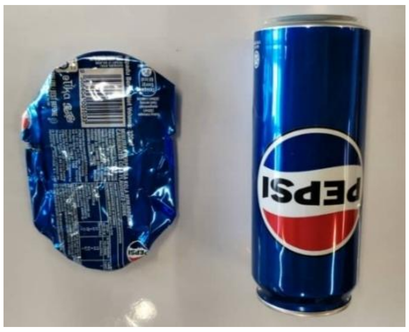


Figure 4.1: Picture of Thin and Thick Aluminium Cans

4.3 Performance of Machine Vision

The machine vision subsystem was implemented using the YOLOv8 deep learning model trained on a custom dataset of plastic bottles, aluminium cans and paper. The performance was evaluated using both training metrics and confusion matrix analysis to assess accuracy, robustness and generalization capability.

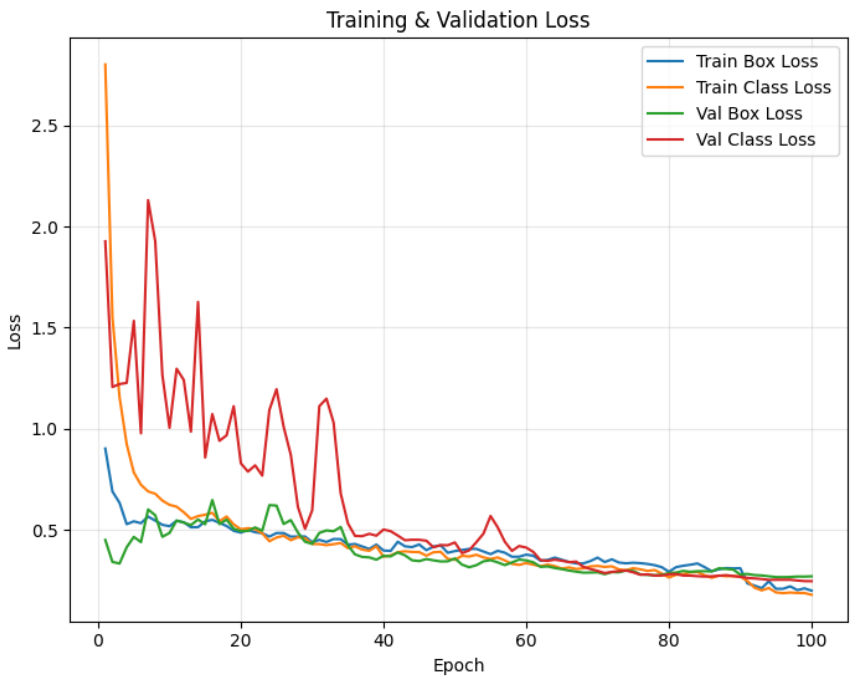


Figure 4.2: Training and Validation Loss Curve

Figure 4.2 shows the training and validation loss curves across 100 epochs. Both the training and validation box loss, as well as the class loss, decreased steadily over time, indicating stable learning and good convergence of the YOLOv8 model. By the end of the training at epoch 100, all losses were reduced to well below 0.5, demonstrating that the model achieved a strong fit to the dataset without signs of underfitting or overfitting.

This convergence implies that the model is capable of accurately localizing waste items (low box loss) and correctly classifying them into their respective categories (low class loss). As a result, the trained model is reliable for real-time waste detection tasks, providing consistent bounding box precision and high classification accuracy across different types of waste.

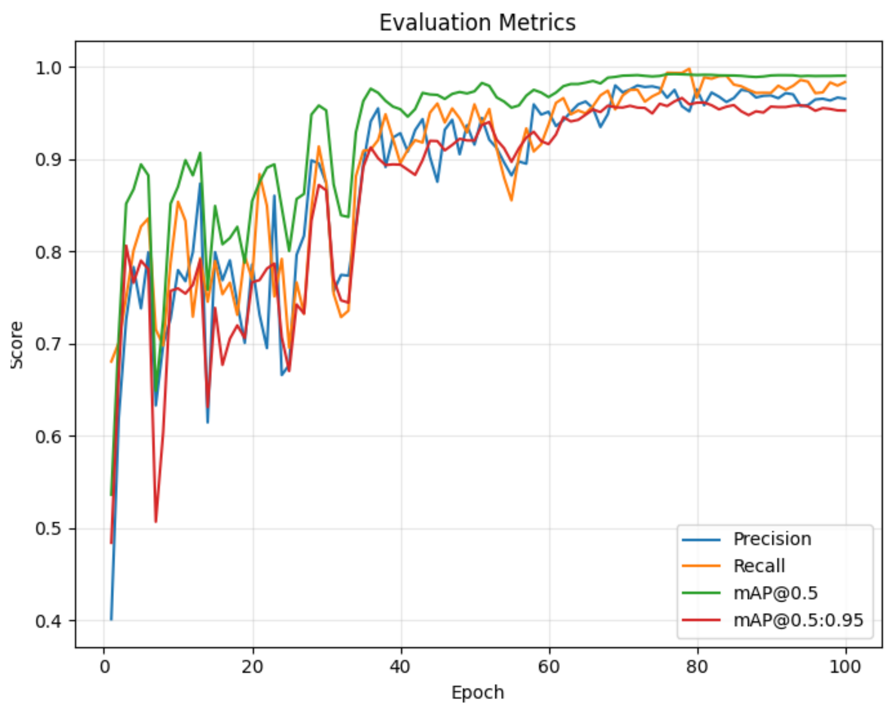


Figure 4.3: Evaluation Metrics Curve

Figure 4.3 shows the evaluation metrics of the YOLOv8 model across 100 epochs, including precision, recall, mAP@50 and mAP@50–95. Precision reached approximately 0.98, indicating that the model produced very few false positives. This means that the model rarely misclassifying background or other materials as waste. Recall improved to 0.97, showing that nearly all waste items present in the dataset were correctly detected. The mAP@50 value achieved around 0.95, reflecting strong detection accuracy when using a 50% IoU threshold, while the stricter mAP@50–95 reached 0.91, confirming that the model maintained reliable detection performance across a wide range of IoU thresholds.

These results demonstrate that the trained YOLOv8 model not only achieved high detection accuracy but also generalized well across different waste types, sizes and shapes.

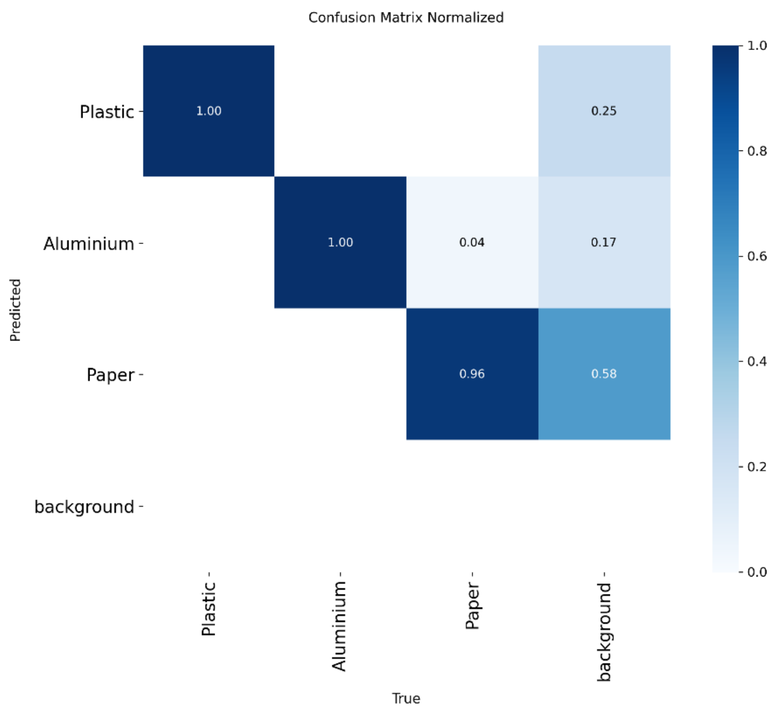


Figure 4.4: Normalized Confusion Matrix

The normalized confusion matrix for the trained model, as shown in Figure 4.4, highlights the classification performance across the three waste categories. The model achieved perfect classification for plastics and aluminium, with 100% accuracy and only negligible confusion with the background. In contrast, paper items were slightly more challenging, with 96% correctly classified but some instances being misclassified as background. This misclassification is likely due to the similarities in colour and texture between paper and the conveyor background, especially under varying lighting conditions. Overall, the results confirm that the YOLOv8 model is highly reliable in distinguishing between plastics, aluminium and paper, with only minor limitations for paper detection. With further dataset augmentation, particularly under different lighting and background scenarios, the model's robustness for paper classification could be further improved.

In real time conveyor testing, the machine vision subsystem achieved an average processing speed of approximately 6 frames per second (FPS) on the laptop. This performance was measured while running the YOLOv8 detection model with a confidence threshold of 0.35 and an IoU threshold of 0.6, balancing detection accuracy with computational efficiency. The achieved FPS was sufficient for the conveyor’s operating speed, ensuring that no waste items passed through undetected.

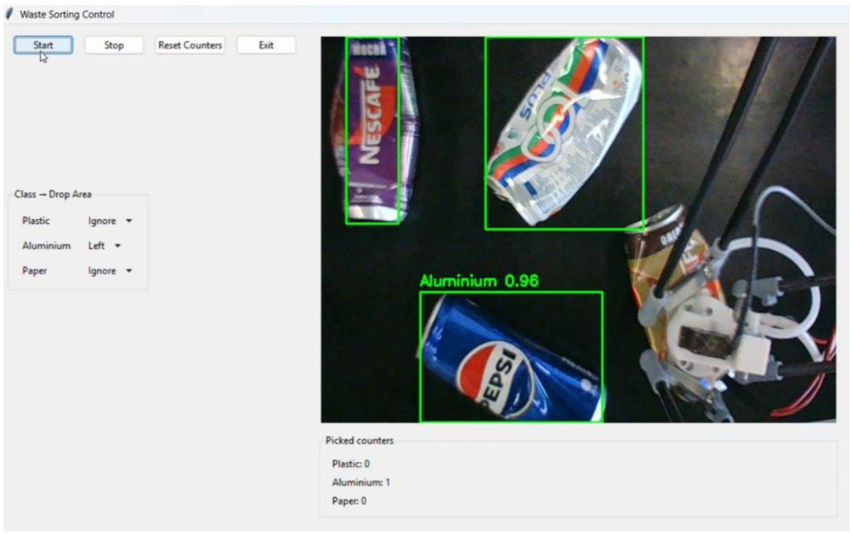
Overall, the real-time performance confirmed that the vision subsystem satisfied the project’s requirements for high accuracy, low-latency inference and reliable multi-object tracking. While the FPS achieved was modest compared to GPU-based implementations reported in the literature, it proved sufficient for the intended application, demonstrating the system’s practicality and scalability under resource-constrained conditions.

4.4 System Testing and Evaluation

The final prototype was tested to evaluate its functionality, accuracy and reliability under both controlled and mixed-waste scenarios. The evaluation focused on three main aspects, which are sorting accuracy, picking speed and long-term reliability.

4.4.1 Sorting Accuracy

Sorting accuracy was assessed through two test setups, including single-item tests, which examined the detection and pick-and-place performance for each waste category and mixed-waste tests, which evaluated the system’s effectiveness under more realistic operating conditions. Two key performance measures were considered, including YOLO detection accuracy and pick-and-place success rate. Detection accuracy was determined by manually observing the real-time GUI feed, such as shown in Figure 4.5 and verifying whether the labels assigned by the model were correct, expressed as the ratio of correctly labelled items to the total number of items observed. The pick-and-place success rate was calculated as the ratio of successful pickups and placements to the total number of pickup attempts.



Class → Drop Area

Plastic	Ignore
Aluminium	Left
Paper	Ignore

Picked counters

Plastic: 0
Aluminium: 1
Paper: 0

Figure 4.5: Real-time GUI Feed with Assigned Label

In the single-item tests, each category was tested over 10 repetitions using 5 items per trial. For aluminium cans, the YOLO detection accuracy reached 100%, while the pick-and-place success rate was 92%. Failures occurred mainly due to the cylindrical cans rolling during pickup. For plastic bottles, the detection accuracy was 96%, but the pick-and-place success rate dropped to 48 percent. This performance indicates that the machine vision is model able to identify items correctly but the pick-and-place operation has low success rate. This was largely due to the IR sensor's inability to detect contact with transparent materials, which reduced pickup reliability. In contrast, paper achieved a detection accuracy of 94% and a pick-and-place success rate of 98%. Although occasional misclassifications occurred under bright lighting conditions, the flat surface of paper allowed for consistently stable gripping.

In the mixed-waste tests, a set of 5 plastics, 5 aluminium cans and 5 paper items was placed together and one class is ignored at a time as there are only two dropping areas. This testing process was repeated 5 times. The detection accuracy achieved was 100% for aluminium, 98.6% for plastic and 98.6% for paper. For pick-and-place performance, aluminium achieved a success rate of around 93.3%, while paper remained the most reliable with over 97.3% success. Plastics, however, continued to pose challenges, with a success rate of 46.6% due to transparency, shape irregularities and sensor detection failures. These results indicate that while the vision system was consistently accurate across all waste categories, the physical handling of plastics remains a limiting factor in overall system performance.

4.4.2 Picking Speed

The average processing speed of the system was evaluated by measuring the time taken from waste detection to successful pickup. This was calculated by dividing the total number of successfully picked items by the total elapsed time and then averaging the results across multiple trials.

The system achieved a throughput of 8 - 15 items per minute, depending largely on the distribution and position of waste on the conveyor. Items located closer to the centreline were picked more quickly, as the slider required less horizontal travel, whereas items near the edges took longer due to the additional motion required. Similarly, item height influenced processing speed. Taller items required less downward travel of the robotic arm, resulting in faster pickups, while shorter items necessitated greater vertical movement, slightly reducing efficiency.

4.4.3 Reliability Testing

A continuous operation test was conducted over a duration of 20 minutes, during which the conveyor, slider, robotic arm and vision system operated without major faults. Minor failures were observed in the pickup of plastic items, but the system recovered without interruption.

However, a significant issue identified was the gradual loss of positional accuracy in both the robotic arm and the slider. Since the system lacks a feedback mechanism, it relies solely on the precision control of the stepper motors. Over extended operation, particularly after 15 - 20 minutes of continuous movement, the coordinates began to drift. This misalignment may have been caused by accumulated step errors or occasional collisions with obstacles, resulting in the robot no longer knowing its exact position. This limitation highlights the need for additional position feedback mechanisms, such as encoders or limit switches, to ensure long-term accuracy and reliable operation.

4.5 Discussion of Findings

The experimental results of the developed waste sorting system were evaluated against both the project objectives and findings from related studies. The discussion focuses on detection accuracy, pick-and-place success rates, throughput and challenges in handling specific waste types.

In single-item tests, the YOLOv8 detection accuracy achieved was 100% for aluminium, 96% for plastic and 94% for paper. The corresponding pick-and-place success rates were 92% for aluminium, 48% for plastic and 98% for paper. These results demonstrate that while the detection system performs at a high level, the physical execution of gripping and sorting plastics is a major limitation.

A comparable study, *PLC-Controlled Intelligent Conveyor System with AI-Enhanced Vision for Efficient Waste Sorting* (Almtireen et al., 2025), also applied YOLOv8 and reported classification accuracies above 95% across plastics, paper and metals. Our system's detection accuracy is consistent with this work. However, the pick-and-place success rate for plastics is notably lower, highlighting that reliable handling of irregular or transparent items is still an unresolved issue.

The system processed between 8 - 15 items per minute depending on waste distribution and positioning on the conveyor. This throughput is lower than that reported in commercial optical sorting systems, which often achieve tens or even hundreds of items per minute by using high-speed conveyors, multiple lanes and industrial-grade actuators. Research systems also tend to simplify operating conditions to boost throughput. While our system prioritised affordability and modularity, its speed limitation reflects the trade-off between academic prototyping and industrial-scale equipment.

The most significant challenge observed was the handling of plastic items, especially transparent bottles and those with irregular surfaces. Detection occasionally misclassified plastics under bright lighting as aluminium and more critically, the IR sensor failed to register contact due to the transparency of the material. This resulted in a pick-and-place success rate of only 48% for plastics.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

This project successfully developed an automated waste sorting system that integrates machine vision, robotic actuation and electronic control into a functional prototype. The first objective, to develop a robotic system equipped with computer vision for accurate detection and classification of various waste types, was achieved through the successful integration of a YOLOv8-based machine vision model with the Delta X robotic arm and a supporting conveyor - slider mechanism. The vision system achieved detection accuracies of up to 100% for aluminium, 96% for plastics and 94% for paper.

The second objective, to automate the sorting process to ensure consistent and accurate waste segregation, was fulfilled by combining vision-based classification with robotic actuation and electronic control. The Delta X robotic arm with vacuum gripper successfully performed pick-and-place operations, achieving 92% success for aluminium and 98% for paper. Although plastics posed challenges due to irregular shapes and transparency, the system nonetheless demonstrated consistent automation of the sorting process under controlled conditions.

The third objective, to evaluate the system's performance in terms of sorting accuracy, processing speed and reliability, was met through systematic testing. The prototype achieved an overall throughput of 8 - 15 items per minute, depending on waste distribution and position and remained operational over a continuous 20-minute test without major faults. While throughput remains modest compared to industrial systems, the results confirm that the prototype is reliable for small-scale applications.

In summary, this project has successfully demonstrated the feasibility of integrating deep-learning based vision with robotic automation for waste sorting. The system achieved its intended aims and provided valuable insights into the challenges of handling plastics and ensuring long-term positional accuracy. Despite the limitations, the work establishes a strong foundation for future improvements and represents a step toward scalable, intelligent waste management solutions.

5.2 Recommendations for future work

To further enhance the performance and reliability of the automated waste sorting system, several improvements are recommended for future development. In terms of end - effector design, adopting a more advanced gripper, such as a vacuum system powered by a stronger motor or a hybrid gripper that combines suction and mechanical grasping, would improve the handling of irregularly shaped or smooth-surfaced materials. The IR sensor currently used for confirming successful pickups could be replaced with an ultrasonic sensor or a force sensor, which would offer more reliable detection across different material types, including transparent plastics.

For mechanical safety and positional accuracy, installing limit switches on all moving parts, including the conveyor, slider and robotic arm, would not only prevent over-travel but also mitigate long-term positional drift by providing consistent reference points, thereby reducing the need for frequent recalibration.

Other than that, the machine vision model could be expanded to include additional waste categories such as glass, rubber and other recyclables, thereby increasing the versatility and practicality of the system in real - world applications. To support this, the dataset should be expanded with a larger number of training images for each class, ensuring balanced representation and improving the model’s ability to generalize under varied lighting, texture and orientation conditions.

REFERENCES

Ackerman, E. (2014). *Electrostatics: Good for Robot Grippers and Lots More*. [online] Available at: https://spectrum.ieee.org/electrostatic-robot-grippers (Accessed: 29 April 2025).

Almtireen, N., Reddy, V., Sutton, M., Nedvidek, A., Karn, C., Ryalat, M., Elmoaqet, H. and Rawashdeh, N. (2025). PLC-Controlled Intelligent Conveyor System with AI-Enhanced Vision for Efficient Waste Sorting. Applied Sciences, [online] 15(3), p.1550. doi:https://doi.org/10.3390/app15031550.

Balluff (n.d.). *The 5 most common types of fixed industrial robots | Balluff*. [online] Available at: https://www.balluff.com/en-us/blog/the-5-most-common-types-of-fixed-industrial-robots (Accessed: 29 April 2025).

Chuah, S. *et al* (2023). ASSESSING RESIDENTS' INTENTION TOWARDS MUNICIPAL SOLID WASTE SOURCE SEPARATION: A CASE STUDY OF MALAYSIA. *PLANNING MALAYSIA*, 21. Available at: https://doi.org/10.21837/pm.v21i25.1245.

DeltaX Robot. (2020). *Delta X 1 - Delta X Robot*. [online] Available at: https://docs.deltaxrobot.com/reference/specifications/sp_x1 (Accessed: 29 April 2025).

Dorna Robotics. (2023). *Types Of Robot Grippers And Their Applications*. [online]. Available at: https://dorna.ai/blog/types-of-grippers-for-robots/ (Accessed: 29 April 2025).

ERI Economic Research Institute (2025). *Recycling and Reclamation Worker*. [online] Erieri.com. Available at: http://erieri.com/salary/job/recycling-and-reclamation-worker/malaysia (Accessed: 29 April 2025).

Esmael Oumer, Melke, A., Wondwosen Teklesilasie and Dejene, G. (2024). Prevalence of work-related injuries and associated factors among municipal solid waste collectors in Hawassa City, Sidama, Ethiopia 2023. *Scientific Reports*, 14(1). Available at: https://doi.org/10.1038/s41598-024-78973-4.

Flexi Bowl. (2014). *SCARA Robot | Flexibowl*. [online] Available at: https://www.flexibowl.com/scara-robot.html (Accessed: 1 May 2025).

Ganes Kesari (2024). Turning Trash Into Treasure: How AI Is Revolutionizing Waste Sorting. *Forbes*. [online]. Available at: https://www.forbes.com/sites/ganeskesari/2024/05/31/turning-trash-into-treasure-how-ai-is-revolutionizing-waste-sorting/ (Accessed: 29 April 2025).

Granta (2023). *What is a Vacuum Gripper? | Granta Automation | Granta Automation*. [online] Available at: https://www.granta-automation.co.uk/news/what-is-a-vacuum-gripper/ (Accessed: 29 April 2025).

International Organization for Standardization (2021) *Robotics — Vocabulary*. ISO 8373. Available at: https://bsol-bsigroup-com (Accessed: 30 April 2025).

International Trade Administration (2024). *Malaysia Waste Management*. [online]. Available at: https://www.trade.gov/market-intelligence/malaysia-waste-management (Accessed: 27 April 2025).

Joseph, R., Divvala Santosh, Ross, G. and Ali, F. (2016). You Only Look Once: Unified, Real-Time Object Detection. *arXiv (Cornell University)*. Available at: https://doi.org/10.48550/arxiv.1506.02640.

KALRA, K. (2023). *OBJECT TRACKING*. [online]. Available at: https://medium.com/@khwabkalra1/object-tracking-2fe4127e58bf (Accessed: 29 April 2025).

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A.C. (2016). SSD: Single Shot MultiBox Detector. *Computer Vision – ECCV 2016*, 9905, pp.21–37. Available at: https://doi.org/10.1007/978-3-319-46448-0_2.

Potrimba, P. (2023). *What is R-CNN?* [online] Roboflow Blog. Available at: https://blog.roboflow.com/what-is-r-cnn/ (Accessed: 29 April 2025).

Robots Done Right. (2022). *SCARA Robots vs Delta Robots*. [online] Available at: https://robotsdoneright.com/Articles/scara-robots-vs-delta-robots.html?srsltid=AfmBOoqbGEUMvPlcYOfPqlVyg5JuCU7V8iJ_Ig4ORjaY1ANcfmLWwHDQ (Accessed: 29 April 2025).

Robots Done Right. (2025). *What is a Delta Robot?* [online] Available at: https://robotsdoneright.com/Articles/what-is-a-delta-robot.html?srsltid=AfmBOooIJ9c8Ssc4Snw5TqH5uOk3dUY4XjfrL9kLu0p3v43liA_g8Oeo (Accessed: 1 May 2025).

Sanyam (2022). *Understanding Multiple Object Tracking using DeepSORT*. [online] Available at: https://learnopencv.com/understanding-multiple-object-tracking-using-deepsort/#Simple-Online-Realtime-Tracking-(SORT) (Accessed: 3 May 2025)

Shanuka Dodampegama, Hou, L., Asadi, E., Zhang, K. and Sujeeva Setunge (2024). Revolutionizing construction and demolition waste sorting: Insights from artificial intelligence and robotic applications. *Resources, Conservation and Recycling*, 202, pp.107375–107375. Available at: https://doi.org/10.1016/j.resconrec.2023.107375.

SM Tech (2019). *All about 775 Motor- Full specification in detail*. [online]. Available at: https://somanytech.com/what-is-775-motor (Accessed: 29 April 2025).

Standard Bot. (2025). *What do robot grippers do? - Standard Bots*. [online] Available at: https://standardbots.com/blog/what-do-robot-grippers-do (Accessed: 29 April 2025).

Standard Bots. (2025). *What is a Cartesian robot? A newbie-friendly guide - Standard Bots*. [online] Available at: https://standardbots.com/blog/what-is-a-cartesian-robot-a-newbie-friendly-guide?srsltid=AfmBOooJ1opLu2BxSQBf5f4i2Q6A0U9i4fZGgZZ1nrv6X-qF30sfF17Y (Accessed: 29 April 2025)

Standard Bots. (2025). *What is a SCARA robot? A brief introduction - Standard Bots*. [online] Available at: https://standardbots.com/blog/what-is-a-scara-robot-a-brief-introduction?srsltid=AfmBOoqbk1NX3B812Ko-M99GZ3wsDx8cAh0wMOnfCj43YYg0IaoesV8G [Accessed 10 May 2025] (Accessed: 29 April 2025).

Stanford University. (2015). *Gripper device using shear-controlled dry adhesive film | Explore Technologies*. [online] Available at: https://techfinder.stanford.edu/technology/gripper-device-using-shear-controlled-dry-adhesive-film (Accessed: 29 April 2025).

Techman Robot. (2023). *Types of Robotic Arms*. [online] Available at: https://www.tm-robot.com/en/robotic-arms/ (Accessed: 27 April 2025).

Worldbank.org. (2022). *Malaysia Waste, parings and scrap, of other plastics, ne imports by country | 2022 | Data*. [online] Available at: https://wits.worldbank.org/trade/comtrade/en/country/MYS/year/2022/tradeflow/Imports/partner/ALL/product/391590? (Accessed: 26 April 2025).

Zainal, F. (2024). *39,000 tonnes of solid waste daily*. [online] The Star. Available at: https://www.thestar.com.my/news/nation/2024/01/02/39000-tonnes-of-solid-waste-daily (Accessed: 26 April 2025).

Zhang, Y., Sun, P., Jiang, Y., Yu, D., Yuan, Z., Luo, P., Liu, W. and Wang, X. (2021). ByteTrack: Multi-Object Tracking by Associating Every Detection Box. *arXiv (Cornell University)*. doi: https://doi.org/10.48550/arxiv.2110.06864.

APPENDICES

Appendix A: Static Simulation of Motor Bracket

This appendix documents the static simulation performed on the 3D-printed motor bracket used to mount the JGB37-545 gear motor on the conveyor frame. The motor bracket was fabricated using PETG. Table A.1 tabulates the material properties used in the simulation.

Table A.1: Material Properties Used in Simulation

Property	**Value**
Young’s modulus	0.8 GPa
Poisson’s ratio	0.38
Density	350 kgm^{-3}
Yield strength	150 MPa

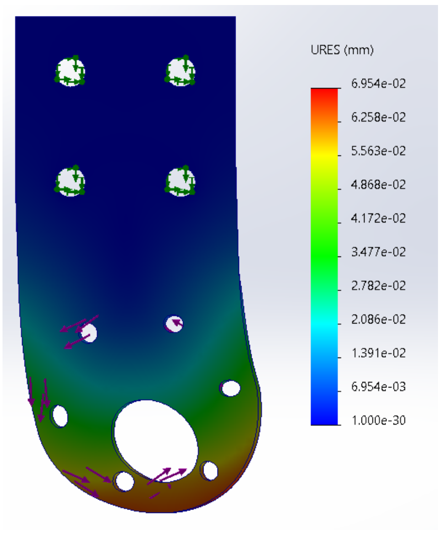


Figure A.1: Displacement Distribution of the Motor Bracket

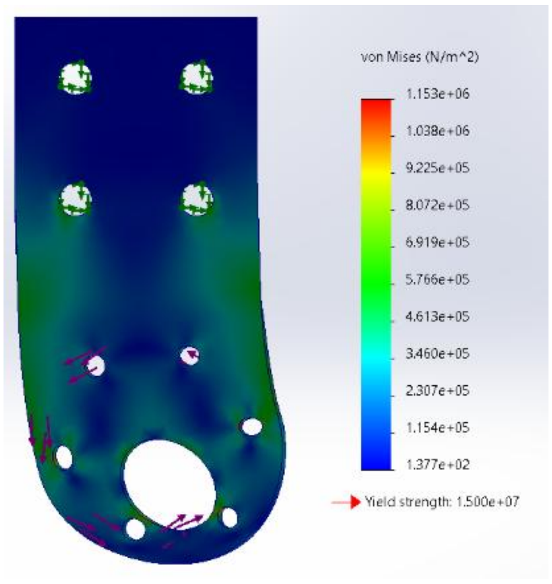


Figure A.2: Von Mises Stress Distribution of the Motor Bracket

Figures A.1 and A.2 show the displacement and von Mises stress distribution of the motor bracket. It can be observed that the maximum displacement and von Mises stress are 0.0695 mm and 1.153 MPa. The maximum von Mises stress is much lower than the assumed yield strength (150 MPa). This indicates that the motor bracket is structurally adequate to withstand the applied motor loads with large safety margin. Furthermore, the maximum deflection (0.0695 mm) is also negligible.

Appendix B: Static Simulation of Slider Shaft

This appendix presents the static simulation analysis of the stainless steel shafts, which are 10 mm in diameter and 500 mm in length, used in the slider mechanism. The shafts support the Delta X robotic arm via three SC10UU linear bearings and their stiffness is critical to ensuring precise pick-and-place performance. The maximum allowable shaft deflection was set at 1 mm, based on accuracy requirements for robotic positioning.

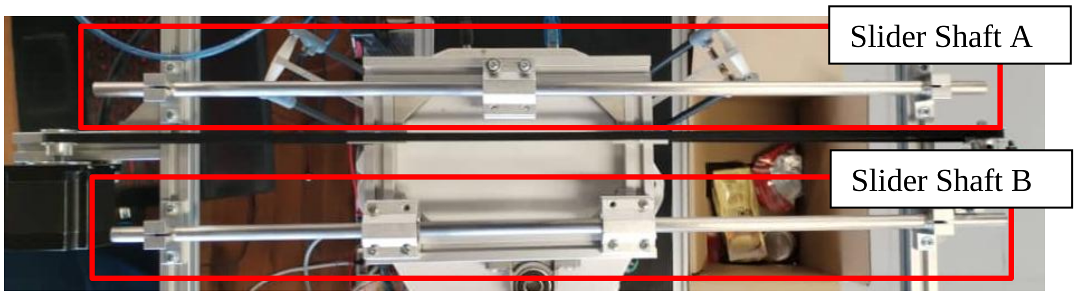


Figure A.3: Slider Shafts with Different Loading Scenarios

There are two different slider shafts with two different loading scenarios as shown in Figure A.3. For slider shaft A, which only have a single SC10UU load, the load condition corresponds to one SC10UU bearing supporting approximately one-third of the Delta X robotic arm's weight (4 kg / 3 ≈ 1.33 kg). The load was applied across a 35 mm contact range positioned at the midpoint of the 500 mm shaft, where maximum deflection is expected.

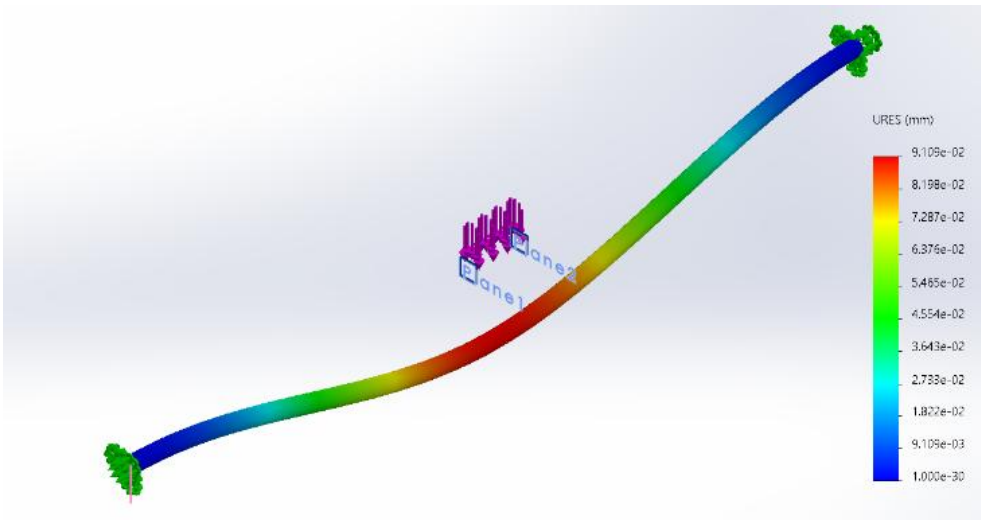


Figure A.4: Displacement Distribution of the Slider Shaft A

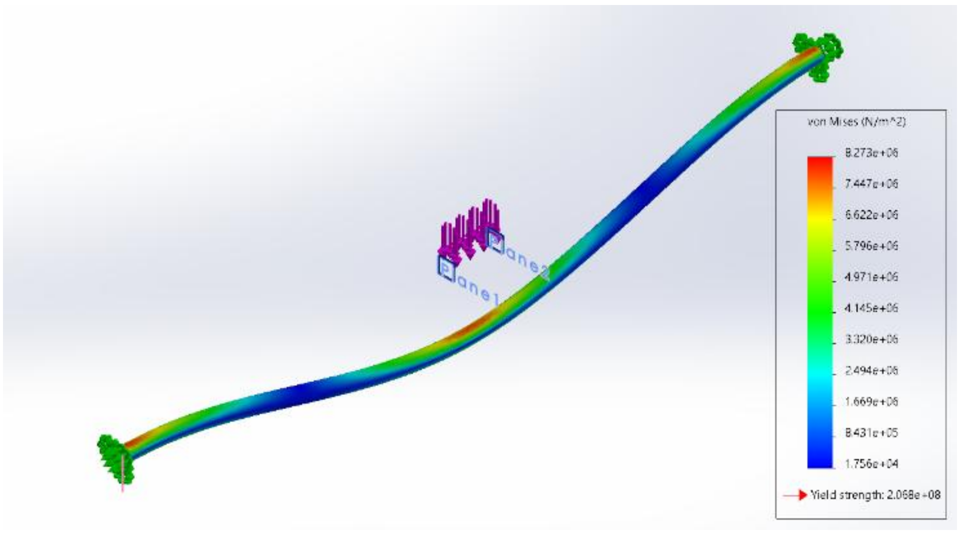


Figure A.5: Von Mises Stress Distribution of the Slider Shaft A

Figures A.4 and A.5 show the displacement and von Mises stress distribution of the slider shaft A. It can be observed that the maximum displacement and von Mises stress are 0.091 mm and 8.273 MPa. The maximum von Mises stress is significantly lower than the assumed yield strength of SUS304 stainless steel (200 MPa), providing a large safety margin. Moreover, the maximum deflection of 0.091 mm is well below the allowable deflection limit of 1 mm.

For Slider Shaft B, the shaft is supported by two SC10UU bearings separated by 130 mm. Each bearing carries approximately one-third of the Delta X robotic arm’s weight (1.33 kg each) distributed over a 35 mm contact range. The load was applied across a 200 mm contact range positioned at the midpoint of the 500 mm shaft, where maximum deflection is expected.

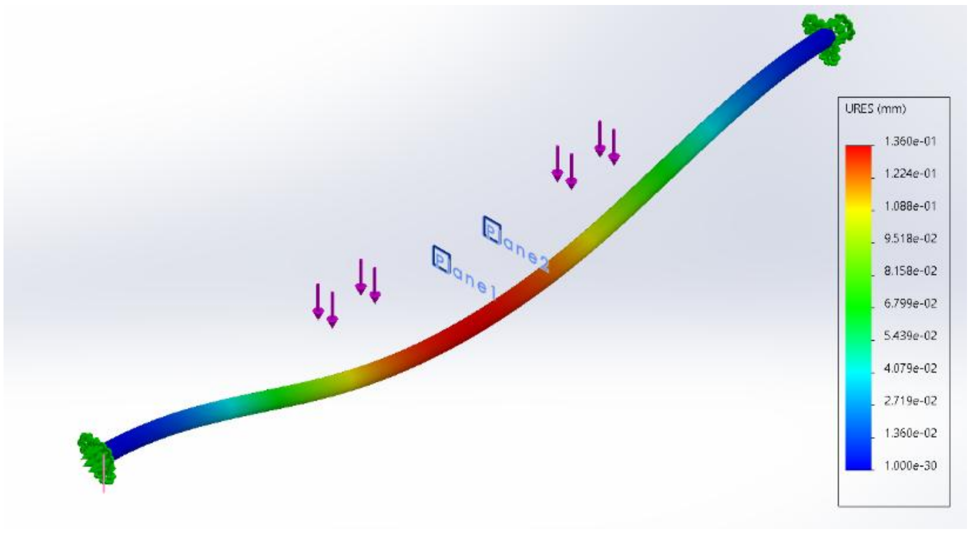


Figure A.6: Displacement Distribution of the Slider Shaft B

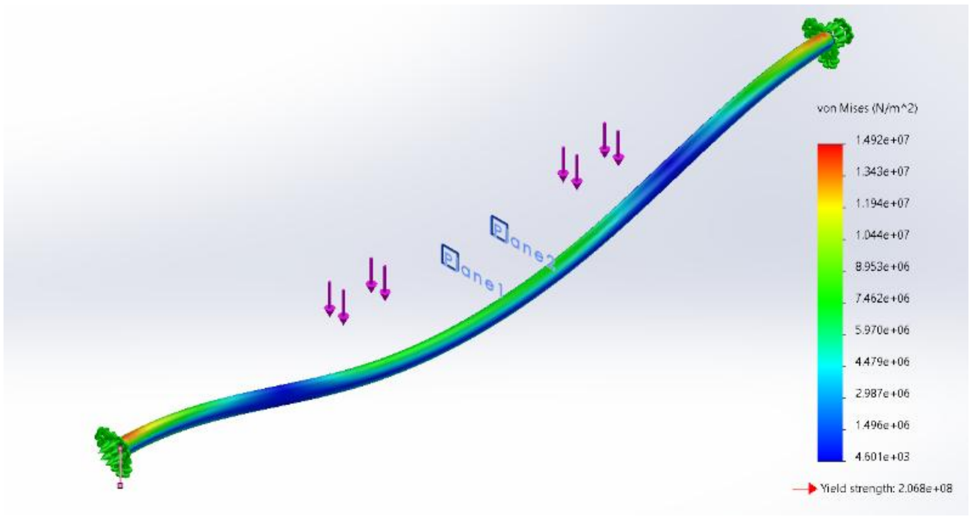


Figure A.7: Von Mises Stress Distribution of the Slider Shaft B

Figures A.6 and A.7 show the displacement and von Mises stress distribution of the slider shaft B. It can be observed that the maximum displacement and von Mises stress are 0.136 mm and 14.92 MPa. The maximum von Mises stress is significantly lower than the assumed yield strength of SUS304 stainless steel, providing a large safety margin. Moreover, the maximum deflection of 0.136 mm is well below the allowable deflection limit of 1 mm.

Appendix C: Full Code

This appendix contains the source code used for the automated waste sorting system. The system consists of two main components. The first component is C++ code, which controls the motor systems and receives feedback from the IR sensor using the ESP32. The second component is Python code, which manages the machine vision system, including YOLOv8 object detection, robot arm control for pick-and-place coordination and the graphical user interface (GUI).

C++ Code (PlatformIO / ESP32):

```
#include <Arduino.h>

// === Conveyor Motor (BTS7960) ===
#define MOTOR_PWM 25   // RPWM to GPIO25
#define IR_PIN 35      // IR sensor pin (digital, HIGH = no
object, LOW = object)

// === Stepper Motor (TMC2209 via STEP/DIR/EN) ===
#define STEP_PIN 27
#define DIR_PIN 26
#define EN_PIN 14

// === Stepper Constants ===
const int stepsPerRevolution = 6500; // Adjust based on your
mechanical setup
bool stepperEnabled = false;

// Ramp parameters (tune for smoothness)
int maxDelay = 200;   // slowest step (microseconds)
int minDelay = 80;   // fastest step (microseconds)
int rampSteps = 1000;  // steps for acceleration/deceleration

void setup() {
  Serial.begin(115200);

  // Conveyor setup
  ledcAttachPin(MOTOR_PWM, 0);
  ledcSetup(0, 20000, 8); // 20kHz, 8-bit PWM

  // IR setup
  pinMode(IR_PIN, INPUT);

  // Stepper setup
  pinMode(STEP_PIN, OUTPUT);
  pinMode(DIR_PIN, OUTPUT);
  pinMode(EN_PIN, OUTPUT);
  digitalWrite(EN_PIN, HIGH); // disable stepper at start

  Serial.println("System Ready");
}

// Function to move stepper one pulse
void stepMotor(int delayTime) {
  digitalWrite(STEP_PIN, HIGH);
  delayMicroseconds(delayTime);
  digitalWrite(STEP_PIN, LOW);
  delayMicroseconds(delayTime);
}

// Function to move stepper with acceleration ramp
void moveStepperRamp(int dir, int steps) {
  digitalWrite(EN_PIN, LOW);   // enable driver
  digitalWrite(DIR_PIN, dir);  // set direction


  int actualRamp = rampSteps;
  if (rampSteps > steps / 2) actualRamp = steps / 2;

  for (int i = 0; i < steps; i++) {
    int currentDelay;

    if (i < actualRamp) { // Acceleration
      currentDelay = maxDelay - ((maxDelay - minDelay) * i /
actualRamp);
    }
    else if (i >= steps - actualRamp) { // Deceleration
      int rampDownIndex = i - (steps - actualRamp);
      currentDelay = minDelay + ((maxDelay - minDelay) *
rampDownIndex / actualRamp);
    }
    else { // Constant speed
      currentDelay = minDelay;
    }

    stepMotor(currentDelay);
  }
  digitalWrite(EN_PIN, HIGH);
}

void loop() {
  // === Conveyor Serial Control ===
  if (Serial.available()) {
    char cmd = Serial.read();

    if (cmd == 'R') { // Full speed
      ledcWrite(0, 255);
    }
    else if (cmd == 'L') { // Half speed
      ledcWrite(0, 127);
    }
    else if (cmd == 'S') { // Stop
      ledcWrite(0, 0);
    }
    else if (cmd == '1') { // Move left
      moveStepperRamp(LOW, stepsPerRevolution);
    }
    else if (cmd == '3') { // Move right
      moveStepperRamp(HIGH, stepsPerRevolution);
    }
  }


  // === IR Sensor Feedback ===
  int ir_state = digitalRead(IR_PIN);
  if (ir_state == LOW) {
    Serial.println("TOUCHED");
  } else {
    Serial.println("NOT_TOUCHED");
  }

  delay(50);
}
```

Python Code (Machine Vision, Pick-and-Place & GUI):

```
import threading
import tkinter as tk
from tkinter import ttk, messagebox
from PIL import Image, ImageTk
import cv2
import serial
import time
from queue import Queue
from ultralytics import YOLO
import numpy as np
```

```
# === Setup Delta X robot serial ===
robot = serial.Serial("COM10", 115200, timeout=1)
time.sleep(2)

# === Setup ESP32 serial ===
esp = serial.Serial("COM12", 115200, timeout=1)
time.sleep(2)

def send_gcode(cmd):
    robot.write((cmd + '\n').encode())
    while robot.in_waiting:
        print(robot.readline().decode().strip())

# === Reads IR state from ESP32 ===
last_ir_state = "NOT_TOUCHED"
def read_ir():
    global last_ir_state
    while esp.in_waiting > 0:
        msg = esp.readline().decode(errors="ignore").strip()
        if msg in ["TOUCHED", "NOT_TOUCHED"]:
            last_ir_state = msg
    return last_ir_state

def conveyor_start():
    esp.write(b'R')

def conveyor_low():
    esp.write(b'L')

def conveyor_stop():
    esp.write(b'S')

# === YOLO model with tracking ===
model = YOLO("runs/detect/train3/weights/newbest2.pt")
class_names = model.names

# === Webcam ===
cap = cv2.VideoCapture(0)
cap.set(3, 640)
cap.set(4, 480)
cap.set(cv2.CAP_PROP_AUTO_EXPOSURE, 0.25)
cap.set(cv2.CAP_PROP_EXPOSURE, -3)
cap.set(cv2.CAP_PROP_GAIN, 0)

# === Calibration matrix (initial identity) ===
M_affine = np.array([[0.0000, 0.5889, -141.7406],
                     [0.6870, -0.0059, -401.4029]],
dtype=np.float32)
```

```python
# === Calibration ===
def webcam_to_robot(x, y):
    global M_affine
    pt = np.array([x, y, 1], dtype=np.float32)
    X, Y = M_affine @ pt
    return int(-X), int(-Y)   # keep your sign convention

# === Constants ===
Z_SAFE = -280
Z_PICK = -300
IDLE_X, IDLE_Y, IDLE_Z = 0, 0, -264
WORKSPACE_LIMIT = 120
WASTE_SPEED = 10
T_PICK = 2.8

DROP_LOCATIONS = {
    "left":   (150, 0, -290),
    "right": (-150, 0, -290)
}

# === Shared state ===
shared = {
    "run": False,
    "mapping": {"plastic": "Left", "aluminium": "Right", "paper":
"Ignore"},
    "counters": {"plastic": 0, "aluminium": 0, "paper": 0},
    "frame": None,
    "program_running": True,
    "calibrating": False
}

lock = threading.Lock()

# Detection memory + queue + flags
detection_memory = {}
target_queue = Queue()
robot_busy = False
conveyor_fast = 1

# Home
send_gcode("G28")


# === Vision task ===
def vision_task():
    global detection_memory, waste_count, robot_busy
    while shared["program_running"]:
        ret, frame = cap.read()
        if not ret:
            break

        if shared["calibrating"]:   #  skip vision when
calibrating
            time.sleep(0.2)
            continue

        now = time.time()

        conveyor_fast = (0 if len(detection_memory) >= 2 else 1)
#change the number to vary the threshold value

        # Conveyor control
        if shared["run"]:
            if conveyor_fast == 0:
                conveyor_low()
            else:
                conveyor_start()
        else:
            conveyor_stop()

        # Run YOLO tracking
        results = model.track(
            frame,
            conf=0.35,
            iou=0.6,
            agnostic_nms=True,
            persist=True,
            tracker="bytetrack.yaml",
            verbose=False
        )

        # Draw detections
        if results and results[0].boxes is not None and
len(results[0].boxes) > 0:
            for box in results[0].boxes:
                x1, y1, x2, y2 =
box.xyxy[0].cpu().numpy().astype(int)
                cls_id = int(box.cls[0].cpu().numpy())
                cls_name = class_names[cls_id]
                conf = float(box.conf[0].cpu().numpy()) if
box.conf is not None else 0.0
                cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255,
0), 2)
                label = f"{cls_name} {conf:.2f}"
                cv2.putText(frame, label, (x1, y1 - 8),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0,
255, 0), 2)

        # Update frame for GUI
        with lock:
            shared["frame"] = frame.copy()

        if not robot_busy and results and results[0].boxes is not
None:
            for box in results[0].boxes:
                x1, y1, x2, y2 =
box.xyxy[0].cpu().numpy().astype(int)
                cx, cy = (x1 + x2) // 2, (y1 + y2) // 2
                cls_id = int(box.cls[0].cpu().numpy())
                cls_name = class_names[cls_id].lower()
                track_id = int(box.id.cpu().numpy()) if box.id is
not None else None
                if track_id is None:
                    continue

                if track_id not in detection_memory:
                    detection_memory[track_id] = {"first_seen":
now, "last_seen": now,
                                                  "pos": (cx,
cy), "cls": cls_name}
                else:
                    detection_memory[track_id]["last_seen"] = now
                    detection_memory[track_id]["pos"] = (cx, cy)
                    detection_memory[track_id]["cls"] = cls_name

            for obj_id, data in list(detection_memory.items()):
                if now - data["first_seen"] >= 1.0:
                    cx, cy = data["pos"]
                    rx, ry = webcam_to_robot(cx, cy)

                    ry_pred = ry - (WASTE_SPEED * (T_PICK / 2 if
conveyor_fast == 0 else T_PICK))

                    if abs(rx) <= WORKSPACE_LIMIT and
abs(ry_pred) <= WORKSPACE_LIMIT:
                        target_queue.put({"id": obj_id, "rx": rx,
"ry": ry_pred, "cls": data["cls"]})
                        print(f"[LOCKED] Target {data['cls']}
ID={obj_id} at ({rx},{ry_pred})")
                        detection_memory.pop(obj_id, None)
                        break

                if now - data["last_seen"] > 2:
                    detection_memory.pop(obj_id, None)


# === Robot task ===
def robot_task():
    global robot_busy
    while shared["program_running"]:
        if shared["calibrating"]:   # ✅ skip vision when
calibrating
            time.sleep(0.2)
            continue
        try:
            target = target_queue.get(timeout=0.2)  # ✅ blocking
queue
        except:
            continue

        robot_busy = True
        rx, ry, cls_name = target["rx"], target["ry"],
target["cls"]

        with lock:
            mapping = shared["mapping"].copy()
            counters = shared["counters"]

        mapped = mapping.get(cls_name, "Ignore").lower()
        if mapped == "ignore":
            print(f"[ROBOT] Ignoring {cls_name}")
            robot_busy = False
            time.sleep(0.5)
            continue

        drop_key = "left" if mapped == "left" else "right"
        drop_x, drop_y, drop_z = DROP_LOCATIONS.get(drop_key, (-
120, 120, -270))

        print(f"[INFO] Picking {cls_name} at X={rx}, Y={ry}")
        send_gcode(f"G01 X{rx} Y{ry} Z{Z_SAFE}")
        send_gcode("M3")

        # Approach with IR
        z, y, touched = Z_PICK, ry, False
        while z > -350:
            z -= 5
            send_gcode(f"G01 Y{y} Z{z-15}")
            y -= (3 if conveyor_fast == 0 else 6)
            time.sleep(0.1)
            print(read_ir())
            if read_ir() == "TOUCHED":
                touched = True
                print("[IR] Object touched!")
                send_gcode(f"G01 Z{Z_SAFE}")
                break
        print(read_ir(),"...................")
        if not touched:
            send_gcode("M5")
            print(f"[INFO] Missed {cls_name.upper()}")
            send_gcode(f"G01 X{IDLE_X} Y{IDLE_Y} Z{IDLE_Z}")
        else:
            time.sleep(0.2)
            esp.write(b'1' if mapped == "left" else b'3')
            send_gcode(f"G01 X{drop_x} Y{drop_y} Z{drop_z}")
            time.sleep(1.5)
            send_gcode("M5")
            print(f"[INFO] Dropped {cls_name.upper()}")
            esp.write(b'3' if mapped == "left" else b'1')
            send_gcode(f"G01 X{IDLE_X} Y{IDLE_Y} Z{IDLE_Z}")
            with lock:
                counters[cls_name] = counters.get(cls_name, 0) +
1

        time.sleep(1.5)
        robot_busy = False


# === GUI thread ===
def run_calibration():

    # Pause main tasks
    with lock:
        shared["calibrating"] = True
        shared["run"] = False

    conveyor_stop()

    print("[CALIBRATION] System paused. Starting calibration...")

    global M_affine

    # Preset real robot coords
    robot_coords = [
        (-120,   0), (0,   0), (120,   0),
        (-120, -120), (0, -120), (120, -120),
        (-120, -240), (0, -240), (120, -240)
    ]
    cam_points = []
    real_points = []
    current_index = 0
    calibration_started = False

    def mouse_callback(event, x, y, flags, param):
        nonlocal current_index, calibration_started
        if calibration_started and event == cv2.EVENT_LBUTTONDOWN
and current_index < len(robot_coords):
            cam_points.append([x, y])
            real_points.append(robot_coords[current_index])
            print(f"[OK] Captured: Camera=({x}, {y}) ↔
Robot={robot_coords[current_index]}")
            current_index += 1

    # Open new calibration window
    # ✅ reuse the existing camera
    global cap
    cap.set(3, 640)
    cap.set(4, 480)
    cap.set(cv2.CAP_PROP_AUTO_EXPOSURE, 0.25)
    cap.set(cv2.CAP_PROP_EXPOSURE, -3)
    cap.set(cv2.CAP_PROP_GAIN, 0)
    cap_calib = cap

    cv2.namedWindow("Calibration")
    cv2.setMouseCallback("Calibration", mouse_callback)

    print("[CALIBRATION] Starting...")
    send_gcode("G28")
    send_gcode("G01 X0 Y0 Z-368")

    while True:
        ret, frame = cap_calib.read()
        if not ret:
            break

        for i, (cx, cy) in enumerate(cam_points):
            cv2.circle(frame, (int(cx), int(cy)), 5, (0, 0, 255),
-1)
            cv2.putText(frame, str(i+1), (cx+5, cy-5),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0,255,0),
2)

        msg = (f"[{current_index+1}/9] Click for
{robot_coords[current_index]}"
               if calibration_started and current_index <
len(robot_coords)
```

```
               else "Press U to start calibration" if not
calibration_started
               else "Done")
        cv2.putText(frame, msg, (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255,255,0), 2)

        cv2.imshow("Calibration", frame)
        key = cv2.waitKey(1) & 0xFF

        if key == ord("u"):
            send_gcode("G28")
            calibration_started = True
            print("[CALIBRATION] Started. Click the 9 points in
order.")

        if key == ord("q") or (calibration_started and
current_index >= len(robot_coords)):
            break


    cv2.destroyAllWindows()

    if len(cam_points) >= 3:
        cam_points_np = np.array(cam_points, dtype=np.float32)
        real_points_np = np.array(real_points, dtype=np.float32)

        M_new, _ = cv2.estimateAffine2D(cam_points_np,
real_points_np)
        if M_new is not None:
            M_affine = M_new
            print("\n[RESULT] Updated Affine Matrix:")
            print(M_affine)
        else:
            print("[ERROR] Calibration failed.")

        send_gcode("G28")

    with lock:
        shared["calibrating"] = False
    print("[CALIBRATION] Done. System resumed.")
    send_gcode("G28")

def gui_thread():
    root = tk.Tk()
    root.title("Waste Sorting Control")
    root.geometry("980x560")

    # Top buttons
    frame_top = ttk.Frame(root, padding=8)
    frame_top.grid(row=0, column=0, sticky="nw")

    def on_start():
        with lock:
            shared["run"] = True
        print("[GUI] Start pressed")

    def on_stop():
        with lock:
            shared["run"] = False
            send_gcode("G28")
            conveyor_stop()
        print("[GUI] Stop pressed")

    def on_reset():
        with lock:
            for k in shared["counters"]:
                shared["counters"][k] = 0
        update_counters()
        print("[GUI] Counters reset")

    def on_exit():
        if not messagebox.askokcancel("Exit", "Stop system and 
exit?"):
            return
        with lock:
            shared["program_running"] = False
            shared["run"] = False
        root.quit()
        root.destroy()


    ttk.Button(frame_top, text="Start", 
command=on_start).grid(row=0, column=0, padx=6, pady=6)
    ttk.Button(frame_top, text="Stop", 
command=on_stop).grid(row=0, column=1, padx=6, pady=6)
    ttk.Button(frame_top, text="Reset Counters", 
command=on_reset).grid(row=0, column=2, padx=6, pady=6)
    ttk.Button(frame_top, text="Exit", 
command=on_exit).grid(row=0, column=3, padx=6, pady=6)
    ttk.Button(frame_top, text="Calibrate", 
command=run_calibration).grid(row=0, column=4, padx=6, pady=6)

    # Mapping controls (class -> drop area)
    frame_map = ttk.LabelFrame(root, text="Class → Drop Area", 
padding=8)
    frame_map.grid(row=1, column=0, sticky="nw", padx=8, pady=6)
    options = ["Left", "Right", "Ignore"]

    mapping_vars = {}
    def make_callback(cls):
        def cb(v):
            with lock:
                shared["mapping"][cls] = v
            print(f"[GUI] mapping {cls} -> {v}")
        return cb

    row_idx = 0
    for cls in shared["mapping"].keys():
        lbl = ttk.Label(frame_map, text=cls.capitalize())
        lbl.grid(row=row_idx, column=0, sticky="w", padx=6,
pady=4)
        var = tk.StringVar(value=shared["mapping"][cls])
        mapping_vars[cls] = var
        cb = ttk.OptionMenu(frame_map, var, var.get(), *options,
command=make_callback(cls))
        cb.grid(row=row_idx, column=1, sticky="w", padx=6,
pady=4)
        row_idx += 1

    # Right side: video and counters
    frame_right = ttk.Frame(root, padding=8)
    frame_right.grid(row=0, column=1, rowspan=3, sticky="nsew")

    video_label = ttk.Label(frame_right)
    video_label.grid(row=0, column=0, padx=6, pady=6)

    counter_frame = ttk.LabelFrame(frame_right, text="Picked
counters", padding=8)
    counter_frame.grid(row=1, column=0, sticky="nsew", padx=6,
pady=6)

    counter_labels = {}
    for i, cls in enumerate(shared["counters"].keys()):
        lbl = ttk.Label(counter_frame, text=f"{cls.capitalize()}:
0")
        lbl.grid(row=i, column=0, sticky="w", padx=4, pady=2)
        counter_labels[cls] = lbl

    def update_counters():
        with lock:
            counters = shared["counters"].copy()
        for k, lbl in counter_labels.items():
            lbl.config(text=f"{k.capitalize()}:
{counters.get(k,0)}")
```

```
    def update_video():
        with lock:
            frame = shared["frame"].copy() if shared["frame"] is
not None else None
        if frame is not None:
            try:
                rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
                im = Image.fromarray(rgb).resize((640, 480))
                imgtk = ImageTk.PhotoImage(image=im)
                video_label.imgtk = imgtk
                video_label.config(image=imgtk)
            except Exception as e:
                print("[GUI VIDEO ERR]", e)
        update_counters()
        root.after(30, update_video)

    update_video()
    root.protocol("WM_DELETE_WINDOW", on_exit)
    root.mainloop()


# Start threads
t1 = threading.Thread(target=vision_task, daemon=True)
t2 = threading.Thread(target=robot_task, daemon=True)
t3 = threading.Thread(target=gui_thread, daemon=True)

t1.start()
t2.start()
t3.start()

try:
    while shared["program_running"]:
        time.sleep(0.5)
finally:
    print("Closing...")
    cap.release()
    cv2.destroyAllWindows()
    conveyor_stop()
    send_gcode("G28")
    send_gcode("M5")
    robot.close()
    esp.close()
    print("Closed.")
```