Enhancing Waste Management at Landfill Sites in South Africa: A more accessable solution

Zara Kara, Shayur Misra, Ruan De Beer, Terry Chitter, Mackyle Naidoo, Augustin Martinez
Python Robotics
6/18/23

Table of Contents

Table of Figures

Abstract

We have set out to tackle the litter problem at landfills sites as in South Africa the main problem to litter is that the landfill sites are that there is too much litter that is not being organized to solve this problem, we have decided to build a conveyor belt system that will be stable and be place at the landfill site location. The conveyor belt will have a robotic arm to be able to detect different litter object and sort them out based on an algorithm that is able to categorize different litter objects. The system itself needs to be autonomous as the landfills sites are very hazardous to humans as they contain dangerous gasses and infections that causes danger to the people on site and in the area.

Chapter 1: Introduction

For many years, South Africa has struggled with littering and inappropriate trash disposal, which has had negative effects on the environment and society. Significant difficulties have been created by the exponential expansion of litter, which has caused a sharp rise in the number of landfills around the nation. Beyond environmental issues, the effects of this problem also have an impact on the health and welfare of the general populace. Pests drawn to landfills can spread disease and pose a threat to both human and animal health. Additionally, a serious issue for the country is waste management due to how urbanization and an increasing population have compounded the problem. Communities have been compelled to turn to illegal dumping on public property due to a lack of an efficient waste management system, which has made the issue of disposal sites worse. Only 7.5% of waste is currently recycled in South Africa, while a startling 90% of it ends up in landfills there.

The DEA trash study estimates that South Africa produces an astounding 107.7 million tons of trash annually, which is disposed of at approximately 826 landfill sites. However, as waste site area becomes rarer, the country now faces an even bigger dilemma. It is essential to create an effective recycling management system that can handle the rising demand and efficiently sort vast amounts of waste while still staying relatively affordable to encourage more widespread usage.

This study aims to find ways to improve recycling management practices by examining the causes and effects of incorrect garbage disposal. The study will examine different methods or technologies used to automate litter sorting that can help create a waste management system that is more effective and environmentally benign. The main objective is to lessen the negative effects landfill sites have on the environment, public health, and the general wellbeing of South African communities.

In order to fully understand the complexity of trash management in South Africa, this dissertation will concentrate on the urgent need for creative and affordable litter sorting solutions. This research attempts to provide workable and sustainable solutions by examining present waste management procedures, looking at successful projects in other areas, and identifying the unique difficulties faced by South African dump sites. The study's conclusions and suggestions could lead to a healthier and cleaner environment, enhancing both the prosperity of nearby communities and the protection of the nation's natural ecosystems.

Background and Overview

Current State of Recycling in South Africa

South Africa produces an astounding 54,425 tonnes of garbage every day, making waste management a major concern for the nation. According to this sobering figure, South Africa ranks as the world's 15th largest producer of trash. Unbelievably, only 10% of the 59 million tonnes of trash produced in the nation overall is recycled, with the other 90% ending up in landfills or on the streets where they finally end up in the ocean.

Beyond aesthetic issues, the effects of this pervasive trash are detrimental to the ecosystem and wildlife. When animals try to eat non-digestible items after being attracted to the strewn debris, they frequently die as a result. Given that plastic debris, a large portion of litter, can take anywhere from 20 to 500 years to degrade, recycling and correct disposal are crucial to reducing the environmental impact of plastic waste.

The problem has been made worse by the rise in landfill locations around South Africa. The number of trash sites increased dramatically from 42 to 1,086 between 2008 and 2015. For those who live nearby, the fast spread of landfills has created grave health problems. Households are no longer as close to these areas as they once were; some are now only 2.8 kilometres away. Living near landfills puts residents at greater risk for health problems, including a 42% spike in asthma cases, an 18% rise in depression cases, and a 25% increase in tuberculosis infections.

Additionally, these dumps' faulty waste management and overflow of trash endanger both land-based and marine ecosystems. The environmental and ecological harm brought on by littering is further exacerbated by the polluting of land and the oceans. Additionally, landfills disturb wildlife habitats, which causes some

species to migrate. Landfills also contribute to climate change because biodegradable trash generates large amounts of methane gas, a powerful greenhouse gas with negative effects.

The creation and implementation of efficient waste management plans are necessary to address these urgent problems, particularly in the form of cost-effective technologies for sorting litter at South African disposal sites. It is possible to lessen the need for landfills, reduce pollution, and lessen the health hazards posed to local communities by enhancing garbage sorting and recycling processes.

The Recycling Process in South Africa

Recycling in South Africa is an industry that faces several challenges including inadequate infrastructure, lack of funding, and low levels of awareness among the public. However, despite these obstacles, South Africa currently recycles 46% of plastic and 70% of paper which is higher than most countries.

The cost of recycling in South Africa varies depending on the material being recycled and the location of the recycling facility. For example, a 10-kilogram carton of white paper costs 56 cents, and a kilogram of green glass costs R1, with prices differing from place to place. The EPR fee for all glass packaging is R86.64 per ton. The prices for different materials can vary depending on the demand for the material and the cost of processing it.

To recycle materials in South Africa, households and businesses collect recyclable materials such as paper, plastic, glass, and metal. The materials are then sorted into different categories and sent to recycling facilities for processing. Sorting involves separating different types of materials, while shredding breaks them down into small pieces for easy transportation and processing. Smelting and refining require specialized machinery to transform the materials into new products.

Recycled materials in South Africa are used to make new products such as paper, cardboard boxes, plastic bags, and bottles. However, waste that cannot be recycled or deemed unprofitable is often dumped in landfills, while other countries have equipment such as incinerator technologies that can be used to process plastics into oil or energy.

Overall, recycling in South Africa is a complex process involving both humans and machines, and the cost of recycling varies depending on several factors.

Problem Statement

To handle the rising trash creation and efficient sorting and recycling of litter, South Africa's current waste management and recycling infrastructure is insufficient. The issue is made worse by the absence of accessible and effective mechanisms for sorting trash at dump sites. In order to address the serious problems caused by littering, incorrect garbage disposal, and overcrowded landfill sites, immediate attention and action are required. The goal of this dissertation is to create a litter sorting system that is both affordable and effective for South African dump sites.

Objectives

The Bedfordview team has come up with a solution to the problems associated with waste sorting in landfills.

1. Investigate the current flaws at local recycling and waste management sites.
2. Propose the creation of a robotic arm and conveyor belt that can quickly sort trash onto a conveyor belt is our proposed answer.
3. We aim to answer to the question whether the implementation of relatively affordable litter sorting systems in local South African waste management sites are achievable.

Importance of the Study: Overall

The adoption of our system will play a significant role in resolving South Africa's urgent waste management problems. It is critical to find effective and affordable methods of classifying and managing the growing amounts of waste produced by the nation as landfills near their limits.

With the introduction of this novel approach, we hope to accomplish several important goals. First, we anticipate a significant decrease in the amount of trash taking up landfill space. A more arranged and space-

efficient landfill site will result from the systems more effective sorting skills, which will ensure that waste is effectively categorized and disposed of in the proper containers.

Additionally, putting in place the system will help lower the health dangers connected to landfills. We can safeguard the health of landfill workers as well as the local population by reducing the need for human workers to manually sort through rubbish. Communities near landfills will benefit from an improvement in the general public's health and well-being as a result.

Furthermore, we want to lessen the environmental damage brought on by the excess waste by managing and lowering the amount of litter in landfills. Among other things, this entails minimizing the quantity of trash that is dumped or washed into the ocean and harms marine life.

Importance of the Study: Robotics

The implementation of our garbage sorting technology will be crucial for both the advancement of robots and for addressing South Africa's urgent waste management concerns. Our work, which demonstrates the efficacy and promise of robotic technology in waste management, contributes to the field as robotics continues to develop and find applications in a variety of industries.

We want to show through our research that robotics can be a more practical and affordable alternative. We work to reduce the overall cost of deployment, maintenance, and operation by improving the robotic trash sorting system's design, functionality, and operational effectiveness. The technology must be affordable in order to be available to a wider range of communities and waste management facilities.

We hope to accomplish a number of important goals with this innovative method that will have larger effects on the robotics industry. First off, the capability of our system to drastically minimize the amount of waste taking up landfill space exemplifies the real-world application of robots in maximizing resource utilization and solving environmental issues. This accomplishment demonstrates how robotics has the ability to completely alter how garbage is managed around the planet.

Furthermore, our discovery has important environmental ramifications. We support the preservation of ecosystems and the protection of natural resources by efficiently managing and minimizing the quantity of litter in landfills. Robotics can be used to reduce environmental harm and promote sustainable practices by preventing garbage pollution in seas, which is caused by inefficient disposal methods.

Limitations of this Study

In order to clearly convey the scope and focus of our investigation, it is crucial to identify its constraints and bounds. It is important to understand that, despite the creative and encouraging nature of our research and development of the robotic arm system for waste sorting in dump sites, our study does not seek to offer a comprehensive and fully scalable solution for waste management on a regional, national, or international scale.

Our study's concentration on the creation and application of the sorting system as a starting point for tackling the difficulties of waste sorting in South African dump sites is one of its limitations. Our study aims to establish the system's viability and usefulness in a safe setting. However, in order for such systems to be scalable and widely used, additional factors would need to be taken into account, such as monetary investments, infrastructure needs, and legal frameworks that go outside the purview of our study.

Another restriction is that the majority of our research is devoted to classifying litter into broad categories like paper, glass, cans, and plastic. While a large majority of waste materials are covered by this method, there are a number of different waste types, such as hazardous or electronic garbage, that may need specialist sorting methods or other solutions. These particular waste streams and their distinct sorting needs are not covered by our study.

Furthermore, our study does not cover all aspects of waste management, such as collection, transportation, and recycling or final disposal of sorted garbage. The sorting stage is where the robotic arm system is most important, however there are many other stages and parties involved in the total trash management process.

As a result, our system's installation would need to be incorporated into a comprehensive waste management framework that also contains other essential elements.

It is crucial to consider our work as an important foundation and proof of concept for the application of cutting-edge trash sorting technology in landfill facilities. Future studies and projects must investigate the adaptability, affordability, and integration of such technologies into more comprehensive waste management plans. Our research offers perceptions and suggestions on the technological viability and potential advantages of robotic garbage sorting, laying the groundwork for future developments and real-world applications in the waste management industry.

Research Methodology

The methods used to conduct our research are as follows:

- Surveys and interviews with residents of our local areas were conducted as well as with waste management workers, and government officials regarding waste management and existing recycling processes.
- Site-visits were conducted to observe current waste management practices and infrastructure, and analysis of existing data on waste generation, disposal, and recycling in South Africa.
- Additionally, we conducted a gathering of opinions and ideas from different vendors and community leaders and waste management workers.

The surveys and interviews that were used are standardized questionnaires and pilot tests to ensure that questions are clear and unbiased to citizens.

These methods will allow us as to gather relevant information and insights that are guiding the design and deployment of an effective waste management solution.

Image 1: Dump site at Bellevue East were interviews with waste management workers were conducts

Outline of this Dissertation

In this dissertation we will discuss and go through a literature review to understand how recycling is approached by well-known companies as well as local ones. Additionally, we take a look at recycling within South Africa from a personal perspective or residents, government officials, and waste management workers.

We will also look into current methods used in recycling, diving deep into how they work as well as pointing out any shortcomings. From here we look into the usage of robotic arms and why they may be used. We also touch on different machine learning algorithms used for object detection and classification in recycling.

Furthermore, we critically analyse the methods used in recycling and point out any research gaps or future directions.

In Chapter 3 we look at a recycling system to be developed from a systematically approach, looking at considerations for system modelling, how the system will work and what technologies or aspects will be included in the prototype such as colour detection, object classification, and object localisation. From here, we look into the system's architectural modelling, going through proposed designs and all the components used in such a system. We also touch on the mathematical aspects of the prototype such as the arm and conveyor belts kinematics and schematics.

Chapter 4 goes into detail about the physical components used in the construction of the prototype, the reason for choosing them, as well the components' industry considerations. We also gain a better understanding of the hardware and software integration from the user manual. The development of the colour detection algorithm is explained in detail as well as the development of the robotic arm's movement, from serial connection to physical movement. Following this we discuss other developments considered while constructing the prototype such as brand detection and object classification using machine learning. Performance tests are discussed and results from these tests are announced.

Chapter 2: Literature Review

Waste management is a serious problem that requires ethical waste management techniques and sustainable solutions. The environmental effect of large firms like Coca-Cola has been significantly reduced because to measures like glass and plastic bottle recycling. Robotics integration in garbage and recycling management has also showed promise in terms of improving productivity, cutting costs, and fostering sustainability. This evaluation of the literature intends to investigate the efficacy of Coca-Cola's recycling initiatives and the use of robotics in trash management, with an emphasis on improving waste management at South African disposal sites. It also goes through technologies used in litter sorting such as machine learning and colour detection.

How is Recycling Approached

Coca Cola Glass Recycling

As the biggest beverage corporation in the world, Coca-Cola has long been dedicated to reducing its environmental effect through sustainable packaging and ethical waste management. The glass bottle recycling program started by the company has helped prevent millions of tons of rubbish from ending up in landfills and conserve precious resources. Because it is infinitely recyclable glass is the perfect material for recycling.

The first step in the glass bottle recycling program is collecting used bottles, and Coca-Cola has partnered with regional waste management firms in several nations to collect and transport old bottles to recycling facilities. After bottles are gathered, they are sorted via colour and checked for any impurities or contaminants. This is important to ensure that recycled glass lasts long and has increased quality. Once sorting is done, the bottles are reduced to tiny fragments, or a "cullet," which are then delivered to a glass recycling facility. In a furnace there, the cullet is combined with raw ingredients like sand, ash, and limestone before being melted down and shaped into new glass bottles. Utilizing cullet in the manufacture of glass lowers the demand for raw materials, conserves energy, and lowers greenhouse gas emissions.

The organization and its partners still face some obstacles and gaps in executing and growing the program, though. The accessibility and availability of recycling infrastructure in some areas is one of the biggest problems. It is challenging to collect and recycle used glass bottles in poorer nations due to inadequate waste management infrastructure.

Coca Cola Plastic Bottle Recycling

Like glass recycling, the gathering of used bottles is the first step in Coca-Cola's plastic bottle recycling operation.

The plastic bottles are then broken down into tiny fragments, or "flakes," in the following stage. To get rid of any contaminants left behind, these flakes are then cleaned and dried. To ensure that the flakes have purity and consistency, the flakes are separated again by kind and colour.

Following sorting, the flakes are melted down and used to create new products like plastic bottles, containers, and other items. Coca-Cola has also created technology that makes it possible to produce bottles made entirely of recycled plastic without sacrificing the product's quality or safety.

The recycling program for Coca-Cola plastic bottles provides several advantages. The first benefit is that it decreases the quantity of plastic garbage that pollutes landfills and oceans. Second, it prevents the creation of virgin plastic, saving vital resources.

Much like the glass recycling program, the absence of infrastructure and collection methods in many areas, particularly in developing nations, is a significant flaw in the program. It is challenging to gather, and process used plastic bottles due to poor or non-existent waste management infrastructure in many regions of the world.

Mpact Recycling Process

The sorting process guarantees that the recovered material fulfils the necessary standards and criteria, making it a crucial step in the waste recycling process. At Mpact, the recyclable waste material is first received before being sorted. Transported to the Mpact waste processing facility, the collected material passes through a preliminary sorting procedure.

Separating distinct trash categories, such as plastics, paper, metals, and glass, is the first step in sorting waste. The garbage is fed into a sorting device that separates the various materials using a variety of techniques. By utilizing sensors, magnets, and air currents, the device separates the materials according to their physical characteristics. For instance, magnets are employed to remove any metals that may have been mixed in with the plastic after the machine utilizes sensors to identify the various types of plastics.

Following the initial sorting, the various materials are segregated and taken to designated sorting locations where the garbage is physically sorted once again. Trained personnel visually inspect the waste at this point and eliminate any materials that do not belong in the recycling stream. As an illustration, PVC and other polymers that are not recycled are deleted.

After being sorted, the trash is put into shredders, where it is cut into smaller bits. The smaller waste items are then washed to get rid of any impurities like glue, labels, and dirt. High-pressure water jets are used throughout the washing procedure to clean the trash. Following washing, the trash is once more separated using specialized machinery that divides the components by density.

The waste that has been sorted must next be examined to make sure it complies with the necessary requirements. The purity, size, and presence of any impurities are checked on the material. Any material that does not adhere to the necessary requirements is taken out and either sent for further processing or discarded.

One potential gap in the sorting process at Mpact is the possibility of human error in the manual sorting stage. Although trained operators visually inspect the waste, there is still the possibility of errors in identifying the specific recycling stream or removing contaminants.

The risk of human error during the manual sorting stage is one potential flaw in Impact's sorting procedure. Despite skilled personnel visually inspecting the waste, the chance of mistakes in identifying the precise recycling stream or removing impurities still exists.

Research Methodology on local waste management sites

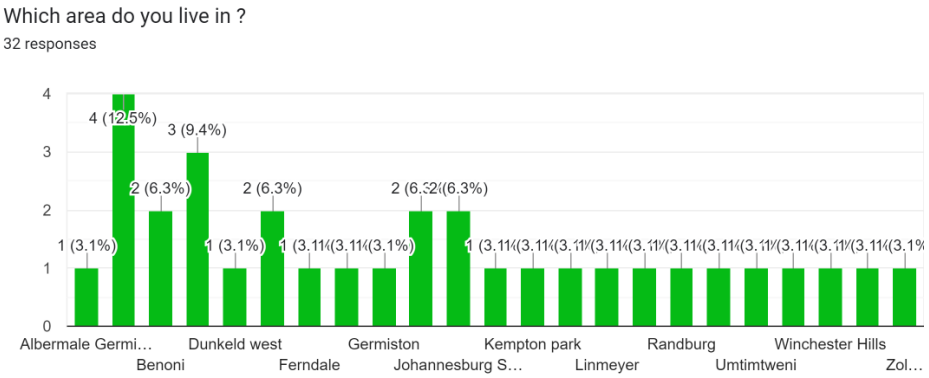
We visited the pick it up area in Bruma to see how they categorize the trash that they bring into the area, they mostly delt with garden objects as well as car tires. They usually bring about 2 trucks worth of trash per day.

This is the area that sorts out mostly books as well as glass bottles, when we got to the area, they told us that this area is a separate area the other area that deal with rubber and garden trash, this area was only for papers and books as well as glass.

The image above shows a container which holds where all the garden trash goes in, the container can hold about 200-400 kg of trash per container.

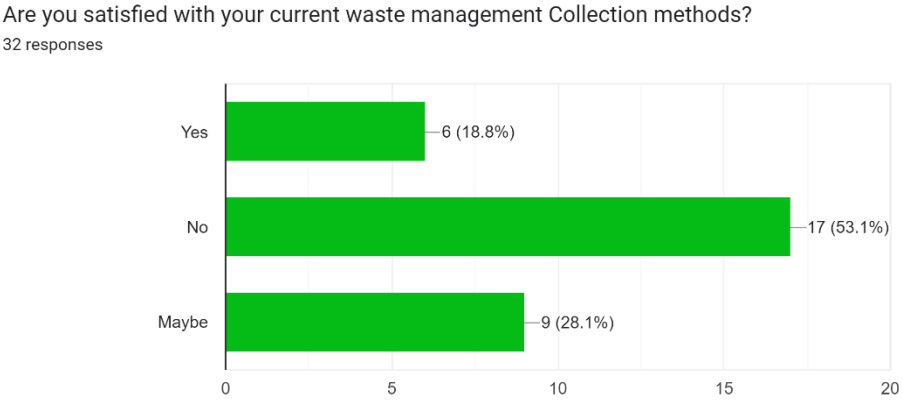
Th image shows the amount of car tires piled up and waiting to be sorted out at a pick it up zone. We unfortunately could not get access to a landfill site to ask questions and take pictures as we were not authorized, so instead we went to a pick it up zone.

Statics from Interviews and Surveys:

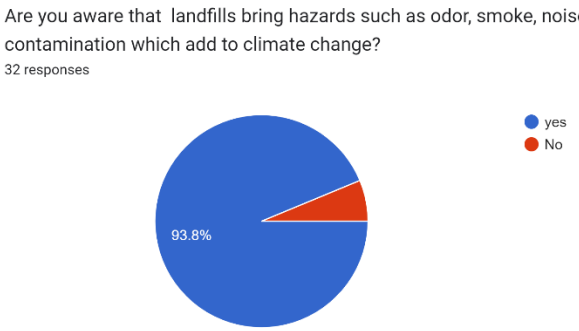


We did a survey online where we created the forum and sent it to different people to get feedback on the survey. Based on the image illustrated above we can see that we have received feedback from 32 different people online. We then printed a different survey with similar questions, and we took it to different pick it up sites to get the people on site to fill out the survey to see their feedback.

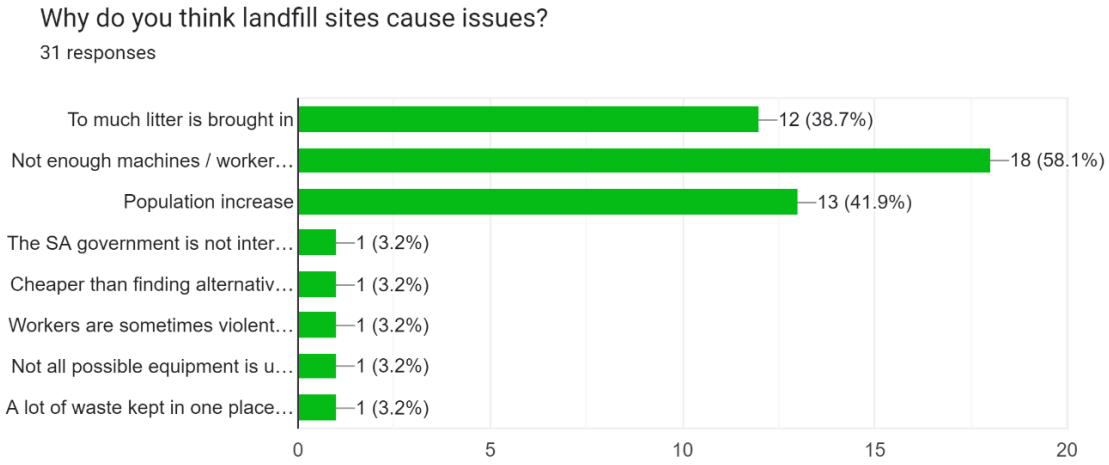
In this image we have a pie chart graph based on the amount of percentage of people that have heard of waste management in contrast to people that have not heard of waste management. 90,6 % people have heard of waste management.



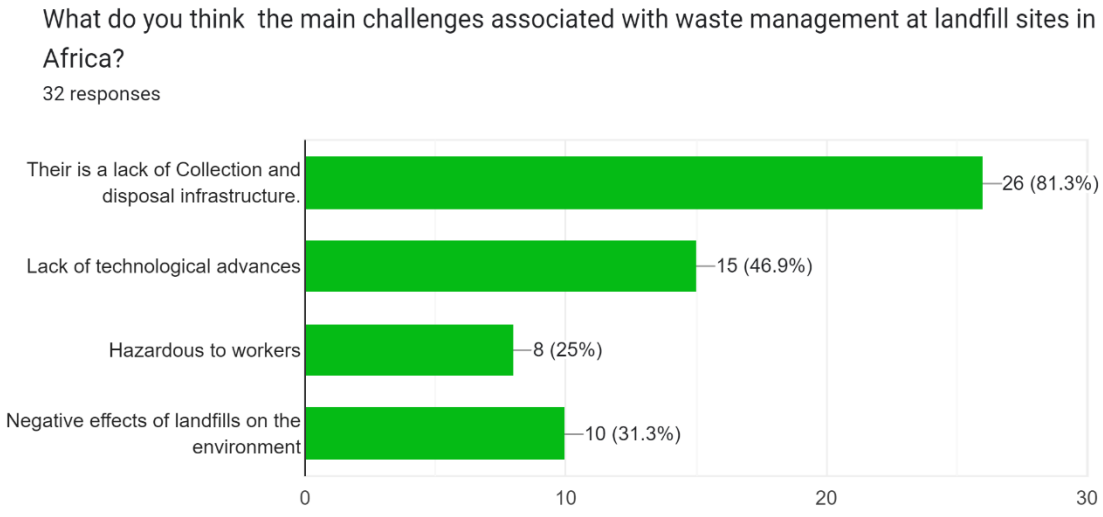
Above we have an image of a graph based on the amount of people that are satisfied with the collection method, 18,8% are happy, 53,1 % are unhappy and 28% are unsure.



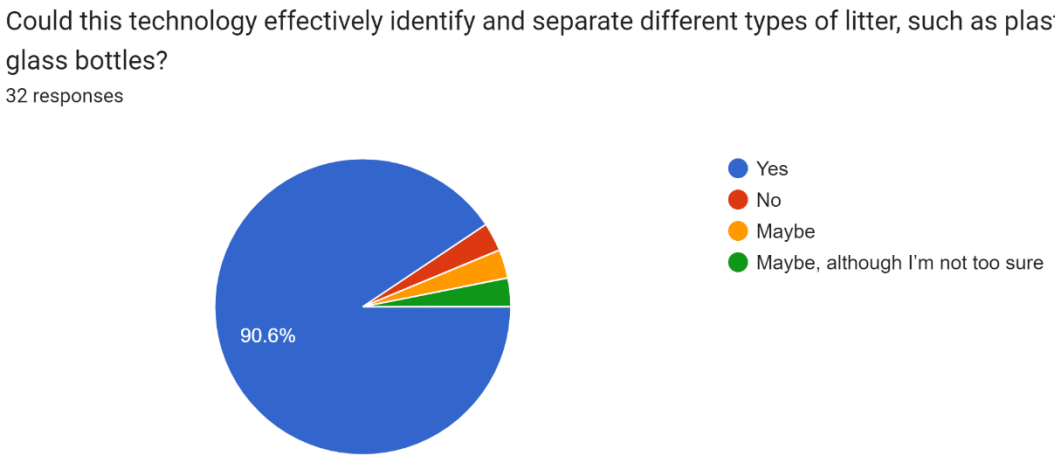
Above is an image of a pie chart of people that are aware of landfill sites being a hazard zone in contrast to people that are unaware. 93.8 % are aware of landfill sites being a hazard zone to the people that are around as well as the people that are working on site.



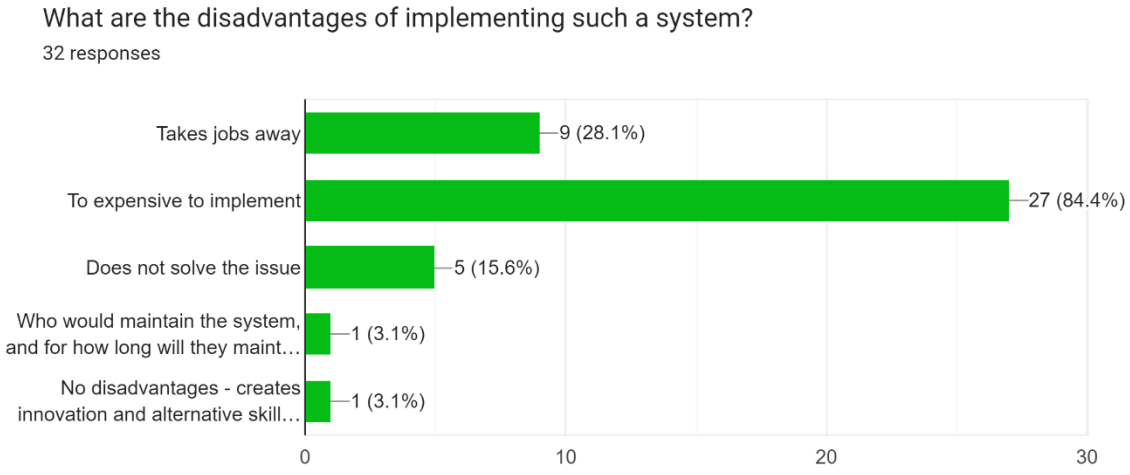
Above is an image based on graphs that contain the causes of landfill contribution. The biggest contribution according to the people is that landfill sites do not have enough machines/ workers implemented on the site.



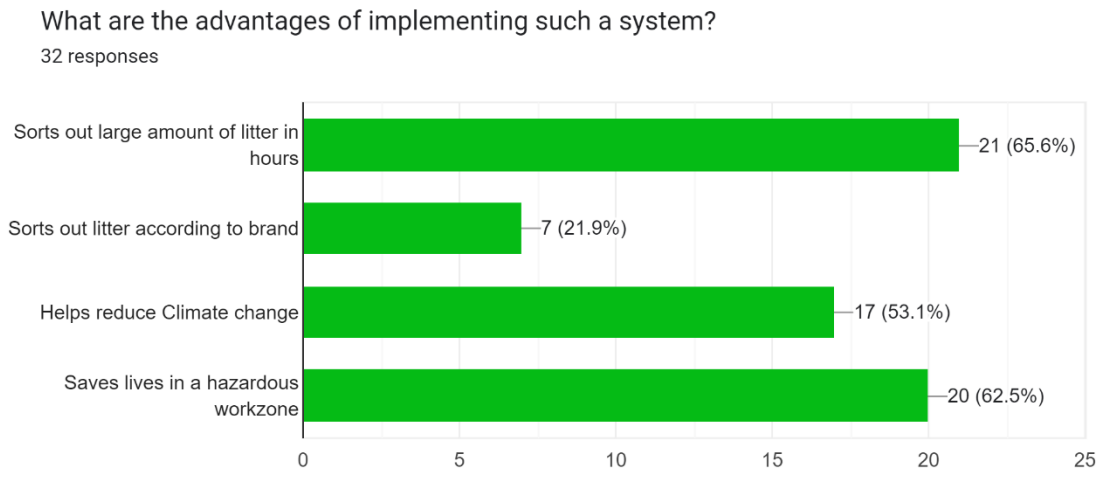
The graph above shows a how waste management is associated in South Africa, 81% say that there is a lack of collection and disposal infrastructure. And 46,9% of people say that there is a lack of technology advances.



The pie chart above shows that 90,6% of people agree that technology could effectively separate the litter into categories.



84,4% of people think that by implementing such a system would be expensive to implement. 28,1% people are worried that the system might take away such jobs.



65,6% of people say that by implementing such a system it would sort a large amount of litter in a few hours, the other advantage is that it saves people from working in a hazardous zone.

Robotics in Waste and Recycle Management

Bin-e

Image 2: Bin-e

Bin-e is a recycling-separating robot that sorts and separates several types of rubbish automatically using artificial intelligence and computer vision technologies. Bin-e is intended to be deployed in public areas where there is a lot of garbage production, such as airports, train stations, malls, and office buildings.

Users are urged to throw their waste in the robot Bin-e, which is placed next to a regular trash can. To detect and classify the type of garbage, such as plastic, paper, metal, or glass, the robot uses sensors and cameras. A mechanical arm and conveyor belt system is then used to sort and segregate the waste, after which it is placed in the proper recycling bin.

Advantages:

- Better Waste Management: By utilizing AI to classify and recycle various sorts of waste, Bin-e improves waste management. As a result, less waste is dumped in landfills, lowering environmental contamination.
- Bin-e saves time and money by doing the manual sorting for you, which can be expensive and time-consuming. Large amounts of garbage may be quickly handled by the automated system, saving time, and lowering personnel expenses.
- Higher Recycling Rates: By classifying waste into distinct categories, such as paper, plastic, and glass, Bin-e facilitates recycling and raises recycling rates while lowering the amount of waste that ends up in landfills.
- Data gathering: Bin-e can provide insightful information about the different sorts of waste being created, which can be used to improve waste management practices for the future.

Disadvantages:

- Cost: Buying and installing bin-e can be expensive initially. Some businesses or people who might not have the money to invest in technology may find this to be a hindrance.
- Upkeep: Bin-e is a complicated system that needs frequent maintenance to make sure it is operating correctly. It can take a lot of money and time to do this.
- Limited Capacity: Due to the types and quantity of waste that can be handled at once, Bin-e might not be able to handle all sorts of waste.
- Technology Dependence: Bin-e is dependent on technology, which is sometimes prone to glitches or other issues. Processes for trash disposal may be delayed or disrupted because of this.

AMP Robotics' Clarke

Image 3: AMP Robotics' Clarke

The Clarke garbage sorting and recycling system by AMP Robotics automates the procedure using robotics, artificial intelligence, and computer vision. The technology is made to assist recycling operations in increasing their productivity and accuracy while lowering their expenses and negative environmental effects.

Clarke is essentially a robotic arm that is connected to a conveyor belt that transports rubbish through the system. Advanced sensors and cameras on the robot arm take pictures of the materials as they move through, and artificial intelligence (AI) algorithms analyses the pictures to identify each piece and determine whether it can be recycled or not.

Overall, Clarke is an innovative garbage sorting and recycling system that makes use of the most recent advancements in robotics and artificial intelligence (AI) technology to assist recycling facilities in streamlining their operations, cutting costs, and promoting sustainability.

Advantages:

- Greater Efficiency: Clarke can efficiently sort enormous amounts of garbage, saving time and money compared to hand sorting.
- Increased Recycling Rates: Because the Clarke can recognize and classify a wide range of materials, more recyclable materials are retrieved, and less garbage ends up in landfills.
- Improved Sustainability: Clarke can contribute to the advancement of a more sustainable future by raising recycling rates and lowering the quantity of garbage disposed of in landfills.

Disadvantages:

- Data Analytics: To enhance waste management procedures and optimize recycling operations, Clarke offers real-time data analytics.
- Less Manual Labor Needed: By reducing the amount of manual labour required in sorting facilities, Clarke can improve working conditions for employees.

Ramudroid v8

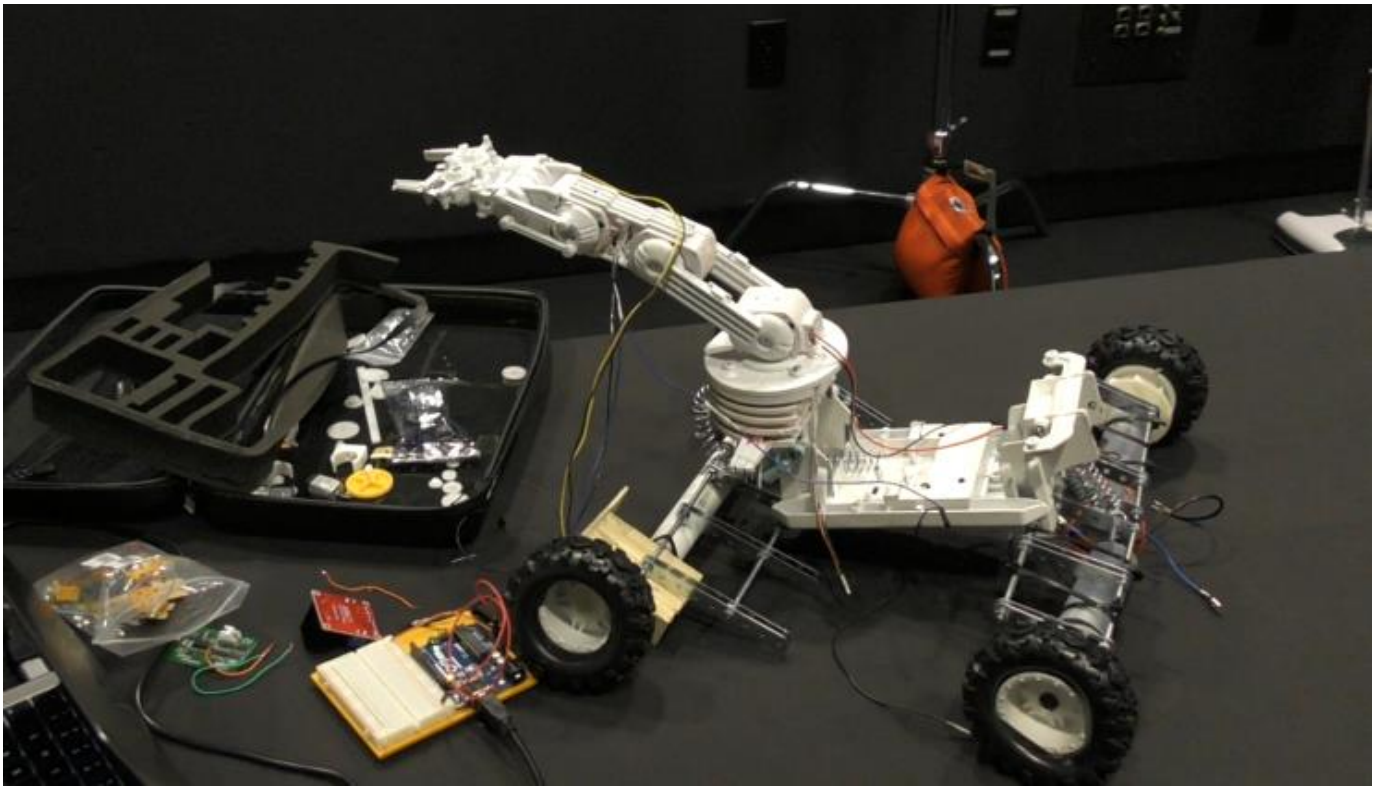


Image 4: Ramudroid

The Ramudroid v8 robot is a trash-collecting device that spots and gathers waste in public areas using sensors and computer vision technologies. The robot can be used in a variety of locations, including parks, malls, and public plazas, and is built to function independently without human assistance.

The Ramudroid v8 robot scans the area and detects litter using a variety of sensors, including cameras and lidar. When the robot spots a piece of litter, it walks toward it, picks it up with a robotic arm, and drops it into its internal trash can. The robot can run for several hours on a single battery and can be configured to use its onboard GPS system to travel paths or through regions.

Advantages

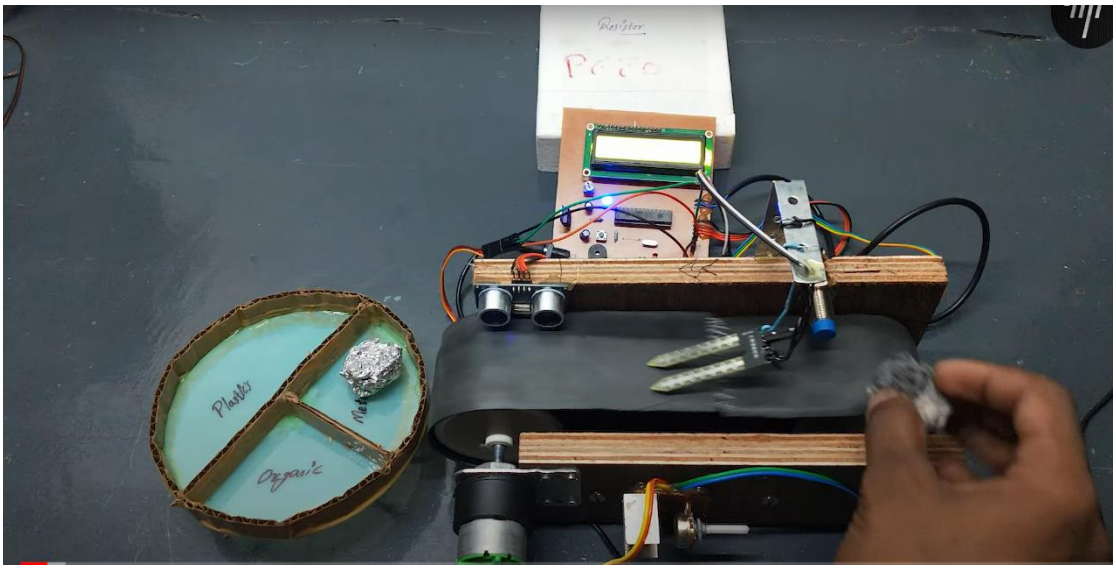
- Built to function autonomously, requiring less human involvement. This can increase the effectiveness and affordability of waste collection, especially in big public areas.
- Can help lessen the negative effects of waste on the environment by collecting rubbish and litter from public areas. Also, the robot can aid in keeping litter out of natural areas like streams.
- Can contribute to better public health and safety by collecting garbage and waste. Litter can attract bugs and rodents, as well as pose a risk to drivers and pedestrians.

Disadvantages:

- Due to its small waste bin capacity, the robot may need to be often emptied in busy places.
- Susceptible to technical issues and malfunctions, much like any robotic system, which could cause delays or downtime.
- For certain organizations, especially smaller firms and governments, the initial costs of purchasing and deploying the robot may be prohibitive.

The user adoption and acceptance of all three robotic technologies could present problems. For instance, there can be issues about data security and privacy, or about how automation would affect human jobs in the waste management sector. To efficiently deploy and maintain all three systems, a considerable investment in time, money, and skill may be necessary.

Garbage sorting machine using Atmega 328/Arduino



The inductive proximity sensor can be used to detect metallic objects in the litter. It can differentiate between metallic and non-metallic items, allowing for the separation of metallic waste for recycling or proper disposal. The ultrasonic sensor can be used to measure the distance or level of waste in a bin or container. It can help determine if the bin is fully or partially filled, enabling efficient waste management and sorting. While the soil moisture sensor is typically used in agricultural applications, it might be repurposed in the garbage sorting machine to detect the presence of liquid or moisture in waste materials. This can be useful for separating wet or liquid waste from dry waste.

Components used:

- Atmega 328/Arduino
- Inductive proximity sensor
- Ultrasonic sensor
- Soil moisture sensor
- 16x2 LCD display
- Servo SG90
- Buzzer 5v
- 16mhz crystal
- 7805 voltage regulator
- 10k preset
- 22pf x2
- 10uf x2
- 10k resistor
- LED
- Male header pins

Inductive Proximity Sensor: An inductive proximity sensor utilizes electromagnetic fields to detect the presence of metallic objects. It works based on the principle of electromagnetic induction, where a coil generates an electromagnetic field. When a metallic object enters the sensing range, it disrupts the field, triggering a detection signal.

Ultrasonic Sensor: Ultrasonic sensors use sound waves in the ultrasonic frequency range to detect the presence and distance of objects. They emit ultrasonic waves and measure the time it takes for the waves to bounce back after hitting an object. This information is then used to determine the distance to the object.

Soil Moisture Sensor: Soil moisture sensors are typically designed for agricultural purposes to measure the moisture content in soil. They utilize electrical conductivity or capacitance to determine the moisture level. In the context of a garbage sorting machine, they might be repurposed to detect the presence of liquid or moisture in waste materials.

16x2 LCD Display: The 16x2 LCD (Liquid Crystal Display) is a commonly used visual output device that can display text and simple graphics. It provides a user interface for displaying information, system status, sensor readings, or prompts during the garbage sorting process.

Servo SG90: The SG90 servo motor is a small, low-cost motor capable of precise angular positioning. It can be used to control movements, such as manipulating a mechanical arm or controlling a conveyor belt in the garbage sorting machine.

Type of equipment used:

Microcontroller Board: Atmega 328/Arduino - The microcontroller board acts as the brain of the garbage sorting machine. It runs the control logic, receives sensor inputs, and sends commands to other components. The Atmega 328 and Arduino boards are commonly used microcontroller platforms.

Sensors: Inductive Proximity Sensor, Ultrasonic Sensor, Soil Moisture Sensor - These sensors detect and measure specific characteristics of the litter, such as metallic presence, distance, or moisture level.

Display: 16x2 LCD Display - The LCD display provides visual feedback, displaying information about the garbage sorting process or system status.

Servo Motor: Servo SG90 - The SG90 servo motor is used for precise control of movements in the garbage sorting machine.

Other Components: Buzzer, Crystal, Voltage Regulator, Preset, Capacitors, Resistors, LED, Male Header Pins - These components are part of the electronic circuitry and infrastructure of the system, including power regulation, signal conditioning, and user interface elements.

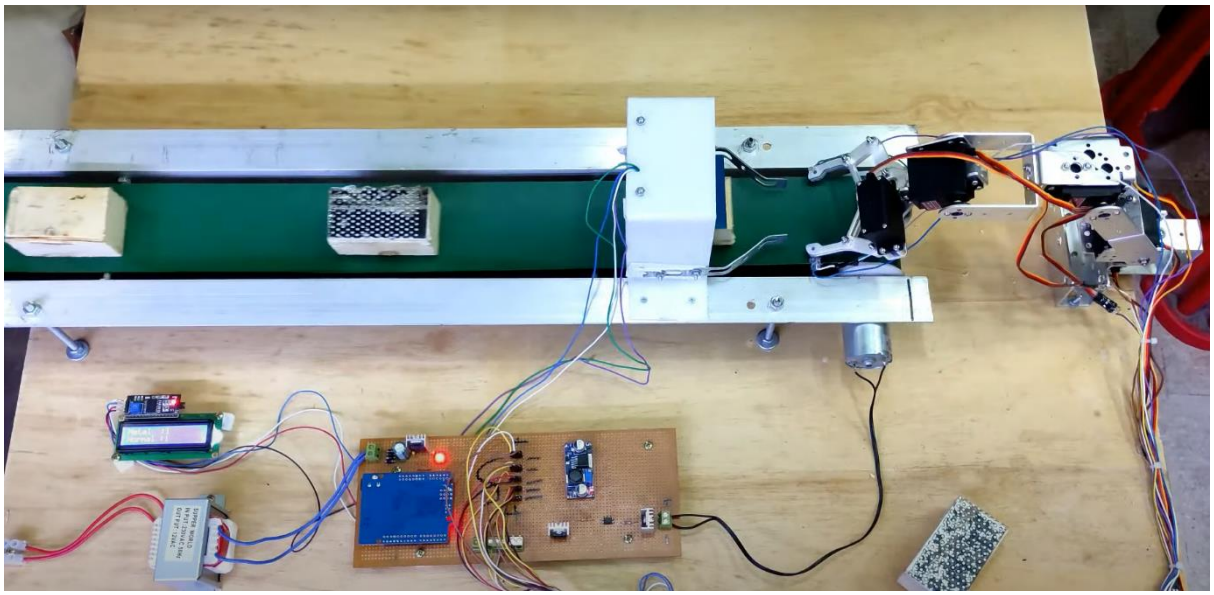
Advantages:

- Cost-effective: Using Atmega 328/Arduino is a cost-effective solution for designing a garbage sorting machine.
- User-friendly: The Atmega 328/Arduino microcontroller makes it simple for anyone to control and program the device, making it user-friendly.
- Effective sorting: Different types of trash can be sorted efficiently using sensors and programming, which lessens the workload placed on workers and improves sorting accuracy.
- Low maintenance: By using Atmega 328/Arduino, the requirement for frequent maintenance is minimized, which lowers operating costs overall.

Disadvantages:

- Limited functionality: The garbage sorting machine's functionality may be restricted by the Atmega 328/Arduino's low processing speed and memory.
- Reliability: The Atmega 328/Arduino may not be able to handle unforeseen mistakes or system breakdowns, raising questions about the machine's dependability.
- Compatibility problems: If the machine needs to be connected to other hardware or software that doesn't support Atmega 328/Arduino, compatibility problems may occur.
- Complexity: For certain people, the design and operation of the waste sorting machine using Atmega 328/Arduino may be challenging due to the technical knowledge and skill required.

Metal and Nonmetal product sorting Robotic arm



This system used a conveyor belt and robotic arm to sort metal and non-mental objects. In order to detect and sort out the litter, the metal sensor (inductive proximity sensor) can detect metallic objects, while the IR sensor can be used to detect the presence or proximity of objects in general. Once the sensors detect an object, the Arduino Uno receives the sensor inputs and processes the data. By analysing the characteristics of the detected objects, such as their shape, colour, or material composition, a classification algorithm can be implemented. This algorithm can determine whether the object is litter or not and classify it accordingly. Upon classifying the object as litter, the robotic arm equipped with the MG996 servo motors can be used to manipulate and sort the objects. The arm can pick up the litter and place it in a designated location for further processing or disposal. The 16x2 LCD display and I2C module can be utilized to provide a user interface. It can display relevant information about the sorting process, instructions, or any user interaction options. The 12V power supply and DC to DC buck converter ensure that the system components receive the required power and voltage levels for operation.

Components used:

- Arduino Uno
- 16x2 LCD Display
- i2c module
- 5 Axis Robotic Arm
- MG996 Servo motor x 5
- 12v DC gear motor
- DC to DC buck converter
- Metal Sensor
- IR Sensor
- 12v power supply

Types of equipment used:

Microcontroller Board: Arduino Uno - The microcontroller board serves as the central control unit for the system, responsible for executing the sorting algorithm and coordinating the operation of other components.

Display: 16x2 LCD Display - The LCD display provides visual feedback and information display during the sorting process.

Communication Module: I2C Module - The I2C module facilitates communication between the microcontroller board and the LCD display, allowing for simplified wiring and control.

Robotic Arm: 5 Axis Robotic Arm - The robotic arm is the mechanical component responsible for manipulating and sorting objects. It typically consists of joints, linkages, grippers, and end-effectors.

Servo Motors: MG996 Servo Motor x 5 - Servo motors are used to control the movement and positioning of the robotic arm. The MG996 servo motors are commonly used due to their ability to provide precise angular control.

DC Gear Motor: 12V DC Gear Motor - The DC gear motor is utilized to drive the conveyor belt, or any other mechanical components involved in the transportation of objects through the sorting process.

Power Converter: DC to DC Buck Converter - The DC-to-DC buck converter is used to regulate the voltage from the 12V power supply to a lower voltage required by the microcontroller board and other components.

Sensors: Metal Sensor, IR Sensor - The metal sensor (inductive proximity sensor) is employed to detect metallic objects, while the IR sensor is utilized to detect the presence or proximity of objects in general.

Power Supply: 12V Power Supply - The 12V power supply provides electrical power to operate the entire system.

These types of equipment, when combined and properly integrated, form a functional system for sorting metal and non-metal products using a robotic arm. It's worth noting that specific brands, models, or variations of these components may exist, and the actual equipment used can vary based on the system design, requirements, and availability.

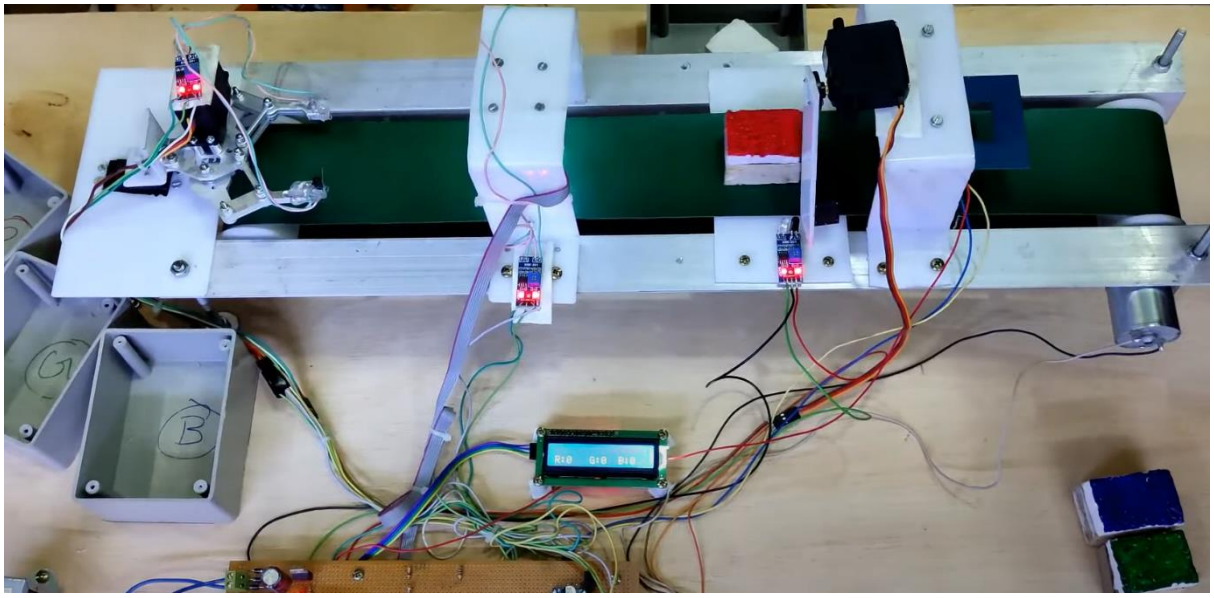
Advantages:

- Precision: By using a robotic arm to sort metal and non-metal objects precisely, the sorting process can be made more accurate and effective.
- Speed: By sporting goods more quickly than humans can, the robotic arm cuts down on the time needed.
- Cost-effective: Sorting metal and non-metal products with an Arduino-based automation project can save money by cutting labour expenditures.
- Consistency: Since the robotic arm can sort products repeatedly without growing fatigued, the outcomes of the sorting process are more reliable.

Disadvantages:

- Initial Cost: Due to the price of the robotic arm and other hardware components needed for the automation project, the project's initial cost may be significant.
- Installation that requires technical expertise: The installation of the robotic arm and the automation project can be difficult.
- Limited capability: The robotic arm's capability for other sorting activities may be limited to metal and non-metal product separation.
- Reliability: The robotic arm's dependability could be a problem because it could eventually develop mechanical or electrical problems.

Colour sensor and metal sensor product sorting conveyor belt Arduino automation project



Colour Sensing: The TCS230 or TCS3200 Colour Sensor is used to detect and identify the colours of the objects in the litter. This sensor typically consists of an array of photodiodes and filters that allow it to measure the intensity of red, green, and blue (RGB) light reflected from the objects. By analysing the RGB values, the colour sensor can determine the colour of each object.

Metal Sensing: The Metal Sensor is utilized to detect the presence of metallic objects in the litter. It employs an inductive proximity sensor principle to identify objects with metal properties based on their electromagnetic response. When a metallic object is detected, it can be classified as metal for further processing or separation.

Robotic Manipulation: The combination of the Robotic Gripper and the MG996 Servo Motors provides the ability to manipulate and handle objects. The robotic gripper is attached to the servo motors, which allow precise control over the gripping action and movement of the gripper. This mechanism enables the sorting system to pick up and transfer objects based on their colour and metal properties.

Conveyor Belt: The 12V DC gear motor and DC to DC buck converter are likely used to drive the conveyor belt, which facilitates the movement of objects through the sorting system. The conveyor belt helps transport the litter to the appropriate sorting stations based on the colour and metal properties detected.

IR Sensors: The three IR sensors are mentioned in your components list, but their specific application in the litter sorting process is not clear. However, IR sensors are commonly used for proximity sensing, object detection, or as part of feedback mechanisms in automated systems. They could be integrated into the sorting system to provide additional functionality, such as detecting the presence of objects, verifying sorting accuracy, or assisting in conveyor belt control.

User Interface: The Arduino Uno and 16x2 LCD Display, along with the I2C module, are used for the system's control and user interface. The Arduino Uno acts as the main control unit, processing the sensor inputs and controlling the actions of the sorting system. The LCD display provides visual feedback, displaying relevant information about the sorting process, system status, or prompts for user interaction.

Components used:

- Arduino Uno
- 16x2 LCD Display
- i2c module

- Robotic Gripper
- MG996 Servo motor x 3
- 12v DC gear motor
- DC to DC buck converter
- TCS230 TCS3200 Colour Sensor
- Metal Sensor
- IR Sensorx3
- 12v power supply

Types of technology used:

Arduino Uno: Arduino Uno is a microcontroller board based on the ATmega328P chip. It provides the processing power and I/O capabilities to control and coordinate the operation of various components in the litter sorting system.

16x2 LCD Display: The 16x2 LCD (Liquid Crystal Display) is a visual output device that can display text and simple graphics. It serves as the user interface, providing information about the sorting process and system status.

I2C Module: The I2C (Inter-Integrated Circuit) module facilitates communication between the Arduino Uno and the LCD display. It allows for simplified wiring and control, using a two-wire serial interface.

Robotic Gripper: The robotic gripper is a mechanical component that mimics a hand or clamp and is used to grasp and manipulate objects. It enables the system to pick up and transfer litter based on its properties.

MG996 Servo Motors: MG996 servo motors are high-torque motors that provide precise angular control. They are used to drive the robotic gripper, allowing for controlled movement and gripping actions.

12V DC Gear Motor: The 12V DC gear motor is utilized to drive the conveyor belt in the sorting system. It provides the necessary mechanical power to move the litter along the sorting process.

DC to DC Buck Converter: The DC-to-DC buck converter is used to regulate the voltage from the 12V power supply to a lower voltage required by the Arduino Uno and other components in the system.

TCS230/TCS3200 Colour Sensor: The TCS230 or TCS3200 colour sensor is an integrated circuit that can detect and measure the intensity of red, green, and blue light reflected from objects. It allows the system to identify the colour of the litter.

Metal Sensor: The metal sensor, which is likely an inductive proximity sensor, detects the presence of metallic objects in the litter. It can differentiate between metal and non-metal materials.

IR Sensors: The three IR (Infrared) sensors are used in the litter sorting system, although their specific application is not clear from the provided information. IR sensors are commonly used for proximity sensing, object detection, or as part of feedback mechanisms in automation systems.

Buzzer: The buzzer is an audio output device that can generate audible alerts or signals. It may be used in the system to indicate certain events or notifications during the sorting process.

Types of equipment used:

- Microcontroller Board: Arduino Uno
- Display: 16x2 LCD Display
- Communication Module: I2C Module
- Mechanical Component: Robotic Gripper

- Servo Motors: MG996 Servo Motors
- DC Motor: 12V DC Gear Motor
- Power Converter: DC to DC Buck Converter
- Colour Sensor: TCS230/TCS3200 Colour Sensor
- Metal Sensor: Inductive Proximity Sensor
- IR Sensors: Infrared Sensors
- Audio Device: Buzzer

Advantages:

- High precision: The use of colour and metal sensors enables exact product sorting based on colour and metal characteristics, increasing the sorting process's accuracy.
- Speed: The conveyor belt can sort goods more quickly than people can, cutting down on the time needed for sorting.
- Efficiency: The sorting process is made more efficient by automation, which also lowers labour costs and boosts productivity.
- Versatility: The conveyor belt can be used for sorting a wide range of products, making it a versatile solution for sorting tasks.

Disadvantages:

- Complexity: The installation process of the conveyor belt and the automation project can be complex and require technical expertise.
- High initial cost: Due to the price of the conveyor belt, sensors, and other hardware components needed for the automation project, the project's initial cost may be expensive.
- Limited functionality: The conveyor belt's capability to perform further sorting operations may be restricted to colour and metal qualities.
- Reliability: The conveyor belt and sensors' dependability may be a problem because they could eventually develop mechanical or electrical breakdowns, which would result in downtime and maintenance expenses.

Litter Classification

Waste management relies heavily on trash classification since it enables effective sorting and proper disposal of various forms of litter. Litter sorting robots can independently identify and classify a variety of things using machine learning algorithms, easing the segregation process at disposal sites. This review explores the possibilities of using machine learning algorithms to improve South African waste management procedures.

VGG-16 Model

The VGG-16 model, put out by Oxford University academics (Simonyan & Zisserman, 2014), uses a novel strategy by sequentially employing several 33 kernel-sized filters. Although this model has a high level of image categorization accuracy, it has some drawbacks. The VGG-16 model takes a long time to train and uses a lot of storage space and bandwidth because to its numerous parameters. In addition, the model's many parameters may cause the gradients to explode (Ferguson, 2014).

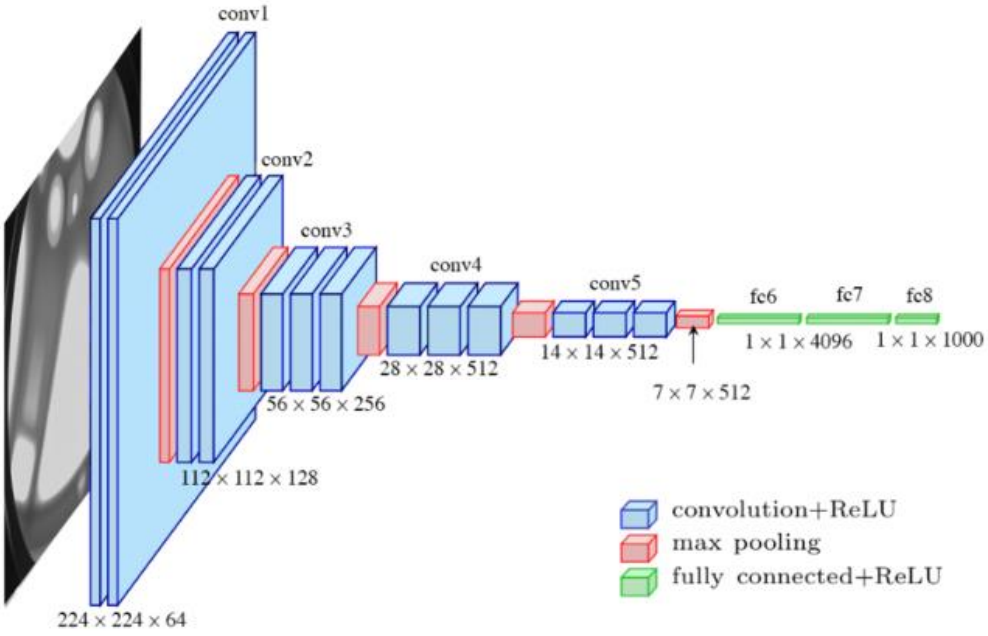


Diagram 1: VGG-16 Model

ResNet Model

The Microsoft Research team created the ResNet model to address the vanishing gradient issue that was present in earlier models (He et al., 2015). It introduces skip connections and residual blocks, which link a layer's activations to skip over lower-level layers. By stacking these residual blocks, this design enables the construction of the ResNet model. The main benefit of ResNet is its capacity to reduce the adverse effects of any one layer on the architecture's overall performance. According to Mahmood et al. (n.d.), this method considerably boosts model performance and training effectiveness.

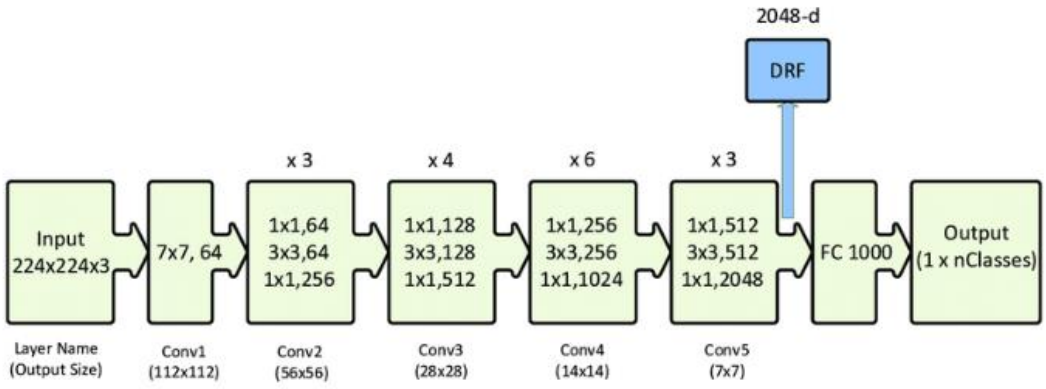


Diagram 2: ResNet Model

PNASNet-5 Model

The PNASNet-5 model (Liu et al., 2018) presents a novel method for understanding the structure of Convolutional Neural Networks (CNNs), which was developed by researchers from Johns Hopkins University, Google AI, and Stanford University. For model training, it uses evolutionary and reinforcement learning methods, producing a quicker and more effective output than earlier models. The PNASNet-5 model exhibits potential capabilities for precise rubbish classification by utilizing these cutting-edge methods (Tsang, 2020).

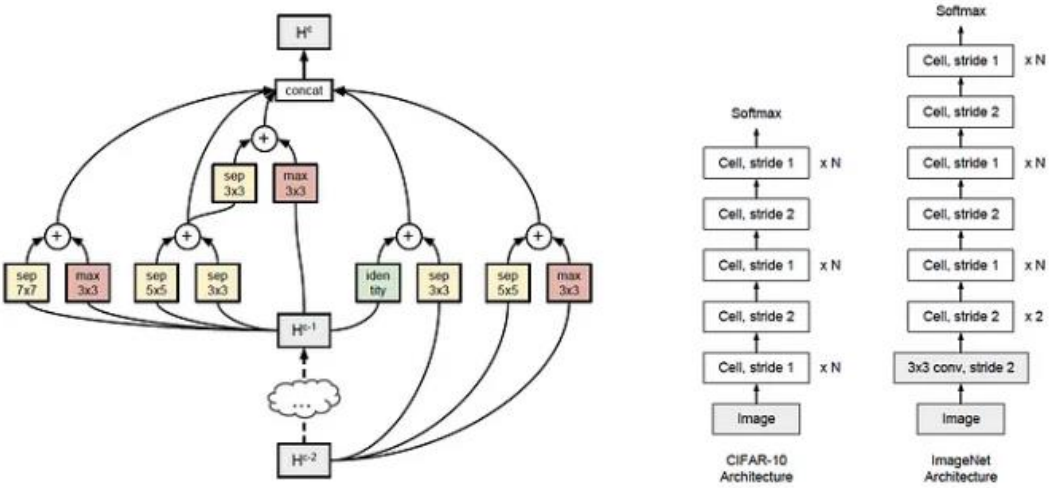


Diagram 3: PNA SNet-5 Model

Evaluation Metrics

Evaluation metrics are crucial for evaluating the effectiveness of the models mentioned. While error rates give a general picture, it is also important to take into account other measures like precision, recall, and F1-score. The models' performance in object identification and classification tasks can be better understood using these indicators.

Sorting using Robotic Arms

Choosing the right robotic arm is essential for effective and precise categorization in the field of garbage sorting. This section will examine several robotic arm types that are frequently employed in a variety of applications while taking into account their benefits and drawbacks. We can choose an appropriate design for our waste sorting robotic arm by comprehending the specifics of each type.

Scara Robotic Arm

The Scara arm is used in applications where it is necessary to focus on high speed as well as high precision situations. The Scara arm can move well along the lateral axis. The Scara arm is extremely useful in scenarios where the robot needs to assemble an object or disassemble objects, as well as PCB soldering as that requires high precision robots which the Scara robot can perform. For example, the SCARA robot arm would not perform well in a situation where it needs to weld a car frame as it would need to weld in both lateral and horizontal motions, this is where the SCARA robot arm would not perform well.

Image 5: Scara Robotic Arm

Articulated Robotic Arm

The articulated robotic arm can have up to 10 or more axes of movement, the articulated arm is able to provide to a degree a lot more freedom of movement that would almost mimic a human arm. Th articulated arm is most used in the manufacturing industry due to the following reasons.

The articulated arm is great as it can handle tasks in a wide area, but they are great at dealing with

- Machine assembly.
- Welding.
- Handling material.

The flexibility of the joint allows the robot to move in directions that otherwise would be difficult for other robots to move. The articulated robot can move at specified angles that would be difficult for other robots such as the SCARA to achieve.

Electronics, in the electronics industry precision is highly required for the small components to be placed correctly so that it can be soldered correctly. For this to occur correctly we will need the robotic arm to be able to move at precise angles. This is also a weakness in the articulated arm as in most cases the articulated arm is not able to reach this level of precision due to the mechanical build of the robotic arm.

Image 6: Articulated Robotic Arm

Cartesian Robotic Arm

These robotic arms are known as linear robots, as this robotic arm moves along three linear axes using the 3D axes plane. This type of robotic arm is usually referred to as a 3D printer as it is only able to move in a linear direction in the 3D plane such as the X, Y and Z just like a 3D printer. Unlike other robotic arms the Cartesian arm has a limited set of movement. Due to the limited amount of freedom the Cartesian arm can perform a high level of precise movement. The Cartesian arm is usually built on a conveyor which will pull the gantry of the arm to allow the arm to be pulled in the specified direction.

The cartesian arm excels in task where precision is required such as PCB assembly, 3D printing, as well as CNC applications. The reason why this is precise as the movement is more rigid, when you have more degree of freedom there is more chances of moving parts to move slightly which will affect the precision movement.

Image 7: Cartesian Robotic Arm

Critical Analysis and Synthesis

Various robotic technologies and automation initiatives created for trash management and rubbish sorting were examined in this review. These technologies provide different ways to boost recycling rates, boost productivity, and encourage sustainability in waste management operations. To completely comprehend the benefits, drawbacks, and ramifications of these technologies, it is crucial to assess and evaluate them.

The Clarke system by AMP robots automates recycling and rubbish sorting through the use of robots, artificial intelligence, and computer vision. It offers better sustainability, higher recycling rates, and greater efficiency. The ability of Clarke to efficiently sort vast amounts of rubbish, recover more recyclable materials, and decrease waste disposal in landfills are among of its key benefits. The impact on human jobs in the waste management industry as well as worries about data security and privacy are however possible drawbacks.

The Ramudroid v8 robot is used to pick up trash in public spaces. It runs without any human intervention and autonomously. The Ramudroid v8's benefits include its capacity to increase public health and safety, lessen environmental effects, and improve waste collection efficiency. Its widespread use, however, may be hampered by restrictions like tiny waste bin capacity, technical problems, and initial expenses.

The Atmega 328/Arduino-powered garbage sorting device provides an affordable method for automating waste sorting. Effective sorting, little maintenance needs, and user-friendly management are all provided. However, when developing this system, it is important to take into account constraints such restricted functionality, dependability worries, compatibility problems, and technological complexity.

The robotic arm used to sort metal and nonmetal products is fast, accurate, economical, and consistent. However, when evaluating this technology, it is important to carefully assess the initial cost, installation difficulty, limited capabilities, and dependability difficulties if it were to be brought into the industry.

The garbage sorting conveyor belt uses metal and colour sensors. The detection and classification of objects according to their colour and metal composition is made possible by an Arduino automation project. It has capabilities for conveyor belts, robotic manipulation, metal detection, and precise colour sensing. However, the system's shortcomings, such as the lack of details regarding the precise use of IR sensors and potential compatibility problems, need to be solved.

Overall, by enhancing efficiency, accuracy, and sustainability, these robotic technologies and automation initiatives have the potential to revolutionize waste management procedures. They have benefits like higher recycling rates, less negative environmental effects, and better working conditions. The requirement for expertise in implementation and maintenance, as well as issues with data security and privacy, employment displacement, early costs, and technological difficulties should all be taken into consideration.

In summary, the technologies under consideration offer good methods for waste management automation and recycling improvement. To overcome their drawbacks and guarantee their successful application in actual waste management systems, a thorough analysis and evaluation are needed. These technologies can help create a waste management ecosystem that is more efficient and sustainable with continuing study, development, and cooperation.

Research Gaps and Future Directions

Research Gaps

Impact evaluation: Additional study is required to fully evaluate the environmental, financial, and social effects of incorporating robotic technologies into garbage management. This involves assessing the decrease in waste production, energy use, and carbon emissions, as well as any potential employment losses and socioeconomic effects.

Performance and dependability over the long term: Many of the assessed technologies are still in the planning and development stages. To assess their long-term effectiveness, dependability, and durability in actual waste management settings, more investigation is required. Assessing the need for maintenance, system downtime, and potential difficulties that can emerge after prolonged periods of operation are all part of this process.

Future Directions

Advanced AI and machine learning algorithms development: Ongoing research and development of AI and machine learning algorithms can improve the efficiency of robotic waste management systems. This entails increasing the object recognition and sorting process' speed and accuracy as well as offering adaptive and self-learning capabilities to manage various waste streams.

Integration of sensor technologies: The effectiveness and efficiency of robotic systems for sorting can be improved by integrating modern sensor technologies such as multispectral imaging, hyperspectral sensing, and enhanced material identification techniques. In order to enable more accurate material identification and separation, future research should examine the possibility of incorporating these sensors into robotic waste management systems.

Chapter 3: System Modelling and Architectural Design

We will go over the design factors and selection process for our suggested litter removal system in this chapter. We'll examine numerous design concepts, weigh their benefits and drawbacks, and then finally justify our final design decision. We will also discuss significant issues and factors pertaining to the application of the selected design.

System Modelling

Considerations for system modelling

The machine will operate in a landfill site as that were our system make the most impact in terms of tackling the problem. Our machine will be a conveyor belt with a robotic arm. The robotic arm will need to work as efficiently as possible to ensure that the litter does get sorted and that our system is able to operate smoothly. To do this our machine will need to incorporate motion control theory and kinematics into it. The motos will have to be adequate for the conveyor belt so that it has enough strength to run the conveyor and that it does not run the system at top speed so that the robotic arm is not able to operate smoothly.

We need to ensure that the robotic arm has enough control of the speed so it does not operate at a high-speed output that it does grab the litter properly, this will also help the robotic arm from breaking down and that the components do not break. The robotic arm needs to calculate at what angle it should position itself so that it is able to pick up the litter on the conveyor belt to do this we will need the ESP 32 camera to locate the litter so that when the robotic arm is at the right angel it is able to close on the litter.

We need to ensure that the axis of the robotic arm is positioned correctly otherwise when the wrist goes to pick up the litter the robotic arm will miss the object and not function as it is intended. To ensure this we will need to look at the base of the arm and make sure that the base is sturdy enough to hold the rest of the arm, the base axis should be positioned at the right angle so that it does not cause too much constraint on the rest of the arm.

If the base is too far apart from the object and is not positioned at the right distance, we will find that the rest of the arm will have to extend further and at a quicker pace in order to reach the object this will then cause for the arm to experience a faster wear of components than if the robotic arms base and the rest of the arms axis are positioned at just the right angle to reach the object and perform its intended function. Our robotic arm will be positioned at the frame of the conveyor belt to make it so that the arm is much closer to the litter that it is supposed to pick up. If the conveyor belts speed is not controlled and it is running at a high velocity speed that will mean that the speed of the robotic arms will need to be equal to the velocity speed that the conveyor belt is running at, this will only cause harm to the system and will complicate the system even more as then we will need to ensure that even though the conveyor belt is running at a high velocity and the robotic arm is also it is still able to perform its function of picking up the litter. We might need to consider making use of the potentiometer on the robotic arm to ensure that we are able to increase or decrease the speed at which the robotic arm operates at.

How the system functions

- Camera Input: The robotic arm is equipped with a camera. The camera captures images or video of the surrounding environment.
- Image Processing: The captured images or video frames are processed using computer vision such as the open-source platform OpenCV. The image processing algorithms analyse visual data to identify and locate objects within the camera’s view.
- Object Detection: The processed images are analysed to detect objects of interest. The goal is to accurately identify and locate objects that the robotic arm needs to interact with.
- Object Localization: Once the objects are detected, the camera provides information about their position and orientation relative to the robotic arm.

- Trajectory Planning: The robotic arm uses the detected object's position and other relevant information to plan a trajectory for reaching and picking up the object.
- Arm Control: Based on the planned trajectory, the robotic arm's control system generates commands to actuate the individual motors or joints.
- Object Manipulation: The robotic arm moves its joints according to the control commands, following the planned trajectory. It reaches the object, adjusts its gripper to securely pick up the object.
- Object Transport or Task Execution: Once the object is secured, the robotic arm can move and transport it to a desired location.

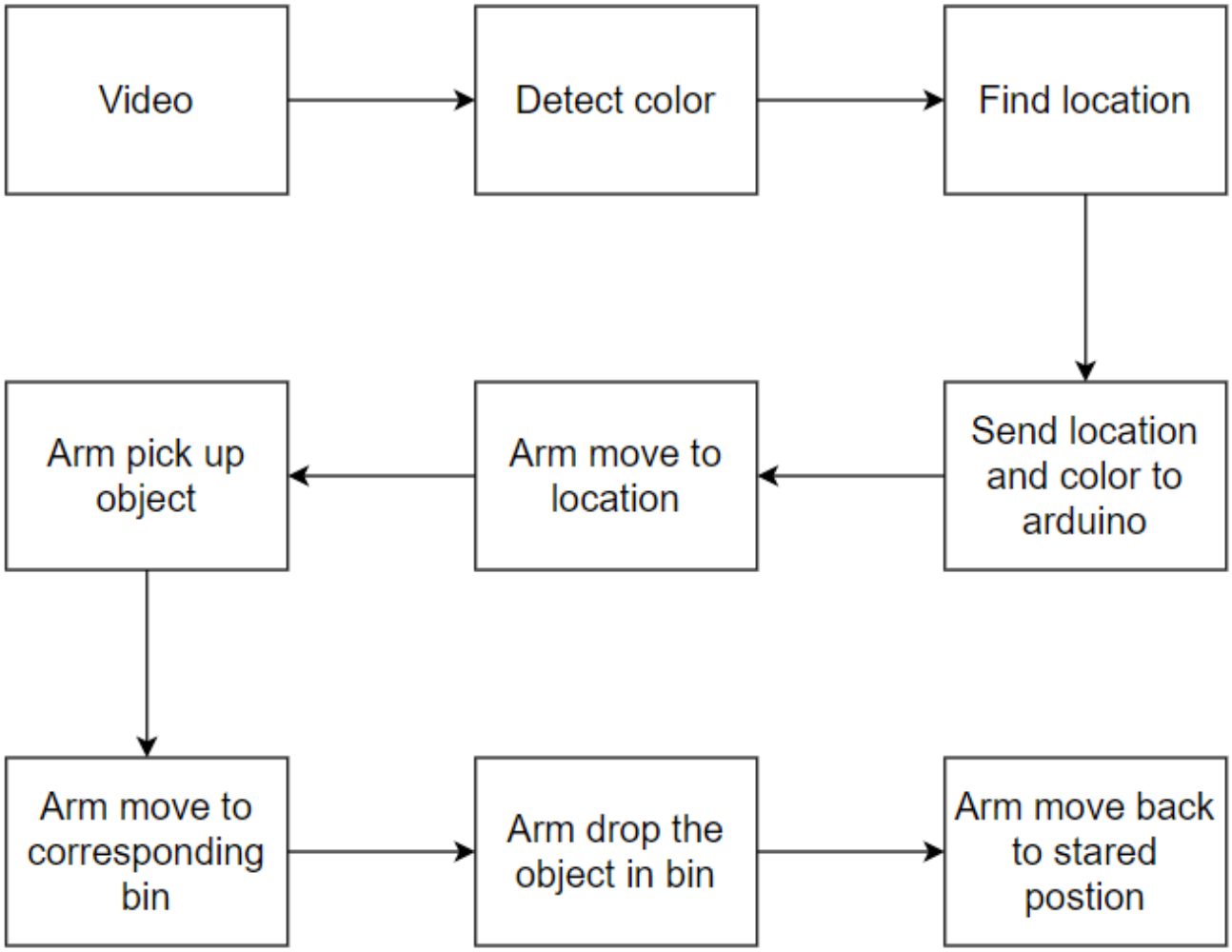


Image 8: Flow or litter sorting

Computer vision

Computer vision refers to the field of study and application of techniques that enable computers to gain a high-level understanding of visual information from images or videos. It involves extracting meaningful information and making interpretations from visual data, like how humans perceive and understand the visual world. We will be looking at the different topics of computer to be used in the prototype so that effective litter sorting is achieved. These techniques include object detection, colour detection, and object localisation.

Object Detection

Object detection is a technique used in computer vision for locating instances of objects in images or videos. Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results. Before looking into object detection, we have studied multiple machine learning algorithms to be considered seen in Chapter 2: Literature Review. For object detection the VGG-16 model will be used due to its relative simplicity against other models looked into.

Colour detection

In robotics to visualize the environment, alongside object detection, is the detection of the object colour in real time extremely important. For this prototype we also make use of colours as a representation of different materials that could be spotted on the conveyor belt. These materials could include paper, plastic, metal, and glass.

The colour detection works by converting the video frames from RGB into HSV colour space. We use HSV colour space because RGB colour space is co-related to the colour of luminance therefore colour information cannot be separated. HSV is used to separate the luminance from the colour information therefore the colour information can be separated (pintusaini, 2023). To detect the different colour the HSV colour range values must be specify. For example, HSV value for red is (0%,100%,100%) (RapidTables, 2023).

Image 9: Colour Detection

Object Classification

Addition to colour detection, this study also touches on object classification using the deep learning CNN algorithms to classify objects into different categories. In the development of the protype, this model is used to classify objects by material types such as paper, cardboard, metal, glass, and non-recyclable items.

Object detection and localization

This refers to the indentification of one or more objects in an image and drawing a bounding box around their extent. Object detection comes the identification and the drawing of the bounding box to determine where the object is located in space. In this study, object localisation is used to determine where objects are on the conveyor so that it can be picked up and places in designated containers.

The image below shows the result of using OpenCV for object localization

Expectations

The Main goal is to build Robotic Arm that sort Litter to its respective areas. The systems will pick and drop the litter in a category that it should be in. This can be monitored through certain waring messages and basic manual overrides and controlling speed. However, is it an automated system that does not need human help.

After we have fulfilled our main goal and implemented the features to it, our group will focus on achieving in the completion of our main goal and adding the features that we want to the system to further enhance the system to be as optimal as it can to tackle the problem, we would like for the robotic arm and the conveyor belt to work as one system. We would also like to add a web module if we have the time to send live data to a mobile app that we would like to make to keep track of progress regarding how much plastic has been sorted out.

Limitations:

The limitation of this prototype is that we will design and build both the robotic arm and the conveyor belt, but they will not be joined as a single system at the start we will work to see if we can interjoin both to be a single system so that the robotic arm and the conveyor belt system will work as one system.

Training our dataset, for this prototype we will use coloured cubes so that we are able to make use of a sensor that detects colour on the conveyor belt. Once we manage this, we will move on to train our prototype to be able to detect the different types of litter that is on the conveyor belt.

Experience, as we are students, we are sure that we can complete this project, but our skills will be put to a test in regards if we are able to join the two systems into one. We would like to join the robotic arm and the conveyor belt system to work as one.

Architectural Design

To create a system that removes litter effectively, we investigated a number of design concepts. Each concept was assessed according to its technical complexity, cost, influence on the trash problem, and viability. This section goes over the designs proposed, their advantages and disadvantages as well as materials used for the chosen design.

Proposed Designs

Design 1: Rover tank

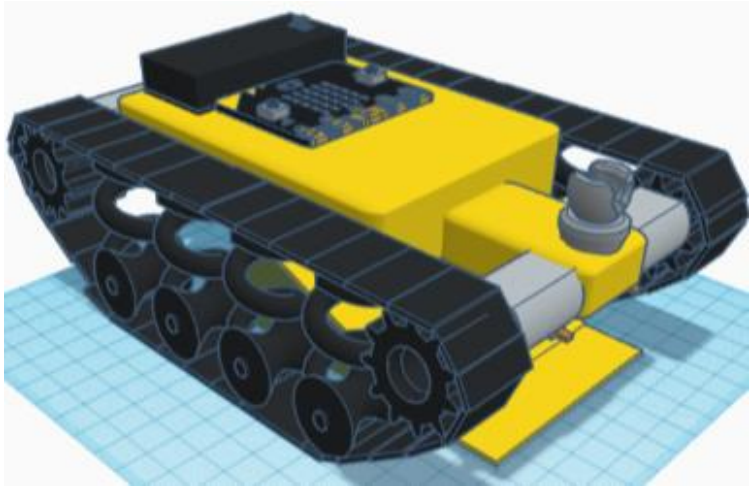


Image 10: Rover tank

For this design we thought of having a rover with tank track to roam around the landfill and sorting out the litter this would be ideal as the tracks would help with overcoming different obstacles but we choose to go against this idea as in the landfill there is a lot of trash and to make an impact on the problem you would need a swarm of these rovers which would increase the cost and make little impact to the problem.

Design 2: Rover

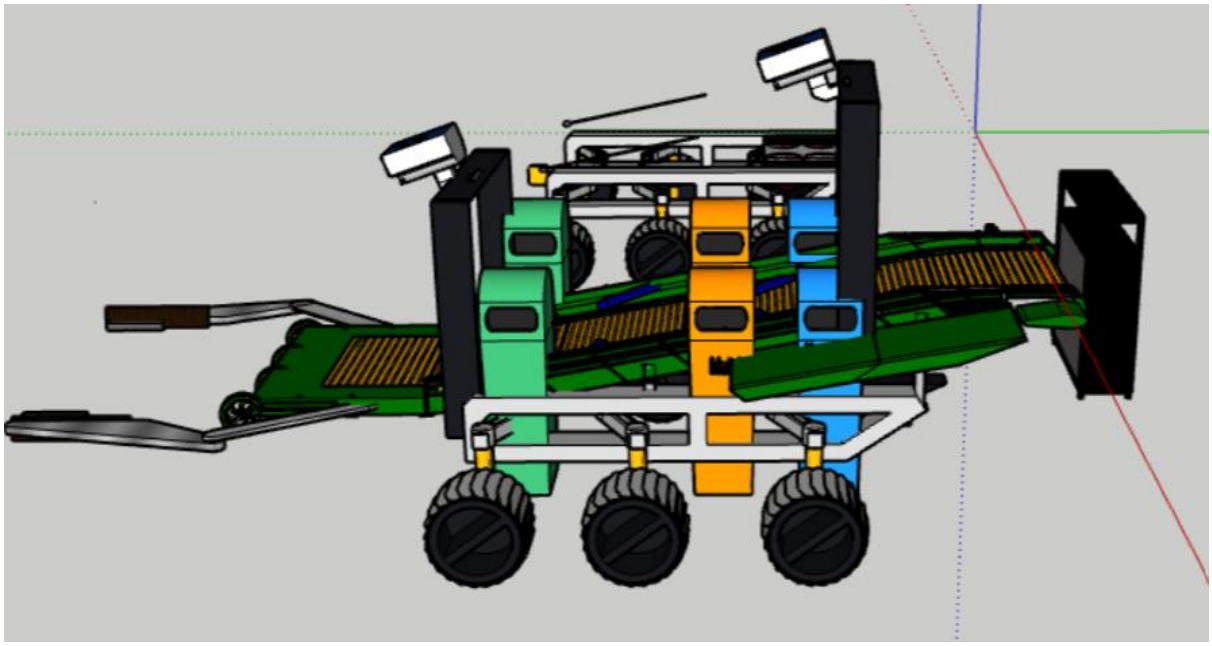


Image 11: Rover side-view

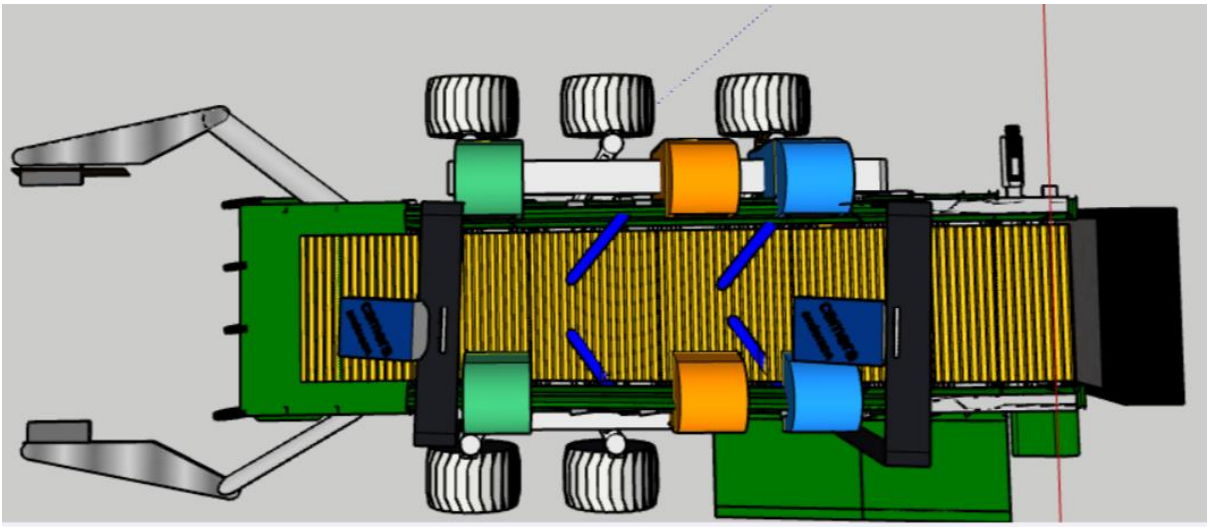


Image 12: Rover top-view

Another rover such as the one above has been considered. This would have been ideal as it would contain a camera that is able to recognize obstacles and what is classified as litter, the clippers would have small brushes and pick up the litter with its clippers. The litter would then be loaded onto a conveyor belt which would pass through a section with a camera on tip to recognize the type of litter and it would move the brushes on the side to the container it needs to go to. This would have been an ideal project as it makes use of various of difficult components but we were advised against this idea, we reviewed the idea again and we choose tackle the project again from scratch as this design was designed for road use and would not have made an impact on litter as an overall problem, it would have reduced the litter on the streets but not on the main problem which is located on landfills. Another problem with this was that we are still students, and our skills may not have been sufficient to overcome this obstacle.

Design 3: Double Conveyor Belt and Robotic Arm

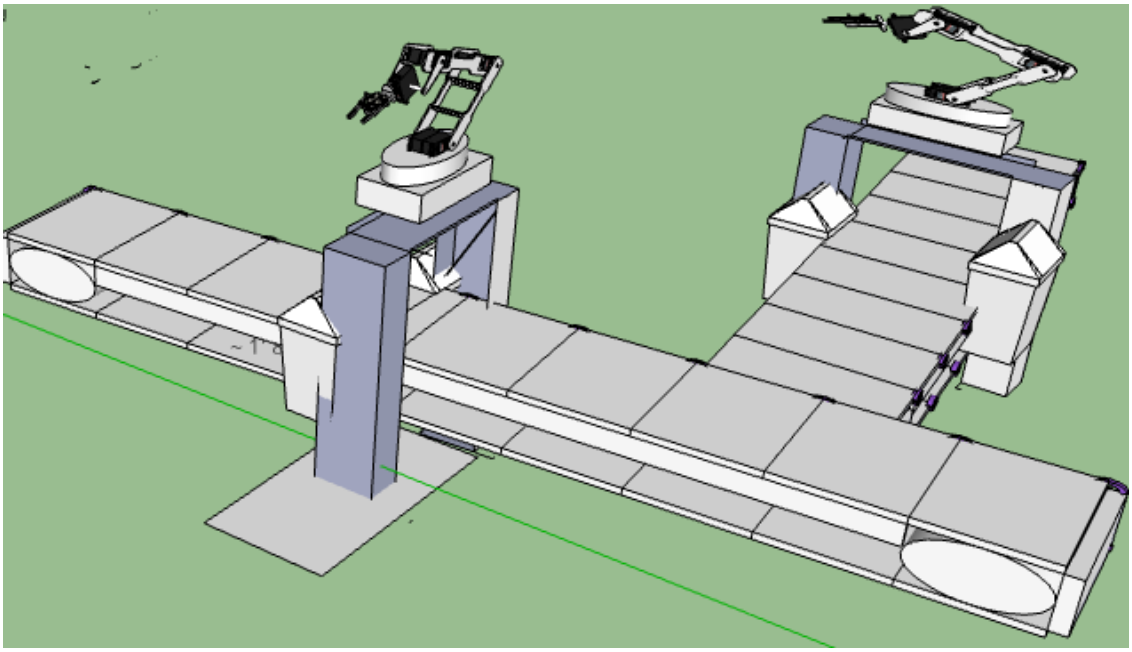


Image 13: Double Conveyor Belt and Robotic Arms

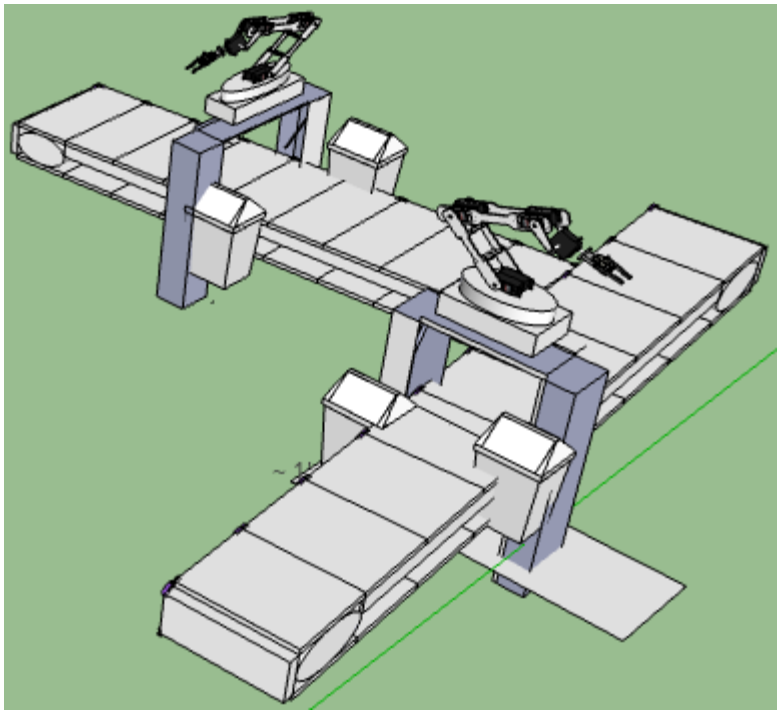


Image 14: Double Conveyor Belt and Robotic Arms

For this design we have designed 2 conveyor belt systems with 2 robotic arms this, the reason we choose to design this system is so that the litter sorting system would be sped up with 2 systems working as one system, we could even divide the system so that the one system is in charge of only sorting through glass and paper and the other system is responsible for sorting through metal and plastic. This was a great idea as this system is complex and would challenge our skills and would motivate us to complete this complex system but due to budget reasons, we have decided to cut the idea to one conveyor belt with one robotic arm that would sort through the litter.

Final Design 4: Conveyor belt and Robotic Arm

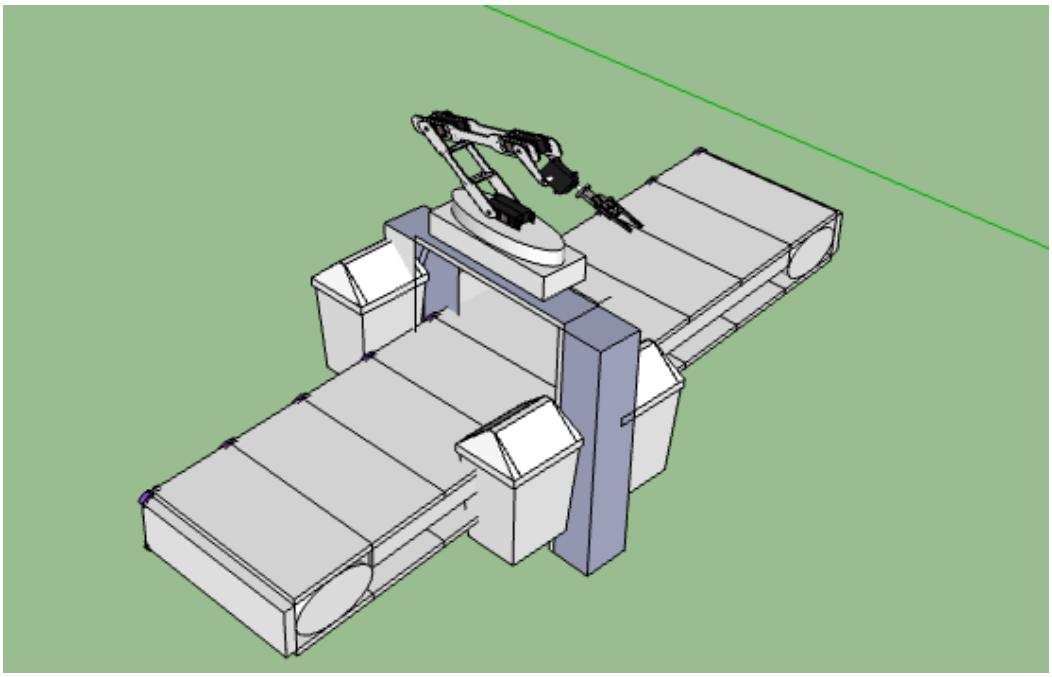


Image 15: Design 4

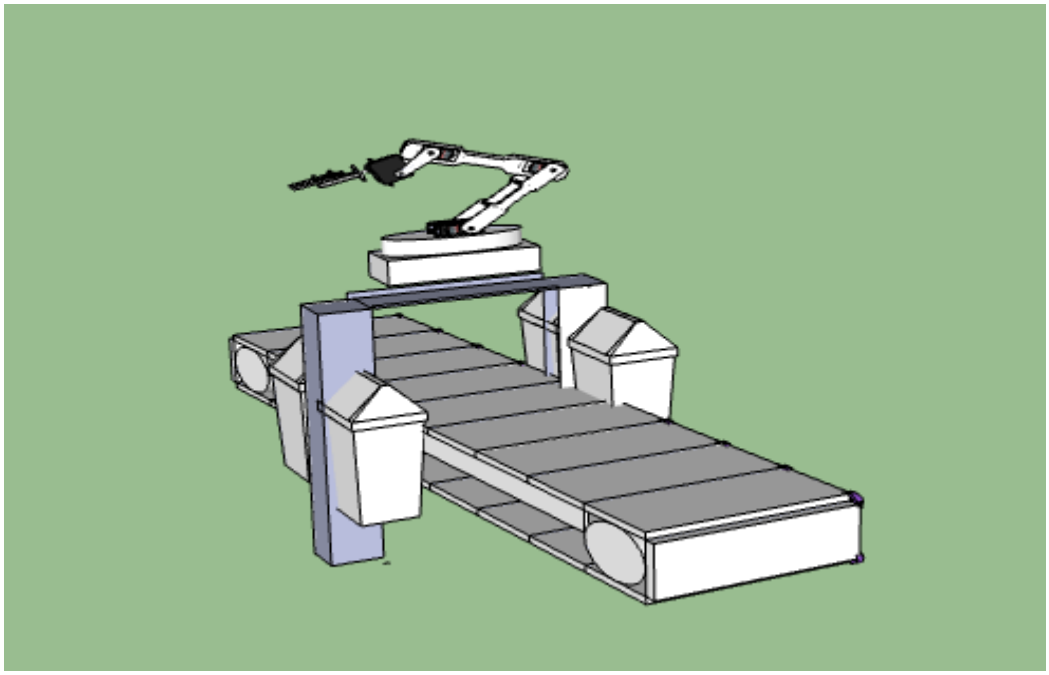


Image 16: Design 4

We have a prototype design which is still in phase of creation where the robotic arm should be tested in various collections settings to ensure its effectiveness and efficiency in sorting waste. We will need to test the robotic arm to ensure that it is working as it should and that it is built of light and yet durable material. To ensure that the robotic arm is as effective as it should be we will need to test to see that the end effectors of the robotic arm are able to grab the different size of items and hold it in place.

We have chosen to go with this design for our final design as it makes use of one conveyor belt system and one robotic arm system which is feasible in terms of skills, time, and budget. We have decided to control the conveyor belt and join the robotic arm and conveyor belt to one system we will attempt to make the robotic arm send a signal to the conveyor belt. When the camera detects the litter, it will send a signal to the conveyor belt to stop, and the camera will send a signal to the robotic arm to sort it according to its category. Once the robotic arm has sorted the litter it needs to send feedback to resume the conveyor belt.

For this we aim to refine our system by researching the best available systems for the problem, designing a system for the problem, and checking which is the best approach. Where a detailed component list is given and why those required parts are best for our system which will contain the budget amount.

From that we build on to a protype and go through stages of achieving our vision by testing and trailing on different stages of building and coding.

We will have set of trained litter items such as a Can, plastic bottle, paper ball and a glass item which should go through a system that allows for the splitting of litter.

This is then placed onto our conveyor. Where is moves at a manageable speed for the two robotic arms that are placed to identify the different types of litter using the cameras and sensors to allow the conveyor blet to stop or slow down and allow the robotic arms to pick up the items and place it in its categories as explained above.

System Components:

The following are the fundamental components used in the operation of the litter sorting system. Each component will be discussed in detail

- Robotic Arm: A mechanical arm with multiple joints and a gripper that can be controlled for precise movements.
- Conveyor Belt: A motorized belt for transporting objects.
- Arduino Board: The control unit that provides power and processes commands for the robotic arm and conveyor belt.
- Power Supply: Supplies the necessary electrical power to the system.
- Raspberry Pi/Computer: Used to run the Python colour detection code.

Robotic Arm Design

Diagram 4: Robotic Arm Design

This is the design we have chosen to go with for the robotic arm. It would be comprised of a 6-servo motor system which contains a base servo motor, a servo motor to control the motion axis of the base of the robot to rotate around.

The shoulder of the robotic arm with another motor which would allow for movement of the shoulder. The elbow and the wrist of the robot both contain motors to allow for the extension of the arm and movement around.

The end effectors are made in a way that looks like clippers; this will allow to close the clippers with a motor so that the arm is able to grab the litter object.

We are still working on the clippers and we are still looking to redesign the clippers we might add another clipper to allow for a sturdier grip and we are looking to add finger like ridges on the side of the clippers or add a material that allows for more friction when picking up object, this is to make it so that the clippers have some surface to cause friction on the object so that it is able to grip more. If we had a smooth surface on the clippers, then the surface friction between the clippers and the object would not be much and the robotic arm would find it difficult to pick up the object.

Each servo is responsible for a specific action used in picking up the object and moving it to the corresponding container.

- Motor 1 (Base): This motor controls the rotation of the base of the robotic arm. It allows 180 degrees of rotation, which means it can rotate the arm in a semicircular motion.
- Motor 2 (Shoulder): The shoulder motor is responsible for controlling the elevation of the arm. It allows the arm to move up and down, changing the angle between the arm and the base.
- Motor 3 (Elbow): The elbow motor controls the bending of the arm at the elbow joint. It enables the arm to articulate and extend, changing the angle between the lower and upper arm segments.
- Motor 4 (Wrist Rotation): This motor enables the rotation of the wrist joint. It allows the end effector to rotate around its axis, providing additional flexibility in manipulating objects.
- Motor 5 (Wrist Tilt): The wrist extension motor controls the extension or contraction of the wrist joint. It allows the end effector to move closer or farther from the arm, changing the distance between the end effector and the elbow joint.
- Motor 6 (Claw): The claw motor controls the gripping mechanism of the end effector. It enables the arm to open or close its claws to grip or release objects.

3D Printed Parts

One of the main objectives for creating the litter sorting system is affordability. Thus, all 3D parts have been printed. All parts are 3D-printed and are shown below

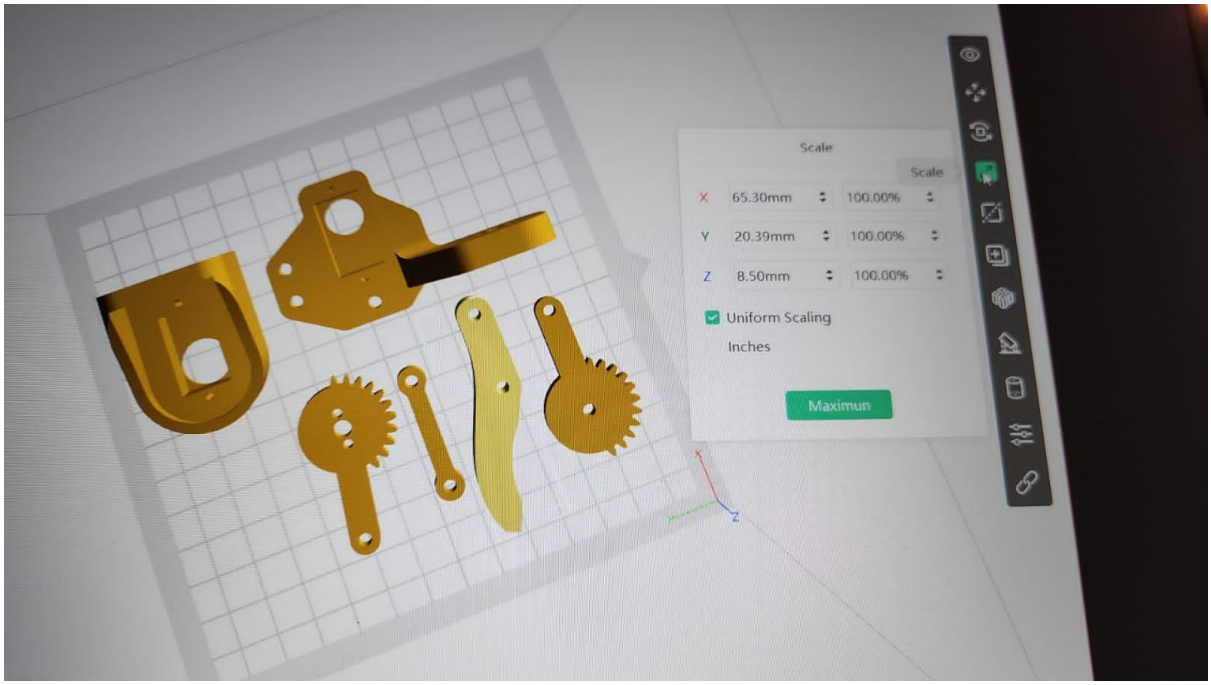


Image 17: 3D Printed Parts

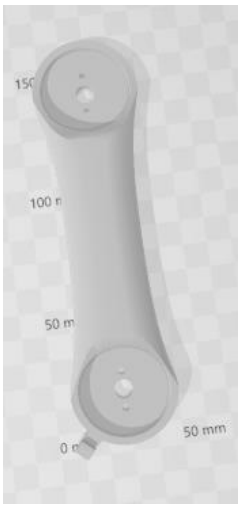


Image 18: Shoulder

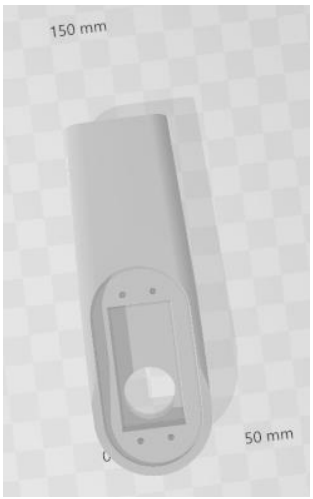


Image 19: Elbow

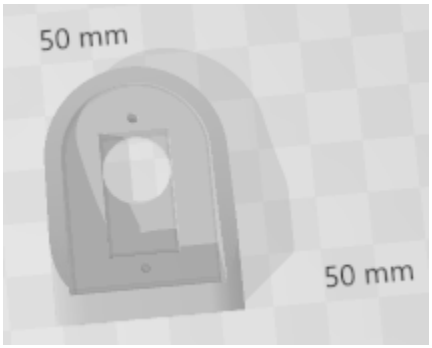


Image 20: Wrist

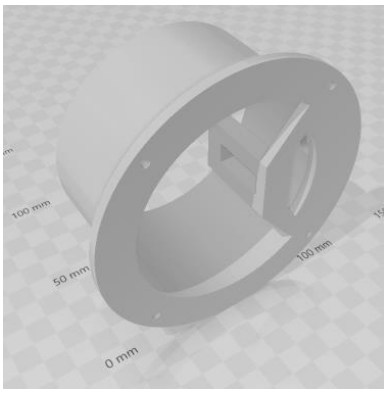


Image 21: Base

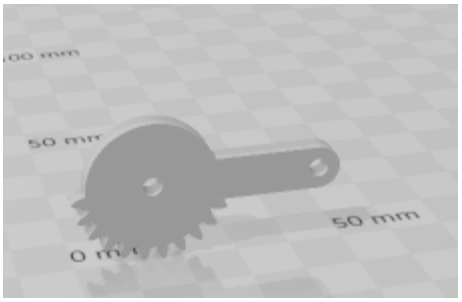


Image 22: Gripper gear 1

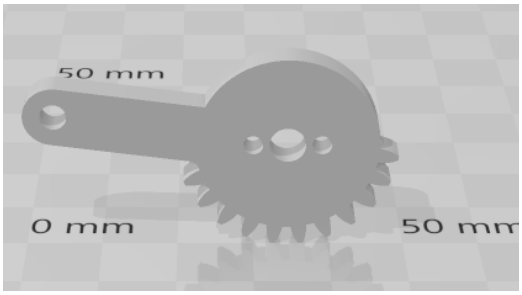


Image 23: Gripper gear 2

Image 24: Gripper gear link

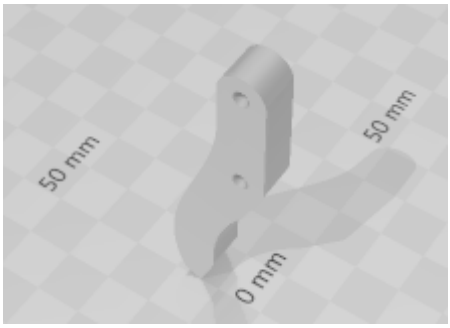


Image 25: Gripper end-effector

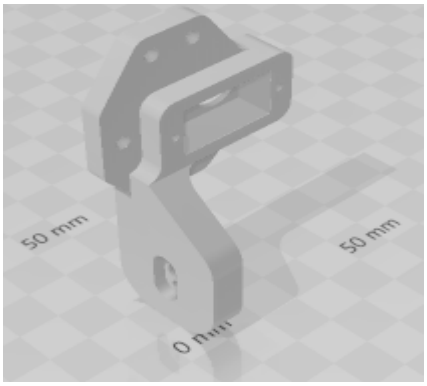


Image 26: Gripper base

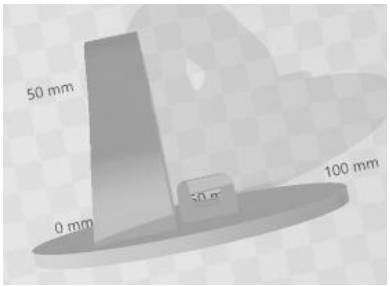


Image 27: Waist

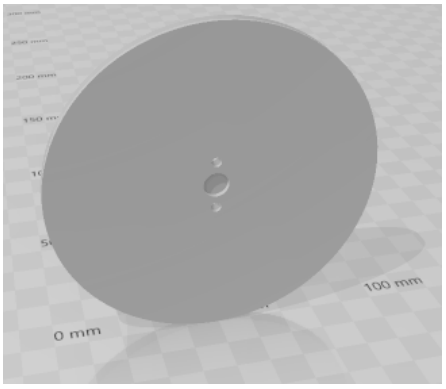


Image 28: Base plate

Image 29: Completed 3D printed parts for robotic arm

Schematic

Below is the schematic for the arm. From the schematic we have 7 servo motors, the signals from the 7 servo motors connect to the Arduino uno pins, the positive and negative of the 7 servo motors all connect to the positive and negative on the breadboard reels. The servo motors will be used for the following components in the robotic arm, the base, the shoulder, the elbow, wrist1, wrist 2, gripper and conveyor belt.

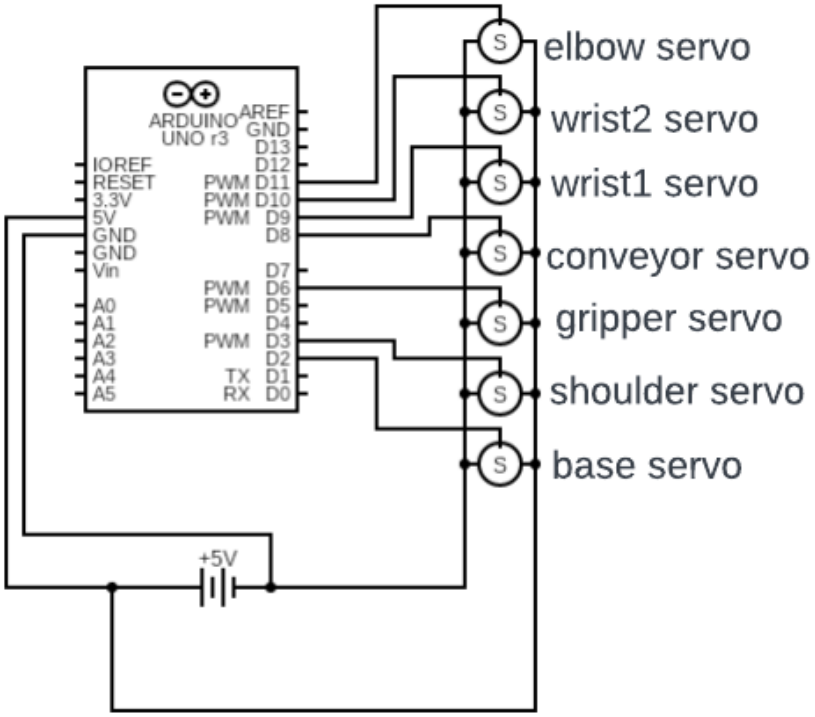
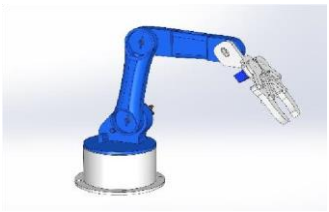


Diagram 5: Robotic Arm schematic

Our Design vs. other designs.

		
Uses 6 motors to get more range of motion and allows more points to be accurately captured	This arm uses hydraulics and a magnet instead of a claw and cannot rotate on its own base. The hydraulics are more suited towards more strength required objects.	This example makes use of motors on a fixed base with three claws to pick up objects. This arm has a limited set of degree movement.
Best suited to pick up objects if the objects are smaller than the claws width.	This arm is best suited to sorting metal. Mental sensors have a very limited range to the point where it needs to be less than 10m to detect metal objects.	With this arm objects of a smaller size are better suited to be picked up

Robotic Arm Kinematics

This System is using kinematics known as geometric kinematics or kinematics based on servo control. This sort of kinematics is commonly utilized in specialist automated arm ventures where the joint points are controlled straightforwardly utilizing servo engines.

In geometric kinematics, the kinematics of the robot arm are determined based on the physical measurements and geometry of the arm. The servo engines incite the joints of the arm, and by controlling the points of the servos, you'll be able decide the position and introduction of the conclusion effector.

The key components of the kinematics in your code incorporate:

Forward Kinematics:
The move Servos () from Arduino code work calculates and controls the points of the servo engines based on the required conclusion effector position. It uses incremental changes in servo points to move the arm to the required setup.

Converse Kinematics:
Although not unequivocally actualized the Arduino code, there's a verifiable shape of inverse kinematics within the execute Action () work. It maps the required activities (protest drop positions) to the joint points required to realize those positions.

Denavit-Hartenberg Parameters:
Arduino code does not unequivocally utilize Denavit-Hartenberg parameters since it expects disentangled kinematics, but it in a roundabout way characterizes the joint points and positions (DH parameters) through the clusters HOME_POSITION, PICKUP_POSITION, and DROP_POSITIONS. These clusters speak to the joint points for different arm arrangements.

DH Parameters Matrix

We will look at the DH (Denavit-Hartenberg) parameters of the matrices that make up our system in this part. We shall pay special attention to the matrices connected to the conveyor belt and robotic arm. We will examine the parameters that define the interfaces and joint points.

The following elements make up the DH parameters:

Link lengths (a): These variables show how long the links are that connect joints. All link lengths in the provided example (a1, a2, a3, a4, a5, a6) are set to zero, indicating that all joints are in the same location.

Connect turns (α): The rotation around the common normal between the old and new joint axes is represented by the twist angle (). The alignment of the joint axes and the absence of any rotational offset are indicated by the fact that all link twists (α1, α2, α3, α4, α5, α6) in this instance are set to zero.

Interface Offsets (d): along the shared normal between the previous and current joint axes are represented by the offset parameter d. The joints are perfectly aligned along the z-axis in this case since all interface offsets (d1, d2, d3, d4, d5, d6) are set to zero.

Joint angles (θ): These parameters speak to the revolution points connected to each joint. θ1, θ2, θ3, θ4, and θ5 compared to the points of the bear, elbow, wrist1, wrist2, and hand servos, separately. θ6 is settled at zero, demonstrating that the final joint does not contribute any extra revolution.

The change lattices T_01, T_12, T_23, T_34, T_45, T_56, and T_6e portray the homogeneous changes between successive joint outlines. Each change network comprises of a 4x4 network speaking to the turn and interpretation between the outlines.

For case, T_01 speaks to the change network from the base outline to the primary joint outline. It combines the turn cosθ1, sinθ1 and the interpretation along the z-axis (0) to speak to the change. So also, T_12, T_23, T_34, T_45, T_56, and T_6e speak to the changes between consequent joint outlines.

To get the in general change network T_0e from the base outline to the conclusion effector outline, you increase the person change lattices T_01, T_12, T_23, T_34, T_45, T_56, and T_6e within the indicated arrange.

Transformation matrix for each joint and length

Link Lengths (a):	**Link Twists (α) – Around z-axis:**
$a1 = 0(base\ to\ shoulder)$ $a2 = 0\ (shoulder\ to\ elbow)$ $a3 = 0\ (elbow\ to\ wrist1)$ $a4 = 0\ (wrist1\ to\ wrist2)$ $a5 = 0\ (wrist2\ to\ gripper)$ $a6 = 0\ (from\ gripper\ to\ end\ effector$	$a1 = 0$ $a2 = 0$ $a3 = 0$ $a4 = 0$ $a5 = 0$ $a6 = 0$
Link offsets (d) – Along z-axis: $d1 = 0$ $d2 = 0$ $d3 = 0$ $d4 = 0$ $d5 = 0$ $d6 = 0$	Joint angles (θ) – Along z-axis: $\theta1 = shoulder\ angle$ $\theta2 = elbow\ angle$ $\theta3 = wrist1\ angle$ $\theta4 = wrist2\ angle$ $\theta5 = gripper\ angle$ $\theta6 = 0$

To calculate the overall transformation matrix from the base frame to the end effector frame, you would multiply the individual transformation matrices for each joint.

$$T_{01} = \begin{bmatrix} \cos\theta1 & -\sin\theta1 & 0 & 0 \\ \sin\theta1 & \cos\theta1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{12} = \begin{bmatrix} \cos\theta2 & -\sin\theta2 & 0 & 0 \\ \sin\theta2 & \cos\theta2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{23} = \begin{bmatrix} \cos\theta3 & -\sin\theta3 & 0 & 0 \\ \sin\theta3 & \cos\theta3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{34} = \begin{bmatrix} \cos\theta4 & -\sin\theta4 & 0 & 0 \\ \sin\theta4 & \cos\theta4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{45} = \begin{bmatrix} \cos\theta5 & -\sin\theta5 & 0 & 0 \\ \sin\theta5 & \cos\theta5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{56} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{6e} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{0e} = T_{01} \cdot T_{12} \cdot T_{23} \cdot T_{34} \cdot T_{45} \cdot T_{56} \cdot T_{6e}$$

Kinematics for inverse (Raspberry pi)

This is the inverse kinematics theory that was worked out for the raspberry pi, as we did not have the time to implement the system, we have implemented all the theory regarding it.

Inverse Kinematics

Base: The base of the arm moves in the x and y axis. The diagram below demonstrates the inverse kinematics for the base's rotation.

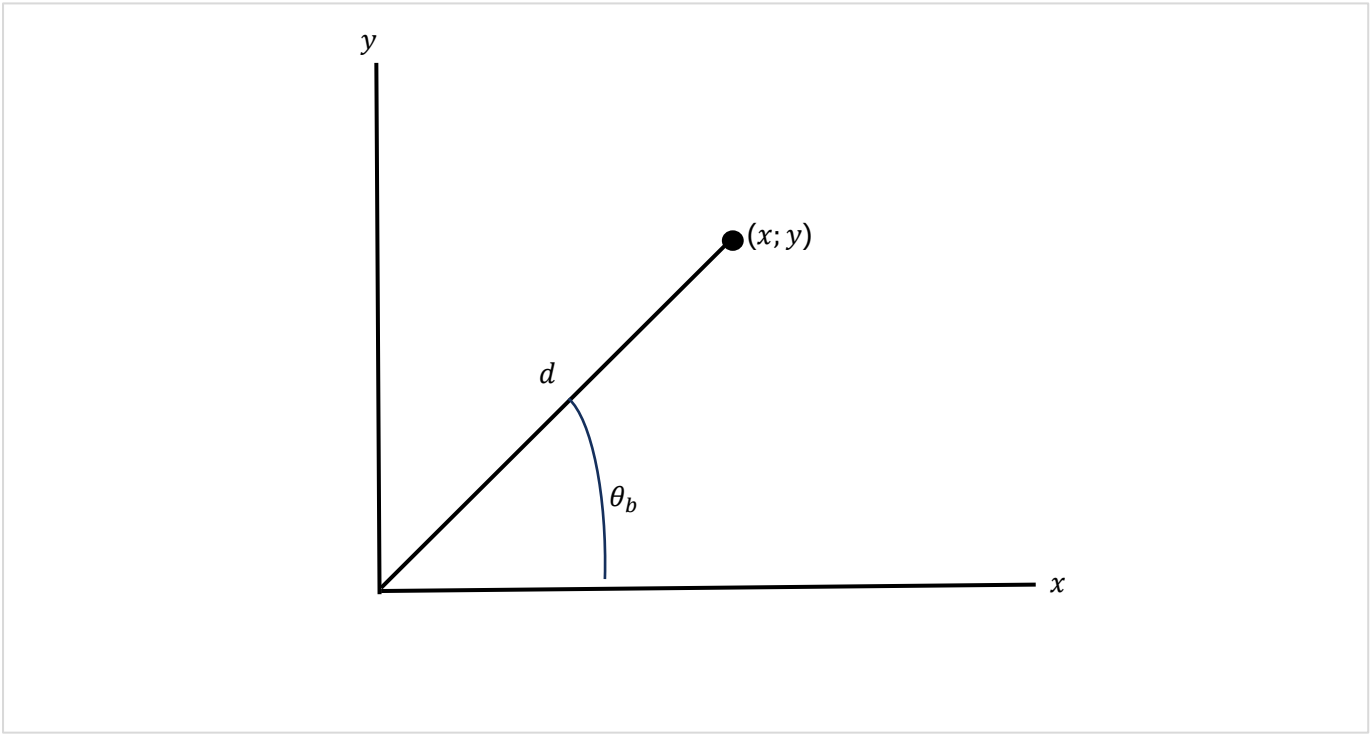


Diagram 6: Inverse kinematics of robotic arm base (rotation)

$$\tan(\theta_b) = \frac{y}{x}$$

$$\theta_b = \arctan\left(\frac{y}{x}\right)$$

$$d^2 = x^2 + y^2$$

$$d = \sqrt{x^2 + y^2}$$

Arm: The rest of the arm moves in the z and l_1 axis where l_1 is the distance between the arm and the center of the base.

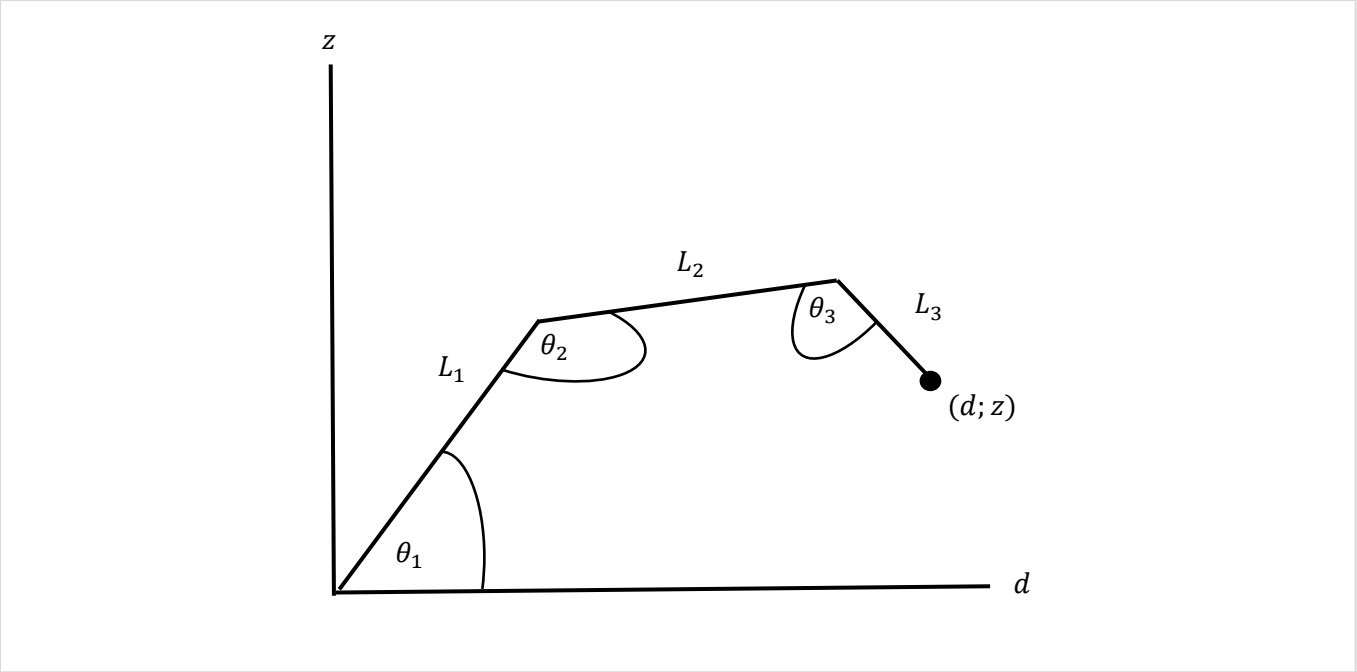


Diagram 7: Robotic arm inverse kinematics

The program used to calculate the kinematics can be found below with its output:

```
from numpy import *

# length of links in cm
l1 = 12
l2 = 11.25
l3 = 3

# prefer position of end effector
pd = -14
pz = 3

phi = 180
phi = deg2rad(phi)

# equation for inverse kinematics
wd = pd - l3*cos(phi)
wz = pz - l3*sin(phi)

delta = wd**2 + wz**2
c2 = (delta - l1**2 - l2**2)/(2*l1*l2)
s2 = sqrt(1-c2**2)
theta_2 = arctan2(s2, c2)

sl = ((l1+l2*c2)*wz-l2*s2*wd)/delta
cl = ((l1+l2*c2)*wd+l2*s2*wz)/delta
theta_1 = arctan2(sl, cl)
theta_3 = phi-theta_1-theta_2

print('theta_1: ', rad2deg(theta_1))
print('theta_2: ', rad2deg(theta_2))
print('theta_3: ', rad2deg(theta_3))
```

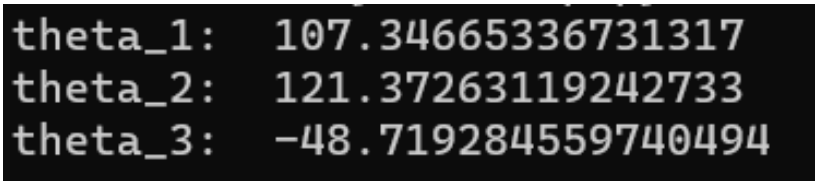


Image 30: Output of Arm Kinematics

Control Theory for Robotic Arm

Kinematic Modelling:

We created kinematics show of the arm. It relates the joint points or positions of the arm's servos to the position and introduction of the conclusion effector (gripper).

Diverse strategies can be utilized, such as the Denavit-Hartenberg tradition or the item of exponentials (POE) strategy, to determine the forward and converse kinematics conditions which we explained previously.

Control Goals:

We chose control destinations for the arm in the Arduino code as there is arrays for pick up, drop 1, 2, 3 and 4 position as well as home point (this is explained in more detail in Chapter4).

Controller Plan:

Based on the control destinations and the mechanical arm's kinematic show, plan a reasonable controller. Common control procedures for automated arms incorporate reverse kinematics control, PID control, or model-based control. The controller creates control signals for each servo to realize the required arm position or movement.

Actuator Control:

Each servo motor within the mechanical arm is controlled independently. It utilizes servo motor control methods such as position commands particular to the servo motor's communication convention (serial communication which we have used in the python and Arduino code).

Tuning:

For Fine-tuning the controller parameters where optimized to the arm's execution, stability, and response time.

Testing and Confirmation:

Testing the arm’s execution beneath diverse scenarios, such as position following, direction taking after, and protest control was done. Confirmation that the arm accomplishes the specified positions and movements precisely and reliably was shown as see in the project’s GitHub repository.

Control Theory for Conveyor Belt

Modelling:

This portrays the relationship between the input (control flag) and the yield (belt speed). It takes into consideration components such as motor characteristics, stack on the belt, grinding, and idleness.

The belt framework with a 12V DC motor, we are going consider an essential direct demonstrate that relates the Motor input voltage to the belt speed.

Motor Voltage:

Input voltage is denoted as V_{in} and the precise motor speed is denoted as ω_m. The relationship between the input voltage and motor speed can be approximated by a direct condition:

$$\omega_m = k_{motor} \cdot V_{in}$$

Here K_{motor} is the engine consistent, communicating with the motor's torque-speed characteristics.

Belt Speed:

The direct speed of the belt is denoted with V_b, where l represents the length of the belt. The belt speed is related to the motor speed through the belt circumference:

$$V_b = \omega_m \cdot R$$

Here R is the compelling span of the transport belt's pulley or drive wheel.

Framework:

Combining the motor and belt models, we are able express the relationship between the input voltage and belt speed:

$$V_b = k \cdot V_{in}$$

Here k speaks to the combined stability of the motor and conveyor belt framework, given by:

$$k = k_{motor} \cdot R$$

The constant k decides the proportionality between the input voltage and the coming about belt speed. The above ignores different variables like contact, stack varieties, and nonlinearities.

Control Destinations:

The control targets for the belt framework. This includes keeping up a consistent speed, accomplishing a particular increasing speed or deceleration, or reacting to changes in stack.

Risk Management

When building this system, we have taken risk management into consideration to avoid injuries as well as to troubleshoot the system a lot easier.

Due to the landfill sites being a dangerous place for humans because of the gasses that can explode, minimal human is implemented into the landfill sites but that reduces the sorting speed at the landfill sites. This system is designed to be autonomous to increase the sorting speed at the landfill sites and to minimize the humans that are located on the actual sites.

- The conveyor system is designed to run as soon as a colour is detected the system will then proceed to run for a certain amount of time once the colour is detected before turning off, this is a risk management that is taken in place so that the system is not running when there are no objects on the conveyor.
- Cable management, we have labelled the cables on the robotic arm to see which cable is for which servo of the robotic arm and bundled it up according to its group. This is done to troubleshoot a lot quicker if there is a connection error, we are easily able to identify the cable connection.
- The conveyor belt has a speed regulator to regulate the speed that the conveyor runs at. This is done to reduce the speed to a manageable one so that the object does not go flying at the robotic arm and break the components or to not strain the robotic arm when it needs to run at max speed.
- We have stepped down the voltage that the conveyor belt needs to run at to be able to test it at campus without tripping the db. board at campus every time, this is done so that we manage to not trip the campus DB board every single time we test the system.
- We have added a clear PCB box to hold the speed regulator unit as well as other boards with a fan on top to cool the boards as to not overheat the boards. The box is also created to prevent the boards from touching the metal frame that they rest o.
- We have put the robotic arm on top of a frame so that the claw does not rest near the conveyor in case the conveyor or the robotic arm comes across an error, and they do not react the way that they are supposed to, this will prevent objects from hitting and breaking the claw.
- The robotic arm is located near the middle considered of the distance to not be able to hit humans that are working in front of the conveyor belt.
- We will also mark the areas on the floor that people should not walk near to keep them safe in the event of an emergency occurs and the robotic arm malfunctions it does not injure anyone that is near the system.

Chapter 4: System Prototyping, Development and Testing

In this section we will look in more detail on how to develop the prototype and perform various tests such as durability under different circumstances to determine what needs to be improved on the system. We will also look at the development process of the system and an information that was obtained during the production of the system. All code and algorithm discussed in this section can be viewed on our GitHub repository here.

Components Used

In this section we explain, in detail, the numerous components used in the physical design of this system, the reasons for choosing them, as well as any industry considerations for component choice.

Power Supply

A 220V to 12V power supply is the power source used in the robotic trash sorting system. Its main job is to reduce the electrical input's high voltage (220V) to a lower voltage (12V) output that can power the system's components. It provides a 100-240V AC input voltage range and outputs a steady 12V DC with a 3A maximum current rating. The power supply guarantees effective power conversion with a 90% efficiency rating while providing overload and short circuit safety. A standard DC power cable with the proper connectors is used to integrate the power supply, allowing it to be connected to the control panel and other components. This power supply is crucial in ensuring the control panel and other system components have a dependable and stable power source.

Input	100-240v, 1.5A
Output	12v, 3A
Surge Protection	Yes
Plug type	IEC-C14, IEC-C8, IEC-C6
Efficiency	90%

For this project, the Sabre Security power supply was selected because of its performance, dependability, and safety features. It is compatible with a variety of power sources due to its broad input voltage range of 100-240V AC.A 90% efficiency rating on the power supply assures effective power conversion, reduces energy loss, and increases the amount of power available for the system's components. The system is

additionally shielded from any electrical faults by the power supply's overload and short circuit safety mechanisms, which also guarantee the durability of the components.

Real-world considerations

It is important to note that in real-world applications an Uninterruptible Power Supply would be more suited for the system, more specifically in South Africa due to frequent power cuts. To supply backup power in the event of power outages or fluctuations, UPS devices are frequently used. They often come with a battery that keeps the system running for a specific amount of time. In essential systems, UPS units guarantee uninterrupted functioning and avoid data loss. It has been seen that the effects of loadshedding result in power surges and electrical shorts damaging components (ECOFLOW, 2023). Component integrity aside, the frequent power cuts will all reduce efficiency. This makes a UPS, more viable if this system is introduced into the industry.

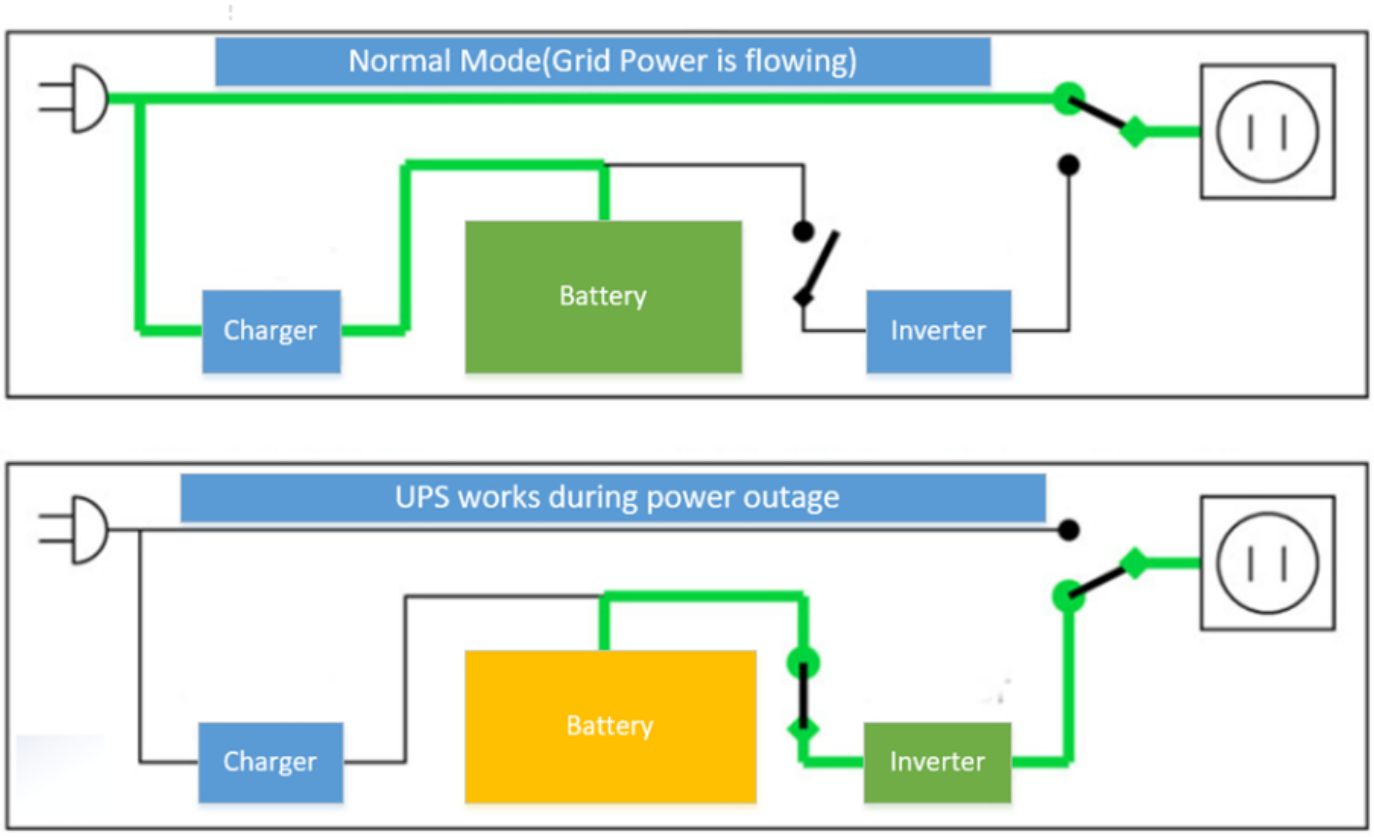


Diagram 8: Flow of power with a UPS during a power outage.

Computer vision

Image 31: USB Webcam

The trash collection system uses computer vision for object recognition, making it possible to efficiently sort various things on the conveyor belt. Currently, to take pictures of objects, a Raspberry Pi is connected to a USB webcam. Computer vision algorithms and analysis are carried out on the Raspberry Pi. This configuration enables the identification of various trash stream objects and real-time image analysis. This project makes use of “720P HD Digital Webcam Free Driver USB PC Web Cam Computer Camera”. This has been chosen for its simplicity and ease of use.

Head Rotation	45 degrees
Resolution	720P HD
White Balance and Colour Correction	Auto
Focal Length	Adjustable
Supports	Windows 2000 / XP / window7 / win8 / window10 / Vista 32-bit etc. Support CC2000, web conferencing, ICQ, MSN, Yahoo, Messenger and Skype

The Raspberry Pi can receive image data from the USB webcam by connecting it to the Raspberry Pi via a USB connection. Using Python libraries like OpenCV, the Raspberry Pi executes computer vision algorithms and analysis, enabling the ability to identify objects, visual analysis, and make decisions for the trash sorting system. The Raspberry Pi processes the recorded photos, analysing them to recognize different objects on the conveyor belt, classify them, and adjust the sorting mechanism appropriately. The trash collection system's present configuration, which uses a USB webcam and a Raspberry Pi, finds a mix between affordability and functionality, offering adequate object identification performance and ease of integration.

Alternatives: ESP32-CAM

Image 32: ESP32 CAM

Other considerations for camera vision for this project include the ESP32-CAM from Leobot. The ESP32-CAM is a Wi-Fi and Bluetooth enabled camera module development board. It is an affordable and low-power solution for a number of uses, such as Internet of Things initiatives, home automation, and remote monitoring. A camera module built into the ESP32-CAM may capture still images or video, and the data can be saved on the board or sent to an attached device through Wi-Fi or Bluetooth. The board also features an integrated microphone and image and video compression capability. The ESP32-CAM may be used with a variety of development environments, including Arduino, MicroPython, and Espressif's own (Leobot Electronics, 2019). We recommend this camera module for the project if no Raspberry Pi is used, as it may be used with an Arduino.

Wi-Fi Protocols	802.11b/g/n
Resolution	2 Megapixel
Storage	Built-in: 520KB SRAM External: 4M PSRAM
Supported Interfaces	UART/SPI/I2C/PWM/ADC/DAC

Dimensions	40.5 × 27 × 4.5 mm

Industry Alternatives

Industry alternatives for computer vision include specialized smart cameras with integrated processing and power supply modules. These cameras provide benefits like better image analysis algorithms, higher resolution, and quicker processing times. For reliable and effective object detection, several industrial applications also use specialized image recognition systems with GPUs or dedicated AI accelerators. The selection of a computer vision solution is influenced by various elements, including system requirements, resources that are available, and economic concerns.

Robotic Arm Frame

Image 33: Robotic Arm Frame

Steel is used in the construction of the frame that holds the robotic arm above the conveyor belt and is known for its lightweight yet durable qualities. Because the frame has a welded structure, stability and toughness are guaranteed throughout the sorting procedure. This is 60cm by 39cm frame with a centre square plate 15cm by 15cm, this is used to mount the robotic on as well as the Esp32 camera underneath the square plate for object detection. The frame has a simple design and provide the arm plenty of room to move around and efficiently sort the things.

Key components of the frame design are compatibility and integration. It has mounting holes that connect easily to the robotic arm. By reducing any unwelcome vibrations or movement during operation, this guarantees a secure and solid connection.

Industry Considerations: Safety and Adjustable Brackets

Several important factors in the context of industrial applications were taken into account when building the frame that supports the robotic arm above the conveyor belt. These factors seek to guarantee the best performance, security, and adaptability in the sorting procedure.

To safeguard operators and avoid accidents while in use, the frame should have strong safety features. Safety barriers, which serve as barriers to stop unintentional contact with the robotic arm's moving parts, should be placed strategically on all sides of the frame. These barriers successfully separate the sorting area and add an extra measure of security for workers who are near the system.

The frame should incorporate movable brackets along its vertical supports to make it easier to place and align the robotic arm precisely. These brackets make it simple to modify the arm's height and location in relation to the conveyor belt. The arm's height-adjustability guarantees compatibility with varied object sizes and streamlines the sorting procedure for various requirements. As a result of the flexible alignment options offered by the adjustable brackets, various sorting scenarios can be accommodated.

Components Platform/Housing

Image 34: Platform Side

Image 35: Platform Top

A special component table constructed of steel and is used to arrange and combine the many parts of the robotic trash sorting system. The system's various modules and components are housed and organized on the component table, which acts as a central platform. During installation, maintenance, and operation, it offers stability, organization, and simple access to the parts.

The aluminium material selected for the component table has a number of beneficial qualities. The system may be placed anywhere in the workspace because to its lightweight design, which also makes it easy to move and carry. Aluminium also offers enough rigidity and strength to handle the components' weight without sacrificing stability.

In addition to its structural advantages, aluminium is also corrosion-resistant, ensuring the component table's endurance and durability. In environments where the sorting system may be exposed to moisture, dust, or other potentially harmful materials, corrosion resistance is essential (Aluminum.org, 2021).

Industrial Considerations

Additionally, the component table design takes into account industry factors to improve functionality and adhere to strict specifications. Based on the particular requirements of the environment, alternate solutions for the component table may be suggested in some industrial settings:

- Modular Racks: Modular racks can be used to hold the system's components instead of a single table. Each module has a specific slot or compartment in these racks, making organization, cable management, and accessibility easier. The shelves or trays on modular racks are frequently movable, accommodating various component sizes and promoting quick installation and upkeep.
- Cabinets or Enclosures: For housing the system's components, cabinets or enclosures provide a secure, contained environment. To assure component security, temperature control, and neat wiring, these can be fitted with lockable doors, ventilation systems, and cable management features. For situations with rigorous safety rules or when the system needs to be secured against unauthorized access, enclosures or cabinets are especially ideal.

DC Motor

Image 36: 3 Phase motor

The 3-phase motor is preferred because of their reliability, simplicity and long service life. The 3-phase motor is a strong motor and is widely used in the industry. The 3-phase motor will be used to turn the conveyor belt. Due to the project need to be used at campus that don’t have 3 phase power the motor won’t work.

Alternative: 12v dc motor

An alternative to the 3-phase motor is a 12v dc motor. The 12v dc motor can be used at campus unlike the 3-phase motor. The 12v dc motor have high start torque which enables them to overcome heavy loads on startup. This feature make dc motors better than as motors. Another distinguishing feature is that dc motors are more compact to 3 phase motors. This motor is perfected for turning the conveyer.

Speed controller

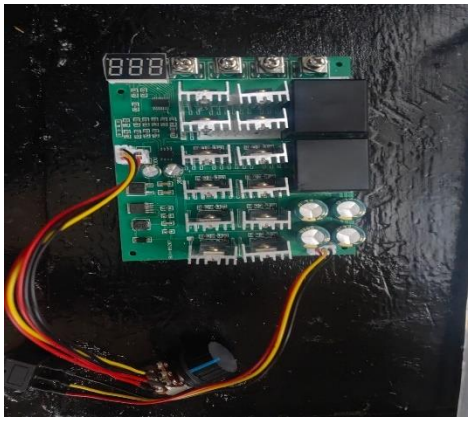


Image 37: 100-amp speed controller

The 100amp speed controller from Bot shop is used to control the speed on the 12v dc motor. The speed controller let us control the speed of the conveyor belt accurate in the speed range from 0 – 100% with a Display. The compact dimensions of the speed controller can fit neatly into the project.

The 100amp speed controller can be controlled more accurately. The 100amp speed controller can control the flow of any DC10-55v motor up to 300W. This make the 100amp speed controller from bot shop be used with a wider range of DC motor with different volt.

Servo 9 grams

This 9g servo motor is light weight, low cost and be controlled with precision. The 9g servo motor has stall torque of 1.2kg so the servo motor can't move heavy objects. The 9g servo motor is used in the project in the arm to close the claw and the wrist of the robotic arm. Those parts of the arm don't need a heavy-duty servo motor. Therefore, the light weight 9g servo motor is preferred.

Image 38: Servo motor 9grams

TOWERPRO MG996R SERVO MOTOR

Image 39: MG996R SERVO MOTOR

This servo motor is light weight, high torque servo motor. The servo motor is 55g and has 15kg stalling torque. This servo motor is used in the project for the heavy duty lifting. This servo motor is used at the bottom of the arm, so it must move the arm with the object, and this requires more torque. This servo motor also offers improved dead bandwidth and cantering precision.

SUNFOUNDER 20kg Motor High Torque Servo

This is an alternative to the TOWERPRO MG996R SERVO MOTOR. The main difference is that this servo motor has 20kg – 22.8kg of torque. This servo motor has 5kg more torque. This servo motor is preferred for this project but due to high cost and unavailability we decided to use the TOWERPRO MG996R SERVO MOTOR.

Image 40: SUNFOUNDER 20kg Motor High Torque Servo

Microcontroller: Arduino

Image 41: Arduino Uno rx3

The Arduino uno rx3 is a microcontroller that is mostly used for prototyping. It is a small easy to use microcontroller that is widely available and low cost. This microcontroller is used in the project to control the arm by controlling the 6-servo motor. The Arduino uno rx3 are the preferred microcontroller for this project.

Microcontroller: Raspberry Pi

This prototype makes use of a Raspberry Pi that will be used to perform object detection and localisation. The microcontroller makes use of the Raspberry Pi OS lite (32-bit) operating system. This operating system uses a command line interface for ease of use and increased performance. The controller is streamed remotely to an external PC through SSH during the development process for object detection and localisation.

Robotic Arm and Motor Weights

The table below provides a detailed list of the robotic arm's 3D printed parts including their dimensions and weights.

Component	Length	Width	Height	Weight
Shoulder joint				31g
Shoulder	150mm	20mm	50mm	25g
Wrist	130mm	20mm	40mm	9g
Gear1	50mm	4mm	5mm	2g
Gear2	50mm	4mm	5mm	1g
Claw attach	40mm	10mm	15mm	8g
Strut				Less than 1g
Claw 1	40mm	10mm	50mm	3g
Claw 2	40mm	10mm	50mm	3g
Motor				2,5KG
Conveyor				
Steel Frame				
Servo sg90	22mm	11.5mm	27mm	9g

Table 1: Robotic Arm and Motor Weights

Components contrast

Our idea is changing at a constant rate that our documentation was not able to fully keep up with the progress, as we are inexperienced students our best option is to do as much research as possible and try out ideas for

ourselves. This section will highlight components we have brought but not have implemented due to time constraint or an evolved version of idea.

Our initial idea was to use an ESP 32 module camera to create a master and slave relationship between Arduino and the camera itself. The camera was responsible to process the image that it detects and sends the processed image to Arduino for feedback but due to time constraints we have discarded that idea and instead just used a normal webcam that was mentioned earlier.

We initially had a 3-phase motor that came with the donated conveyor belt but when we tested it our it immediately tripped the DB board and stepped down the motor, but this proved to be an expensive method as we did not have the required components to do this, instead we looked for a 12v DC motor and replaced the 3-phase motor. This motor still had too much power, so we stepped it down and the components to do this were a lot cheaper, thus this motor was chosen for prototype.

Image 42: Phase-3 Motor

We decided to go from utilizing a breadboard to a prototype board to address the problem of loose connections as we increased the number of connections to the moving elements. Although the breadboard originally made things easier, as the system became more complicated, connections were more likely to come loose. We made the decision to buy a prototype board and soldered the jumper wires to ensure safe and dependable connections. This method improved the connections' stability while also proving to be a practical and affordable technique. Additionally, this change was effectively implemented within the specified time frame, allowing us to continue moving forward with the project without sacrificing effectiveness.

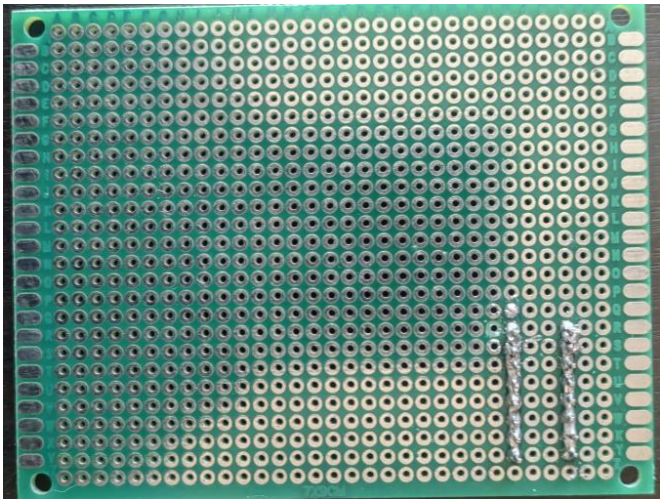


Image 43: Prototyping Board

User Manual: Robotic Arm and Conveyor Belt Using Arduino with Python Color Detection Code

System Components

The Robotic Arm and Conveyor Belt System using Python colour detecting code is the subject of this section, and focuses on its prototyping, development, and testing. The following details the system's configuration, use, programming, safety precautions, troubleshooting, and maintenance. The following components are included in the robotic arm and conveyor belt system:

- Robotic Arm: equipped with a gripper and several joints, this mechanical arm allows for accurate object manipulation.
- A conveyor belt: a motorized belt used to move items along a predetermined path.
- The Arduino Board: the main controller in charge of providing power and processing commands for the conveyor belt and robotic arm.
- USB Camera: responsible for camera feed and colour detection, connected to the Arduino Uno via the USB port.
- Power supply: Offers the system the necessary electrical power.
- Computer: Used to run the Python code for colour detection.

System Configuration

Follow these steps to set up the Robotic Arm and Conveyor Belt System with Python colour detection code:

- Step 1: Connect the power supply to the Arduino board.
- Step 2: Connect the Arduino board to the robotic arm and conveyor belt, ensuring proper wiring and connections.
- Step 3: Mount the components securely in their correct positions.
- Step 4: Check the power supply and ensure it is turned on.
- Step 5: Connect the Arduino board to your computer using a USB cable.
- Step 6: Install Python and the necessary libraries for colour detection on your computer. The necessary libraries include “cv2”, “numpy”, “serial”, and “time”.

System Operation

To operate the Robotic Arm and Conveyor Belt System with Python colour detection code, follow these steps:

- Step 1: Run the Python code on your computer.
- Step 2: The code will detect the specified colour from the camera feed.
- Step 3: Based on the colour detection, the Python code will send commands to the Arduino board.
- Step 4: The Arduino board will control the robotic arm and conveyor belt accordingly.

Modifications to the System

In some cases, you may wish to modify the existing program’s colour detection algorithm, arm-sweep speed, or servo positions.

- Step 1: Open the Python code in a text editor or any other integrated development environment (IDE) of your choice.
- Step 2: Modify the colour detection parameters as needed, such as the target colour range and action to be taken.
- Step 3: Save the modified code.
- Step 4: Run the updated Python code on your computer.
- Step 5: Verify that the Arduino board receives the appropriate commands, and the system responds accordingly.

Troubleshooting

If you encounter any issues with the Robotic Arm and Conveyor Belt System, consider the following troubleshooting steps:

- Check all connections and wiring to ensure they are properly connected.
- Verify that the power supply is functioning correctly.
- Review the Python code for any errors or bugs.
- Ensure that the Arduino board is properly programmed and receives commands from the Python code.

Safety Guidelines

Additionally, to ensure safe operation of the arm and conveyor belt system, we have provided the necessary safety guidelines:

- Keep hands and other objects away from the robotic arm and conveyor belt while in operation.
- Do not overload the conveyor belt beyond its recommended capacity.
- Disconnect the power supply before making any adjustments or modifications to the system.
- Follow all electrical safety precautions while working with the system.
- Be cautious when modifying the Python code and ensure it does not cause unexpected or unsafe movements.

Maintenance

To maintain the Robotic Arm and Conveyor Belt System, follow these guidelines:

- Regularly clean the robotic arm and conveyor belt to remove any debris or dust that may affect their performance.
- Inspect all wiring and connections periodically to ensure they are secure and undamaged.
- Lubricate mechanical joints of the robotic arm as recommended by the manufacturer of the motors used.
- Keep the system in a clean and dry environment to prevent moisture or corrosion.
- Ensure amperage requirements are not overloaded.

This section offers thorough instructions for the Robotic Arm and Conveyor Belt System using Python colour detecting code prototyping, development, and testing phases. Users can successfully implement and automate object manipulation based on colour recognition by following the provided instructions.

Development of Colour Detection Algorithm

In this section we look at how colour detection is implemented as well as how it is tested. The colour detection makes use of the python programming language to classify objects as colour. This algorithm classifies the colour by yellow, black, blue, or red. In practice this would be substituted for object classification where colours could be replaced by materials such as paper, plastic, metal, or glass.

Understanding Colour Detection

The procedures for colour space conversion, color thresholding, morphological operations, and contour analysis are described. Understanding the future implementation and performance assessment of the colour detection algorithm depends on having this fundamental knowledge.

Colour Conversion

Converting the collected frame from the default BGR (Blue-Green-Red) colour system to the HSV (Hue-Saturation-Value) colour space is the first step in the colour detection process. The establishment of colour limits based on hue, saturation, and value components is made easier by this conversion, which also makes it simpler to analyse colours.

Colour Thresholding

The colour detection technique applies predetermined colour boundaries to the frame once it is in the HSV colour space to produce a binary mask. This mask keeps pixels that fall inside the defined colour range and filters out pixels that do not. The binary mask is created using the OpenCV function `cv2.inRange()` based on the provided lower and higher colour limits.

Morphological Operations

Morphological procedures are used to polish the binary mask and remove noise. The opening operation (`cv2.MORPH_OPEN`) has been selected as the morphological operation for noise reduction. This process helps to smooth out the detected colour regions within the binary mask and remove small undesired spots.

Contour Analysis

After morphological procedures have been applied, contour analysis is carried out to locate and examine the boundaries of related components or objects that are present in the binary mask. For the purpose of finding contours in the binary mask, the OpenCV method `cv2.findContours()` is used. The method eliminates contours that don't fit certain size and form requirements by computing properties like contour area and aspect ratio.

Form Location:

The calculation utilizes form discovery to recognize boundaries of connected white locales within the parallel veil. Forms are gotten utilizing the `cv2.findContours()` work. Forms speak to the boundaries of colour districts within the picture.

Form Investigation:

Each form is exclusively analysed to channel out undesirable districts and distinguish substantial colour districts. The calculation calculates the form region utilizing `cv2.contourArea()` and the bounding rectangle utilizing `cv2.boundingRect()`. Forms with a zone more prominent than an edge (200 pixels) and an angle proportion inside a particular extend (0.8 to 1.2) are considered substantial colour locales.

These calculations collectively work together to identify colours within the video stream. The colour thresholding separates pixels of intrigued, morphological operations help refine the colour locales, and form examination empowers the identification of substantial colour districts based on estimate and perspective proportion criteria.

Furthermore, it's worth specifying that the calculation coordinating serial communication with an Arduino board to send signals from python based on the identified colours.

Overall, these computer vision techniques enable the code to effectively detect colours by segmenting specific colour regions from the captured frames. The combination of colour space conversion, colour thresholding, morphological operations, and contour analysis allows the code to identify and distinguish colours based on their visual characteristics in the HSV colour space.

We will now be looking at the program used to detect these colours, showing the implementation of these techniques, as well as the visual and hardware outputs

```
#Declaring Libraries being Used
import cv2
import numpy as np
import serial
import time

# Defining the RGB bounds low and high
color_bounds = {
    "yellow": ([20, 100, 100], [30, 255, 255]),
    "black": ([0, 0, 0], [180, 255, 30]),
    "blue": ([100, 100, 0], [140, 255, 255]),
    "red": ([0, 100, 100], [10, 255, 255])
}

def detect_color(frame):
    #Converts BGR color space to the HSV (Hue, Saturation, Value)color space
    hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    #Takes the color bounds and process to get color detecion for the system
    for color, (lower, upper) in color_bounds.items():
        lower = np.array(lower, dtype=np.uint8)
        upper = np.array(upper, dtype=np.uint8)
        mask = cv2.inRange(hsv_frame, lower, upper)

        #Morphological operations to remove noise
        kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (5, 5))
        mask = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel, iterations=2)

        # This is to find contours in the mask
        contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

        for contour in contours:
            # Calculate contour area and aspect ratio for the objects
            area = cv2.contourArea(contour)
            x, y, w, h = cv2.boundingRect(contour)
            aspect_ratio = w / float(h)

            # Filter out contours based on size and aspect ratio of the object
            if area > 200 and 0.8 <= aspect_ratio <= 1.2:
                return color

    return None

def send_signal_to_arduino(position):
    # Connection to the Arduino via serial port
    ser = serial.Serial('COM11', 9600)
```

```
    time.sleep(2)  #This is a time for connection to established

    # Send the position to Arduino
    ser.write(str(position).encode())

    # Wait for Arduino to process the command
    time.sleep(2)

    # Close the serial connection
    ser.close()

def main():
    # This opens the webcam
    cap = cv2.VideoCapture(1)

    while True:
        ret, frame = cap.read()
        if not ret:
            break
         # runs the function and returns the colour
        color = detect_color(frame)

        cv2.imshow('Frame', frame)
        #Checks if the colour detects and assigns a position either 1,2,3 or 4
        if color:
            print(f"Detected color: {color}")
            position = None
            if color == "yellow":
                #Assigning a postion of 1 for the drop off
                position = 1
            elif color == "black":
                #Assigning a postion of 2 for the drop off
                position = 2
            elif color == "blue":
                #Assigning a postion of 3 for the drop off
                position = 3
            elif color == "red":
                #Assigning a postion of 4 for the drop off
                position = 4

            if position is not None:
                # sends the Positions assigned 1 2, 3 or 4
                send_signal_to_arduino(position)
                break

        # closes the system if 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
#closes the webcam
    cap.release()
    cv2.destroyAllWindows()
```

```
if __name__ == '__main__':
    main()
```

Code Snippet 1: Colour detection algorithm

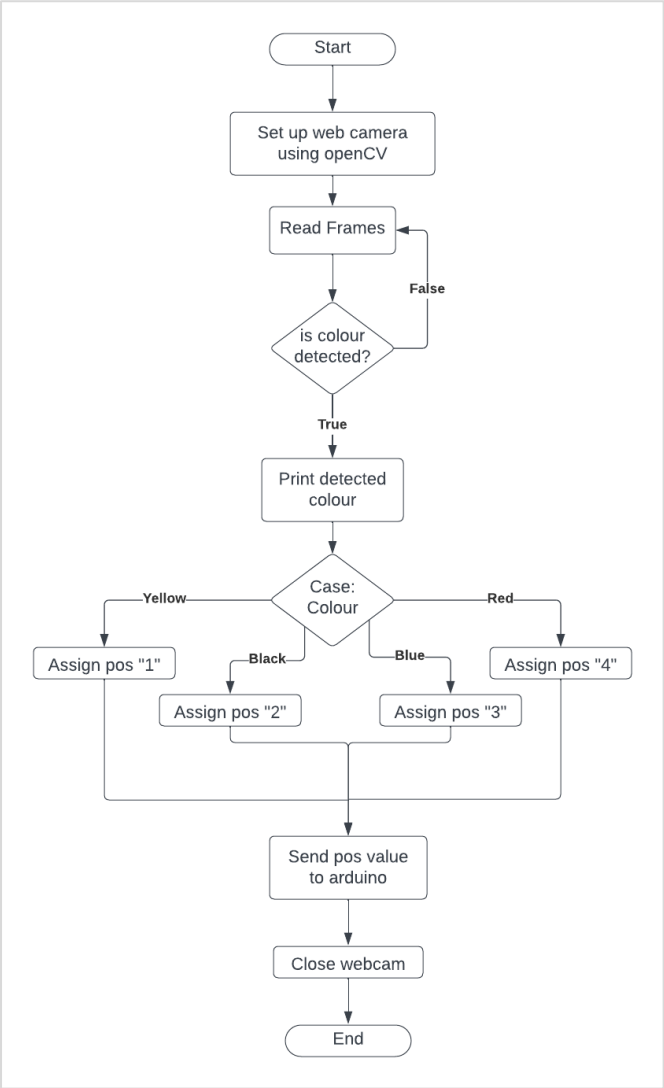


Diagram 9: Colour detection algorithm flow

The code presented above shows how to merge the Arduino and Python color detecting system for controlling a robotic arm.

The first line of the code imports the required libraries, including OpenCV (cv2), numpy, serial, and time. These libraries offer functions for managing time, serial connection, array operations, and image processing, respectively.

To identify colour, it processes each frame by converting it from the BGR colour space to the HSV colour space using `cv2.cvtColor()`. The HSV colour space is more suitable for colour analysis.

For each colour defined in the colour bounds dictionary, the code applies a colour threshold using `cv2.inRange()` to create a binary mask where the detected colour appears as white and other colours appear as black. Morphological operations (`cv2.morphologyEx()`) are then applied to remove noise and improve the quality of the mask.

Contours are extracted from the mask using `cv2.findContours()` and filtered based on their area and aspect ratio. If a contour meets the criteria of sufficient area and aspect ratio close to a square shape, the code considers the colour detected.

Using the supplied port and baud rate, the `send_signal_to_arduino(position)` method creates a serial connection with the Arduino board. The position parameter, which represents the color that was detected, is provided to the Arduino. To give the connection time to establish and the Arduino time to process the command, a delay is added. The serial connection is finally terminated.

The `detect_color` function takes a frame as input and performs color detection using the provided color bounds. It converts the frame from BGR color space to HSV color space, applies a mask to isolate the desired colours, performs morphological operations for noise removal, and finds contours in the resulting mask. It then filters the contours based on size and aspect ratio to detect the desired colours.

The `send_signal_to_arduino(position)` function is used to send the position to the Arduino for robotic arm control if a valid position has been assigned. If the '`q`' key is hit to stop the program or when a valid color is recognized and processed, the loop ends.

The webcam capture is released at the end of the code, and all open windows are closed.

Colour Bounds

The colour bounds below are used when classifying colours. The classification makes use of the HSV format.

Colour Classification	Hue	Saturation	Value
Yellow	[20, 30]	[100, 255]	[100, 255]
Black	[0, 180]	[0, 255]	[0, 30]

Blue	[100, 140]	[100, 255]	[0, 255]
Red	[0, 10]	[100, 255]	[100, 255]

Table 2: Colour Classification using HSV colour bounds

Output

While there are no explicit outputs returned from this program, there are visual feedbacks and hardware outputs.

Visual Outputs: The developer is given feedback through the usage of the following visual outputs.

- **The webcam**: This is activated by the system, which records live video frames. The screenshot frames are shown in a window with the label "Frame." Developers may view the objects the camera has caught and watch the color identification process in action due to this visual feedback.
- **Detected Colour Message**: A message specifying the discovered color is output to the console or command line interface if the system successfully detects a color using the defined color boundaries and object filtering criteria. It could say, for instance, "Detected color: yellow" or "Detected color: blue." Users are informed of the hue the machine has detected through this information.

The example below shows the result if a yellow object is detected

```
                #Assigning a postion of 3 for the drop off
                position = 3
            elif color == "red":
                #Assigning a postion of 4 for the drop off
                position = 4

            if position is not None:
                # sends the Positions assigned 1 2, 3 or 4
                send_signal_to_arduino(position)
                break

        # closes the system if 'q' is pressed
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
#closes the webcam
    cap.release()
    cv2.destroyAllWindows()

if __name__ == '__main__':
    main()
```

```
Detected color: yellow
```

Image 44: Yellow object is detected

Hardware Output:

Based on the detected color, the system communicates with an Arduino board to drive the robotic arm. A position value designating the designated drop-off location. The following are the location values that are provided to the Arduino:

- The position value `1` is sent to the Arduino if the color "yellow" is detected.
- The position value `2` is provided to the Arduino if the color "black" is detected.
- The position value `3` is communicated to the Arduino if the color "blue" is detected.
- The position value `4` is communicated to the Arduino if the color "red" is detected.

Other Developments and Notes

Live camera footage

We had implemented a function to open a window on the computer, providing constant video feed and real-time bounding box calculation. While this feature was functional, it was unfortunately removed. This is because the code was constantly running, causing the code to be less accurate as the slight movements would cause a change in incoming light, consequently causing inaccurate detection of different colours. In the example below, this can be seen as a red object is detected as red, yellow, and black in different intervals.

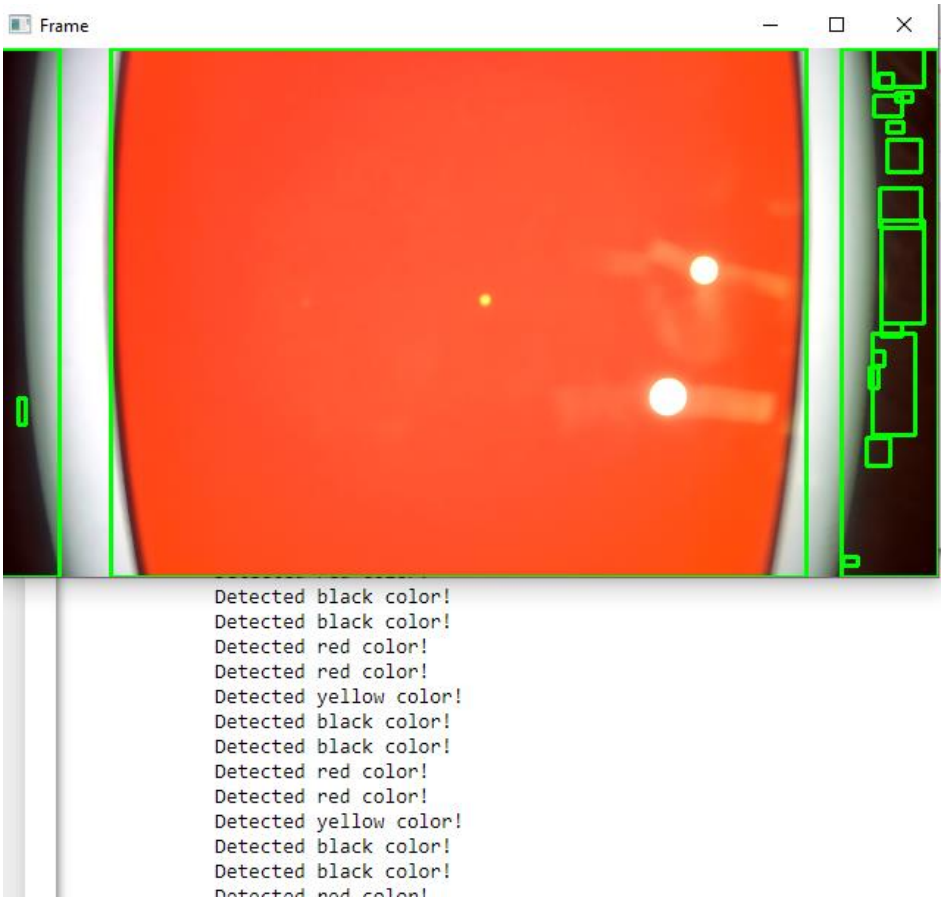


Image 45: Inaccurate colour detection

Real-time distance tracking

With the introduction of a bounding box, this was further developed where each box would classify the distance and give the distance in length and width of the object as well as the object it detected. However, the program became delayed as more process had to be done. We deemed this implementation to be inefficient in our system as in practical use this system such a system relies heavily on performance.

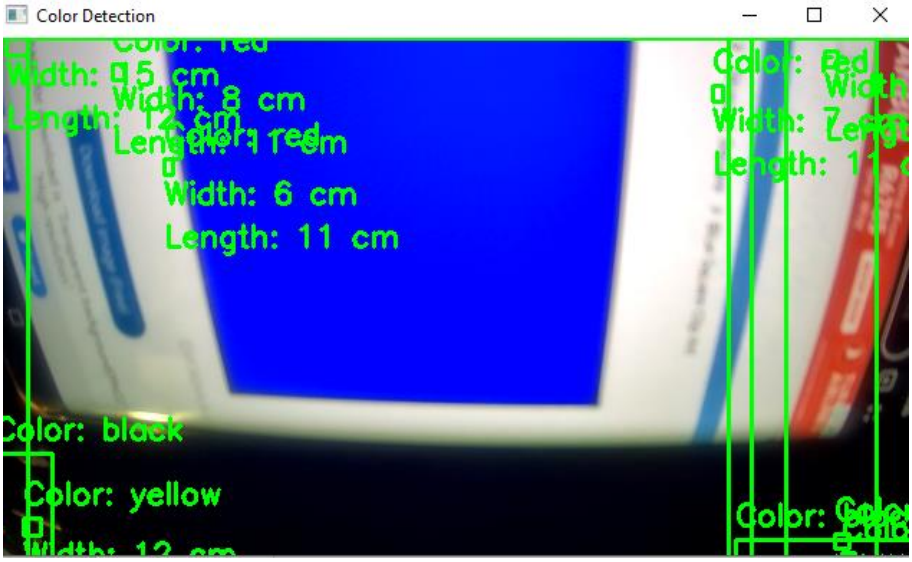


Image 46: Implementation of distance tracking

Development of the Robotic Arm’s Movements

In this section we provide an understanding on how the positions outputted from the colour detection translate into the robotic arm’s movement and how the drop off positions are correctly decided.

```
#include <Servo.h>
Servo hand;
Servo wrist2;
Servo wrist1;
Servo shoulder;
Servo elbow;
Servo base;
Servo conveyor;

const int HOME_POSITION[] = {180, 40, 0, 30, 150, 90}; // Adjusted base angle
to 40
const int PICKUP_POSITION[] = {100, 180, 0, 30, 150, 90};
const int DROP_POSITIONS[][6] = {
  {100, 180, 0, 30, 150, 20},
  {100, 180, 0, 30, 150, 40},
  {100, 180, 0, 30, 150, 20},
  {100, 180, 0, 30, 150, 40}
};
const int NUM_DROP_POSITIONS = sizeof(DROP_POSITIONS) /
sizeof(DROP_POSITIONS[0]);

bool isInitialized = false;
bool conveyorReset = false;

void setup() {
  Serial.begin(9600);
  initializeServos();
}

void loop() {
  if (!isInitialized) {
    initializeArm();
    isInitialized = true;
  }

  if (!conveyorReset) {
    resetConveyor();
    conveyorReset = true;
  }

  if (Serial.available() > 0) {
    int input = Serial.parseInt();
    executeAction(input);
  }
}

void initializeServos() {
  hand.attach(6);
  wrist2.attach(9);
```

```
  wrist1.attach(10);
  shoulder.attach(3);
  elbow.attach(11);
  base.attach(2);
  conveyor.attach(8);
}

void initializeArm() {
  moveServos(HOME_POSITION);
}

void resetConveyor() {
  moveConveyor(155);
  delay(700); // Change delay to 0.5 seconds (500 milliseconds)
  moveConveyor(0);
}

void executeAction(int action) {
  if (action >= 1 && action <= 4) {
    int dropIndex = action - 1;
    moveConveyor(155);
    delay(700); // Delay before conveyor stops (0.5 seconds)
    moveConveyor(0);
    moveServos(PICKUP_POSITION);
    delay(700); // Delay after reaching PICKUP_POSITION (0.5 seconds)
    moveServos(DROP_POSITIONS[dropIndex]);
    moveServos(HOME_POSITION);
  } else {
    Serial.println("Invalid input");
  }
}

void moveServos(const int angles[]) {
  int currentAngles[] = {hand.read(), wrist2.read(), wrist1.read(),
shoulder.read(), elbow.read(), base.read()};
  const int speedDelay = 15;  // Adjust this value for faster or slower
movement

  for (int i = 0; i < 6; i++) {
    int currentAngle = currentAngles[i];
    int targetAngle = angles[i];

    for (int angle = currentAngle; angle != targetAngle; angle += (targetAngle
> currentAngle) ? 1 : -1) {
      setServoAngle(i, angle);
      delay(speedDelay);
    }
  }
}

void moveConveyor(int angle) {
  conveyor.write(angle);
  delay(700); // Change delay to 0.5 seconds (500 milliseconds)
```

```
}

void setServoAngle(int servoIndex, int angle) {
  switch (servoIndex) {
    case 0:
      hand.write(angle);
      break;
    case 1:
      wrist2.write(angle);
      break;
    case 2:
      wrist1.write(angle);
      break;
    case 3:
      shoulder.write(angle);
      break;
    case 4:
      elbow.write(angle);
      break;
    case 5:
      base.write(angle);
      break;
    default:
      break;
  }
}
```

Code Snippet 2: Movement of the robotic arm using Arduino

The program above is an Arduino sketch created for servo motor control of a robotic arm. This code enables the precise manipulation of the hand, wrist, shoulder, elbow, and base, among other parts of the arm. The arm's ability to transfer objects is implemented by addition of a conveyor belt control.

The `Servo` library, which offers the required functions and methods for servo initialization and angle manipulation, is used by the code to enable servo motor operation. The first line of code creates instances of the Servo class for each servo motor in the system, connecting the servos to specific Arduino board pins.

The code contains a number of constant arrays to specify the intended places for the arm movements. The default home location for each servo is represented by the `HOME_POSITION` array. The position necessary for picking up objects is specified by the `PICKUP_POSITION` array. Additionally, the two-dimensional `DROP_POSITIONS` array maintains different drop positions for various places. The total number of drop positions is shown by the `NUM_DROP_POSITIONS` constant.

The serial communication is initially initialized with a baud rate of 9600 in the `setup()` method, which is called only once at the beginning of the program. This enables communication between the python program and the Arduino board. Additionally, the `initializeServos()` function is used to link the servo connections by attaching the servo objects to the appropriate pins.

The `loop()` function, which implements the main program loop, runs continually. The code runs the `initializeArm()` function to set all the servos to the home position specified in the `HOME_POSITION` array on the first iteration. The conveyor belt is also turned on by calling the `resetConveyor()` function. For a predetermined amount of time, this function sets the conveyor servo motor angle to 155. It then stops the conveyor by setting the angle to 0.

The code waits for input from the Serial port during the loop. The `executeAction()` function is used to carry out the required action after receiving input. This function checks to see if the input received from the

colour detection is between 1 and 4. In that case, it chooses the appropriate drop position from the `DROP_POSITIONS` array and makes a series of motions. This means turning on the conveyor belt for a predetermined amount of time, moving the arm to the `PICKUP_POSITION`, moving the arm to the chosen drop point, and then bringing the arm back to the `HOME_POSITION` array-defined home position. An "Invalid input" message is transmitted via the Serial port if the input is outside of the range.

The code uses the `moveServos()` function to make servo motions easier. The servo angles are gradually adjusted by this function to meet the intended target angles. It cycles through each servo, measuring the current and target angles, and then gradually increasing or decreasing the angle until it reaches the desired angle. The arm will move more smoothly and steadily due to this implementation.

The operations `moveConveyor()`, which modifies the angle of the servo motor driving the conveyor belt's movement, and `setServoAngle()`, which adjusts a specific servo motor's angle based on the desired angle and the servo index.

In conclusion, this code offers a thorough foundation for the robotic arm's control. It features capabilities for object manipulation and transportation through the integration of a conveyor belt and permits exact positioning of the arm's component parts. Developers can transmit precise commands to the Arduino board to carry out activities by utilizing serial connection, which enables the arm to carry out a variety of jobs requiring precise and regulated movements.

Position Mapping Considerations

It is important to note that position mapping in the above code snippet is matched to our development of the litter sorting system. Factors such as bin placement, servo motor range, as well as the dimensions of the robotic arm and conveyor belt play a significant role in the decision of position mapping. It is suggested that modular approaches are taken such to enable adjustments in other systems where system architecture is different from our own.

Raspberry Pi Remote Connection and Video Streaming

We have taken a different approach to object detection and localisation using a Raspberry Pi as well. For ease of use, the controller is connected remotely to an external PC via an SSH controlled by the PuTTY software. Additionally, video feed is sent from the controller remotely using a client-server approach where all code is run and compiled using Jupyter Notebook. The below code snippets show the remote connection between the controller and external PC used to stream the video captured by the controller's web camera.

```
import socket
import cv2
import pickle
import struct



s = socket.socket()
port = 12345
s.connect(('192.168.0.134', port))



data = b""
payload_size = struct.calcsize("Q")
```

```
while True:
    while len(data) < payload_size:
        packet = s.recv(4*1024) # 4K
        if not packet: break
        data+=packet
    packed_msg_size = data[:payload_size]
    data = data[payload_size:]
    msg_size = struct.unpack("Q",packed_msg_size)[0]

    while len(data) < msg_size:
        data += s.recv(4*1024)
    frame_data = data[:msg_size]
    data  = data[msg_size:]
    frame = pickle.loads(frame_data)
    cv2.imshow("RECEIVING VIDEO",frame)
    key = cv2.waitKey(1) & 0xFF
    if key  == ord('q'):
        break
client_socket.close()
```

Code Snippet 3: Client-side video streaming

The code snippet above shows the actions taken to form the connection with the server-side to stream video footage remotely. The majority of the work involved in showing the received video frames in a window is provided in this half. The "RECEIVING VIDEO" window, which shows the video frames obtained from the server, would be the result of this algorithm.

When the user pushes the 'q' key, the code will quit the loop and the window will close, ending the program. Until then, the window will display a continuous stream of video frames.

```
import socket
import cv2
import pickle
import struct



s = socket.socket()
print ("Socket successfully created")



port = 12345



s.bind(('', port))
print ("socket binded to %s" %(port))

s.listen(5)
print ("socket is listening")



while True:
    c, addr = s.accept()
```

```
    print ('Got connection from', addr )

    if c:
        vid = cv2.VideoCapture(-1)

        while (vid.isOpened()):
            img, frame = vid.read()

            a = pickle.dumps(frame)
            message = struct.pack("Q",len(a))+a
            c.sendall(message)

            #cv2.imshow('Transmitting Video',frame)
            key = cv2.waitKey(1) & 0xFF
            if key ==ord('q'):
                c.close()
```

Code Snippet 4: Server-side video streaming

The code above creates a video streaming server using sockets and records camera video frames with OpenCV. It starts by importing the required modules, such as pickle, socket, cv2 (OpenCV), and struct.

Using `socket.socket()`, a socket object named s is then constructed. Using `s.bind('', port)`, the socket is then bound to a particular IP address and port. The socket can accept connections from any accessible network interface if the `bind()` function returns an empty string.

`S.listen(5)`, where 5 is the maximum quantity of connections in the queue, causes the server to begin listening for incoming connections. Using `s.accept()`, it enters a loop to accept incoming connections. The client's address is printed each time a connection is made.

The default webcam is launched when the video capture object vid is initialized in the loop using `cv2.VideoCapture(-1)`. Using `vid.read()`, a new loop is started to read video frames from the camera. The variables image and frame contain the captured frames' data.

Pickle.dumps(frame) is used to serialize the frame and create a byte stream from it. The length of the serialized frame is computed, and using the syntax `struct.pack("Q", len(a)) + a)`, the length and frame data are combined.

Using `c.sendall(message)`, where c is the connection socket, the serialized frame is then transmitted over the network to the client. To make sure all the data is sent, use the `sendall()` function.

`cv2.waitKey(1) & 0xFF` are used in the code to check for the 'q' key stroke as well. When the 'q' key is depressed, `c.close()` is used to terminate the connection. This enables the server to end the loop and cease streaming video whenever needed.

Development of Object Detection and Localisation

We outline the object localization and detection methods that we employed in this section of the paper. In developing a litter sorting system, object detection and localisation are key. The approach used, the experimental setup, and the outcomes of our application are described in the following subsections.

Methodology

Object detection

We used the Python OpenCV module to find objects in the video stream. A video capture object (`cv2.VideoCapture`) was used to capture the video frames. The actions listed below were carried out for every frame:

Preprocessing: To enhance the image's quality, the captured frame was treated. We used Gaussian blurring (`cv2.GaussianBlur`) with a kernel size of (7, 7), and bilateral filtering (`cv2.bilateralFilter`) with a kernel size of 5x5.

Morphological dilation: To improve the object boundaries, we applied morphological dilation (`cv2.dilate`) using a 5x5 kernel.

Edge detection: Using `cv2.cvtColor` to convert the result image to grayscale, canny edge detection (`cv2.Canny`) was applied with threshold values of 40 and 10.

Thresholding: To obtain a binary image, a binary thresholding operation (`cv2.threshold`) with a threshold value of 128 was applied.

Extraction of Contours: From the binary image, we extracted contours using `cv2.findContours`. Only the external contours were retrieved using the `cv2.RETR_EXTERNAL` parameter, and the contour representation was compressed using `cv2.CHAIN_APPROX_SIMPLE`.

Bounding Rectangles: Using `cv2.boundingRect`, bounding rectangles were calculated for each contour. Rectangles with the smallest possible area were also calculated using `cv2.minAreaRect`. The detected objects in the image are represented by these rectangles.

Visualization: Using `cv2.drawContours`, the rectangles of the identified objects were drawn on the original frame, and the resulting image was presented.

Object Localisation

By computing the minimal area rectangles (`cv2.minAreaRect`) for each contour produced during object detection, we were able to localize objects in our implementation. The bounding box coordinates were then determined using the vertices of these rectangles. To depict the localized objects graphically on the initial frame, the bounding box was constructed.

Results and Analysis

This subsection presents the outcomes of our object detection and localization implementation. Successful localization allowed for the representation of the detected objects as bounding boxes in the video frames. The qualities of the objects and the intricacy of the settings have an impact on the localization accuracy.

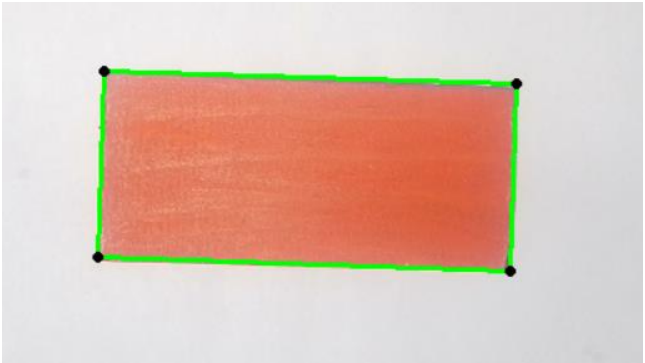


Image 47: Example output of object detection and localisation algorithm

Object detection and localisation algorithm

```
# import the opencv library
import cv2
import numpy as np

# define a video capture object
vid = cv2.VideoCapture(1)

while (True):
```

```
# Capture the video frame
# by frame
ret, frame = vid.read()

image = frame
bilateral = cv2.bilateralFilter(image, 15, 75, 75)
Gaussian = cv2.GaussianBlur(bilateral, (7, 7), 0)
# Taking a matrix of size 5 as the kernel
kernel = np.ones((5, 5), np.uint8)

img_dilation = cv2.dilate(Gaussian, kernel, iterations=1)

# Convert image to grayscale
gray = cv2.cvtColor(img_dilation, cv2.COLOR_BGR2GRAY)

# Use canny edge detection
edges = cv2.Canny(gray, 40, 10, apertureSize=3)

# threshold
thresh = cv2.threshold(edges, 128, 255, cv2.THRESH_BINARY)[1]

# get contours
contours = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
contours = contours[0] if len(contours) == 2 else contours[1]
for cntr in contours:
x, y, w, h = cv2.boundingRect(cntr)
rect = cv2.minAreaRect(cntr)
box = cv2.boxPoints(rect)
print(box)
box = np.int0(box)
cv2.drawContours(image, [box], 0, (0, 255, 0), 2)

# Display the resulting frame
cv2.imshow('frame', frame)

# the 'q' button is set as the
# quitting button you may use any
# desired button of your choice
if cv2.waitKey(1) & 0xFF == ord('q'):
break

# After the loop release the cap object
vid.release()
# Destroy all the windows
cv2.destroyAllWindows()
```

Code Snippet 5: Object detection and localisation algorithm

Development of Object Classification

We describe the creation of the object classification system utilizing the VGG16 convolutional neural network architecture in this part. Sorting items into six distinct classes was the goal. Data collection, model configuration, training, and evaluation made up the development process.

Data Gathering

A dataset was assembled, consisting of photos from various sources such as the trashnet database provided on Kaggle, for the purposes of testing and training the object classification system. A training set and a test set were created from the dataset using 83% and 17% respectively. The test set was utilized for evaluation, whereas the training set was used to train the model. To comply with the input specifications of the VGG16 model, the photos were downsized to a standard size of 224x224 pixels. The dataset consists of images in six different classes, namely, cardboard, paper, plastic, glass, metal, and non-recyclable. The table below provides the number of images used in their respective classes.

Category	Number of Images
Cardboard	549
Glass	601
Metal	492
Paper	712
Plastic	578
Non-recyclable	167

Table 3: Image distribution in different categories

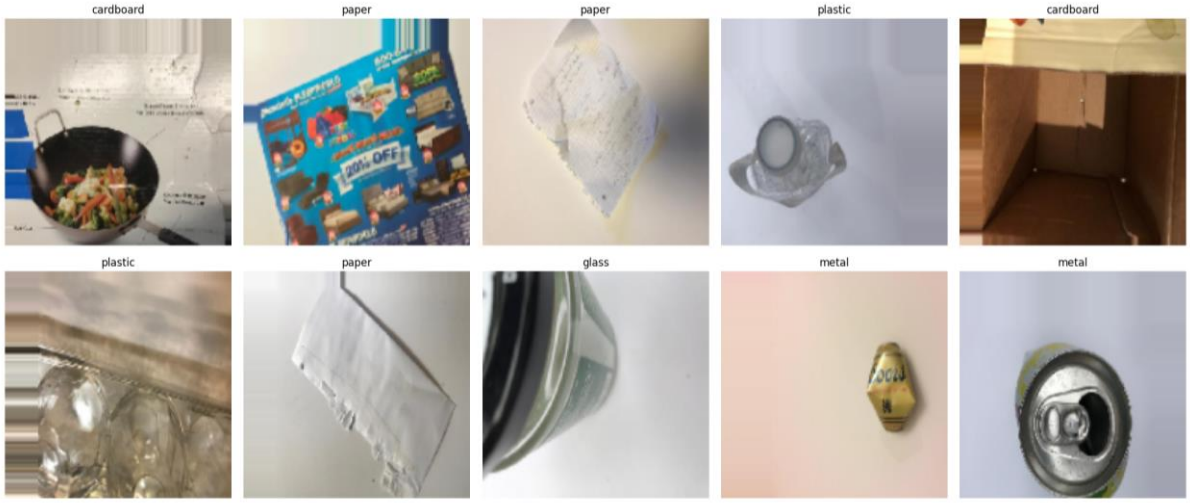


Image 48: Sample of training and testing images

Model Configuration

We started the VGG16 architecture using ImageNet weights and left out the top classification layer in order to use the pre-trained VGG16 model for object classification. We froze the pre-trained weights in the VGG16 model and only trained the newly added classification layers by setting the trainable parameter of all layers to False. Three completely interconnected layers with ReLU activation functions made up the classification layers.

Model Training

We used the ImageDataGenerator class from the Keras library to supplement the training data when training the object classification system. The diversity of the training samples was increased using horizontal flip augmentation. The categorical cross-entropy loss function and the Adam optimizer were used to build the model. Using a batch size of 32, we trained the model over the course of 20 epochs.

20 epochs were selected after deciding on the training loss's convergence seen in the image below. The loss function is often minimized during training, and the number of epochs is frequently decided by watching when the loss stops reducing considerably or reaches a plateau. The model has picked up on the patterns and characteristics included in the training data if the training loss has decreased to a desirable level after 20 epochs.

The model's performance on the training data is evaluated by a training accuracy metric. Insights into the model's learning progress can be gained by tracking the training accuracy during the training process. The model has properly learned from the training data as the training accuracy increased with each iteration and reaches a suitable level after 20 epochs.

A hyperparameter called learning rate controls how frequently the model updates its parameters during training. It regulates how quickly the model absorbs new information from the data. If a greater learning rate is selected, the model converges more quickly and may need fewer epochs to reach a particular level of performance. On the other hand, a slower learning rate may result in slower convergence and increase the number of epochs needed for the model to attain its ideal state.

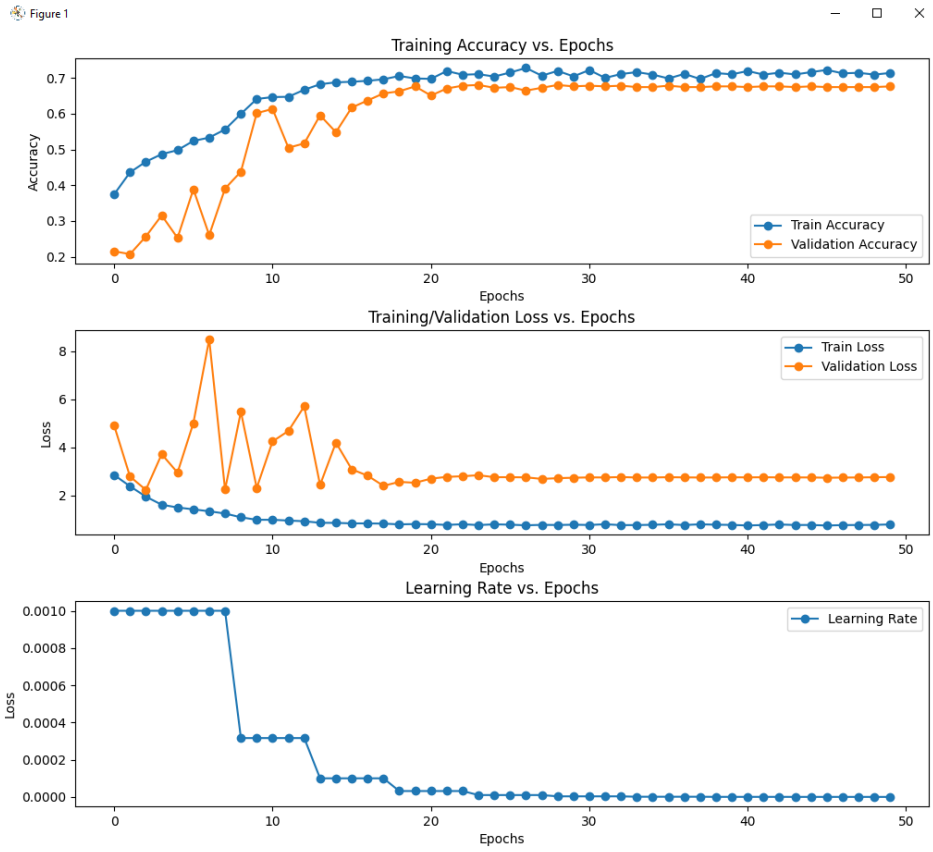


Image 49: Training accuracy, Validation loss, and Learning rate against epochs

Model Evaluation

We used the test set to assess how well the item classification system performed. The class labels of the test photos were predicted using the training model. We calculated numerous measures, such as accuracy, precision, recall, and F1-score, to evaluate the model's performance. The scikit-learn library's `classification_report` function offered a thorough analysis of the model's performance.

Using the test set's actual class labels as well as the anticipated class labels, the confusion matrix was also computed. This matrix offered insights into any model misclassifications and helped show the distribution of projected classes.

Results and Discussions

On the test set, the trained object classification model had an accuracy of 70%. A thorough evaluation of the model's performance for each class was made possible by the classification report's disclosure of the precision, recall, and F1-score for each class. The confusion matrix revealed further instances when the model had trouble correctly identifying particular objects.

Due to the relatively limited size of the training dataset and the potential for overfitting, the proposed object classification system has certain drawbacks. Potential bias imposed by the training dataset is one of them. Expanding the dataset and investigating cutting-edge methods like transfer learning or fine-tuning to improve the model's performance could be the main goals of future study.

	precision	recall	f1-score	support
0	0.81	0.90	0.85	81
1	0.96	0.47	0.63	100
2	0.56	0.96	0.71	82
3	0.77	0.82	0.80	118
4	0.65	0.61	0.63	96
5	0.22	0.07	0.10	30
accuracy			0.70	507
macro avg	0.66	0.64	0.62	507
weighted avg	0.72	0.70	0.69	507

Image 50: Classification report

```
[[73  0  0  7  1  0]
 [ 2 47 23  5 18  5]
 [ 0  1 79  1  1  0]
 [11  0  4 97  6  0]
 [ 3  1 31  0 59  2]
 [ 1  0  5 16  6  2]]
```

Image 51: Confusion matrix

Conclusion

In conclusion, the development of the VGG16 model-based object classification system has produced encouraging results, correctly classifying items into six different groups. However, the model's evaluation revealed some flaws, such as the classification of plastic bottles as glass, the misclassification of some brown papers as cardboard, and the mistake of metallic-looking materials with metal. These results show that the model has to be improved and further refined.

Future research could think about a variety of strategies to improve the model's performance and solve these limitations. More varied samples of the incorrectly categorized materials, including plastic bottles and brown papers, might be added to the training dataset to improve the model's ability to distinguish between them. Additionally, investigating methods like transfer learning or fine-tuning, which use pre-trained models and adapt them to particular tasks, could increase the accuracy of object classification.

The results of this development process offer insightful information for next studies in the area of object classification. It is possible to improve the model's performance and make a positive impact on the development of more precise and dependable object categorization systems by addressing the model's observed shortcomings and using these approaches.

Object Classification Algorithm

```
import numpy as np
import tensorflow as tf
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Flatten, Dense
from tensorflow.keras.applications import VGG16

num_classes=6
IMAGE_SHAPE = [224, 224]
batch_size=32
epochs = 5

vgg = VGG16(input_shape = (224,224,3), weights = 'imagenet', include_top =
False)
for layer in vgg.layers:
layer.trainable = False
x = Flatten()(vgg.output)
x = Dense(128, activation = 'relu')(x)
x = Dense(64, activation = 'relu')(x)
```

```
x = Dense(num_classes, activation = 'softmax')(x)
model = Model(inputs = vgg.input, outputs = x)
model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])

trdata = ImageDataGenerator(horizontal_flip=True)
train_data_gen =
trdata.flow_from_directory(directory="my_data/train",target_size=(224,224),
shuffle=False, class_mode='categorical')
tsdata = ImageDataGenerator()
test_data_gen = tsdata.flow_from_directory(directory="my_data/test",
target_size=(224,224),shuffle=False, class_mode='categorical')

training_steps_per_epoch = np.ceil(train_data_gen.samples / batch_size)
validation_steps_per_epoch = np.ceil(test_data_gen.samples / batch_size)
model.fit_generator(train_data_gen, steps_per_epoch = training_steps_per_epoch,
validation_data=test_data_gen,
validation_steps=validation_steps_per_epoch,epochs=epochs, verbose=1)
print('Training Completed!')

Y_pred = model.predict(test_data_gen, test_data_gen.samples / batch_size)
val_preds = np.argmax(Y_pred, axis=1)
import sklearn.metrics as metrics
val_trues =test_data_gen.classes
from sklearn.metrics import classification_report
print(classification_report(val_trues, val_preds))

Y_pred = model.predict(test_data_gen, test_data_gen.samples / batch_size)
val_preds = np.argmax(Y_pred, axis=1)
val_trues =test_data_gen.classes
cm = metrics.confusion_matrix(val_trues, val_preds)
print(cm)

keras_file="model.h5"
tf.keras.models.save_model(model,keras_file)
```

Code Snippet 6: Object classification algorithm

Other Developments: Brand Detection

Other than colour detection, we has also looked into the detection of brands for any items found on the belt. Brand detection in trash sorting systems has several uses and advantages. By making businesses answerable for their garbage and encouraging appropriate waste management, it encourages product responsibility. By recognizing and giving recyclable products top priority, the system also streamlines recycling activities. It offers useful marketing insights that help businesses make decisions and adjust their strategy in light of consumer behaviour and market changes.

Additionally, trademark detection makes sure that regulations are followed, especially in areas where producer responsibility laws are more stringent. It makes it easier to comply with rules and encourages sustainable product lifecycle management by assisting in the identification of those who are responsible for litter. By examining the frequency and distribution of litter from certain brands and assisting efforts to reduce trash output and pollution, the technology also supports environmental impact assessments.

Initially this would have been used to serve as a guide for any readers looking into implementing a process such as this; however due to time constraints this was not further developed.

```python
import numpy as np
import pytesseract

def preprocess_image(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    _, thresholded = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)
    return thresholded

def detect_logo(image):
    logos = {
        "cocacola": {
            "template": cv2.imread("cocacola_logo.png", cv2.IMREAD_GRAYSCALE),
            "label": "Coca Cola"
        },
        "koo": {
            "template": cv2.imread("koo_logo.png", cv2.IMREAD_GRAYSCALE),
            "label": "Koo"
        },
        "blacklabel": {
            "template": cv2.imread("blacklabel_logo.png", cv2.IMREAD_GRAYSCALE),
            "label": "Black Label"
        }
    }

    orb = cv2.ORB_create()

    for logo, data in logos.items():
        template = data["template"]
        label = data["label"]
```

Image 52: Brand detection example code

Image 53: Logo detection output example

Simulation of Brand Detection

Due to budget constraints, we could not get access to Matplotlib so we consulted with our lecture and has agreed we could make use of pygame to create the simulates.

```python
#This are the Liabries we are using
import mediapipe as mp
import cv2
import numpy as np
from mediapipe.framework.formats import landmark_pb2
import time
from math import sqrt
import win32api
import pyautogui

# modules from the mediapipe library and  mp_drawing

mp_drawing = mp.solutions.drawing_utils
mp_hands = mp.solutions.hands
click=0
#Starting the Webcam
video = cv2.VideoCapture(1)
#This line initializes the hand tracking module from the Mediapipe library
(mp_hands),
#specifying the minimum detection and tracking confidence thresholds as 0.8.
```

```
#The with statement ensures that the resources used by the hands object are
properly released after its usage.
with mp_hands.Hands(min_detection_confidence=0.8, min_tracking_confidence=0.8)
as hands:
    while video.isOpened():
        _, frame = video.read()
        #converts the color space of the frame from BGR (OpenCV default)
        image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
         #horizontally flips the image
        image = cv2.flip(image, 1)
         #retrieves the dimensions (height and width)
        imageHeight, imageWidth, _ = image.shape
       # performs hand detection and tracking, returning the results.
        results = hands.process(image)


        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

        if results.multi_hand_landmarks:
            for num, hand in enumerate(results.multi_hand_landmarks):
                mp_drawing.draw_landmarks(image, hand,
mp_hands.HAND_CONNECTIONS,
                                          mp_drawing.DrawingSpec(color=(250, 44,
250), thickness=2, circle_radius=2),
                                           )
 #draws the landmarks and connections on the image
        if results.multi_hand_landmarks != None:
          for handLandmarks in results.multi_hand_landmarks:
            for point in mp_hands.HandLandmark:


                normalizedLandmark = handLandmarks.landmark[point]
                pixelCoordinatesLandmark =
mp_drawing._normalized_to_pixel_coordinates(normalizedLandmark.x,
normalizedLandmark.y, imageWidth, imageHeight)

                point=str(point)
 #checks if there are any detected hand landmarks
                if point=='HandLandmark.INDEX_FINGER_TIP':
                 try:
                    indexfingertip_x=pixelCoordinatesLandmark[0]
                    indexfingertip_y=pixelCoordinatesLandmark[1]

win32api.SetCursorPos((indexfingertip_x*4,indexfingertip_y*5))

                 except:
                    pass

                elif point=='HandLandmark.THUMB_TIP':
                 try:
                    thumbfingertip_x=pixelCoordinatesLandmark[0]
                    thumbfingertip_y=pixelCoordinatesLandmark[1]
                    #print("thumb",thumbfingertip_x)
```

```
                except:
                    pass

                try:
                    #pyautogui.moveTo(indexfingertip_x,indexfingertip_y)
                    Distance_x= sqrt((indexfingertip_x-thumbfingertip_x)**2 +
(indexfingertip_x-thumbfingertip_x)**2)
                    Distance_y= sqrt((indexfingertip_y-thumbfingertip_y)**2 +
(indexfingertip_y-thumbfingertip_y)**2)
                    if Distance_x<5 or Distance_x<-5:
                        if Distance_y<5 or Distance_y<-5:
                            click=click+1
                            if click%5==0:
                                print("single click")
                                pyautogui.click()

                except:
                    pass
 #Shows image with hand tracking
        cv2.imshow('Hand Tracking', image)

        if cv2.waitKey(10) & 0xFF == ord('q'):
            break

video.release()
```

Code Snippet 7: Simulation of Brand Detection

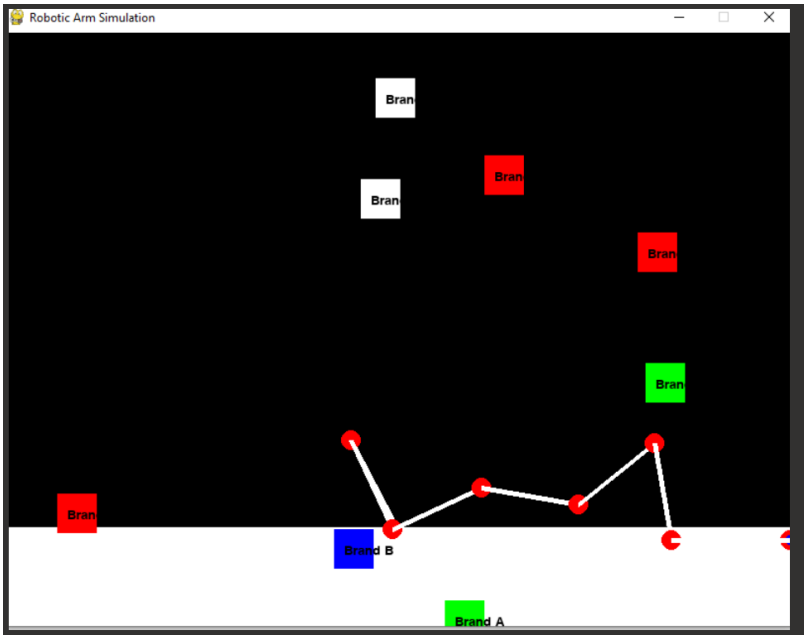


Image 54: Simulation of Brand Detection

In this simulation we have different brands such as brand A, B and C we also have different colours such as green, blue red. The image above shows blocks coming down labelled with different brands and colours, the image also contains a robotic arm with red dots representing servo motors to move the arm. The simulation shows how the robotic arm moves towards the different brands to sort it out based on brands and colour. The simulation is a representation of our prototype system on how it works to be able to detect the different colours and classify the object based on colour.

System and Software Integration

The integration of the hardware and software components is implied in the code but not explicitly provided. The code assumes the webcam is connected to the system, and it uses the video capture from the webcam to process frames and perform color detection.

The code also assumes the Arduino board is connected via the serial port (COM11). It establishes a serial connection using serial.Serial and sends signals to the Arduino using the send_signal_to_arduino function.

The integration process would involve connecting the webcam to the system and ensuring the proper functioning of the video capture. Additionally, the Arduino board should be connected to the system, and the appropriate drivers and libraries should be installed. The code assumes the Arduino is programmed to receive and process the position signals sent from the software.

Power Supply Testing with Arduino

During the development of the system, we investigated different power supply alternatives and examined the servos' current, and voltage needs in our system.

We connect all seven servo signals to the Arduino and power the servos using the Arduino's power output (from the 5V pin), the available current will be shared among all the servos. The Arduino's 5V pin can provide a limited amount of current, typically around 0.5A in total. This means that the total current consumed by all the servos should not exceed this limit to avoid potential issues with the Arduino board.

In our case, the available current was distributed among the servos based on their power requirements and the overall load. The actual current drawn by each servo will depend on factors such as its operating conditions, mechanical load, and other specifications.

Since the total current available from the Arduino's 5V pin is limited, it is crucial to ensure that the combined current draw of all the servos connected does not exceed the Arduino's maximum capacity (around 500mA).

Since the total current consumption exceeded the Arduino's capability, it led to unstable servo behaviour, erratic operation, or even potential damage to the Arduino board.

Each servo is estimated to draw about 83.33mA, and we have a total of six servo motors connected to the Arduino which combined current would be 500mA.

$$500mA = 83.33mA \cdot 6$$

To determine the total amperage required for our robot arm with three SG90 and four MG996 servo motors, we needed to consider the current draw of each servo motor.

The current draw of a servo motor varies depending on the load and the movement being performed. However, we can estimate an approximate current range for the SG90 and MG996 servos.

The SG90 servo motor typically draws around 150-250mA (0.15-0.25A) when operating under a load. We assume an average current draw of 200mA for each SG90 servo motor.

The MG996 servo motor, being a high-torque servo, can draw higher currents, especially when operating under heavy loads or performing quick movements. It typically draws around 500-900mA (0.5-0.9A). We assume an average current draw of 700mA for each MG996 servo motor.

Calculating the estimated total voltage and current:

Voltage: Since both the SG90 and MG996 servos have an operating voltage range of 4.8V to 7.2V. A common choice is 5V, as it falls within the specified operating range of both servo types.

Current: To determine the required current capacity of the power supply, we considered the maximum torque current draw of the servos. Assuming the maximum torque current required for the SG90 servo is 0.2A and for the MG996 servo is 0.7A, we calculated the total estimated current draw as follows:

$$
\begin{aligned}
total_{current} &= (num_{SG90} \times current_{SG90}) + (num_{MG996} \times current_{MG996}) \\
&= (4 \times 0.2A) + (3 \times 0.7A) \\
&= 2.9A
\end{aligned}
$$

This means the power supply needs to be capable of providing around 2.9A (or more) to operate the robot arm with all the servo motors under high torque conditions.

We determined that a 5A constant current is suitable for powering the servos. With a 5A current rating, it provides ample capacity to handle the estimated total current draw of the servos, which is 2.9A. This will ensure that the servos receive the necessary current to operate properly, even under heavy load or high torque conditions.

Using a 5A constant current driver provides a significant margin of safety and allows for stable and reliable operation of the servos. It ensures that the power supply can handle the peak current demands and fluctuations that may occur during servo movements.

DC-DC XL6009 Auto Boost/Buck

The DC-DC XL6009 Auto Boost/Buck Module is a versatile module that can both step up (boost) and step down (buck) voltage. It has an adjustable output voltage range, making it suitable for various applications. Using this module, we can step down the voltage from 14V to 5V. By adjusting the module's output voltage to the desired 5V, we can obtain a stable 5V output from the module. The DC-DC XL6009 module has a maximum current rating, typically around 4A or 5A, depending on the specific variant being used.

Components:

Image 55: 12volt battery 2A

Image 56: 220-volt AC to 12DC and switch

Image 57: 12-volt 5A PSU limited to change Amps to max of 1.49

Image 58: 12-volt 5A PSU

Image 59: 12-volt 3A PSU

Image 60: 12- volt 1.5A PSU

Image 61: Boost Buck Module

Image 62: 5A constant current driver

Image 63: Voltage regulator 3A

Methods

It is important to recognize that voltage and current play different roles in the performance of servo motors. Where voltage affects the speed of a servo, current affects the maximum torque a servo reaches. From this understanding we looked into performance of the servos and deciding on whether to connect them in parallel or series.

In a series connection is understood that voltage supply between servos are sufficient and increases as opposed to parallel connections, however, the current draw remains the same. In a parallel connections we observed that voltage remains the same, however, our current supply to each servo motor is increased. Therefore, we had decided to connect our servos via a prototype board in parallel.

Due the fact that when you connect the positive and negative terminals of the power supply in parallel, each servo will have access to the same voltage and current from the power supply. This configuration ensures that the voltage and current available to each servo are equal.

Note:

Our Amps output is dependent on the PSU. We have tested 5 PSUs with different AMPs the methods below have been tested with all 5 PSUs and two of them given us the highest Amps from 1,50 – 1,41 Amps.

1st method:

We connected the 220-volt AC to 12DC and switch where the input is from a battery charger to the 220volt wall socket, this is then stepped down to 12 DC and switched to store power in a 12V battery. When power is out the system switches to the battery. This is then connected to the voltage regulator (3Amps) where the voltage is now stepped down to 5 volts. Then this connected to the 5A constant driver where voltage and current can be controlled by limiting it. This this goes to the protype board's positive and negative terminals.

2nd method:

We connected the 220-volt AC to 12DC and switch where the input is from a battery charger to the 220volt wall socket, this is then stepped down to 12 DC and switched to store power in a 12 battery. When power is out the system uses the battery. This is then connected to the voltage regulator boost buck where the voltage is now stepped down to 5 volts. Then this then connected to the 5A constant driver where voltage and current can be controlled by limiting it. This this goes to the protype board's positive and negative terminals.

3rd method:

We connected the 220-volt AC to 12DC and switch where the input is from a battery charger to the 220volt wall socket, this is then stepped down to 12 DC. Then this then connected to the 5A constant driver where voltage and current can be controlled by limiting it. This this goes to the protype board positive and negative.

Findings and observations

We connected power (5V) to an SG90 servo without providing any control signal, the servo remained in its default or neutral position.

When a servo receives power but no control signal, it typically does not have any specific instructions on how to position itself. As a result, it will tend to hold its neutral position, which is typically around the midpoint of its range (approximately 90 degrees for a 180-degree servo like the SG90).

Without a control signal, the servo will not respond to any movement commands or positional changes. It will stay stationary until it receives appropriate control signals that instruct it to move to a different position.

- The input voltage given to the system can be controlled however the amps given in the system at the start determine how many amps are in the system this current starts from the PSU whatever the current is at that point goes through the entire system.
- If we want more amps, we need a PSU that can supply that amount as our system's max torque requires 2,9Amps for all the servos motors.
- The SG90 servo motor typically draws around 150-250mA (0.15-0.25A) when operating under a load. While the MG996 servo motor. Typically draws around 500-900mA (0.5-0.9A).

- In theory SG90(0.15-0.25A) and MG996 (0.5-0.9A) servo motors can have constant of 4-5 volts and Amps of 1,50 then system will run and perform as the Arduino did run the system on 0.5 Amps/500mA.
- Using a 5A PSU of 1.5A allowed us to provide three times as much power to the servos than the Arduino and if we wanted to utilize maximum torque then we had to double the Amperage to 2.9A.
- In conclusion currently the external power source for the robot arm we built is should run the servos at half of its maximum rate.

It is important to note that since the servos have their own external power supply, we need to attach a ground to the external source to the ground on Arduino. The soldering can affect the flow of current through the system and the thickness of the cables allow for the max amps to pass through preferred is a 1,5mm single core.

New Robotic arm schematic

This is the new schematic for the robotic arm, we had to make changes to the schematic due to the implications discussed above. We added an external power supply that will power the robotic arm servo motors with 5 V and 6 Amps this in turn gave the servo motor enough power and torque to be able to pick up blocks smoothly and a coke glass bottle for the max weight currently. We have connected our servos in parallel to increase the amps and to feed all our positive and negatives on two lines that are being fed from the external power supply.

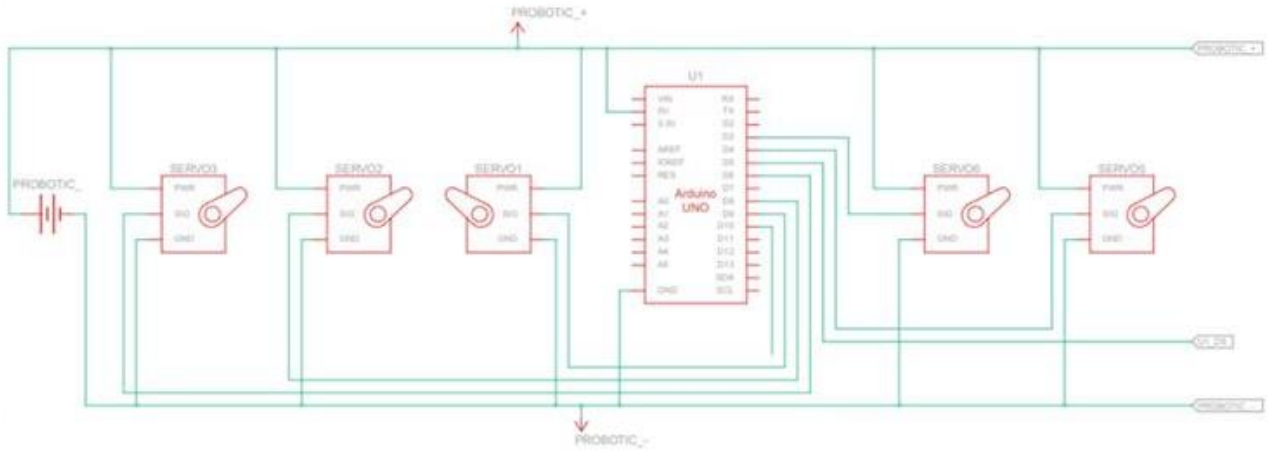


Image 64: New Robotic Arm Schematic

New conveyor belt schematic

This is our new circuit diagram for the conveyor belt, we noticed that the conveyor belt was inconsistent in timing as it would sometimes suddenly speed up and other times it would hardly move then we noticed that we were not providing the conveyor belt with enough voltage our amps, our motor is able to take 12V and 11A so we got a power supply that would provide the motor with the required voltage as well as the required amps. We noticed that our motor was now able to move about 8 kg easily and it was more consistent. This is a simulation of our circuit diagram for the servo motors as well as the conveyor motor.

Image 65: New conveyor belt schematic

The image below shows the simulation developed using Tinkercad.

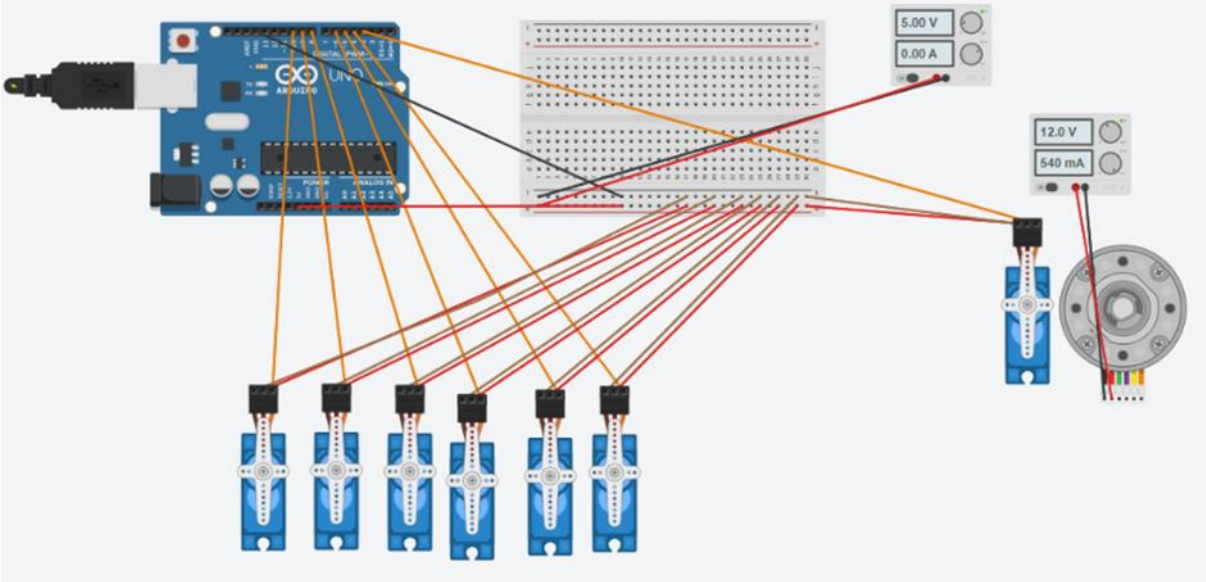


Image 66: Simulation using Tinkercad

Performance Testing: Robotic Arm

Stress testing

A stress test was conducted to test the durability of the robotic arm under abnormal circumstances to see what the max operating weight for the robotic arm as well the operating time based as how long the robotic can operate.

A Stress test for a mechanical arm with six servos would regularly include subjecting the arm to different challenging conditions to survey its execution, toughness, and unwavering quality. Here's a layout of a push test strategy for a six-servo mechanical arm: With the arm at 5V and 6amps it is at maximum torque for the small SG 90s and MG99R6

Initially the arm had been powered by the Arduino alone and we were not receiving enough power to pick up a sponge—as a test object—with ease. We also found jittering movements and instability on the arm. We later realized that we were not giving the system enough power to work with, we were not giving the servo motors the amps and volts needed to increase the torque of the robotic arm. Once we added an external power supply that provided the correct amps and voltage that the servo motors required, we found a significant improvement on the stability of the robotic arm and the weight we were able to pick up. We tested the max weight that the robotic arm can pick up safely we can say that the max weight right now would be ±400 grams.

If we had to improve the frame of the robotic arm and pump up the voltage to the max voltage and amps that the servo motors can take than we would probably be able to pick up heavier objects.

Stack testing

The arm is tested to decide its greatest load-bearing capacity. By slowly incrementing the weight being lifted by the arm until it comes to its stop. This test makes a difference in distinguishing the most extreme weight the arm can handle without compromising its basic judgment. This it can pick a maximum of a glass coke coco cola bottle of 400grams. Where the small SG90s struggle, the MG99R6 servo motors can move it with ease.

Speed testing

The arm's reaction time and speed by commanding it to perform at diverse speeds. This test its capacity to move easily and precisely at both moderate and quick speeds. This makes a difference assess the arm's control framework and its capacity to handle real-time developments.

The arm performs well at slower speeds set to 15, however we can make the robot move much faster depending on the speed given however it is not so accurate being fast due to limitations of the design.

It is also not recommended to operate at higher speeds as it would increase the wear and tear occurring on the frame.

Perseverance Testing:

This test runs the mechanical arm persistently for an expanded period, recreating a real-world implementation of a system like this. This test points to decide on the off chance that the arm can work without its processor overheating or encountering execution corruption over time.

After long periods of time, it becomes inaccurate and moves to places that it was not supposed to, as it runs repeatably for a long time where it does tend to warm up the servos depending on the task to pick up heavy weights.

The issue that we encountered with testing over long periods of time is that our frame for the 3d arm is not the most suitable it would have to be investigated as the base keeps breaking and cracking. It would be ideal to redesign the frame with stronger components that are light yet more durable.

Collision Testing:

This test in collisions or deterrents within the arm's way whereas it's in operation. This test assesses the arm's capacity to identify and react to unforeseen circumstances, such as halting or altering its way when experiencing an obstacle.

The arm is set to pick up at a set position so if there is a collision, it does not know about it as it is not programmed to take that into consideration.

Precision Testing:

The arm performs the exact developments, such as picking up objects of shifting sizes or setting them areas. At a certain degree the arm's precision and repeatability in performing these assignments, guaranteeing it can accomplish the specified level of accuracy.

We can guarantee a 99% pick up and shifting if location or different servo angles are the robot takes instructions very well, it performs as given task according to the array servo degrees.

It moves to the areas it is given we high torque and accuracy.

The conveyor belt would have to be redesigned or worked on as the motor of the conveyor is not fully stable so we may experience inconstancy on the speed from time to time.

Natural Testing:

This arm’s pick-up and drop-off functions were tested in diverse natural conditions, such as temperature extremes. This test makes a difference in the arm's execution and unwavering quality beneath different conditions that it may experience in real-world applications.

For this we tested different objects such as wooden objects, sponges, glass, and thin lights, steel boxes. It picks and moves with ease unless it is beyond its maximum pick-up weight.

We have not tested over extreme conditions as the arm build is fragile due to the 3D printing filament.

This system is recommended to be set in an indoor environment as there is not protection from external conditions such as rain and sand.

Stretch Testing:

The stretch test refers to the is the mechanical arm’s load that go past its typical working run. This test makes a difference in recognizing the arm's breaking point or any potential shortcomings in its structure or servos.

This was done and the outcomes will well be presented as we tried the different objects such as wooden objects, sponges, glass, and thin lights, steal boxes. Which it picks and moves. Certain weights do strip the

gears on the small servos however the big ones can handle the weight as we have 3 of each, this system is designed for small simple objects.

Performance Testing: Conveyor Belt

Stress test

A Stress test for a Conveyor belt is outlined to assess its execution and strength beneath challenging conditions. It is a 12V DC motor with 11Amps.

Stack Capacity Testing:

The Conveyor belt is subjected to stack levels to decide its most extreme stack capacity. Slowly incrementing the weight being transported on the belt until it comes to complete stop. This test makes a difference in distinguishing the most extreme weight the Conveyor belt can handle without stopping.

We have testing with real life objects such as a 5kg bag of putty and a G-clamp of 1kg, this as speed of 40RPM still moves with ease. We can add more weight such as and additional 2kg bag of onions and it will still move just a bit slower. The motor experiences a dead stop with a SUV Battery of 20Kg at 40RPM.

The conveyor can handle a max load with ease up to 8kg anything beyond would require increasing up to the max speed and increase the voltage as well as the amps to the max to move up to 10 kg with ease.

We operate the conveyor at a continuous run for about 2 minutes straight without stopping and we identified that the potentiometer begins to heat and smoke, as this is a prototype it is a natural outcome as we are working with a potentiometer that cannot handle working with high current over a long period of time, so the components start to heat up around 2 minutes.

Speed and Proficiency Testing:

The Conveyor belt's reaction time and effectiveness by running it at diverse speeds. This test its capacity to move easily and precisely speeds. This makes a difference in the Conveyor belt’s control framework and its capability to handle real-time fabric taking care of operations.

Start and Stop is accurate to a point due to certain mechanical issues it is not 100% accurate, however with the servo motor controlling the movement at 155 Degrees which allows us to control it well for the system as it shows to be the most effective.

Under long and continuously running the belt with heavy loads at speed 40 – 100 we experienced heating of the PWM speed controller.

Life span and Durability Testing:

We have conducted long-term testing to assess the conveyor belt’s solidity and durability by observing the belt's execution over an amplified period, watching any signs of debasement, or untimely wear. This test helps assess the belt's life expectancy and its capacity to preserve steady execution over time.

Artificial Intelligence and Machine Learning

The colour detection and robotic arm movements do not involve AI or machine learning techniques. It uses computer vision techniques to detect colours based on predefined colour bounds. The colour detection process is implemented by converting the frame from the webcam into the HSV colour space and applying thresholding operations to identify specific colours within the given ranges. The detection is rule-based and does not involve any learning or training of models.

Training and Learning:

The colour detection and robotic arm movements programs do not include any training or learning processes. The colour bounds and positions for the robotic arm are predefined within the code. If you want to improve the colour detection capabilities or perform more complex tasks, you might consider using machine learning techniques. For instance, you could collect a dataset of labelled colour samples, train a

machine learning model (such as a neural network) on this data, and then use the trained model for colour recognition. However, the given code does not incorporate such training or learning processes.

Budget

In this section we briefly go over the budget of this project. The budgets are drawn up in the below tables which includes a budget for the project overall, another which excludes a conveyor belt as well as any other necessary tools used in the development of the physical system.

Budget with motor and conveyor included		
Item	**Quantity**	**Price**
100 A PWM Motor speed controller	1	R 653,02
Arduino uno R3	2	R 480
ESP32 with Cam OV2640 Dev Board	1	R 209.53
Motor 12 V		
Conveyor belt +Guiders	1	R 7500
PWM DC Motor Speed Controller 8A With Display	1	R 171.35
MG996 Servo	4	R 325.16
40A DC Motor Speed Controller PWM	1	R 366.99
L28p Arduino Dual motor driver	1	R 199
Webcam	1	R 500
Stainless steel Table	1	R2540
Stainless steel Frame	1	R1000
Power supply 12V 12A	1	R428.00
Power Supply 5V 6A	1	R128.00
	Total Price	**R 14164,06**

Table 4: Budget including all components

Budget excluding conveyor		
Item	**Amount**	**Price**
100 A PWM Motor speed controller	1	R 653,02
Arduino uno R3	2	R 480
ESP32 with Cam OV2640 Dev Board	1	R 209.53
Motor 12 V	1	R700
Conveyor belt +Guiders	1	
PWM DC Motor Speed Controller 8A With Display	1	R 171.35
MG996 Servo	4	R 325.16
40A DC Motor Speed Controller PWM	1	R 366.99
L28p Arduino Dual motor driver	1	R 199
Webcam	1	R 500
Stain a steel Table	1	R2540
Stain a steel Frame	1	R1000
Power supply 12V 12A	1	R428.00
Power supply 5V 6A	1	R128.00
	Total Price:	**R 6134,00**

Table 5: Budget excluding conveyor belt

Tools budget		
Item	**Quantity**	**Price**
Universal prototyping PCB 70x90mm - Single sided	2	R 62.10
Female to Male Jumper Cable Dupont 30cm	3	R 196.32
Tin-Plated Fly Jumper Wire Cable 24AWG 20CM 100pcs	1	R 57.50
NASCO 2mm Copper Solder Wire	1	R249
NASCO 2mm Copper Solder Wire	1	R350
Filament Wanhao	1kg	R 299

6m wood 30mm by 40mm		R200
	Total Price:	**R 1386,92**

Table 6: Tools budget

Project Schedule and Tasks

This section gives a thorough summary of the team's tasks and the project schedule. This section highlights the progress made in completing various project milestones while outlining the timetable and duties for each activity. The timetable displays the team's planned strategy for ensuring the project's successful completion. This part sheds light on the project's development and the team members' contributions by providing the tasks in a logical order and noting their current state.

Project Schedule

- **Research Phase:**
  - Task 1: Research and gather reference designs for waste management solutions.
  - Task 2: Evaluate the advantages and disadvantages of existing designs.
  - Task 3: Conduct a literature review for further research.
- **Design Phase:**
  - Task 4: Develop and discuss prototype designs within the group.
  - Task 5: Select the main prototype design.
  - Task 6: Finalize the design for the chosen solution.
- **Data Collection Phase:**
  - Task 7: Create a questionnaire and identify companies and target audience for data collection.
  - Task 8: Collect and compile data from interviews and surveys.
- **Component Procurement and Preparation:**
  - Task 9: Procure necessary components for the robotic arm and conveyor.
  - Task 10: 3D print and assemble the robotic arm.
  - Task 11: Modify and test the conveyor belt.
- **Algorithm Development:**
  - Task 12: Research and select appropriate algorithms for object detection.
  - Task 13: Train and refine the algorithm using a dataset.
  - Task 14: Finalize the code for the algorithm.
- **Robotic Arm Implementation:**
  - Task 15: Code the robotic arm movements and integrate it with the algorithm.
  - Task 16: Set up camera linkage and test its functionality.
  - Task 17: Program the arm to sort objects into designated containers.
- **Documentation and Review:**
  - Task 18: Document the entire process, including development, testing, and implementation phases.
  - Task 19: Review and refine the documentation based on feedback.

Tasks

Research (Whole Group) - 2 weeks (Completed)
• Investigating the chosen problem • Understanding the impact of the problem on people • Researching existing solutions to the litter problem • Examining statistics and the overall impact of litter • Design and Create a Layout (Whole Group) - 3-4 weeks (Completed)
Design and Create a Layout (Whole Group) - 3-4 weeks (Completed)
• Reviewing designs of other solutions for reference • Listing advantages and disadvantages of existing designs • Creating prototype designs and discussing them • Comparing prototype designs and selecting the main design • Documenting ideas, incorporating feedback, and conducting further research on the literature review • Designing multiple solutions and evaluating their pros and cons • Finalizing the design of the chosen solution
Survey Research and Interviews (Mackyle and Agustin) - 3 days (Completed)
• Designing a questionnaire • Identifying companies and target audience for interviews • Collecting data through surveys and interviews • Incorporating the gathered data into the documentation
3D Printing Robotic Arm and Assemble (Shayur) - 1 week (Completed)
• Procuring filament, screws, and servo motors for the robotic arm • 3D printing the robotic arm based on the design • Checking the fitting of all printed pieces • Assembling the robotic arm and attaching servo motors
Buying Parts (Mackyle and Agustin) - 2 days (Completed)
• Researching necessary components • Setting a budget limit • Identifying reliable sources for purchasing the required parts • Procuring and testing the components
Conveyor Belt Prototype (Mackyle and Agustin) - 4 days (Completed)
• Researching conveyor belt options • Acquiring a functional conveyor belt • Analysing the components and modifying the belt • Adding a frame and testing connections • Replacing the motor and incorporating a speed control unit • Testing the functionality of the conveyor belt prototype
Algorithm Unit (Ruan) - 3 weeks (Completed)
• Researching different algorithms • Evaluating algorithm options and packages • Implementing an appropriate algorithm in a small project for testing • Loading relevant litter images for training • Training and testing the algorithm • Retraining and retesting the algorithm as needed
Robotic Arm Prototype Movement (Terry, Shayur) - 2 weeks (Completed)
• Researching different robotic arm movements • Acquiring final components, including Arduino Uno • Assembling and configuring the robotic arm • Coding the arm's axis and movements (e.g., open/close of end effectors, pickup motion)
Robotic Arm and Camera Linkage (Terry, Shayur, Ruan) - 1 week (Not accomplished)
• Researching camera linkage with Arduino

- Procuring necessary components, such as ESP32 CAM
- Integrating the camera with the robotic arm system
- Reviewing and modifying the code for arm movements
- Ensuring proper execution of the algorithm on the arm
- Sorting objects based on the algorithm's output

Train Model to Detect Different Colours (Mackyle) – 1 week (Completed)

- Developing a model to detect different colours
- Sending signals to Arduino based on color detection
- Executing robotic arm movements based on classified colours
- Testing basic movements and sequencing
- Dropping objects according to their detected colours

Program the Robotic Arm to Move to Set Sequences (Mackyle and Agustin) - Success

- Determining appropriate angles for arm movement
- Programming the arm to follow instructions upon color classification
- Organizing and labelling connections for easier troubleshooting
- Verifying correct functioning of the robotic arm code

Conveyor Belt Test (Zara, Mackyle, Agustin) - Success - 1 week (Completed)

- Acquiring necessary components
- Researching and testing connection setup
- Ensuring proper operation of the conveyor belt
- Connecting the driver unit to regulate and stop the motor
- Customizing the conveyor belt to support the robotic arm
- Painting the conveyor belt and adding guides
- Verifying the conveyor belt's functionality

Join the Robotic Arm System to the Conveyor Belt System (Mackyle and Agustin) - 1-2 weeks

- Testing the separate systems individually
- Researching methods to integrate the two systems
- Procuring any additional components needed
- Establishing the connection between the two systems
- Verifying proper functionality and retraining the model for the conveyor belt
- Fine-tuning the robotic arm's performance
- Enabling conveyor belt stoppage upon litter detection and resuming after arm returns to the origin
- Add Final Touches – Ongoing
- Managing cables and securing connections
- Implementing a PVC box for added protection
- 3D printing spare parts for emergency situations

Add Final Details and Features (Whole Group) – 1 Week

- Incorporating a web module and LCD screen
- Fine-tuning the entire system
- Addressing any remaining errors and glitches
- Conducting thorough testing and ensuring smooth operation

Prepare Documentation (Whole Group) - 4 days

- Adding schematics based on the system design
- Incorporating any missing data into the documentation
- Ensuring the documentation is up to date and accurate
- Seeking feedback from a lecturer to validate the documentation
- Making necessary corrections and additions

Presentation (Whole Group) - 1 week

- Assigning roles for the presentation
- Creating PowerPoint slides
- Conducting a final system check
- Resolving any last-minute issues
- Ensuring the completeness of documentation

- Planning the presentation and demonstration
- Practicing and delivering the presentation

Chapter 5: Results, Conclusion, and Recommendations

In this section we will look at the results of the system itself, we will also discuss the results of the system and how we could improve the system and the recommendations we would make if we would have to redesign the system and how we would tackle it. We will also talk about the closing statement based on our observations.

Results

We are still working on the system currently but the current result we have right now is that the conveyor belt is running but due to the brackets not being stable and other factors such as the belt still needs fine tuning we can determine that the speed is not constant as yet, when we start the system the speed will suddenly be a bit too slow or the speed will fluctuate and increase suddenly and be regulated back to the desired speed that we want.

The robotic arm can pick up objects, but they need to be light, to overcome this problem we need to add a more stable end effectors so that it is able to grip heavy objects. The web cam can pick up the colours and the Arduino tells us what colour it detects, and it proceeds to give the robotic arm instructions as where it needs to drop off the object. We managed to get to achieve the objective to be able to identify a colour and be able to pick up the colour and drop off the object based on the colour it was classified as.

Recommendations

Based on the results that we currently have and that we have discussed above we would recommend if we had a bit more time as well as the budget we would add more brackets to secure the system from moving from the torque of the motor, we would also advice on getting a new or building a new conveyor belt as our belt is giving us too many issues which causes fluctuation of the conveyor speed which then has to be regulated by the stepper unit itself.

We also have made the system to be a more complex to meet the complexity standard level of this project, instead of using a robotic arm to sort out the litter we would instead use a solenoid system which would increase the speed of sorting instead of using robotic arms which needs to grip the object and move it carefully to the specified place. If this option was not viable then we would increase the number of robotic arms that are implemented to the system to increase the pickup rate of the robotic arm to the litter that is being run on the conveyor belt. We would also change the material of the robotic arm to be more stable and durable, as is we have run multiple tests with our robotic arm and some components have started to wear and tear and break so we would choose a more durable material.

We would also change the motors that are being used to be able to pick up heavier objects as is our robotic arm faces the issues that it is not able to pick up certain objects as it does not have enough torque power at the base and shoulder area which causes the robotic arm to not be able to pick up the robotic arm itself. These are the recommendations we would take and advice on.

Conclusion

In conclusion by doing research on the problem we have identified that litter problem is a huge problem all over the world, we however will tackle the litter problem in South Africa first, but we have looked at potential solutions that other companies from a global scale have produced. By looking at the problem the environment is in, and some potential solutions other companies have produced. We of the Bedfordview campus have come to a potential solution to the problem, we will build a prototype of a conveyor belt system with two robotic arms, these will be two different systems and we will train to implement all the features discussed and make sure that both systems are working as they should.

After we are happy that the system is working, we will try to join both systems so that they are able to work as one system. The conveyor belt will contain the litter and move it towards the robotic arms, the robotic arms will then detect the litter with the camera and indemnify the type of litter through machine learning using OpenCV as well as TensorFlow and feeding it the dataset. Once the litter has been identified the robotic arm

will then proceed to pick up the litter off the conveyor belt and move it towards the categorized container for that type of litter, it will then proceed to go to the initial state and wait to identify the next litter and proceed in this manner.

There is much areas that have room for improvement but overall we have managed to achieve out main goals that we have specified and wanted to achieve, had we had a bit more time we would have like to improve the system itself a bit more and had we had financial support from our University we would have had a bit more budget to work with to increase the durability and certain components in order to enhance our system even more to actually be able to make a more solid impact to our specified problem that our team had set out to achieve.

References

2023/2024. (2022). Bottle Recycling Prices in South Africa - 2023/2024. [online] Available at: https://safacts.co.za/bottle-recycling-prices-in-south-africa/ [Accessed 17 Apr. 2023].

2023/2024. (2022). Paper Recycling Prices in South Africa - 2023/2024. [online] Available at: https://safacts.co.za/paper-recycling-prices-in-south-africa/ [Accessed 17 Apr. 2023].

3SMedia (2022). Challenges within the South African recycling industry | Infrastructure news. [online] Infrastructure news. Available at: https://infrastructurenews.co.za/2022/01/26/challenges-within-the-south-african-recycling-industry/ [Accessed 17 Apr. 2023].

Almanzor, E., Anvo, N.R., Thuruthel, T.G. and Iida, F. (2022). Autonomous detection and sorting of litter using deep learning and soft robotic grippers. *Frontiers in Robotics and AI*, [online] 9. Available at: https://www.frontiersin.org/articles/10.3389/frobt.2022.1064853/full [Accessed 19 Mar. 2023].

Aluminum.org. (2021). *Aluminum Sustainability | The Aluminum Association*. [online] Available at: https://www.aluminum.org/sustainability#:~:text=A%20highly%20durable%20metal%2C%20aluminum,to%2065%25%20less%20than%20steel. [Accessed 7 Jun. 2023].

Available at: https://businesstech.co.za/news/trending/123769/how-much-rubbish-you-dump-in-south-african-each-day/
[Accessed 18 March 2023].

Available at: https://chariotenergy.com/blog/how-long-until-plastic-decomposes/
[Accessed 18 03 2023].

Available at: https://theconversation.com/south-african-study-highlights-growing-number-of-landfill-sites-and-health-risks-141890
[Accessed 23 03 2023].

Available at: https://www.conserve-energy-future.com/advantages-disadvantages-landfills.php
[Accessed 23 03 2023].

Available at: https://www.statssa.gov.za/?p=11527
[Accessed 18 March 2023].

BERNIER, C., 2022. *Robotic Arms: Different Types and When to Use Them.* [Online]
Available at: https://howtorobot.com/expert-insight/robotic-arms
[Accessed 28 05 2023].

BERNIER, C., 2022. *SCARA Robots: Guide to The Most Versatile and Sought After Robot.* [Online]
Available at: https://howtorobot.com/expert-insight/scara-robots
[Accessed 28 05 2023].

Bine.world. (2023). Bin-e | Smart Waste Bin. [online] Available at: https://bine.world/ [Accessed 19 Mar. 2023].

conserv, n.d. *Advantages and Disadvantages of Landfills.* [Online]
Available at: https://www.conserve-energy-future.com/advantages-disadvantages-landfills.php
[Accessed 23 03 2023].

conserv, n.d. Advantages and Disadvantages of Landfills. [Online]

Dffe.gov.za. (2022). Working on Waste | Department of Environmental Affairs. [online] Available at: https://www.dffe.gov.za/projectsprogrammes/workingonwaste [Accessed 17 Apr. 2023].

ECOFLOW (2023). How Does Load-Shedding Affect the Community? [online] EcoFlow. Available at: https://blog.ecoflow.com/za/how-does-load-shedding-affect-community/#:~:text=Security%20%E2%80%93%20The%20absence%20of%20electricity,Without%20electricity%2C%20they%20cannot. [Accessed 7 Jun. 2023].

Energy, C., 2021. How long until plastic decomposes. [Online]

Energy, C., 2021. *How long until plastic decomposes.* [Online]
Available at: https://chariotenergy.com/blog/how-long-until-plastic-decomposes/
[Accessed 18 03 2023].

Ferguson, M., 2014. *ResearchGate.* [Online]
Available at: https://www.researchgate.net/figure/Fig-A1-The-standard-VGG-16-network-architecture-as-proposed-in-32-Note-that-only_fig3_322512435
[Accessed 20 April 2023].

He, K., Zhang, X., Ren, S. & Sun, J., 2015. *Cornell University.* [Online]
Available at: https://arxiv.org/abs/1512.03385
[Accessed 20 April 2023].

Kiger, P.J. (2017). Robotic Sorting Could Be the Efficient Future of Recycling. [online] HowStuffWorks. Available at: https://science.howstuffworks.com/environmental/green-tech/sustainable/efficient-recycling-robotic-sorting-artificial-intelligence.htm [Accessed 19 Mar. 2023].

Leobot Electronics (2019). *ESP32-CAM WiFi + bluetooth Camera Module Development*. [online] Leobot.net. Available at:
https://leobot.net/viewproduct.aspx?id=5163&gclid=CjwKCAjw1YCkBhAOEiwA5aN4AR27HkHOOgbMQge4WMNuBHiKamoiTyL0d0mjB8h9QeX94AHyQVBcmhoCUZQQAvD_BwE [Accessed 7 Jun. 2023].

Li, F.-F., 2006. *IMAGENET.* [Online]
Available at: https://www.image-net.org/challenges/LSVRC/
[Accessed 20 April 2023].

Lihi Gur Arie, P., 2022. *Color Segmentation with K-means Clustering.* [Online]
Available at: https://towardsdatascience.com/image-color-segmentation-by-k-means-clustering-algorithm-5792e563f26e
[Accessed 20 05 2023].

Linkedin.com. (2023). LinkedIn. [online] Available at: https://www.linkedin.com/pulse/robot-pick-garbage-ramudroid-v8-altanai-bisht/ [Accessed 19 Mar. 2023].

Liu, C. et al., 2018. *Cornell University.* [Online]
Available at: https://arxiv.org/abs/1712.00559
[Accessed 20 April 2023].

Mahmood, A., Giraldo-Ospina, A., Bennamoun, M. & A Kendrick, G., n.d. [Online].

Malibongwe Tyilo (2019). Maverick Life: Recycling in South Africa, how are we doing? A very practical guide to getting started. [online] Daily Maverick. Available at: https://www.dailymaverick.co.za/article/2019-07-20-recycling-in-south-africa-how-are-we-doing-a-very-practical-guide-to-getting-started/ [Accessed 17 Apr. 2023].

Market Trends (2022). *Top 100 Robotics Projects for Engineering Students*. [online] Analytics Insight. Available at: https://www.analyticsinsight.net/top-100-robotics-projects-for-engineering-students/ [Accessed 18 Mar. 2023].

Mpact (2023). Mpact. [online] Google.com. Available at:
https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwj754Cf4Ln-AhXEYcAKHT27DdMQFnoECAsQAQ&url=https%3A%2F%2Fwww.mpact.co.za%2Four-products%2Fmpact-brochure.pdf&usg=AOvVaw2RQiSc8ru58gjvln62Wt3I [Accessed 21 Apr. 2023].

Mushiri, T. and Emmison Gocheki (2020). *Design of a Garbage Collection Robot*. [online] ResearchGate. Available at:
https://www.researchgate.net/publication/338320769_Design_of_a_Garbage_Collection_Robot [Accessed 19 Mar. 2023].

Sabre Security Mart. (2023). Power Adapter-220V to 12V DC 3A. [online] Available at: https://securitymart.co.za/products/power-adapter-220v-to-12v [Accessed 7 Jun. 2023].

Simonyan, K. & Zisserman, A., 2014. *Cornell University.* [Online]
Available at: https://arxiv.org/abs/1409.1556
[Accessed 20 April 2023].

stats sa, 2022. stats sa. [Online]

stats sa, 2022. *stats sa.* [Online]
Available at: https://www.statssa.gov.za/?p=11527
[Accessed 18 March 2023].

Stephan (2019). Home | Mpact Recycling. [online] Mpactrecycling.co.za. Available at: https://www.mpactrecycling.co.za/ [Accessed 21 Apr. 2023].

Takealot.com. (2023). Takealot. [online] Available at: https://www.takealot.com/720p-hd-digital-webcam-free-driver-usb-pc-web-cam-computer-camer/PLID91674003 [Accessed 7 Jun. 2023].

Takealot.com. (2023). *Takealot*. [online] Available at: https://www.takealot.com/720p-hd-digital-webcam-free-driver-usb-pc-web-cam-computer-camer/PLID91674003 [Accessed 7 Jun. 2023].

The Coca-Cola Company. (2021). Sustainable Packaging | The Coca-Cola Company. [online] Available at: https://www.coca-colacompany.com/sustainability/packaging-sustainability [Accessed 21 Apr. 2023].

ThemeGrill (2016). *Modular Rack | Delta Power Solution*. [online] POWER AUTOMATION - Power Solutions. Available at: http://deltapowersolutions.co.za/deltapower-products/modular-rack/ [Accessed 7 Jun. 2023].

Tomita, A., 2020. *South African study highlights growing number of landfill sites, and health risks.* [Online]
Available at: https://theconversation.com/south-african-study-highlights-growing-number-of-landfill-sites-and-health-risks-141890
[Accessed 23 03 2023].

Tomita, A., 2020. South African study highlights growing number of landfill sites, and health risks. [Online]

Tsang, S.-H., 2020. *medium.* [Online]
Available at: https://sh-tsang.medium.com/reading-pnasnet-progressive-neural-architecture-search-image-classification-1beb1de06fe6
[Accessed 2023 April 2023].

UNIVERSAL ROBOTS, 2022. [Online]
Available at: https://www.universal-robots.com/in/blog/types-of-robotic-arms/
[Accessed 28 05 2023].

WCCO - CBS Minnesota (2017). Denver Recycling Centre Testing Robotic Sorter. YouTube. Available at: https://www.youtube.com/watch?time_continue=49&v=YCfFFjFkYlc&embeds_euri=https%3A%2F%2Fscience.howstuffworks.com%2F&feature=emb_logo [Accessed 19 Mar. 2023].

World of Bin-e (2020). Bin-e - the smartest waste bin worldwide. YouTube. Available at: https://www.youtube.com/watch?v=LJYWRTRJThY [Accessed 19 Mar. 2023].

Writer, S., 2016. BUSINESSTECH. [Online]

Writer, S., 2016. *BUSINESSTECH.* [Online]
Available at: https://businesstech.co.za/news/trending/123769/how-much-rubbish-you-dump-in-south-african-each-day/
[Accessed 18 March 2023].