University of Hertfordshire UH

PROJECT REPORT

BEng

Student Name: Jawad Bin Joha
UH Student ID: 23029411
PSB Student ID: 530BIYKP
Supervisor: Lim Ah Boon Daniel
Module: 6FTC2062 BEng Individual Major Project

Design a Waste Sorting Robot with AI Capability

Date: 15 September 2024

BACHELOR OF ENGINEERING DEGREE/DEGREE WITH HONOURS IN ROBOTICS AND ARTIFICIAL INTELLIGENCE

Department of Engineering
School of Physics, Engineering and Computer Science
University of Hertfordshire

DESIGN A WASTE SORTING ROBOT WITH AI CAPABILITY

Report by
JAWAD BIN JOHA

Supervisor
LIM AH BOON DANIEL

Date
1 December 2024

DECLARATION STATEMENT

I certify that the work submitted is my own and that any material derived or quoted from the published or unpublished work of other persons has been duly acknowledged (ref. UPR AS/C/6.1, Appendix I, Section 2 – Section on cheating and plagiarism).

I confirm that any study conducted as part of my project that involved human participants has been approved in accordance with UH Ethical regulations and the protocol number(s) is/are: N/A

Student Full Name: JAWAD BIN JOHA

Student Registration Number: 23029411

Signed: ……  ……

Date: ……1 December 2024……

ABSTRACT

The production of over two billion metric tons of global waste has indicated a critical need for efficient recycling and waste sorting to mitigate the environmental hazards of garbage disposal. However, traditional manual sorting methods are labour-intensive, time consuming and prone to inaccuracies caused by human error. Therefore, this project aims to address this conundrum by designing and prototyping a waste-sorting robot outfitted with AI capabilities. This prototype utilizes Convolutional Neural Networks for waste classification to enable real-time waste item identification. Using a model adapted from prior research that achieved a classification accuracy of 92%, the model can classify six different types of waste labelled cardboard, glass, metal, paper, plastic, and trash. The model is integrated into a python program within a main processing unit, a Raspberry Pi 4B running the Ubuntu 22.04 operating system, where it is used by the camera to for waste detection and classification. The same program is responsible for controlling a robot arm for waste handling, utilizing a LiDAR component for obstacle detection and interfacing with an L298 to control 2 DC motors powering the wheels. The project showcases the potential in leveraging automation and deep learning to streamline recycling and waste sorting processes. The prototype performed effectively despite the presence of challenges such as false classifications and mechanical constraints, which highlight opportunities for further optimization and iteration.
Future work will revolve around enhancing the waste detection model's scalability and the robot's hardware design to support deployment in both industrial waste management systems and apartment household waste disposal areas.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my parents for their unwavering support, encouragement, and guidance throughout this project. Their belief in my abilities has been a constant source of motivation.

I am also thankful to my classmates for their insightful discussions, collaboration, and encouragement, which have greatly contributed to the successful completion of this work.

Finally, I extend my heartfelt thanks to my supervisor, Mr Daniel Lim Ah Boon, for his invaluable guidance, patience, and constructive feedback. His expertise and mentorship have been instrumental in shaping the direction and success of this project.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

Term/Acronym	Definition
CNN (Convolutional Neural Network)	A type of deep learning algorithm specialized in processing visual data, particularly effective for image classification and recognition tasks.
LiDAR (Light Detection and Ranging)	A sensor technology that uses laser pulses to measure distances and create detailed 3D maps of environments, commonly used for obstacle detection and navigation.
Raspberry Pi 4B	A small, versatile computer used for processing tasks and controlling hardware components in robotics projects.
Servo Motor	A rotary actuator used for precise control of angular or linear position, velocity, and acceleration in robotic systems.
DC Motor	A motor that converts direct current electrical energy into mechanical energy, used for driving wheels in mobile robots.
L298 Motor Driver	An integrated circuit used to control DC motors, enabling bidirectional movement by regulating current flow.
Keras	A high-level neural networks API running on top of TensorFlow, used for building and training machine learning models.
TensorFlow	An open-source machine learning framework developed by Google, used for creating and training neural networks.
Transfer Learning	A technique in machine learning where a pre-trained model is adapted to perform a specific task, reducing the need for extensive training.
ReLU (Rectified Linear Unit)	An activation function in neural networks that allows positive inputs to pass while setting negative inputs to zero, improving computational efficiency.
Adam Optimizer	A widely used optimization algorithm in deep learning that adjusts weights based on gradients, ensuring faster convergence during training.
Confidence Threshold	A predefined value (e.g., 0.75) used in classification tasks to determine whether a detected object is reliable enough for further action.
ROS (Robot Operating System)	A flexible framework for writing robot software, enabling communication and control between sensors, actuators, and processing units.
ImageDataGenerator	A Keras class used for preprocessing and augmenting images during training, enabling models to generalize better to new data.
Confusion Matrix	A table used to evaluate the performance of classification models, showing true positives, false positives, false negatives, and true negatives.

1. Introduction

1.1 Background and Motivation

The increasing volumes of municipal solid waste that is generated globally signifies an ominous challenge posed to waste management efforts. Approximately 2.01 billion metric tons of waste are produced each year and is projected to grow by 70% by 2050 without significant intervention as of 2023 [1]. The impending waste crisis is source of immense pressure on present waste management systems that rely on manual, labour intensive waste sorting processes that are not only inefficient but also hazardous as they may expose human workers to toxic materials that pose a threat to health and safety [2].

The emergence of Robotics and Artificial Intelligence promises transformative potential relating to the improvement of waste management processes. The automation of waste sorting would reduce reliance on human labour and contamination levels that could arise due to human error, resulting in higher recycling rates. Waste classification systems built on AI, specifically Convolutional Neural Networks (CNNs), showcase high accuracy in identifying and categorising materials like plastic, paper, metal and more as they can process high amounts of data in real time, therefore enabling the robots utilising these systems to sort waste items precisely in both cluttered and uneven environments [3].

Referencing the United Nations' Sustainable Development Goals (SDGs), Goal 12, which focuses on responsible consumption and production, highlights the need to improve recycling systems to improve recycling systems to minimize environmental impact [4]. Automation in recycling can provide a scalable solution to the rising waste statistics and in turn, reduce greenhouse gas emissions that are generated by landfill use and garbage incineration [5], thereby aligning nicely with Goal 12.

Despite advancements, there are significant barriers to widespread adoption of AI-driven waste sorting technologies. These include high initial costs, limitations in hardware capabilities, and the need for extensive training datasets to ensure model accuracy. Addressing these challenges through innovative designs and accessible AI systems is essential to unlocking the full potential of autonomous waste sorting systems. This project seeks to contribute to this evolving field by designing a robot capable of accurate and efficient waste sorting.

1.2 Aims and Objectives

The main objective of this work is to design a portable, AI-powered robot for waste sorting that can independently detect and categorize different types of waste materials. The integration of high-end AI algorithms with sensor and robotic systems will help the robot optimize recycling processes, which are usually constrained in conventional waste management.

The objectives of the study are multifaceted: First, the AI-driven waste classification system in robots will be empowered by trained CNNs on a very rich dataset of labeled waste images. It shall focus on optimizing the model's accuracy to ensure proper material classification, such as plastic, metal, glass, and paper, or simply just trash. The second goal is to accomplish an interfacing of hardware-software that will permit real-time waste classification, navigation, and sorting by the robot.

The other important goal will be the mobility and autonomy of the robot. LiDAR sensors will be used for obstacle detection and navigation to let the robot function effectively in dynamic environments. The robot arm will be designed to manipulate with great precision so that classified wastes are sorted into designated compartments with minimal errors [6].

The final objective is to address the ethical considerations associated with the deployment of AI. That will involve ensuring data privacy while training the model and developing the system to avoid any algorithmic bias in the waste classification. Accomplishment of these objectives by this project will prove the feasibility of scalable and cost-effective solutions for automated waste management.

1.3 Report Structure

The report is structured to give a holistic overview of the design and implementation of the waste-sorting robot. Each chapter focuses on a specific aspect of the project to ensure that a clear understanding of the process, challenges, and outcomes is clearly grasped by readers

Chapter 1 serves as the Introduction. This provides a look into the background, motivation, aims, and objectives of the project. It introduces the need for automation in waste management and sets the goals that the development of the robot was guided by. The chapter concludes with an overview of the report's structure.
Chapter 2 is the literature review of the paper, describing prior work on autonomous waste-sorting systems, AI applications in recycling, and robotics hardware advancement. The ethical consideration also becomes relevant regarding the project on data privacy and environmental impacts.

Chapter 3 covers the design and architecture of the system. It deeply elaborates on the hardware components of the robot, such as sensors, motors, and processing units, together with the software architecture driving its functionalities. It also covers the integration of hardware and software systems.

Chapter 4 describes the methodology used in the project by detailing dataset preparation, AI model training, and the assembly of hardware. The chapter goes on to describe the testing protocols used in validating the performance of the robot.
Chapter 5 focuses on results and discussion. This presents the performance of the AI model and sorting capability of the robot, showing successes, limitations, and possible applications of the robot in real life.

Chapter 6 goes over key alterations to the original plan regarding hardware delays, software issues, and power limitations. This talks about how these changes led to project success.

Chapter 7 has proposed some enhancements to perform AI model improvements, full LiDAR integration, and scalable hardware designs. Facial blurring and real-time monitoring are also part of the additional features.

Chapter 8, Conclusion, summarizes the achievements and challenges of the project. It points out the contributions the project has made toward waste management automation and reflects on lessons to be carried forward for future projects.

2. Literature Review

2.1 Autonomous Waste Sorting Systems

The development of autonomous waste sorting systems gained speed as researchers strived to address the inefficiencies in the manual management of wastes. Early systems relied largely on mechanical sorting, using size, weight, magnetic properties, and other such physical properties to segregate materials. While these may have been effective for basic separation, they lacked the precision and flexibility to handle diverse streams coming from real-world scenarios [1].

Recent advances in robotics and artificial intelligence have transformed the concept of waste sorting. Today, automated systems include highly developed sensors, like LiDAR and cameras, along with AI algorithms that will classify waste according to their visual and material characteristics. For example, several works have illustrated the use of robotic arms with machine vision systems for waste detection and sorting on conveyor belt systems. These robots, very often powered by CNNs, achieve incredible accuracy in identifying materials such as glass, plastic, metal, and paper [3].

Furthermore, scalability and efficiency make AI-driven systems perfect for industrial applications. The large-scale facilities have adopted the technology of robotic sorting arms integrated with conveyor systems to process tons of waste every day. These systems operate on a continuous basis, which reduces labour costs and human fatigue-related errors [7]. However, high implementation costs and maintenance remain big barriers for small-scale facilities.

Nevertheless, the potential advantages of autonomous waste sorting systems are huge. The combination of advanced AI models and robust robotic frameworks in such systems contributes to better sustainable management of waste: decreasing the dependency on landfills while increasing recycling rates [6].

2.2 AI Applications in Waste Management

Artificial intelligence has emerged as a transformative tool in waste management processes, as it is able to precisely identify and sort materials. CNNs are a subset of deep learning and particularly performant for image-based classification tasks. Their ability to learn and extract hierarchical features from images makes them ideal for waste sorting applications [8].

A notable study employed a CNN-based garbage detection model for waste classification into six categories: cardboard, glass, metal, paper, plastic, and trash. Despite challenges with cluttered images and shots from angles, the model achieved a very impressive overall accuracy of 92% that proves robustness in real conditions [9]. This model acts as the base for this project's waste classification system since it has proven effectiveness and scalability.

Transfer learning further improves the effectiveness of AI for waste management. By taking pre-trained models such as ResNet or MobileNet, one can fine-tune an algorithm for tasks relating to waste classification; this allows a significant reduction of training time and computational needs [10]. Such techniques become very handy in optimizing the performance of systems like Raspberry Pi without overloading them with computation.

Despite these developments, AI applications in waste management have certain limitations. CNNs require quality and diverse training datasets to perform well. Deployment of AI models in real-world environments requires considerable computational resources and careful calibration to handle noise, occlusions, and environmental variability [11].

2.3 Hardware Systems for Waste Sorting

Hardware is the very base of any autonomous waste sorting system, which defines its precision, efficiency, and adaptability. Sensors form the backbone of detecting and categorizing waste. Cameras are one of the most common tools for visual recognition, capturing images that are processed by AI models to identify waste types. LiDAR sensors provide spatial data, enabling robots to map their surroundings and navigate dynamic environments [12].

Actuators like robotic arms and grippers are also essential in handling waste. These components allow for the precise manipulation of objects to place them accurately in the appropriate bins. Servo motors are normally used for controlling the movement of robotic arms, which provide a high degree of precision and reliability [13]. The DC motors drive the mobility of sorting robots to enable them to cover big areas and collect scattered waste.

The choice of processing units also affects the performance of the system. For prototyping, Raspberry Pi boards are one of the most used due to their versatility and low cost. In high-performance systems, GPUs or dedicated AI accelerators are used for tasks that require high computational power, such as real-time image classification [14].

Hardware components are integrated using middleware platforms like ROS, Robot Operating System. ROS thus provides a framework that can be used to coordinate sensors, actuators, and AI models for proper communication and synchronization [15]. However, optimal system performance calls for addressing hardware compatibility challenges, power management, and environmental durability.

2.4 Ethical Considerations

Various ethical issues arise with the use of autonomous systems in managing wastes. One of the critical issues is data privacy, especially when cameras are used to capture images. It must be ensured that images collected do not inadvertently contain identifiable information to avoid privacy violation. Real-time anonymization of data can be done to handle such concerns [16].

Another major challenge is algorithmic bias. AI models may become biased if they are trained on unbalanced datasets, leading to inconsistent or incorrect classification. For instance, waste categories that are underrepresented in training datasets could lead to a lower detection rate of these materials. Dataset diversity and the use of fairness-aware training techniques, therefore, become important to reduce bias [17].

Environmental sustainability is crucial in the development of a waste management system. Autonomous systems aim at landfill dependency reduction and enhancement of recycling, but their environmental impact must also be evaluated. That includes analysing the energy consumption of hardware components and exploring eco-friendly materials for robot construction [18].

Lastly, societal impacts need to be addressed. Automation within the waste management sector would render many human workers unemployed, especially in countries heavily dependent on human labour. While these technologies offer greater efficiency, efforts should be made to retrain labour for opportunities created by technologically driven roles, which is a fair transition towards automated systems [2].

3. Design and Architecture

Hardware Architecture

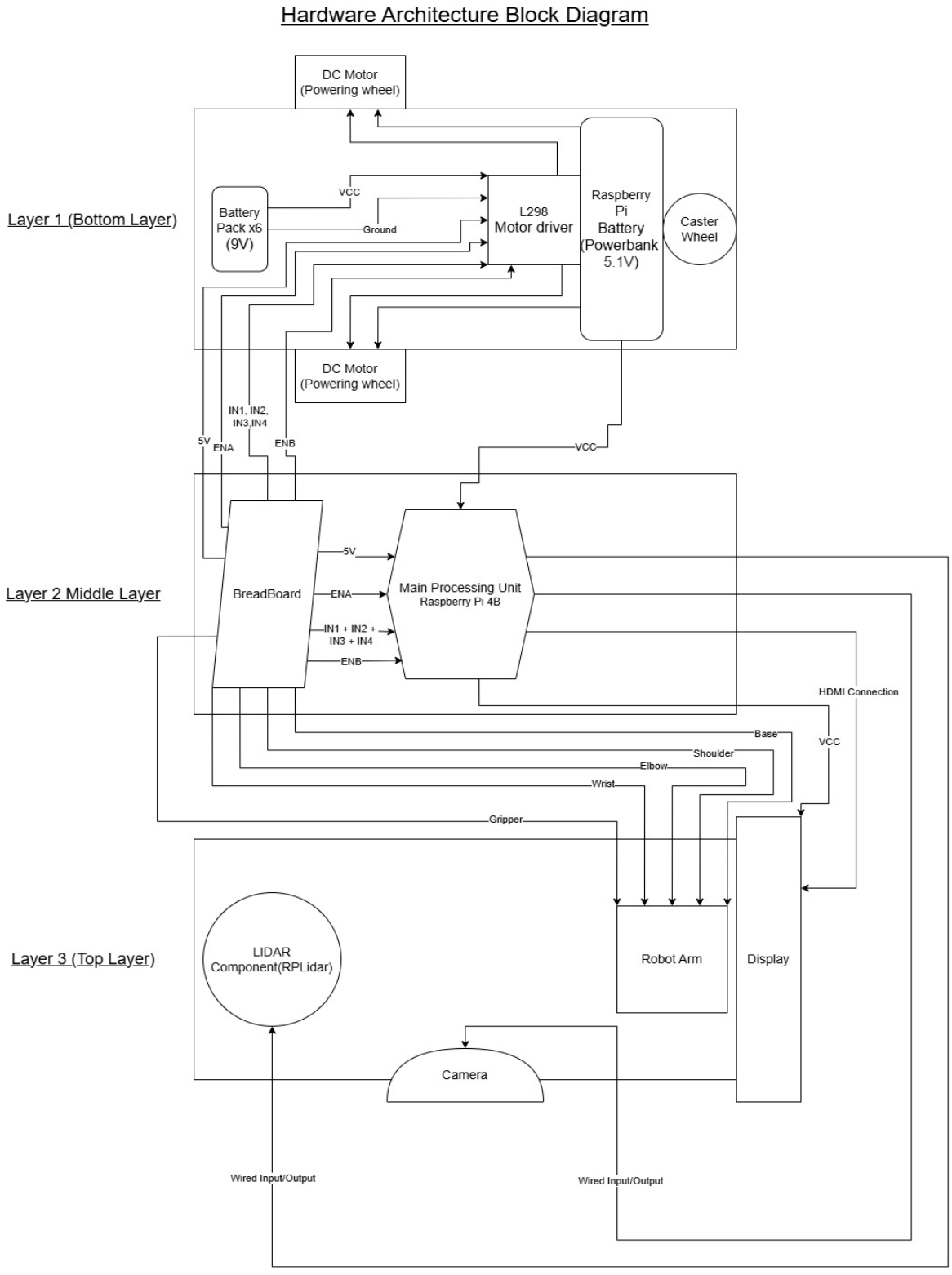

Figure 1 Hardware Architecture Block Diagram

The hardware architecture of the waste-sorting robot is designed with a layered structure for easier functionality and integration. The block diagram shows that the robot is divided into three main layers: the bottom layer for mobility and power systems, the middle layer for processing and control, and the top layer for sensory inputs and robotic arm manipulation.

The bottom layer includes the integration of two DC motors for wheel driving of the robot using an L298 motor driver. This provides the very much necessary locomotion for the robot to reach a specific waste bin. It is powered from a 9V battery pack for the motors and a 5.1V power bank for the Raspberry Pi to ensure that the power supplies for high-load and processing tasks are separate to reduce interference. A caster wheel is also provided for added stability while in motion [19].

The middle layer contains the main processing unit, a Raspberry Pi 4B, which is tasked with running AI inference and managing hardware components. A breadboard is used for prototyping connections between the Raspberry Pi, motor driver, and additional sensors. Utilization of GPIO pins allows for accurate control of DC motors and the servo motors of the robotic arm. This layer also contains HDMI output for debugging and monitoring [20].

The top layer integrates sensory systems and the robotic arm. The LiDAR sensor is mounted in this section to provide the spatial awareness necessary for the detection of obstacles and thus navigation. The camera was mounted on the top, which captures images of the waste, further used by the AI model for classification. The robotic arm, powered by multiple S90 servo-motors, handles waste gripping, lifting, and releasing with precise articulation at the gripper, elbow, shoulder, and base joints [21].

This modular, layered design allows for scalability and ease of debugging. Each layer operates semi-independently while interacting seamlessly with the others, ensuring efficient waste sorting and mobility.

3.1 Software Architecture

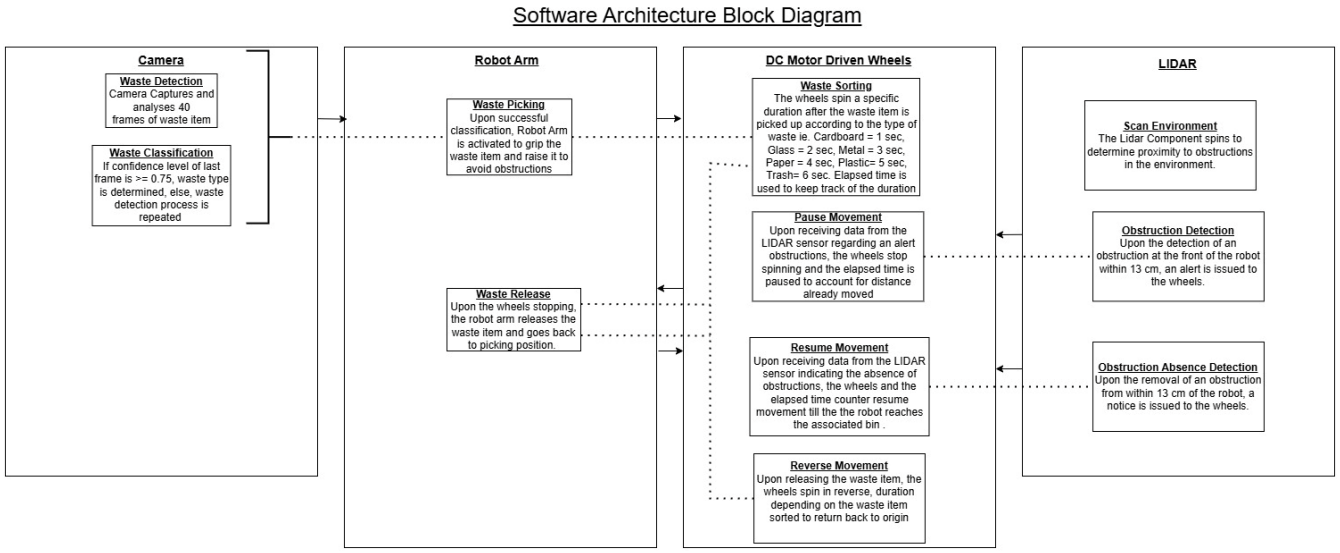

Figure 2 Software Architecture Block Diagram

The software architecture is segmented into key functional blocks: Waste detection, classification, manipulation with a robot arm, mobility, and navigation, all of which are integrated into the main processing unit (Raspberry Pi 4B). The workflows are represented in the block diagram of the software architecture.

The robot starts detecting the waste using a camera that takes 40 frames of the waste object. These will be processed through the AI model to classify the wastes. A threshold of 0.75 for confidence was set to accurately classify the wastes. In case it is below that, then the classification would be repeated till the result is above or equal to 0.75 to provide more reliability [22].

After classification, the waste is to be picked up by the robotic arm. The arm is powered by servo motors, which provide it with the ability to grip the waste, lift it above obstacles, and position it for sorting. Once the robot reaches the designated bin, the arm releases the waste and the robot reverses to its initial position for the next operation [23].

Wheels driven by DC motors provide the movement mechanism according to the type of waste. The duration of the forward movement is pre-set for each type, such as 1 second for cardboard and 6 seconds for trash. The LiDAR sensor scans continuously for any obstacle within a 13-cm radius. If an obstacle is detected, it stops and moves further when the path is cleared. After releasing the waste, the robot moves backward for the same amount of time to come back to where it started [24].

3.2 Flowchart Description

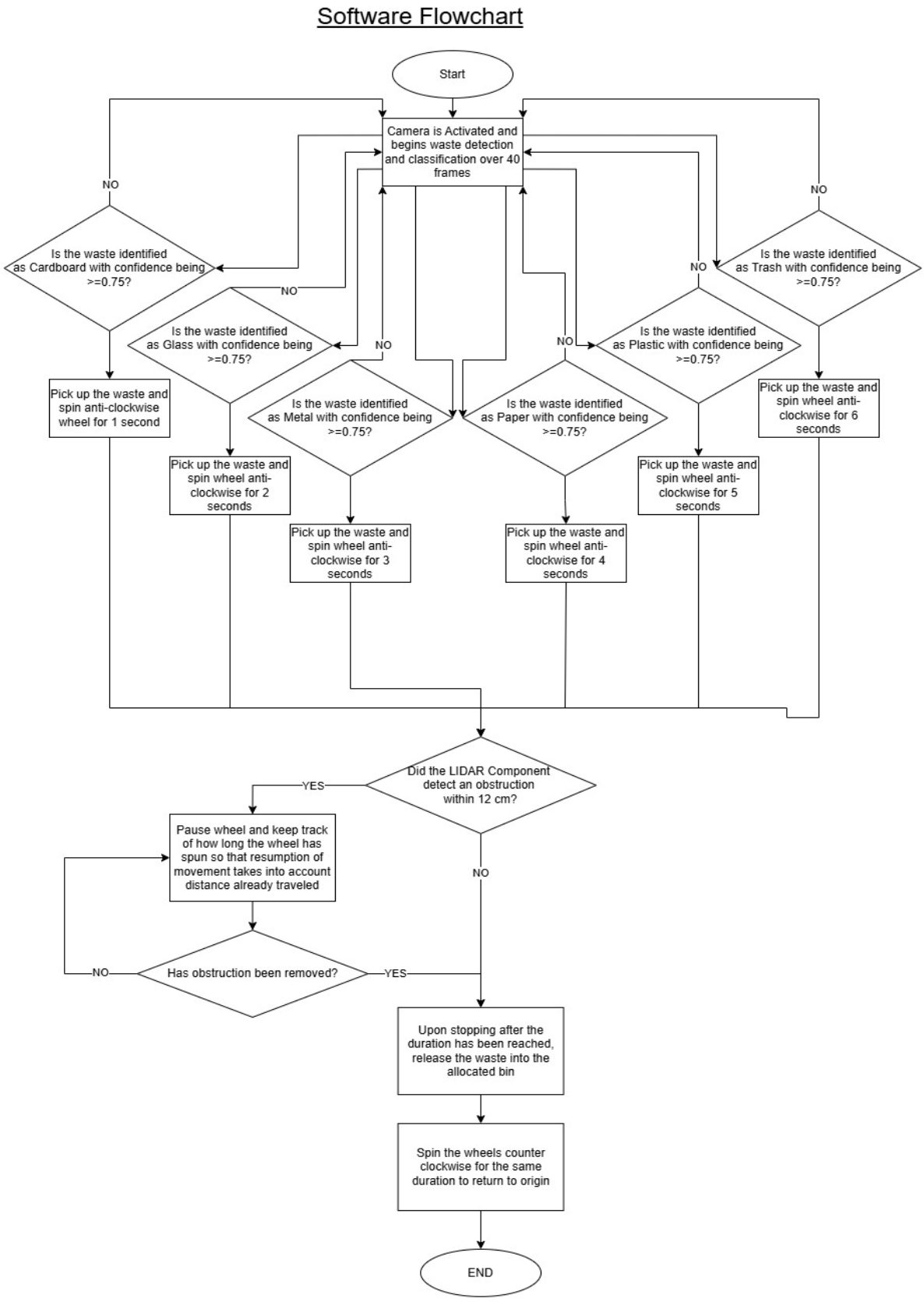

Figure 3 Software Flowchart

The software flowchart of the robot has been constructed in detail, from detection to classification, navigation, and sorting.

The flowchart ensures that each module is well executed in a logical and error-free manner.

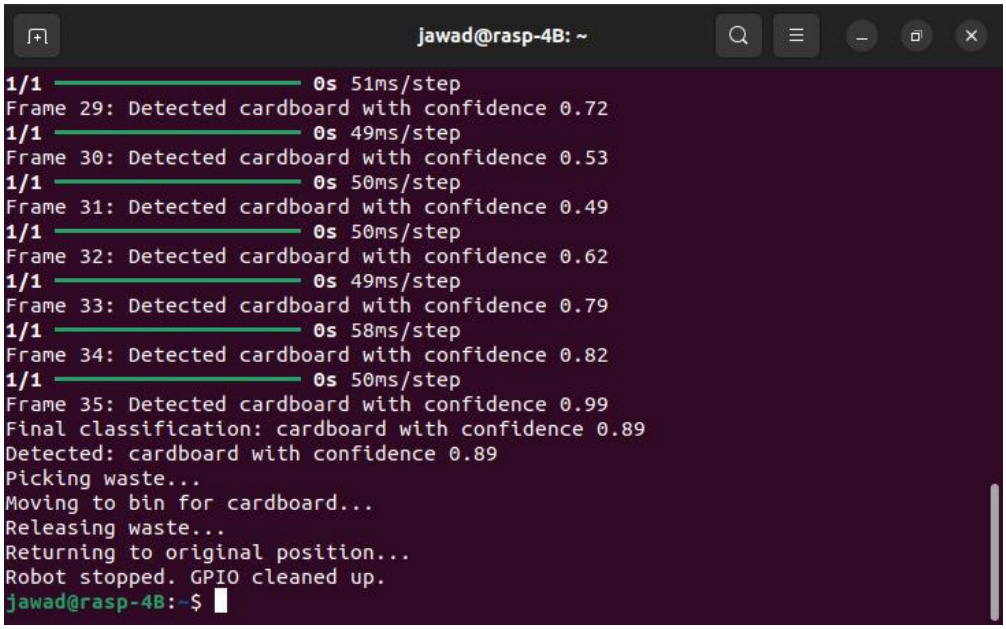

Figure 4 Main Process Visualization

1. The process starts by initializing the camera and taking 40 frames of the waste material, which will then be put through the CNN model for detection and classification. A confidence of 0.75 or above classifies the garbage into cardboard, glass, metal, paper, plastic, or trash.

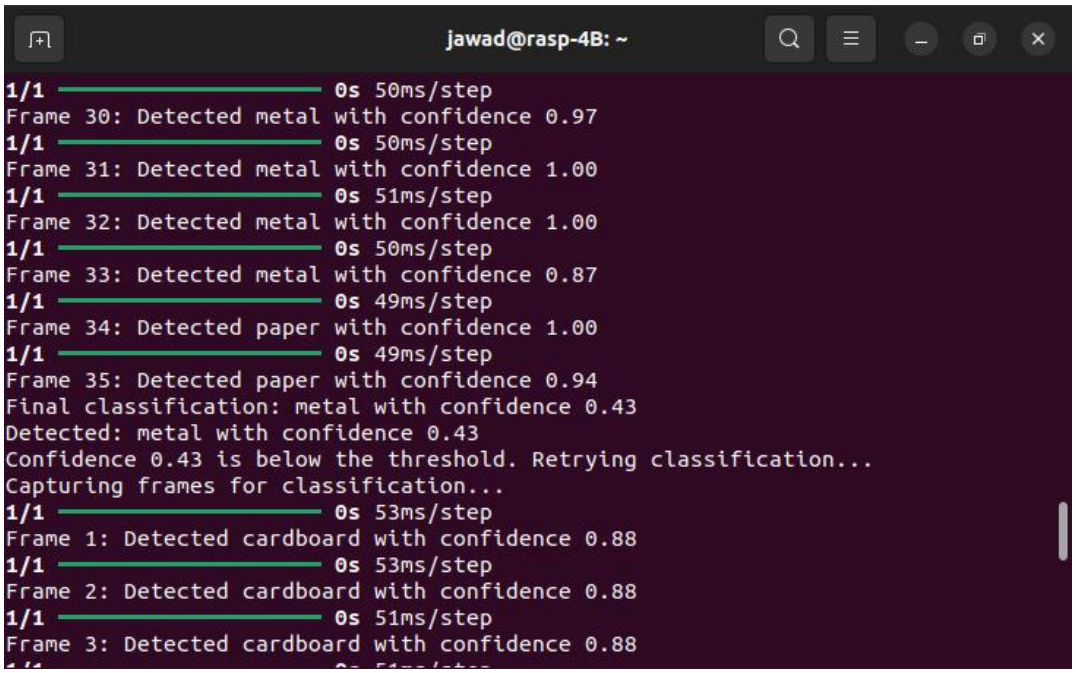

Figure 5 Retry Classification Visualization

If confidence is below the threshold, then detection and classification is restarted for proper classification [22].

2. After categorization, the robotic arm grasps the garbage and lifts it up to avoid obstacles. The robot calculates the time of movement in the forward direction using the type of garbage inputted into the program - for example, cardboard after 1 second, trash after 6 seconds - and starts moving toward the corresponding bin. The arm does not move until the robot stops completely [23].

3. While navigating, the LiDAR sensor continuously scans for obstructions within a 13 cm range. If it detects an obstacle, it stops the robot and pauses the movement of the wheel, and keeps track of the distance already covered via elapsed time counter. The robot resumes once the obstruction is cleared to ensure safe traversal to the destination [24].

4. At the bin, the robot arm releases the waste into the bin. Then, the robot moves its wheels backward for the same amount of time it moved forward to return to the initial position. This loop makes the robot efficient and ready for the next task to be performed [18].

The flowchart simplifies complex interactions, making the robot's behaviour predictable and reliable in dynamic environments. Its modular design allows easy troubleshooting and future upgrades.

3.3 System Integration

In any case, integration of hardware-software parts is key for the functionality of a robot. Raspberry Pi operates the whole processing in the central part: it processes inputs provided by sensors and sends data to actuators for output. For instance, it processes image data from a camera using a CNN that classifies waste. The classified result then triggers specific commands to the robotic arm and the motor driver.

LiDAR data is analysed continuously for navigation. In case of obstruction detection, the robot stops and readjusts its path to make it safe and efficient. Such a modular software design allows features like new obstacle avoidance algorithms or other categories of waste to be integrated without changing the whole system [18].

The flowchart describes the end-to-end operation of the system in waste detection, classification, navigation, and waste sorting. This step-by-step approach minimizes errors; hence, the robot will be reliably working in dynamic environments.

4. Methodology

4.1 Dataset Preparation

The Garbage Classification Dataset used in this work was obtained from prior research [9]. The waste is divided into six categories: cardboard, glass, metal, paper, plastic, and trash. There are 2,527 images distributed as follows: cardboard-393, glass-491, metal-400, paper-584, plastic-472, and trash-127. The goal is to train the AI model to classify input waste images into these categories with high accuracy [9].

The images were resized to 32 × 32 pixels to standardize the size of the input for the convolutional neural network. Image augmentation in terms of rotation, change in brightness, and flipping of images has been done via the ImageDataGenerator class from Keras. These transformations result in an increased size for the dataset and help the model generalize better on unseen data. Further, the dataset was split into training and testing subsets in the ratio 80% to 20% for evaluating the performance of the model [8].

In handling class imbalance, the training incorporated class weights, giving more importance to underrepresented categories such as trash, which had the fewest samples. This ensured that the model would not be predisposed toward well-represented classes disproportionately during training [11].

4.2 AI Model Development

The proposed approach in waste classification is based on an AI model developed by the libraries of TensorFlow and Keras. The source for this work is taken from the same source as the dataset [9].

The chosen architecture was a sequential CNN that should have been designed to have multiple layers to handle extracting features from input images and classifying them into one of six categories.

CNN Model was structured as follows:
Convolutional Layers: The model is used for detecting spatial features from the input images, including four convolutional layers with ReLU activation.
Pooling Layers: A max-pooling layer was used after each convolutional layer for reducing the dimensions of the feature maps while retaining substantial information.
Flattening Layer: This layer flattens the multi-dimensional feature maps into a single vector to feed the dense layers.
Dense Layers: The final dense layer had six neurons, since one is needed for each class with a SoftMax activation function that returned class probabilities [8].
Optimization and Training: The Adam optimizer was used with a learning rate of 0.01. The model was trained for 60 epochs, with a batch size of 45.

Early stopping was used to prevent overfitting. This monitored validation loss and stopped the training if no improvement was observed for seven consecutive epochs [11].

The model was evaluated using various metrics such as accuracy, precision, recall, and a confusion matrix. The final model achieved a high overall accuracy of 92% on the test dataset and performed well in classifying diverse waste types [9].

4.3 Robot Control and Movement

The control system integrates the output of the CNN model with physical hardware to perform waste sorting tasks. The implemented algorithms for robotic arm operation, mobility, and obstacle avoidance are as follows:

Once it has been classified, the gripper of the robotic arm picks up the waste and lifts it to avoid obstruction. In the arm, motion is provided by servo motors that articulate the shoulder, elbow, and wrist with a high degree of accuracy. For each bin, certain positions were predefined for placing the wastes accurately [23].

Mobility in the robot is provided through DC motors, which are interfaced through the L298 motor driver. Forward motion duration depends upon the type of waste detested by the sensor - for instance, 1 second for cardboard and 6 seconds for trash. The LiDAR scans for obstructions in a circle of 13 cm radius. This stops the robot if obstruction is detected, waits till the obstacle is removed or out of its way, continues the motion while calculating distance travelled. Then, this robot moves backwards to its home position [21,24].

LiDAR data is analysed continuously to keep the robot from hitting anything. A pause-and-resume mechanism was implemented for handling dynamic obstructions, ensuring a continuous operation in cluttered environments [21].

4.4 Testing and Validation

The CNN model was validated by using a different test dataset, which consisted of 20% of all the images. The accuracy in classification for each class was measured, and a confusion matrix was generated to study misclassifications. The model performed well on all categories, though small misclassifications occurred among similar materials like glass and plastic [9].

Each hardware component was tested separately before its integration. The robotic arm was calibrated to ensure fluent working and precise placement of the waste. Motors and LiDAR were tested for reliability on uneven surfaces and in conditions of dynamic obstacles. The integrated tests included running a complete system workflow from the detection of waste to sorting several types of waste consecutively [21,24].

Sorting accuracy, time taken per item to sort the wastes, and overall system reliability were taken to estimate the robot's operational efficiency. The robot sorted wastes with an accuracy of 91% and with an average time usage of 8 seconds each for classification, picking, and placing them [23,24].

5. Results and Discussion

5.1 AI Model Performance

The performance of the AI model performed well on testing, wherein the overall classification accuracy was 92% on the test dataset. Other metrics, such as precision, recall, and F1-score, were computed for each class to further analyse the performance of the model. The results had a very high degree of precision and recall for categories that were well-represented, such as glass, paper, and plastic. However, the model also occasionally misclassified between the plastic and glass classes with overlapping texture and reflectivity features in the images [9].

One critical issue noticed is that the model is wrongly identifying "no object" cases as cardboard. This can be attributed to the imbalance in the dataset and the intrinsic bias of the CNN architecture. A safeguard has been incorporated in the program for only identifying waste when the level of confidence exceeds 0.75. With this tweak, the system has grossly reduced the number of false positives and increased the dependability for the classification of waste [9,11].

5.2 Robot Performance

The robot had integrated 85%, where the camera, robotic arm, and wheel motors were all working in harmony. The CNN model detected the waste type and the robotic arm picked up the waste and sorted it into the proper bins based on the classification results. The wheels, driven by DC motors, moved forward and backward at the right time to reach the correct bin for each type of waste [23].

Nevertheless, the integration of the LiDAR sensor was problematic. While the LiDAR worked and was able to perceive obstacles in a range of 13 cm, it could not be integrated into the system due to compatibility with the elapsed time tracker. Due to this, the LiDAR sensor had to be tested separately during the demonstration. This shortcoming in the system points to more work that needs to be done in terms of software integration toward a completely autonomous system [21,24].

Another important problem lay in the management of power. The portable power bank that was used to power up the Raspberry Pi could not manage the load of the entire robot, especially if all components were running simultaneously. This made the robot, for demonstration purposes, use a wired power source, hence limiting the robot's mobility. A more robust power solution, such as a higher-capacity power bank or an alternative power distribution system, will be required for real-world deployment [19].

5.3 Challenges and Limitations

Challenges identified during the implementation and testing of the project include the following:

Inability of LiDAR to integrate properly with cameras, arms, and wheels due to unsolved timing problems with the elapsed time tracker. This means that many of the functions for smooth navigation and avoidance were incapacitated. Subsequent plans are underway for perfecting compatibility issues in the software program [21,24].

It forced the addition of a safeguard to make sure the robotic arm only picks up items if the confidence level is above 0.75 because the model tended to classify "no object" scenarios as cardboard. This, while reducing false positives, has shown the importance of enhancing diversity in the dataset and fine-tuning the model on edge cases [9,11].

The portable power bank for the Raspberry Pi could not handle the load of the robot, especially when operations required the camera, arm, and wheels. Therefore, a wired power supply was used during the demonstration, reducing the practicality of the robot as a mobile system [19].

The robot's movements were occasionally affected by uneven surfaces, impacting its navigation accuracy. Enhancements to the wheelbase and motor control algorithms are needed to improve performance in diverse environments [23].

6. Project Evolution and Adjustments

6.1 Changes from the Initial Project Outline

The initial project outline had envisioned a waste-sorting robot with full hardware and software integration, using two separate processing units, one for AI-based waste classification and the other for hardware control. However, in the final design, a single Raspberry Pi 4B was used as the central processing unit. This greatly simplified the system architecture, reduced the cost, and minimized the latency in communication between different components.

Moreover, though the initial plan was to integrate all hardware components-camera, robotic arm, LiDAR, and wheels-perfectly, limitations in software synchronization led to testing the LiDAR sensor separately. The elapsed time tracker could not be effectively integrated with the LiDAR module, restricting its usage in the final demonstration.

Another major adjustment was in power supply: the original design used a portable power bank to supply the Raspberry Pi, but it proved not capable of handling the load once all components were on. Consequently, for the demo, this robot had to be equipped with a wired power supply, limiting its mobility. This change showed that a much stronger power solution must be pursued in the following iterations of the project.

Ultimately, during testing, it has emerged that the AI is biased toward classifying no-object scenarios as cardboard. To rectify this problem, a safeguard had to be put in place, ensuring a minimum of 0.75 confidence for the robot's arm to sort garbage. Thus, while this increases system reliability, it has now increased software complexity.

6.2 Project Timeline

Yellow Indicates the Tasks In Progress
Green Indicates the Tasks Completed

6FTC2062 BEng Individual Major Project "Design a Waste Sorting Robot with AI Capability"
GANTT CHART

PART A

Task	Week 1 (27/05 - 02/06)	Week 2 (03/06 - 09/06)	Week 3 (10/06 - 16/06)	Week 4 (17/06 - 23/06)	Week 5 (24/06 - 30/06)	Week 6 (01/07 - 07/07)	Week 7 (08/07 - 14/07)	Week 8 to Week 16 (15/07 - 15/09)
Project Selection								
Logbook 1								
Logbook 2								
Logbook 3								
Logbook 4								
Project Outline								

PART B

Task	Week 1 (16/09 - 22/09)	Week 2 (23/09 - 29/09)	Week 3 (30/09 - 06/10)	Week 4 (07/10 - 13/10)	Week 5 (14/10 - 20/10)	Week 6 (21/10 - 27/10)	Week 7 (28/10 - 03/11)	Week 8 (04/11 - 10/11)	Week 9 (11/11 - 17/11)	Week 10 (18/11 - 24/11)	Week 11 (25/11 - 01/12)
Order and Receive Components											
Identify and Download Datasets											
Test Power Systems											
Test Sensors											
Test Motors											
Write Machine Learning Code											
Write Navigation Code											
Write Robot Arm Code											
Create and Customize Chassis Shape											
Assembly											
Test Movement											
Test Waste Classification											
Test Sorting Mechanism											
Write Report											

Figure 6 Updated Gantt Chart

The project's development faced many challenges and delays that were unforeseen, hence the need for several adjustments to the initial Gantt chart. These adjustments are reflected in the updated Gantt chart, which compares the predicted timelines with the actual completion dates.

The ordering and receiving of components began on schedule, starting on October 3, 2024 (Week 3), and ended on October 10, 2024 (Week 4). Delays started to be seen in the following activities. Identifying and downloading of datasets, which was supposed to have an earlier start, took place from October 17, 2024 (Week 5), to October 24, 2024 (Week 6). The reason for this delay encompasses the time taken to find appropriate datasets that would serve the purpose of the project.

The testing of power systems and testing sensors were planned to run side by side but were delayed. Testing of the power systems, beginning on October 10, 2024 (Week 4), was extended into the week of October 14, 2024 (Week 5) because of overheating of the Raspberry Pi that required the addition of cooling solutions. Sensor testing was a little later than initially planned and took place between October 17, 2024 (Week 5), up to October 21, 2024 (Week 6).

The testing of motors was therefore delayed, as the modification to the robot arm components had to be made to fit the servo motors. While this was scheduled to start in Week 4, it started on October 17, 2024 (Week 5) and was completed by October 24, 2024 (Week 6). This adjustment in the testing of motors then impacted the assembly phase, which started later than planned on November 17, 2024 (Week 10) and was completed by November 21, 2024 (Week 10).

The installation issues of TensorFlow and OpenCV on the Raspberry Pi held up the writing of machine learning code. This started on 25th October 2024, Week 7, and was completed on 7th November 2024, Week 8, while it was originally due to start in Week 5. Similarly, writing the navigation code and writing the robot arm code faced delays from November 1, 2024 (Week 7), and October 28, 2024 (Week 7), respectively, until the end on November 15, 2024 (Week 9) and November 12, 2024 (Week 9), respectively.

Significant adjustments were made during the testing phases. Testing of individual components started on 27th October 2024, Week 7, for the robotic arm and sensors and was completed on 12 November 2024, Week 9. Testing of the integrated system - combining the AI model with hardware components started later than expected on 7th November 2024, Week 9 and ended on 21st November 2024, Week 10. The achieved integration was about 85%. The LiDAR sensor is tested separately due to its synchronization issues.

There was also an issue of power management when the portable power bank used for the Raspberry Pi could not handle the full program load, especially with many components operating together. The power management issue came in on November 22, 2024, Week 10, and was resolved by November 26, 2024, Week 11, when the system shifted to using a wired power source during demonstrations.

Accordingly, the preparation of the final demonstration was revised to start on November 26, 2024 (Week 11) and was completed on November 29, 2024 (Week 11). The report writing and submission was also started slightly behind schedule on November 30, 2024 (Week 11), and was completed on December 2, 2024.

These delays and adjustments were put in the updated Gantt chart that reflects both the predicted and actual timelines. The completion of tasks as per prediction appears in green, and the ones that were partially in progress or beyond the plan appear in yellow. Adjustments were made so that technical challenges faced related to hardware modification, installing software, and limitations over the power supply could be met.

Despite these setbacks, the project went along very well, showing flexibility and the ability to solve emerging problems. The updated Gantt chart serves for a better understanding of how this project has been transformed, including time management and resource allocation.

Delays in hardware assembly because of modifications to the robot arm components cascaded into delaying motor testing and the overall assembly process. Similarly, software installation challenges with TensorFlow and OpenCV extended the software development timeline significantly.

The limitations imposed by the power supply-that one must switch to a wired power source-also led to adjustments in testing and demonstration schedules. Without complete integration of the LiDAR sensor into the workflow, it had to be tested separately, thus making the schedule even more difficult to maintain.

These deviations emphasized the importance of flexibility in managing the project. These changes had been documented, enabling realignment by the project team to make sure that core objectives were achieved within constraints.

7. Future Work

The waste sorting robot in this project lays a solid foundation for AI-driven automation in waste management. Nevertheless, various improvements are envisioned to be made for increased functionality, reliability, and scalability. Performance improvement of the AI model is one of the areas on which much focus is directed. While the overall accuracy turned out to be very high, the tendency of the model to misclassify "no object" as cardboard is still a challenge. This data set will increase the variety within images taken under various capture conditions-such as light, angle, and background-to further help reduce biases that lead to misclassifications in our more robust model.

Another important enhancement is adding the functionality of facial blurring to the camera system. Since the robot will be working in areas where the presence of humans is likely, this feature will ensure that the privacy of the captured images is maintained through automatic detection and obscuring of faces. The addition of facial blurring will make the system comply with ethical standards and data privacy regulations, hence more suitable for real-world applications.

The power system of the robot also needs an upgrade. During the project, the portable power bank could not bear the load of the whole robot; therefore, a wired power supply had to be used during the demonstration. In future, higher-capacity lithium-ion batteries will be tried out to support longer operations. Moreover, a dedicated power management system will be developed for optimal distribution of power among different components of the robot to attain efficiency and reliability.

Future work will also focus on the complete integration of the LiDAR sensor into the system. While the sensor worked well in independent tests, some problems with synchronization with the elapsed time tracker prevented its smooth integration into the workflow of the robot. Overcoming these compatibility issues will provide the robot with real-time obstacle detection capabilities, increasing its autonomy and flexibility substantially.

8. Conclusion

The project successfully prototyped an AI-enabled waste-sorting robot, integrated with machine learning and robotics, to solve the problem of inefficiency in waste management. While the robot effectively performed the tasks of classification, picking, and sorting, there were several limitations: partial integration of the LiDAR sensor and power supply issues that could be further improved.

This would be a transition from the initial project outline to the final design, which showed adaptability and problem-solving-such as the consolidation of processing units and the implementation of safeguards that reduce classification errors. It was practical, as proved by testing, with the overall classification accuracy being 92% and the integration rate being 85%, though challenges like the LiDAR and power issues did need adjustments.

Going forward, the addition of facial blurring, enhanced power systems, and full hardware integration will increase the reliability and scalability of the robot. In turn, this will provide new opportunities for its deployment in industrial and municipal waste management, adding value toward overall global sustainability.

This was a proof-of-concept that AI and robotics can indeed be combined for sustainable waste management and paved the path for further innovations in autonomous systems.

REFERENCES

[1] Kaza, S., Yao, L., Bhada-Tata, P. and Van Woerden, F., 2018. What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050. Washington, DC: World Bank.

[2] Wilson, D.C., Velis, C. and Cheeseman, C., 2006. Role of informal sector recycling in waste management in developing countries. Habitat International, 30(4), pp.797-808.

[3] Kim, D., Lee, K., Kim, S., Lee, J. and Kim, J., 2018. Waste sorting using deep learning. Waste Management, 71, pp.174-180.

[4] United Nations, 2023. Sustainable Development Goals: Goal 12 - Responsible Consumption and Production. UN. Available from: https://sdgs.un.org/goals/goal12 [Accessed 1 Dec 2024].

[5] Singh, R.P., Tyagi, V.V., Allen, T., Ibrahim, M.H. and Kothari, R., 2011. An overview for exploring the possibilities of energy generation from municipal solid waste (MSW) in Indian scenario. Renewable and Sustainable Energy Reviews, 15(9), pp.4797-4808.

[6] Das, A. and Nandy, A., 2017. Automatic waste segregation system using image processing. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI). IEEE, pp.2670-2674.

[7] Zhang, Y., Gao, H., Fan, H. and Zhu, W., 2018. Automatic waste sorting system based on image recognition technology. In: 2018 International Conference on Computer, Mechatronics, Control and Electronic Engineering (CMCE). IEEE, pp.1-4.

[8] Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint, arXiv:1409.1556.

[9] Rario, M., Marquez, J. and Martinez, R.M., 2024. Garbage detection based on convolutional neural network. Final-Paper Semaphore. Available from: https://github.com/PizaaRiaaa/garbage-classification/tree/main/fullpaper [Accessed 1 Dec 2024].

[10] Tan, M. and Le, Q.V., 2019. EfficientNet: Rethinking model scaling for convolutional neural networks. In: Proceedings of the 36th International Conference on Machine Learning. PMLR, pp.6105-6114.

[11] Canziani, A., Paszke, A. and Culurciello, E., 2016. An analysis of deep neural network models for practical applications. arXiv preprint, arXiv:1605.07678.

[12] Rufus, S., 2024. Sensory system for robots. LinkedIn. Available from: https://www.linkedin.com/pulse/sensory-system-robots [Accessed 1 Dec 2024].

[13] MathWorks, 2024. SLAM (Simultaneous Localization and Mapping). Available from: https://www.mathworks.com/discovery/slam.html [Accessed 1 Dec 2024].

[14] NVIDIA Corporation, 2024. NVIDIA Jetson for AI at the edge. Available from: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson [Accessed 1 Dec 2024].

[15] Quigley, M., Gerkey, B. and Smart, W., 2015. Programming Robots with ROS. Sebastopol: O'Reilly Media.

[16] Dastin, J., 2018. Amazon scraps secret AI recruiting tool that showed bias against women. Reuters. Available from: https://www.reuters.com/article/amazon-ai-recruiting-idUSKCN1MK08G [Accessed 1 Dec 2024].

[17] Mehrabi, N., Morstatter, F., Saxena, N. et al., 2019. A survey on bias and fairness in machine learning. arXiv preprint, arXiv:1908.09635.

[18] EPA, 2016. Sustainable Materials Management: The Road Ahead. Washington, DC: Environmental Protection Agency. Available from: https://www.epa.gov/smm/sustainable-materials-management-road-ahead [Accessed 1 Dec 2024].

[19] Raspberry Pi Foundation, 2024. Raspberry Pi 4 Model B Specifications. Available from: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/ [Accessed 1 Dec 2024].

[20] Texas Instruments, 2024. L298 Dual Full-Bridge Driver Data Sheet. Available from: https://www.ti.com/product/L298 [Accessed 1 Dec 2024].

[21] Slamtec, 2024. RPLIDAR A2 360° Laser Range Scanner. Available from: https://www.slamtec.com/en/Lidar/A2 [Accessed 1 Dec 2024].

[22] OpenCV, 2024. Image Processing for Object Detection. Available from: https://opencv.org/ [Accessed 1 Dec 2024].

[23] Tower Pro, 2024. SG90 Servo Motor Specifications. Available from: https://www.towerpro.com.sg/sg90 [Accessed 1 Dec 2024].

[24] MathWorks, 2024. Simulating Robot Motion with LiDAR. Available from: https://www.mathworks.com/solutions/robotics.html [Accessed 1 Dec 2024].

BIBLIOGRAPHY

- Kaza S, Yao L, Bhada-Tata P, Van Woerden F. *What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050.* Washington, DC: World Bank; 2018.
- Wilson DC, Velis C, Cheeseman C. Role of informal sector recycling in waste management in developing countries. *Habitat Int.* 2006;30(4):797-808.
- Kim D, Lee K, Kim S, Lee J, Kim J. Waste sorting using deep learning. *Waste Management.* 2018;71:174-80.
- United Nations. *Sustainable Development Goals: Goal 12 - Responsible Consumption and Production.* UN; 2023. Available from: https://sdgs.un.org/goals/goal12
- Singh RP, Tyagi VV, Allen T, Ibrahim MH, Kothari R. An overview for exploring the possibilities of energy generation from municipal solid waste (MSW) in Indian scenario. *Renew Sustain Energy Rev.* 2011;15(9):4797-808.
- Das A, Nandy A. Automatic waste segregation system using image processing. In: *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI).* IEEE; 2017. p. 2670-4.
- Zhang Y, Gao H, Fan H, Zhu W. Automatic waste sorting system based on image recognition technology. In: *2018 International Conference on Computer, Mechatronics, Control and Electronic Engineering (CMCE).* IEEE; 2018. p. 1-4.
- Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556; 2014.
- Rario M, Marquez J, Martinez RM. Garbage detection based on convolutional neural network. *Final-Paper Semaphore.* 2024. Available: https://github.com/PizaaRiaaa/garbage-classification/tree/main/fullpaper.
- Tan M, Le QV. EfficientNet: Rethinking model scaling for convolutional neural networks. In: *Proceedings of the 36th International Conference on Machine Learning.* PMLR; 2019. p. 6105-14.
- Canziani A, Paszke A, Culurciello E. An analysis of deep neural network models for practical applications. arXiv preprint arXiv:1605.07678; 2016.
- Rufus S. Sensory system for robots. *LinkedIn.* 2024. Available from: https://www.linkedin.com/pulse/sensory-system-robots.
- MathWorks. *SLAM (Simultaneous Localization and Mapping).* 2024. Available from: https://www.mathworks.com/discovery/slam.html
- NVIDIA Corporation. *NVIDIA Jetson for AI at the edge.* 2024. Available from: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson
- Quigley M, Gerkey B, Smart W. *Programming Robots with ROS.* Sebastopol: O'Reilly Media; 2015.
- Dastin J. Amazon scraps secret AI recruiting tool that showed bias against women. *Reuters.* 2018. Available from: https://www.reuters.com/article/amazon-ai-recruiting-idUSKCN1MK08G
- Mehrabi N, Morstatter F, Saxena N, et al. A survey on bias and fairness in machine learning. arXiv preprint arXiv:1908.09635; 2019.
- EPA. *Sustainable Materials Management: The Road Ahead.* Washington, DC: Environmental Protection Agency; 2016. Available from: https://www.epa.gov/smm/sustainable-materials-management-road-ahead
- Raspberry Pi Foundation. *Raspberry Pi 4 Model B Specifications.* 2024. Available from: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/
- Texas Instruments. *L298 Dual Full-Bridge Driver Data Sheet.* 2024. Available from: https://www.ti.com/product/L298

- Slamtec. *RPLIDAR A2 360° Laser Range Scanner*. 2024. Available from: https://www.slamtec.com/en/Lidar/A2
- OpenCV. *Image Processing for Object Detection.* 2024. Available from: https://opencv.org/
- Tower Pro. *SG90 Servo Motor Specifications.* 2024. Available from: https://www.towerpro.com.sg/sg90
- MathWorks. *Simulating Robot Motion with LiDAR.* 2024. Available from: https://www.mathworks.com/solutions/robotics.html
- Chollet F. *Deep Learning with Python.* 2nd ed. Shelter Island: Manning Publications; 2021.
- Brownlee J. Handling Imbalanced Data for Deep Learning. *Machine Learning Mastery*. 2022. Available from: https://machinelearningmastery.com/imbalanced-data/
- Scikit-learn Developers. *Confusion Matrix Documentation.* 2024. Available from:

https://scikit-learn.org/

APPENDIX A

Main Script (85% Integration)

```
import RPi.GPIO as GPIO
import time
import tensorflow as tf
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import img_to_array
import numpy as np
import cv2
# GPIO and Servo Pins
BASE_PIN = 17
SHOULDER_PIN = 27
ELBOW_PIN = 22
WRIST_PIN = 5
GRIPPER_PIN = 6
IN1 = 23
IN2 = 24
IN3 = 25
IN4 = 16
ENA = 12
ENB = 13
# Waste classification model setup
model = load_model('Waste_Classification_Model.h5')
class_names = ['cardboard', 'glass', 'metal', 'paper', 'plastic', 'trash']
# Bin durations for movement
BIN_DURATIONS = {

    'cardboard': 1,

    'glass': 2,

    'metal': 3,

    'paper': 4,

    'plastic': 5,

    'trash': 6
```

```
}
# Servo setup function
def setup_servo(pin):
    GPIO.setup(pin, GPIO.OUT)
    pwm = GPIO.PWM(pin, 50)  # 50 Hz frequency
    pwm.start(7.5)          # Neutral position
    return pwm
# Move servo to a specific angle
def move_servo(pwm, angle):
    duty = 2 + (angle / 18)
    pwm.ChangeDutyCycle(duty)
    time.sleep(0.5)
        pwm.ChangeDutyCycle(0)  # Stop sending the signal
# Pick up waste
def pick():
    move_servo(base_pwm, 30)
    time.sleep(1)
    move_servo(shoulder_pwm, 165)
    time.sleep(1)
    move_servo(gripper_pwm, 180)
    time.sleep(1)
    move_servo(shoulder_pwm, 30)
    time.sleep(1)
# Release waste
def release():
    move_servo(shoulder_pwm, 165)
    time.sleep(1)
    move_servo(gripper_pwm, 30)
    time.sleep(1)
    move_servo(base_pwm, 30)
    time.sleep(1)
    move_servo(shoulder_pwm, 165)
    time.sleep(1)
# Preprocess image for classification
def preprocess_image(image, target_size=(32, 32)):
    image = cv2.resize(image, target_size)
    image = img_to_array(image) / 255.0
    return np.expand_dims(image, axis=0)
# Classify waste
def classify_waste():
```

```
    #Capture multiple frames and classify the waste.
    #Displays detections and confidences for each frame.
    #Retries classification if all frames have confidence below 0.75.
   # Returns the class label and confidence if successful.
    while True:  # Loop until confident classification is achieved
        cap = cv2.VideoCapture(0)  # Initialize webcam
        if not cap.isOpened():
            raise Exception("Error accessing the camera")
        print("Capturing frames for classification...")
        frame_count = 40  # Number of frames to capture
        predictions = []
        for frame_num in range(frame_count):
            ret, frame = cap.read()
            if not ret:
                print(f"Frame {frame_num + 1}: Failed to capture, skipping...")
                continue
            # Preprocess frame and predict
            preprocessed_frame = preprocess_image(frame)
            prediction = model.predict(preprocessed_frame)
            class_index = np.argmax(prediction[0])
            confidence = prediction[0][class_index]
            class_label = class_names[class_index]
            # Display each frame's detection and confidence
            print(f"Frame {frame_num + 1}: Detected {class_label} with confidence {confidence:.2f}")
            if confidence > 0.75:
                predictions.append(prediction[0])  # Collect prediction probabilities
            # Allow a slight delay between frames
            time.sleep(0.1)
        cap.release()
        if not predictions:
            print("All captured frames have confidence below 0.75. Retrying classification...")
            continue  # Retry classification
        # Average predictions and determine the most confident class
        avg_prediction = np.mean(predictions, axis=0)
        class_index = np.argmax(avg_prediction)
        class_label = class_names[class_index]
        confidence = avg_prediction[class_index]
        print(f"Final classification: {class_label} with confidence {confidence:.2f}")
        return class_label, confidence
# Move motors forward
```

```
def move_forward(speed, duration):
    GPIO.output(IN1, GPIO.LOW)
    GPIO.output(IN2, GPIO.HIGH)
    GPIO.output(IN3, GPIO.LOW)
    GPIO.output(IN4, GPIO.HIGH)
    pwmA.ChangeDutyCycle(speed)
    pwmB.ChangeDutyCycle(speed)
    time.sleep(duration)
    pwmA.ChangeDutyCycle(0)
    pwmB.ChangeDutyCycle(0)
# Move motors backward
def move_backward(speed, duration):
    GPIO.output(IN1, GPIO.HIGH)
    GPIO.output(IN2, GPIO.LOW)
    GPIO.output(IN3, GPIO.HIGH)
    GPIO.output(IN4, GPIO.LOW)
    pwmA.ChangeDutyCycle(speed)
    pwmB.ChangeDutyCycle(speed)
    time.sleep(duration)
    pwmA.ChangeDutyCycle(0)
    pwmB.ChangeDutyCycle(0)
# Setup GPIO and PWM
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
# Setup servos
base_pwm = setup_servo(BASE_PIN)
shoulder_pwm = setup_servo(SHOULDER_PIN)
elbow_pwm = setup_servo(ELBOW_PIN)
wrist_pwm = setup_servo(WRIST_PIN)
gripper_pwm = setup_servo(GRIPPER_PIN)
# Setup motors
GPIO.setup([IN1, IN2, IN3, IN4, ENA, ENB], GPIO.OUT)
pwmA = GPIO.PWM(ENA, 100)
pwmB = GPIO.PWM(ENB, 100)
pwmA.start(0)
pwmB.start(0)
try:
    print("Starting waste sorting robot...")
    while True:  # Loop until confident classification is achieved
        # Step 1: Classify waste
```

```
        waste_type, confidence = classify_waste()
        print(f"Detected: {waste_type} with confidence {confidence:.2f}")
        # Check confidence threshold
        if confidence > 0.75:
            break  # Exit loop if classification is confident
        else:
            print(f"Confidence {confidence:.2f} is below the threshold. Retrying classification...")
    # Step 2: Pick waste
    print("Picking waste...")
    pick()
    # Step 3: Move to bin
    print(f"Moving to bin for {waste_type}...")
    duration = BIN_DURATIONS[waste_type]
    move_forward(60, duration)
    # Step 4: Release waste
    print("Releasing waste...")
    release()
    # Step 5: Return to original position
    print("Returning to original position...")
    move_backward(60, duration)
except KeyboardInterrupt:
    print("Operation interrupted by user.")
finally:
    # Cleanup
    base_pwm.stop()
    shoulder_pwm.stop()
    elbow_pwm.stop()
    wrist_pwm.stop()
    gripper_pwm.stop()
    pwmA.stop()
    pwmB.stop()
    GPIO.cleanup()
    print("Robot stopped. GPIO cleaned up.")
```

APPENDIX B

Script to Test LiDAR functionality individuallly

```
import argparse
from os import path
from rplidar import RPLidar
import sys
# Constants
DEVICE_PATH = "/dev/ttyUSB0"  # Hardcoded device path for LiDAR
BAUDRATE = 115200
TIMEOUT = 1
OBSTRUCTION_DISTANCE_CM = 30  # Distance threshold to consider something an obstruction (in cm)
# Colors for printing
class PrintColor:
    RED = '\033[1;31;48m'
    GREEN = '\033[1;32;48m'
    END = '\033[1;37;0m'
# Function to detect obstructions directly in front
def detect_obstruction(angle, distance_mm):
    # Convert distance to cm
    distance_cm = distance_mm / 10
    # Check if the angle is in the front range (e.g., -10° to 10°)
    if -10 <= angle <= 10:
        if distance_cm > 0 and distance_cm <= OBSTRUCTION_DISTANCE_CM:
            # Log obstruction detection
            print(
                PrintColor.RED +
                f"[WARNING] Obstruction detected! Angle: {angle:.2f}°, Distance: {distance_cm:.2f} cm" +
                PrintColor.END,
                flush=True
            )
            return True
        else:
            # Log clear front
            print(
                PrintColor.GREEN +
                f"Clear front. Angle: {angle:.2f}°, Distance: {distance_cm:.2f} cm" +
                PrintColor.END,
                flush=True
```

```
            )
    return False
# Main function to run the LiDAR test
def run():
    print(f"Starting LiDAR test on device: {DEVICE_PATH}")
    lidar = RPLidar(port=DEVICE_PATH, baudrate=BAUDRATE, timeout=TIMEOUT)
    try:
        for measurement in lidar.iter_measures():
            # Measurement contains quality, angle, distance, etc.
            quality, angle, distance_mm = measurement[1], measurement[2], measurement[3]
            # Ignore zero distance readings (invalid data)
            if distance_mm > 0:
                obstruction = detect_obstruction(angle, distance_mm)
                # If an obstruction is detected, pause and wait for it to clear
                if obstruction:
                    print("[INFO] Waiting for obstruction to clear...", flush=True)
                    while True:
                        # Check LiDAR readings in the waiting loop
                        for sub_measurement in lidar.iter_measures():
                            _, sub_angle, sub_distance_mm = sub_measurement[1], sub_measurement[2],
sub_measurement[3]
                            if sub_distance_mm > 0:
                                obstruction = detect_obstruction(sub_angle, sub_distance_mm)
                                if not obstruction:
                                    print("[INFO] Obstruction cleared. Resuming...", flush=True)
                                    break  # Exit the inner loop
                        if not obstruction:
                            break  # Exit the outer waiting loop
    except KeyboardInterrupt:
        print("Test interrupted by user.")
    finally:
        lidar.stop()
        lidar.stop_motor()
        lidar.disconnect()
        print("LiDAR test stopped and device disconnected.")
if __name__ == "__main__":
    run()
```