ABSTRACT

The primary objective of this project is to design and develop a trash collection robot capable of detecting, collecting, and disposing of waste materials from the environment using a mobile app interface. The system integrates multiple components including DC geared motors for movement, MG996R and DS3235MG servo motors for actuation of the collection mechanism, metal detector sensors for identifying metallic waste, and a real-time clock for timestamp logging. Control and communication are managed through the ESP32 Dev Module, which connects to the Blynk IoT platform for remote operation. Power distribution is efficiently managed using dual XL4016E1 step-down buck converters. This robot is intended to support cleaner environments, minimize human effort in trash handling, and promote smarter waste management through embedded systems and IoT technology. The modular design allows scalability and adaptation for more complex applications in future developments.

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVES

The primary objective of this project is to design and develop a manually controlled robotic system capable of collecting solid waste and performing basic segregation of metallic and non-metallic trash. The robot should assist in promoting cleaner environments by making waste collection more efficient and contactless. The project also aims to create a low-cost prototype that demonstrates how robotics can be applied to solve real-world problems in public hygiene and waste management.

1.2 MECHATRONICS PERSPECTIVE

The primary objective of this project is to design and develop a manually controlled robotic system capable of collecting solid waste and performing basic segregation of metallic and non-metallic trash. The robot should assist in

promoting cleaner environments by making waste collection more efficient and contactless. The project also aims to create a low-cost prototype that demonstrates how robotics can be applied to solve real-world problems in public hygiene and waste management

1.3 CONTEXT DESCRIPTION

With the rising concern over urban waste management and the health risks associated with manual trash collection, there is a growing need for semi-automated and intelligent systems that can assist in these tasks.

Actuators:

DC gear motors provide drive motion, while MG996R and DS3235MG servos manage collection and sorting with precision.

Sensors:

A metal detector sensor detects metal objects, and the ESP-WROOM-32 enables wireless control via

Wi-Fi/Bluetooth.

Structure:

A balanced chassis supports all components, with a

4-channel relay for safe power switching and control.

Environment:

Designed for indoor/semi-smooth surfaces, powered by a 12V 4.5Ah battery, and supports easy modular upgrades.

CHAPTER II

DESIGN AND FABRICATION

2.1 STRUCTURAL ELEMENTS

The following project has used a few structural elements which involves the use of a wooden plank, some PVC pipes and Trash Bins which are joined together using screws, washers, and nuts.

2.2 MECHANICAL ELEMENTS

2.2.1 DC Motors (12V, 60 RPM):

These motors provide the primary locomotion for the bot. The gear reduction offers high torque and slow, controlled movement—ideal for maneuvering during trash collection.

2.2.2 MG996R & DS3235MG Servo Motors:

These servos are used for actuation in the trash-collecting and sorting mechanisms. MG996R is known for moderate torque and speed, while the DS3235MG offers higher torque (35kg) for heavier movements like bin lifting or arm rotation.

Figure 2.2.1 DC MOTOR Figure 2.2.2 MG996R Figure 2.2.3 DS3235MG

2.2.3 **Chassis and Frame (assumed)**:
An aluminum frame designed to hold all components, with space for the motor mountings, battery placement, and sorting mechanisms. Proper wheelbase and center-of-gravity balance are crucial for stability.

2.3 Electrical Elements

2.3.1 Amptek 12V 4.5Ah Sealed Lead Acid Battery:

Supplies power to the motors, sensors, and control board. It provides enough current for long-duration operations and is well-suited for mobile platforms due to its reliability and rechargeability.

2.4 Electronic Elements

2.4.1 ESP-WROOM-32 (ESP32 Module):

This microcontroller board integrates Wi-Fi and Bluetooth for wireless control and monitoring. It handles all logic, motor commands, sensor input, and communication with a user or base station.

2.4.2 Metal Detector Sensor:

Used to detect the presence of metallic objects. When metallic trash passes near the sensor, it triggers sorting logic, allowing the bot to classify and separate metal from non-metal trash.

Applications:

1. Detects and helps separate metallic items from non-metallic items, enabling automated waste segregation.
2. Detects the presence of gears or metal parts for safety interlocks or to monitor mechanical systems.

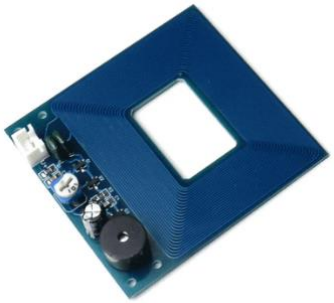


Figure 2.4.2 METAL DETECTOR SENSOR

Figure 2.4.1 ESP-WROOM

2.5 HARDWARE (CAD) DIAGRAM

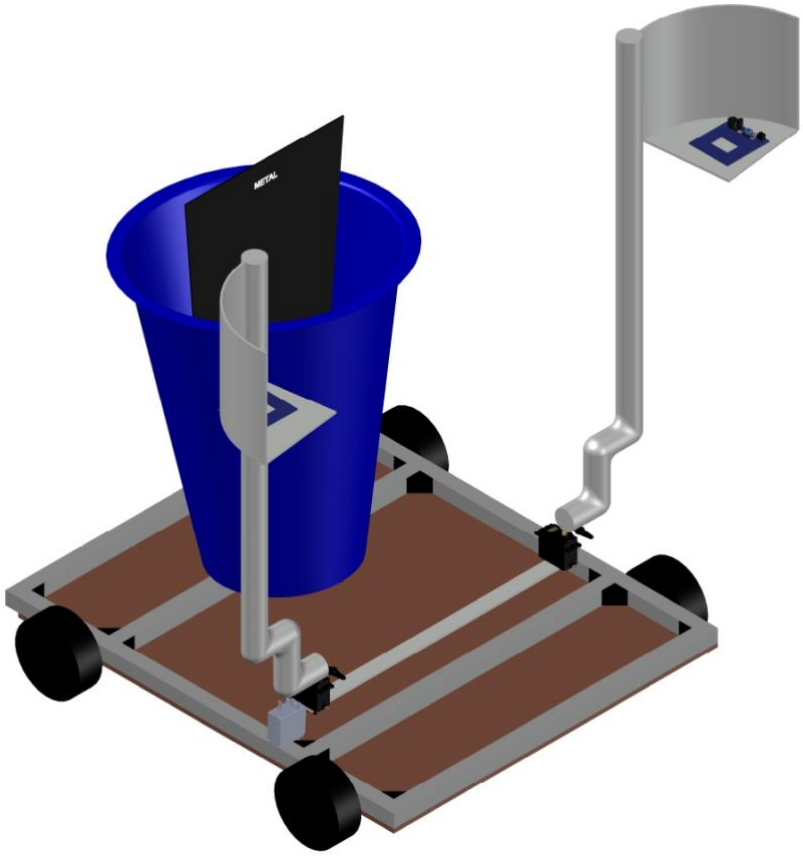


Figure 2.5.1 Completed Design

This CAD image depicts a simplified design of a mobile trash-collecting robot. It features a rectangular base platform with four wheels for locomotion. A blue conical bin is centrally mounted on the platform to hold collected waste. An articulated arm-like structure extends from the base, culminating in a sensor mounted above a designated sorting area. The robot is designed to move around, collect trash into the bin, and then potentially use the metal detector module to identify and sort the collected items.

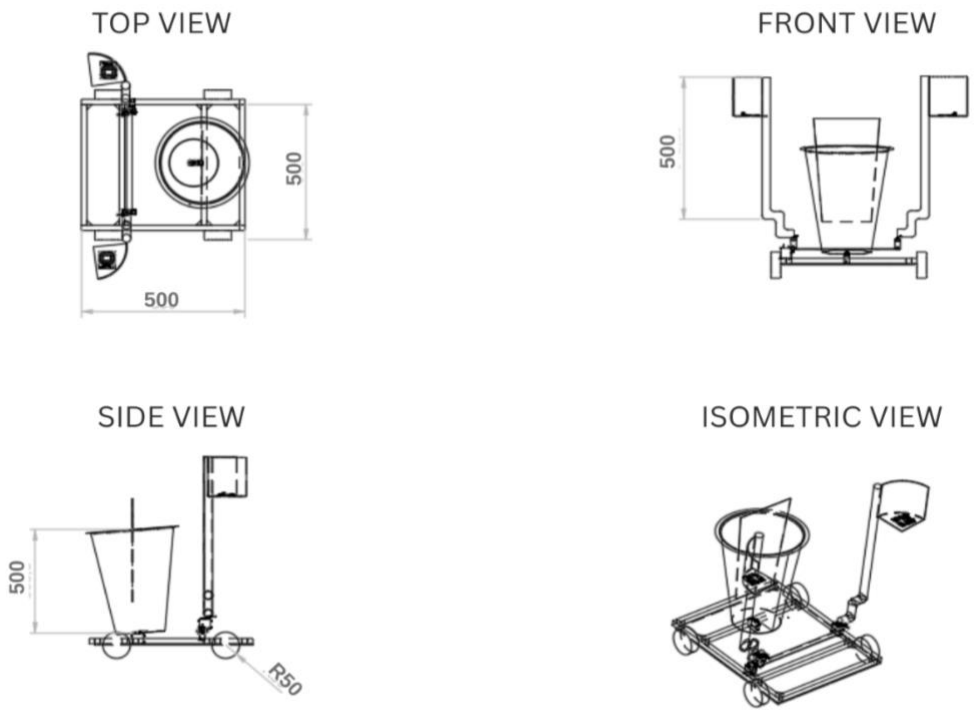


Figure 2.5.2 Sectional View

View	Dimensions	Measurement
Top View	Length	500 mm
	Width	500 mm
Front View	Height	500 mm
Side View	Height	500 mm
	Wheel Radius	R50 mm

Table 2.5.1 Dimensions

2.6 ELECTRICAL CIRCUIT DIAGRAM

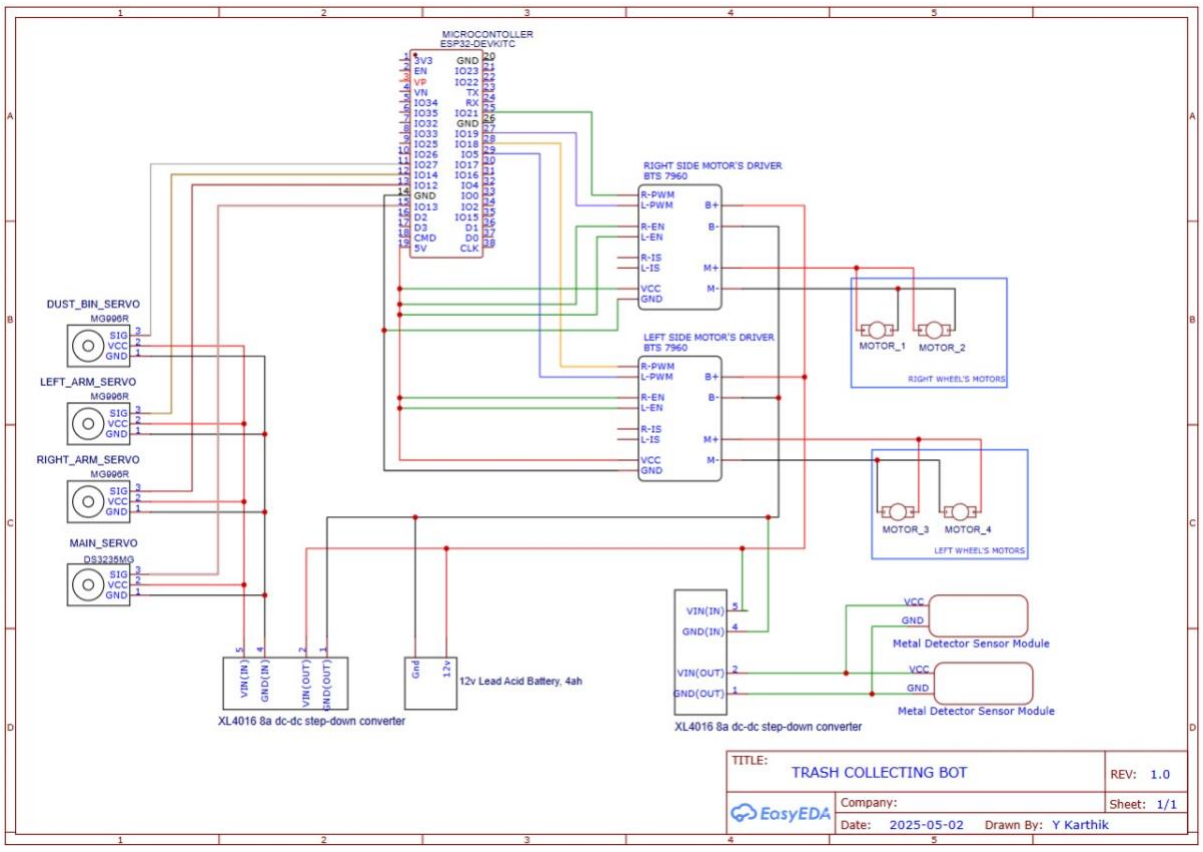


Figure 2.6 Circuit Diagram

This trash collecting robot is controlled by an ESP32 microcontroller, which manages four servo motors (for the dustbin, left arm, right arm, and overall movement) and four DC motors for locomotion, driven by BTS7960 motor drivers. Power is supplied by a 12V lead-acid battery, stepped down to 5V using XL4016 DC-DC converters. The robot also includes a

metal detector sensor. The robot's movement is controlled via a mobile app with a circle pad interface, wirelessly sending commands to adjust motor speed and direction for tasks like turning or moving straight.

CHAPTER III

FUNCTIONAL INTEGRATION

3.1 TYPE OF CONTROL

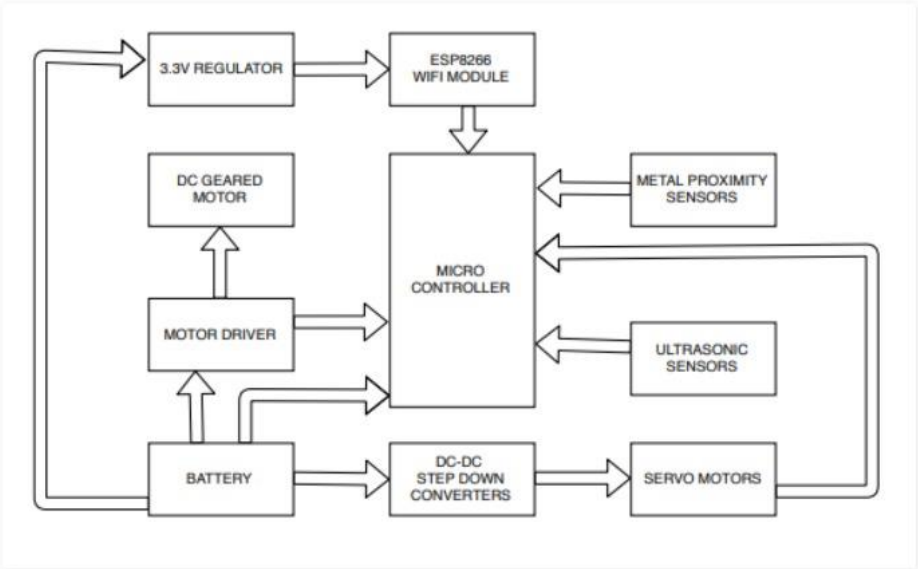


Figure 3.1 Flow of Control

This system follows a control flow where the battery powers the entire setup, with voltage regulated by DC-DC converters. The microcontroller processes data from sensors (ultrasonic for distance, metal proximity for object detection) and sends control signals to actuators like motor drivers (for movement) and servo motors (for precise actions). The ESP8266 Wi-Fi module allows

wireless communication for remote control or data exchange. In short, sensors gather data, the microcontroller processes it, and actuators perform actions, with Wi-Fi enabling remote interaction.

3.2 FLOWCHART

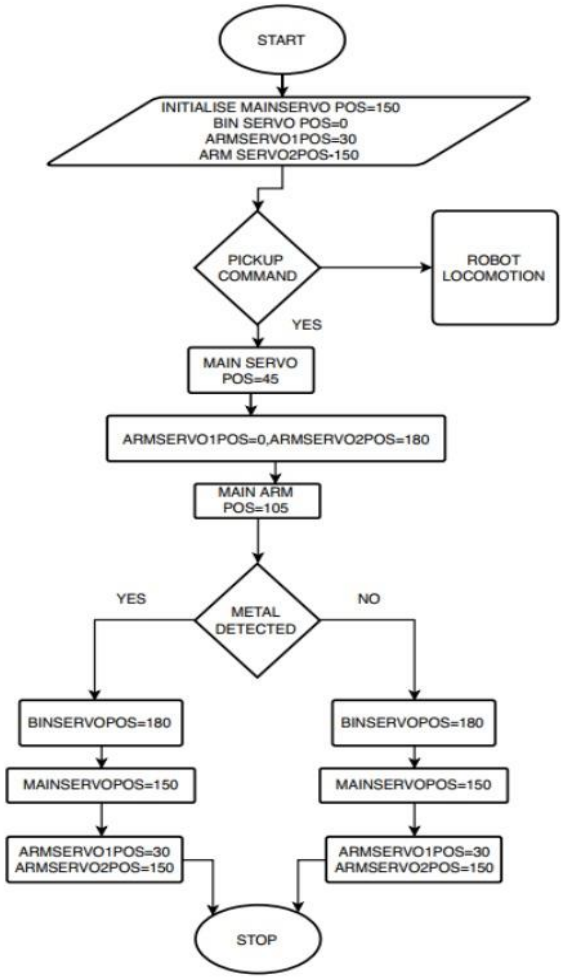


Figure 3.2 Flowchart

This flowchart outlines the control flow for a trash-collecting robot's pickup and sorting sequence. It begins by initializing all servos to default positions. When a pickup command is received, the main and arm servos reposition to collect the trash. The robot

then checks for metal using a sensor. If metal is detected, the bin servo opens to deposit metallic waste; if not, it deposits non-metallic waste. After sorting, all servos return to their original positions. If no pickup command is received, the robot enters a locomotion state, continuing to navigate or waitfor commands. The process loops, enabling continuous operation.

3.3 CHASSIS MODELLING -4 WHEEL DRIVE ROBOT

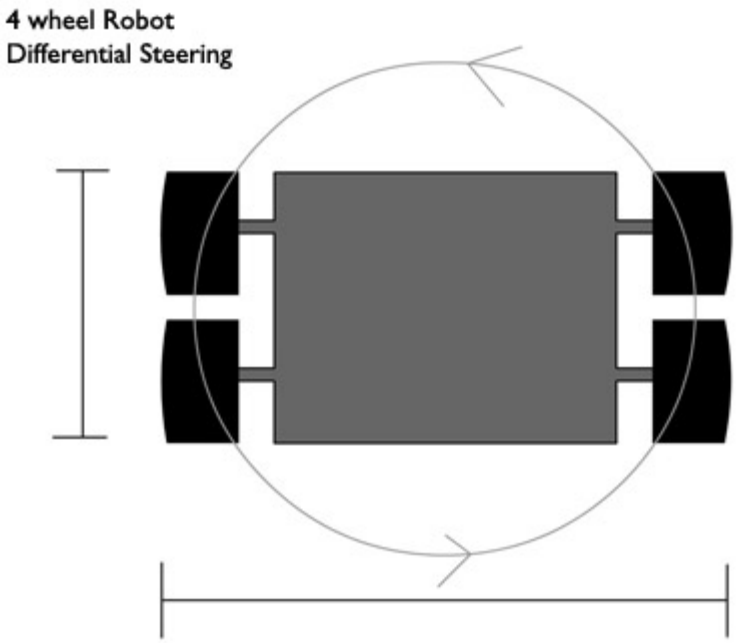


Figure 3.3 Maneuverability

Chassis modelling for a 4-wheel drive robot involves designing a sturdy frame that evenly distributes weight and supports all mounted components. It ensures precise alignment for motorized wheels to achieve balanced traction and mobility. The 4-wheel drive system provides enhanced torque and stability, allowing the robot to navigate rough terrains efficiently.

3.4 COMMENTS ON PERFORMANCE

The project demonstrates excellent integration of hardware and software components, showcasing a clear understanding of robotic control systems. The use of an ESP32 for centralized control, combined with servo and DC motor coordination, reflects efficient design and execution. Incorporating a metal detector for sorting adds a practical, intelligent layer to the system. The mobile app interface provides intuitive and responsive control, enhancing user interaction. Overall, the robot performs its tasks reliably and effectively, highlighting strong problem-solving skills and a solid grasp of embedded systems and automation.

3.5 APP CONTROLLER

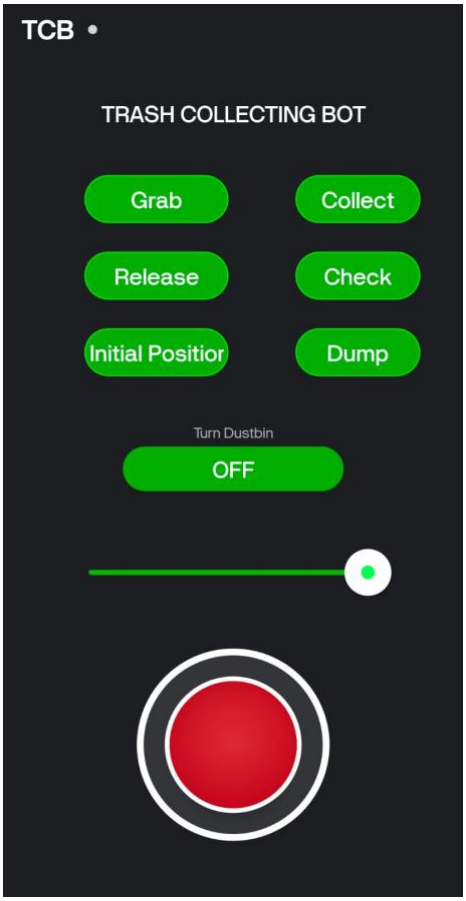


Figure 3.5 Design of the App

This app is designed to control a **Trash Collecting Bot (TCB)**. It has simple buttons for different functions: "Grab" to pick up trash,

"Release" to let it go, "Collect" to start collecting trash, "Check" to inspect something, and "Dump" to empty the collected trash. There is also a switch to turn the dustbin mechanism ON or OFF. A slider with a number (like 1023) likely controls the motor speed or sensor sensitivity. The big red button at the bottom could be for emergency stop or a start/stop command. Overall, it is a clean and user-friendly remote-control interface for operating a trash-collecting robot.

CHAPTER IV

RESULT

Figure 4.1 Completed Design

The trash-collecting robot is designed to autonomously detect, pick up, and sort waste using an ESP32 microcontroller, servo motors, a gripper mechanism, and sensor modules. The process begins with the robot scanning the ground using a camera and IR sensor to identify potential trash. Once detected, the robotic arm precisely aligns above the object, and the gripper carefully lifts it off the ground. The

collected item is then brought near a metal detection sensor, which determines whether the object is metallic or not. Based on this input, the robot sorts the item by rotating its arm and placing it into either the Metallic Waste Bin or the Non-Metallic Waste Bin. This sorting ensures efficient separation of recyclable metals from general waste. After sorting, the robotic arm automatically resets all servo motors to their initial positions, opens the gripper, and signals readiness (via an LED or status message) for the next collection cycle, enabling continuous, hands-free operation.

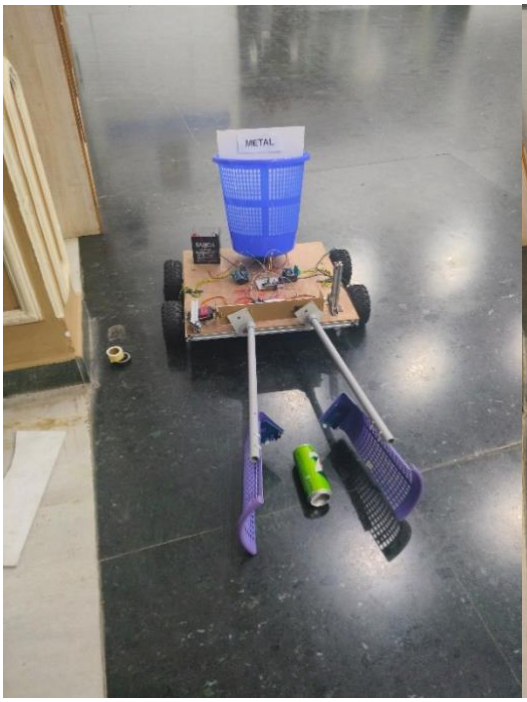
4.1 OVERVIEW OF THE MECHANISM

FIGURE 4.2

The trash-collecting robot is engineered to autonomously identify, pick up, and sort waste using a combination of sensor inputs and motorized components. At its core, it features an ESP32 microcontroller, servo motors for joint movements, and a gripper mechanism to grab items. The robot initially scans the

ground using a camera and IR sensor for potential trash. Once trash is located, it activates its gripper and begins the collection process, preparing for further classification based on material type.

4.2 STEP BY STEP PICKUP PROCESS



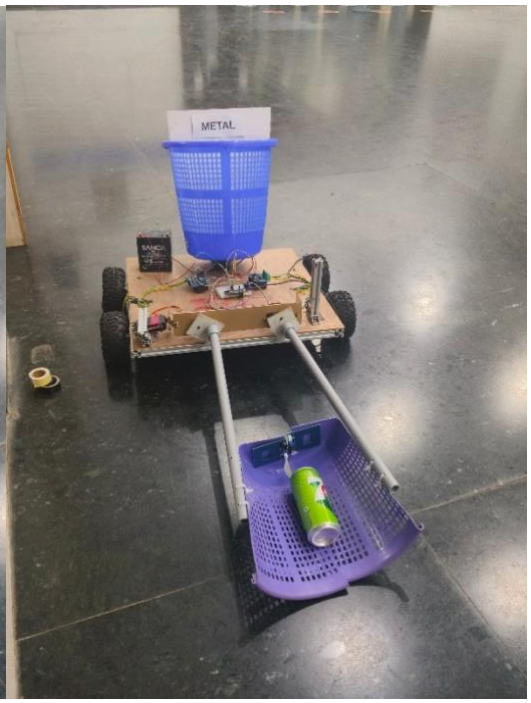


FIGURE 4.3.1 FIGURE 4.3.2

The pickup sequence starts as the robot detects trash using the onboard vision module or IR sensor. The robotic arm aligns itself precisely above the object using servo-controlled joints. The

gripper closes in around the trash and gently lifts it off the ground. Each movement—from detection to lifting—is programmed with precision for reliable performance. The process is completed in just a few seconds, allowing the robot to transition to the sorting stage swiftly.

4.3 METAL DETECTION AND SORTING

FIGURE 4.4

After the trash is lifted, it is brought near the metal detection sensor module. This sensor checks for the presence of metal based on electromagnetic signals. If metal is detected, a specific signal is sent to the microcontroller to guide the robot to the metallic waste bin. If not, it redirects to the non-metallic section. This logic ensures that recyclable metals are separated efficiently from general waste. The decision-making algorithm is built into the ESP32's codebase.

4.4 SORTING OUTCOMES

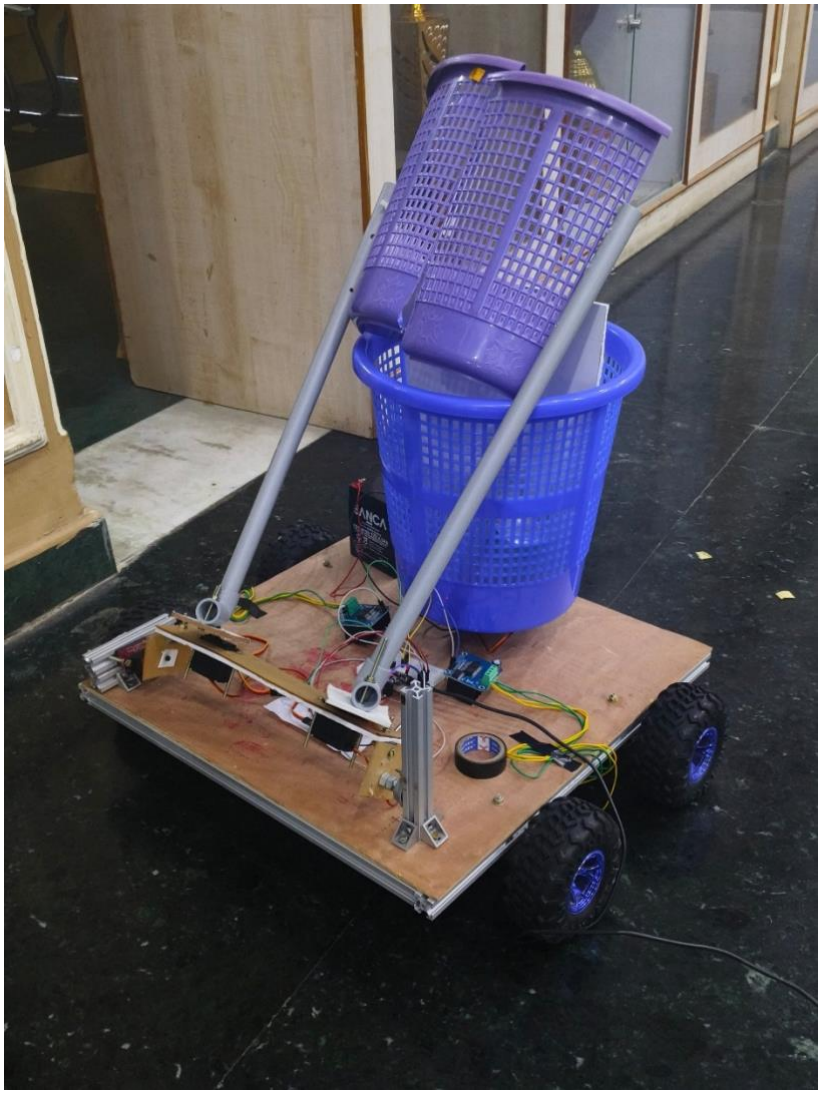


FIGURE 4.5

Once the material type is confirmed, the robot rotates its arm to the respective bin. For metallic waste, it deposits the item into the designated Metallic Waste Bin, and for others, into the Non-Metallic Waste Bin. The placement is accurate due to the coordinated motion of servos and real-time sensor feedback. The sorting mechanism ensures that materials are separated cleanly, reducing manual segregation effort and improving recycling efficiency.

CHAPTER V

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUDING REMARKS

This project effectively embodies the core principles of mechatronics by integrating mechanical design, electronics, control systems, and embedded programming into a single, functional robotic system. Through the successful implementation of sensors, actuators, and wireless communication, the trash-collecting robot highlights the interdisciplinary nature of mechatronics engineering. It not only addresses a real-world environmental challenge but also demonstrates the practical application of theoretical concepts, reinforcing the importance of system integration and intelligent automation in modern engineering solutions.

5.2 FUTURE SCOPE

The trash-collecting robot can be upgraded with AI-based vision systems to identify and sort various waste types more accurately. Adding GPS and autonomous path planning would allow deployment in larger outdoor environments. Solar-powered charging can enhance energy efficiency, while cloud connectivity and data logging can support smart city integration. These improvements would help evolve the robot into a fully autonomous and intelligent waste management solution.

REFERENCES

1) https://iottrashbot.blogspot.com/2018/09/trash-collecting-robot-with-iot.html

2) https://youtu.be/kp7PEkG62i8?si=pZe8QgkF9M84Rt6Z

3) https://youtu.be/iagL5j2l-fA?si=8LVFZEtc_3LNadX1

APPENDIX

```
#define BLYNK_TEMPLATE_ID "TMPL309wN1vU5"
#define BLYNK_TEMPLATE_NAME "TCB"
#define BLYNK_AUTH_TOKEN
"A9pYeQZ2kYm3TK77dx80l2D-gc-BL5uo"

#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <ESP32Servo.h>

// Wi-Fi Credentials
char ssid[] = "YK";
char pass[] = "abcd1234";

// --- Motor Pin Assignments ---
#define RIGHT_RPWM 19
#define RIGHT_LPWM 21
#define LEFT_RPWM 5
#define LEFT_LPWM 18

// --- Servo Constants ---
#define CLOCKWISE 1300
```

```
#define COUNTERCLOCKWISE 1700
#define SERVO_STOP 1500
#define SERVO_DELAY 400

// --- Servo Pins ---
#define SERVO_A_PIN 13  // Arm Servo A
#define SERVO_B_PIN 12  // Arm Servo B
#define SERVO_C_PIN 27  // myServo - Arm Position
#define DUSTBIN_SERVO_PIN 14 // Dustbin Servo

// --- Servo Objects ---
Servo servoA;
Servo servoB;
Servo myServo;
Servo dustbinServo;

// --- Motor Control Variables ---
int x = 0, y = 0;
int rawSpeed = 512;
int mappedSpeed = 128;

void setup() {
  Serial.begin(115200);
  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);

  // --- Motor Pins ---
```

```
  pinMode(RIGHT_RPWM, OUTPUT);
  pinMode(RIGHT_LPWM, OUTPUT);
  pinMode(LEFT_RPWM, OUTPUT);
  pinMode(LEFT_LPWM, OUTPUT);

  // --- Servo Setup ---
  servoA.attach(SERVO_A_PIN);
  servoB.attach(SERVO_B_PIN);
  myServo.attach(SERVO_C_PIN);
  dustbinServo.attach(DUSTBIN_SERVO_PIN);

  // --- Initialize Servos ---
  servoA.writeMicroseconds(SERVO_STOP);
  servoB.writeMicroseconds(SERVO_STOP);
  myServo.write(90);
  dustbinServo.writeMicroseconds(SERVO_STOP);
}

// --- Joystick X (Forward/Backward) ---
BLYNK_WRITE(V0) {
  x = param.asInt();
}

// --- Joystick Y (Left/Right) ---
BLYNK_WRITE(V1) {
  y = param.asInt();
```

```
}

// --- Speed Slider ---
BLYNK_WRITE(V2) {
  rawSpeed = param.asInt();
  mappedSpeed = map(rawSpeed, 0, 1023, 0, 255);
}

// --- Arm Grab ---
BLYNK_WRITE(V3) {
  if (param.asInt() == 1) {
    Serial.println("Grabbing...");
    servoA.writeMicroseconds(CLOCKWISE);
    servoB.writeMicroseconds(COUNTERCLOCKWISE);
    delay(SERVO_DELAY);
    servoA.writeMicroseconds(SERVO_STOP);
    servoB.writeMicroseconds(SERVO_STOP);
  }
}

// --- Arm Release ---
BLYNK_WRITE(V4) {
  if (param.asInt() == 1) {
    Serial.println("Releasing...");
    servoA.writeMicroseconds(COUNTERCLOCKWISE);
    servoB.writeMicroseconds(CLOCKWISE);
```

```
    delay(SERVO_DELAY);
    servoA.writeMicroseconds(SERVO_STOP);
    servoB.writeMicroseconds(SERVO_STOP);
  }
}

// --- Arm Position Control ---
BLYNK_WRITE(V5) {
  if (param.asInt() == 1) {
    myServo.write(180);
    Serial.println("Arm → Collect (180°)");
  }
}

BLYNK_WRITE(V6) {
  if (param.asInt() == 1) {
    myServo.write(120);
    Serial.println("Arm → Check (120°)");
  }
}

BLYNK_WRITE(V8) {
  if (param.asInt() == 1) {
    myServo.write(60);
    Serial.println("Arm → Dump (60°)");
  }
```

```
}

BLYNK_WRITE(V9) {
  if (param.asInt() == 1) {
    myServo.write(90);
    Serial.println("Arm → Initial (90°)");
  }
}

// --- Dustbin Servo (Dump Open/Close) ---
BLYNK_WRITE(V7) {
  if (param.asInt() == 1) {
    Serial.println("Dustbin Dumping...");
    dustbinServo.writeMicroseconds(1300); // Open
    delay(1500);
    dustbinServo.writeMicroseconds(SERVO_STOP); // Stop
  } else {
    dustbinServo.writeMicroseconds(SERVO_STOP); //
Ensure Stop
  }
}

void loop() {
  Blynk.run();

  // Motor Movement Handling
```

```
if (x > 2) {
  // Forward
  analogWrite(RIGHT_RPWM, mappedSpeed);
  analogWrite(RIGHT_LPWM, 0);
  analogWrite(LEFT_RPWM, mappedSpeed);
  analogWrite(LEFT_LPWM, 0);
} else if (x < -2) {
  // Backward
  analogWrite(RIGHT_RPWM, 0);
  analogWrite(RIGHT_LPWM, mappedSpeed);
  analogWrite(LEFT_RPWM, 0);
  analogWrite(LEFT_LPWM, mappedSpeed);
} else if (y > 2) {
  // Right Turn
  analogWrite(RIGHT_RPWM, 0);
  analogWrite(RIGHT_LPWM, mappedSpeed);
  analogWrite(LEFT_RPWM, mappedSpeed);
  analogWrite(LEFT_LPWM, 0);
} else if (y < -2) {
  // Left Turn
  analogWrite(RIGHT_RPWM, mappedSpeed);
  analogWrite(RIGHT_LPWM, 0);
  analogWrite(LEFT_RPWM, 0);
  analogWrite(LEFT_LPWM, mappedSpeed);
} else {
  // Stop
```

```
    analogWrite(RIGHT_RPWM, 0);
    analogWrite(RIGHT_LPWM, 0);
    analogWrite(LEFT_RPWM, 0);
    analogWrite(LEFT_LPWM, 0);
  }
}
```