DESIGN AND CONSTRUCTION OF A MACHINE LEARNING CLASSIFIER-BASED WASTE RETRIEVAL ROBOT



BY

EBUBE OPARA (ENG1603829)

AND

EGHOSA MICHAEL OSAYANDE (ENG1607933)

SUPERVISED BY

(*PROF PATIENCE ORUKPE*)

CO-SUPERVISED BY

(*ENGR EDOSA OSA*)

DEPARTMENT OF ELECTRICAL ELECTRONICS ENGINEERING

UNIVERSITY OF BENIN

DECEMBER 2022

DESIGN AND CONSTRUCTION OF A MACHINE LEARNING CLASSIFIER-BASED WASTE RETRIEVAL ROBOT



BY

EBUBE OPARA (ENG1603829)

AND

EGHOSA MICHAEL OSAYANDE (ENG1607933)

SUPERVISED BY

(*PROF PATIENCE ORUKPE*)

CO-SUPERVISED BY

(*ENGR EDOSA OSA*)

DEPARTMENT OF ELECTRICAL ELECTRONICS ENGINEERING

UNIVERSITY OF BENIN

DECEMBER 2022

**CERTIFICATION**

This is to certify that this project work was carried out by Ebube Opara ENG1603829 and Michael Eghosa Osayande ENG1607933, students of the DEPARTMENT OF ELECTRICAL ELECTRONICS ENGINEERING, UNIVERSITY OF BENIN, in partial fulfillment of the requirements for the award of the BACHELORS of ENGINEERING DEGREE in ELECTRICAL ELECTRONICS ENGINEERING; under the supervision of *PROF PATIENCE ORUKPE and CO-SUPERVISED BY ENGR EDOSA OSA*

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

***(PROF PATIENCE ORUKPE) (PROF PATIENCE ORUKPE)***

PROJECT SUPERVISOR HEAD OF DEPARTMENT

**DEDICATION**

This work is dedicated to Almighty God for his guidance and protection all through this research work and seeing the completion a successful one. This project is also dedicated to our parents and guardians who gave us the opportunity and encouragement to grow and reach the end of an important phase in our lives.

**ACKNOWLEDGEMENT**

We would like to acknowledge Almighty God who gave us the guidance to choose Electrical/Electronic as a course of study and also see us through right to the very end of this bachelor’s degree.

We would like to say that there are so many people to thank, however, we will like to thank the Electrical/Electronic Engineering department, University of Benin, for providing the platform on which we were engaged with this project.

We want to say a big thank you to our project supervisors, Prof. (Mrs.) Patience E. Orukpe and Engr. E. Osa for their constant support, patience and guidance. A big thank you to the Head of Department (Electrical/Electronic Engineering) Prof. (Mrs.) Patience E. Orukpe.

We are grateful to all staff and students of the department of Electrical/Electronic Engineering for making our period in University of Benin a time worthwhile and a successful one.

We want to thank our families for the encouragement and financial support throughout our time spent in the University of Benin, they encouraged us all through our years of learning and also appreciate our siblings for their care and vital support.

We are highly grateful.

**ABSTRACT**

The need to control and curb the litter of plastic waste, specifically plastic bottles, in the environment has risen with the continuous influx of plastic in the ecosystem.

Their non-biodegradable nature makes it difficult for them to be destructible and harms the environment.

To effectively manage and dispose of the excess plastic in the environment, an effective plastic waste collector is necessary. While developed countries have an efficient system of management, developing and underdeveloped countries struggle with setting up an organized system due to cost.

In this project, we designed a simple and cheap machine-learning classifier waste retrieval robot to help collect and retrieve plastic bottles in their immediate environment.

## TABLE OF CONTENTS

[TABLE OF CONTENTS 7](#_Toc122418003)

[CHAPTER ONE 9](#_Toc122418004)

[INTRODUCTION 9](#_Toc122418005)

[1.1 BACKGROUND OF STUDY 9](#_Toc122418006)

[1.2 AIM 10](#_Toc122418007)

[1.3 PROBLEM STATEMENT 10](#_Toc122418008)

[1.4 OBJECTIVES 10](#_Toc122418009)

[1.5 METHODOLOGY 10](#_Toc122418010)

[1.6 SCOPE OF WORK 11](#_Toc122418011)

[1.7. SIGNIFICANCE OF WORK 11](#_Toc122418012)

[CHAPTER TWO 12](#_Toc122418013)

[LITERATURE REVIEW 12](#_Toc122418014)

[2.1 LITERATURE REVIEW 12](#_Toc122418015)

[**2.1.1. PATTERN RECOGNITION IN IMAGE PROCESSING** 13](#_Toc122418016)

[**2.1.2. IMAGE PROCESSING IN COMPUTER VISION** 15](#_Toc122418017)

[**2.1.3. Open CV** 17](#_Toc122418018)

[2.2 RELATED WORKS 18](#_Toc122418019)

[**Development Of Identification Methods for Cans And Bottles By Irsyadi Rani and Ihsan Budiman** 18](#_Toc122418020)

[**Trash And Recyclable Material Identification Using Convolutional Neural Network By Rumana Sultana** 18](#_Toc122418021)

[**Design And Construction Of A Litter Collecting Robot By Karl Kangul et. Al** 20](#_Toc122418022)

[CHAPTER THREE 21](#_Toc122418023)

[METHODOLOGY AND DESIGN 21](#_Toc122418024)

[3.1 METHODOLOGY 21](#_Toc122418025)

[**3.1.1. DESIGN AND CONSTRUCTION OF THE CHASSIS** 21](#_Toc122418026)

[**3.1.2. THE ELECTRONIC CIRCUIT** 21](#_Toc122418027)

[**PARTS OF THE ELECTRONIC CIRCUIT** 22](#_Toc122418028)

[1. H-Bridge Circuit 22](#_Toc122418029)

[2. The Battery Pack 23](#_Toc122418030)

[3. The Battery Management System(BMS) 24](#_Toc122418031)

[4. THE Microcontroller Unit (MCU) And The Buck Converter 24](#_Toc122418032)

[**3.1.3. COMPONENTS USED FOR THE ELECTRONIC CIRCUIT** 27](#_Toc122418033)

[3.1.4. TRAINING AND TESTING DATA ACQUISITION 27](#_Toc122418034)

[3.1.5. TRAINING THE MACHINE LEARNING MODEL 35](#_Toc122418035)

[3.1.6. ROBOT PROGRAMMING METHODOLOGY 39](#_Toc122418036)

[CHAPTER FOUR 50](#_Toc122418037)

[TEST RESULTS AND ANALYSIS 50](#_Toc122418038)

[4.1. TESTING RESULTS 50](#_Toc122418039)

[4.2. OBSERVATION 50](#_Toc122418040)

[CHAPTER FIVE 51](#_Toc122418041)

[CONCLUSIONS AND RECOMMENDATIONS 51](#_Toc122418042)

[5.1. CONCLUSION 51](#_Toc122418043)

[5.2. LIMITATIONS 51](#_Toc122418044)

[5.3. RECOMMENDATIONS 51](#_Toc122418045)

[References 52](#_Toc122418046)

# CHAPTER ONE

# INTRODUCTION

## 1.1 BACKGROUND OF STUDY

Owing to the low cost of production of plastic and its indestructible and nonbiodegradable nature, there has been a consistent increase in the use of plastic for packaging and storage in manufacturing.

The increase in the use of plastic has led to an alarming increase in plastic waste. While this is expected, its non-biodegradable nature makes it last for considerably long periods of years without disintegrating.

This slow disintegrating process has led to plastic saturation in the ecosystem which can only be managed by reducing their production.

To achieve this, organizations would be required to recycle plastic waste instead of producing more. However, the retrieval and collection of plastic waste have been challenging to manage over the years.

One of the most prominent challenges with waste management over the years has been segregating plastic waste from other types of waste from the source of disposal.

In most developing countries, there is a lack of an efficient waste management system, and creating an easy to understand system would require a collective effort of a fully committed government, the general public and private organisations.

There is a need for precise sorting of plastic waste to allow recycling and technology for sorting plastic waste especially in developing countries is rare and expensive by their standards.

The sorting is usually done manually which is not an efficient method of sorting plastic from a waste pile. There is a need for a more efficient alternative of detecting plastic waste from a pile to make sorting easier.

To combat this, an efficient method that can be considered is the use of programmed robots for obstacle detection, especially plastic. (Ejimofor et al, 2022)

Robots are usually used to perform industrial tasks and especially in hazardous environments. Robots usually work with artificial intelligence to have human-like senses such as vision, touch, temperature sensing etc. (Britannica, n.d.)

Although most robots are not self-sufficient, artificial intelligence increases their functionality and they can be useful for plastic detection.

By employing robotics, which is the design and construction of robots to carry out tasks traditionally done by human beings, there is a possibility of creating a cheaper and more efficient method for plastic waste detection and retrieval.

## 

## 1.2 AIM

To design and construct an economic robot that can detect and retrieve plastic waste material objects utilising a machine learning classifier model.

## 1.3 PROBLEM STATEMENT

The increase in plastic waste in the ecosystem is at an alarming rate and there is a need for active detection and retrieval for recycling purposes.

Most developing countries do not have the economic or infrastructural means to build effective waste management systems. (The World Bank)

## 1.4 OBJECTIVES

The objective of this project for plastic waste detection are:

1. To design and construct the robot chassis
2. To design and construct the electronic circuitry
3. To obtain data for training the machine learning classifier
4. Train the machine learning classifier model
5. To design the robot control
6. To test the designed constructed robot

## 1.5 METHODOLOGY

1. Using available metals that are cost effective, the robot chassis will be welded.
2. Necessary electronic components such as relays, electronic motors, printed circuit boards, lithium batteries etc, the electronic circuit will be developed.
3. Data for training the machine learning model will be derived as live data from the environment of the Electrical/Electronic Final Year Class (LT1) in the faculty of Engineering, University Of Benin.
4. The training will be done in a cloud environment i.e., Google collab. Python programming will be used, the Haars-Cascade classifier model. The model will be validated performance metrics.
5. Open CV libraries will be used for image processing and the classifier developed above will be used to identify desired images of plastic waste objects. This intelligence will thereafter be interfaced with Raspberry Pi to activate the robot and its peripherals.
6. The detection and retrieval of the developed robot will be tested using various objects.

## 

## 1.6 SCOPE OF WORK

This work is limited to the retrieval of plastic water bottles in developing countries such as Nigeria.

## 

## 

## 1.7. SIGNIFICANCE OF WORK

The project is designed to contribute to the efforts of plastic waste management and in turn, reduce plastic pollution.

With the use of cot-friendly materials, we want to develop a means for localized plastic waste management which can be improved upon with further work.

# 

# 

# 

# CHAPTER TWO

# LITERATURE REVIEW

## 

## 2.1 LITERATURE REVIEW

Computer Vision is one of the most powerful and compelling types of AI and we’ve almost experienced it in any number of ways without even knowing.

The concept of computer vision is based on teaching computers to process an image at a pixel level and understand it.

Computer vision is the field of computer science that focuses on mimicking human vision by enabling computers to identify and process objects in images and videos in the same way that humans do.

Computers are trained to recognize images from the vast available data of images on the internet. The algorithms were built to mimic the image recognition and processing by the human eye.

Computer vision is all about pattern recognition. In a day, approximately 3 billion images are shared and algorithms have been trained to identify items based on their attributes.

Attributes of an image are defined and stored in datasets and these attributes are compared with the extracted features of images the algorithm has been trained with.

There are common tasks that computer vision can be used for and they are:

1. **Object classification:** Systems with computer vision can inspect visual content like photos and videos and classify the object on a photo/video to a pre-defined category. A typical example is when the system can find a dog among all objects in the image.
2. **Object identification:** For identification, the system scans photos and videos and can point out a particular object in them. For example, the system can find a specific dog among the dogs in the image.
3. **Object tracking:** For this function, the system will process a video, find its object or objects of interest from the search criteria given and track its movement.

### **2.1.1. PATTERN RECOGNITION IN IMAGE PROCESSING**

Pattern recognition is simply the computation of any given data by comparing it with existing knowledge in a database for similarities to determine the object.

It has taken years of development from the 1960s up to this point and it includes a lot of methods and computations. The point of pattern recognition is for a system to be able to accurately describe an object based on predetermined characteristics.

Pattern recognition has different subsections and they are given below:

1. Statistical Pattern Recognition

Statistical decision and estimation is based on feature vector distribution and it is a probability and statistical model. The statistical model is characterised by a family of class-conditional probability density functions.

The features are put in an optional order and the set of features can be regarded as a feature vector. This method of pattern recognition deals with features only and does not consider the relationship between features.

1. Data Clustering

This is an unsupervised method of machine learning and the aim is to find similar clusters from a mass data. There is usually no prior knowledge of the clusters in this method. This method is separated into two - hierarchical clustering and partitional clustering.

1. Fuzzy Sets

This method follows the logic that the thinking process and languages of human beings are usually fuzzy and uncertain. Realistically, it is not always possible to have complete answers or classifications. The use of fuzzy sets in pattern recognition started in 1966, where the two basic operations, i.e. abstraction and generalisation were proposed by Bellan et al.

1. Neural Networks

Neural networks have been developing quickly since the first neural networks model was proposed in 1943. The approach of neural networks applies biological concepts to machines for pattern recognition. This led to the creation of artificial neural networks which is set up by trying to physiologically model the human brain.

Neural networks are made up of different associate units. In addition, genetic algorithms applied in neural networks are statistical optimised algorithms proposed by Holland in 1975.

Neural Pattern Recognition has a higher appeal because it needs minimum prior knowledge. In any situation where it has enough layers and neurons, an ANN can create any complex decision region.

1. Haar’s Cascade Classifier

Haar’s cascade classifiers are classifiers that were first used for real-time face detection then subsequently used for object detection. It is a machine learning object detection algorithm that identifies objects in an image and video.

A detailed description of Haar’s classifiers can be seen in Paul Viola and Michael Jones’s paper “Rapid Object Detection using a Boosted Cascade of Simple Features”. In their paper, Viola and Jones propose an algorithm that is capable of detecting objects in images, regardless of their location and scale in an image. Furthermore, this algorithm can run in real-time, making it possible to detect objects in video streams.

Specifically, Viola and Jones focus on detecting faces in images. Still, the framework can be used to train detectors for arbitrary “objects,” such as cars, buildings, kitchen utensils, and even bananas.

Haar Cascade is a machine learning-based approach where a lot of positive and negative images are used to train the classifier.

• Positive images – These images contain the images which we want our classifier to identify.

• Negative Images – Images of everything else, which do not contain the object we want to detect.

One of the primary benefits of Haar cascades is that they are just so fast — it’s hard to beat their speed

1. Structural Pattern Recognition

This method does not follow feature extraction and segmentation, instead it focuses on the description of the structure. It throws light on how some less complex sub-patterns can make one pattern. The two main methods in structural pattern recognition are syntax analysis and structure matching.

If you take the relationship between each part of the object into consideration, structural pattern recognition is best. It deals with symbol information, and this method can find applications in high-level computation like image interpretation.

Structural pattern recognition is always accompanied by statistical classification or neural networks for dealing with more complex problems of pattern recognition. A simple application is in the recognition of multidimensional objects.

Other forms of pattern recognition include:

1. Syntactic pattern recognition
2. Approximate reasoning approach to pattern recognition
3. A logical combinatorial approach to pattern recognition
4. Applications of Support Vector Machine (SVM) for pattern recognition
5. Using higher-order local autocorrelation coefficients to pattern recognition
6. A novel method and system of pattern recognition using data encoded as Fourier series and Fourier space.

For object detection in image processing for the sake of this project, the machine learning involved is the use of Haar’s cascade classifier as it is more widely used for image processing.

### **2.1.2. IMAGE PROCESSING IN COMPUTER VISION**

Image processing is the transformation of an image into a digital form and carrying out several operations to get relevant information out of it.

Image processing systems treat images as 2D signals to be able to apply predetermined methods of signal processing. There are different types of image processing and some of them are:

1. Object Visualization - This helps the system to find objects that are not visible in the image
2. Object Recognition - This enables the system to differentiate or detect objects in the image
3. Image sharpening and restoration - This is the process of enhancing an original image to a better working quality.
4. Pattern recognition - This is done by simply measuring the various patterns around the objects in the image
5. Image retrieval - In this process, the system browses a database of images for images similar to the original image.

Image processing is a step-by-step process that starts with acquiring an image through camera capture or from an existing image. Several operations are then performed on it to get the desired information out of the image.

Image processing is usually done using machine learning which is a subset of artificial intelligence. Image processing makes use of deep neural networks to process images and the most commonly used Neural Network is the Convolutional Neural Network (CNN). (Klearstack, 2021) (Tripathi, 2021)

CNN is commonly used for image recognition and processing. It processes an image block-by-block, from the upper left corner, and moves pixel-by-pixel till the last pixel to complete the verification.

The final results from the block verification are passed through a convolutional layer, where each data element is connected. Passing the results through the convolutional layer leads to the result from the verification data.

This result is what helps the system to identify the image. For a better understanding, the step-by-step process of image processing is explained in details below.

1. Image Acquisition

This is the first step in image processing. This involves getting the image from its source, usually a hardware-based source such as optical sensors, digital cameras, webcams, from your computer storage or the internet.

1. Image Enhancement

Image enhancement is the process of highlighting the features of interest in the given image that is unclear and this is usually done by enhancing the brightness or saturation of the image.

1. Image Restoration

Image restoration involves improving the way an image appears. However, unlike image enhancement, mathematical and probabilistic models are used to improve its appearance as opposed to increasing brightness and saturation in image enhancement.

1. Colour Image Processing

Colour image processing includes analysing, transforming and interpreting any visual data given in colour. Colour image processing is divided into two major areas:

* Full-colour image processing: in full-colour processing, images are gotten from full colour sensors or cameras. This is the colour image processing that is usually in use.
* Pseudo-colour image processing: For pseudo-colour processing, false colours are assigned to a monochrome image.

The focus here is on assigning a colour to a particular monochrome intensity or range of intensities.

5. Wavelets and Multiresolution Processing

This is the use of wavelets transform to analyse a visual signal i.e., image into different frequency components in various degrees of resolution (i.e. multiresolution). This allows features that are not detected at one resolution to be easily spotted at another resolution.

6. Compression

Compression is reducing the size of an image in order to reduce the storage required to save it or the bandwidth required to transmit it. This is done to most images on the internet.

7. Morphological Processing

Morphological processing is a set of processing operations for processing images based on their shapes. It applies a structuring element to an input image to create an output image of the same size.

8. Segmentation

Segmentation is an essential procedure for object detection, image recognition, feature extraction, and classification tasks. depend on the quality of the segmentation process.

It involves dividing an image into various segments to increase the effectiveness of pattern recognition. Segmentation is one of the most difficult steps of image processing.

9. Representation and Description

Image representation and description is critical for successful detection and recognition of objects in a scene.

After an image has been separated into object and background regions, each region needs to be represented and described in a way that makes it easy for computer processing.

This is especially useful during pattern recognition. Representation handles the characteristics of the image and its regional properties.

While, description deals with extracting information that helps the system to differentiate classes of objects.

10. Recognition

Recognition simply assigns a label to an object based on its description. It is the process of identifying an object or a feature in any given visual data.

These are the steps images have to go through to be recognised by an Artificial Neural Network (ANN) designed for image processing such as CNN.

### **2.1.3. Open CV**

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.

Being a BSD-licensed product, OpenCV makes it easy for businesses to utilise and modify the code. It is a collection of functions and classes that implement many Image Processing and Computer Vision algorithms.

OpenCV is a multi-platform API written in ANSI C and C++, hence is able to run on both Linux and Windows. The library has more than 2500 optimised algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms.

These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

## 2.2 RELATED WORKS

### **Development Of Identification Methods for Cans And Bottles By Irsyadi Rani and Ihsan Budiman**

The procedure for the identification of cans and bottles for this experiment consisted of a series of computational steps such as image acquisition, image pre-processing, feature extraction and identification.

These steps lead to pattern recognition of an actual object. Starting from taking the image from the inspection zone and trimming the unnecessary part of the image.

The background noise is then eliminated from the image using combined operation of threshold and morphological operation with convolution filters.

The features extraction stage consists of quantization of image, calculating co-occurrence matrix and calculates energy in different quantization levels. In the feature extraction phase, both colour and grayscale images are considered.

The image is normalised, its edges are extracted and it is stored as a logical image. The values of the image are compared with the values in the database and it is matched with the closest value.

The project has a success rate of 91.3% and focused on identifying cans and glass bottles based on colour, shape, size of image etc.

The aim of the project was to determine that computer vision can be used to sort multiple types of solid waste - cans and bottles.

**Limitations**

The project did not focus on the detection of any plastic because the values for determining plastic were not accounted for in their database as it was beyond our scope of study.

### **Trash And Recyclable Material Identification Using Convolutional Neural Network By Rumana Sultana**

This project used image processing algorithms and deep learning technologies to detect trash in public places and improve municipal trash collection. The aim of the project was to improve trash management systems and create a smart city.

The research used 2 Convolutional Neural Networks, both based on the AlexNet network architecture.

One was developed to search for trash objects in an image and the other was designed to separate recyclable items from the landfill trash objects.

For the project, they trained and tested the two-stage CNN system on the benchmark TrashNet indoor image dataset and received great performance.

Then the system was trained and tested on outdoor images taken by the authors in the ideal environment for usage.

Using the outdoor image dataset, the first CNN achieved a preliminary 93.6% accuracy to identify trash and non-trash items on an image database of assorted trash items.

A second CNN was then trained to distinguish trash that will go to a landfill from the recyclable items with an accuracy ranging from 89.7% to 93.4% and overall, 92%.

The trash detecting process was divided into three steps which were object detection, object retrieval, and object classification.

The detection was done using a camera and for retrieval and classification, the following were the steps.

They developed an algorithm using a CNN to detect if an object was an item to collect such as trash or recyclable, or an item to leave alone such as an animal or another item.

Another algorithm was developed to detect if a collected item is either trash or a recyclable item. Both CNNs were developed and tested in both indoor and outdoor settings.

When the robot finishes classifying the object, it will either be placed in a trash bin or a recycling bin depending on the determination of the classification algorithm.

**Limitations**

A future goal is to integrate this image processing-based trash identification system in a smart trash can robot with a camera to take real-time photos that can detect and collect the trash all around it.

There were some challenges with identifying some objects such as glass and plastic are often mixed up with an accuracy of less than 90%.

These challenges happened perhaps due to the lighting effect, camera quality, environmental or weather factors, etc.

### **Design And Construction Of A Litter Collecting Robot By Karl Kangul et. Al**

The project was carried out in a controlled mockup of 4 different environments to measure the robot’s performance across different platforms.

The first was a flat surface, the second an artificial grass field, the third a rocky environment, and the fourth an elevated platform to watch obstacle avoidance.

The robot was designed to work without human interaction and make independent decisions based on its programming.

The robot could detect an obstacle using sensors, differentiate between a bottle and other obstacles and decide to retrieve the bottle.

The experiment had a 100% success rate on the flat platform with no other obstacles in the way except the bottles.

On the artificial grass, efficiency with obstacle detection was reduced by the artificial grass sending false positives as detected bottles.

On the rocky surfaces, the robot's efficiency and speed were reduced as a result of the terrain. It was harder to detect plastic bottles as the robot had to avoid constant obstacles.

**Limitations**

While the robot showed robustness on different terrains, efficiency was dependent on the terrain of the experiment.

# 

# CHAPTER THREE

# METHODOLOGY AND DESIGN

## 

## 3.1 METHODOLOGY

This project is divided into two parts - Electronics (hardware aspect) and Intelligence (software aspect).

### **3.1.1. DESIGN AND CONSTRUCTION OF THE CHASSIS**

This is the material that was used to make the chassis of the plastic retrieval robot in this experiment.

The measurement of the Acko board for the robot was 12x24. We chose an ACKO for the chassis because it’s lightweight and low cost.

Picking a lightweight Acko board is to reduce time-lag with steady-state response due to the weight of the robot. Excess weight will cause the robot to respond slowly to command from the Raspberry Pi and Arduino Mega.

Also, with excess weight, the robot will require more electrical power for the motor to function effectively, so this Acko board being lightweight made it a good choice.

When designing the chassis, we had two options of making the chassis encase the robot only with no room for plastic bottle storage after retrieval or design the chassis with a hollow space to store collected plastic bottles.

In the former, the robot retrieves the bottles and moves to a waste disposal can for plastic offloading before searching again. In the latter, the robot stores the bottle till its chassis is full before offloading.

We designed the latter option containing a hollow space for plastic retrieval as this is less time-consuming and more aesthetically pleasing. It had a sweeper brush to sweep the plastic into the chassis.

### 

### **3.1.2. THE ELECTRONIC CIRCUIT**

The electronics circuit is made up of a hydronium microcontroller, the H-bridge and motors.

The H-bridge was made up of relays in order to control the flow of current to enable the motors to move forward and backwards.

We made use of a battery management system (BMS) to prevent overcharging and complete drainage to avoid making the battery sleep.

The machine learning model installed in the Raspberry pi will in turn communicate with the arduino.

The arduino mega will receive its command from the raspberry pi based on the image the raspberry pi sees.

### 

### **PARTS OF THE ELECTRONIC CIRCUIT**

#### H-Bridge Circuit

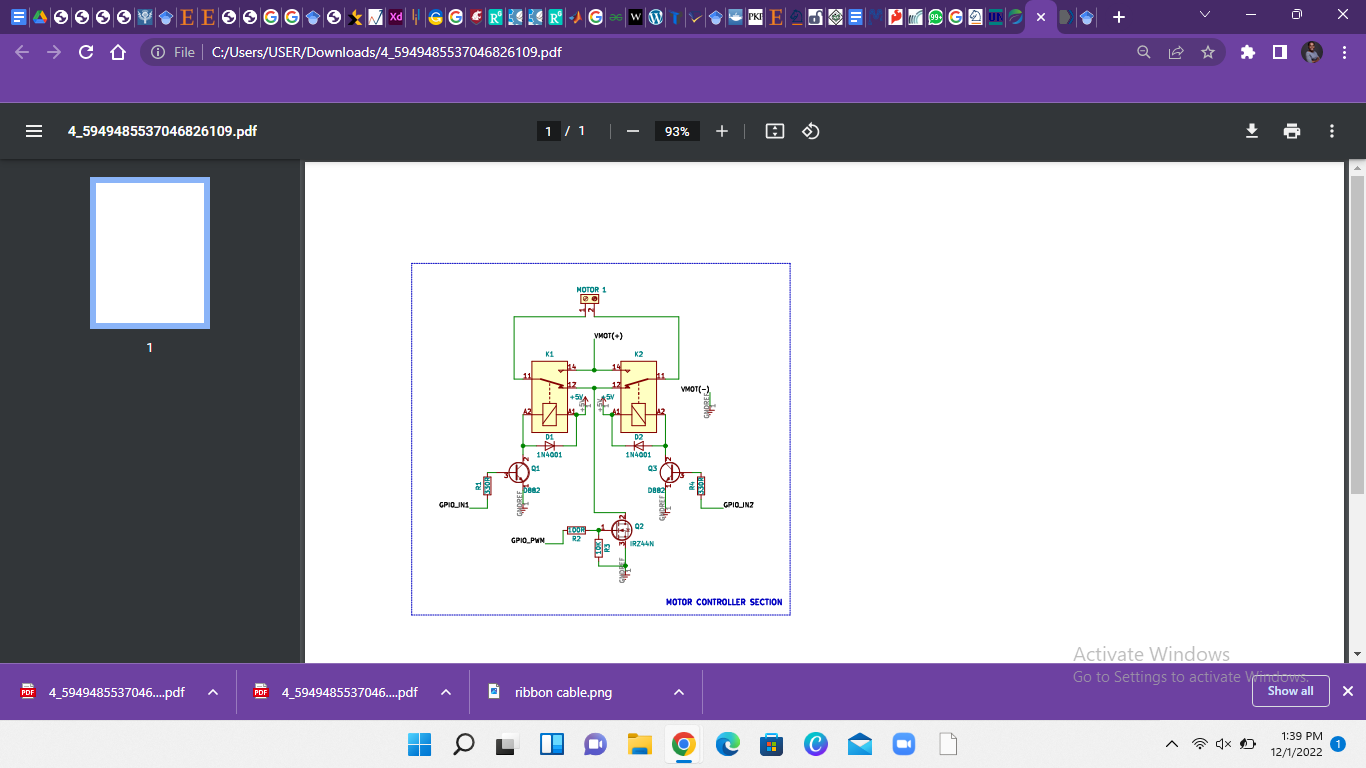


Fig 3.2.1. The Electronic Circuit of the H-Bridge

A H-bridge is usually used with a microcontroller, such as an Arduino, to control motors and this is responsible for the motors of the robot being able to move forward or backward.

An H-bridge is a circuit that switches the voltage/current at both ends of the load or output it is connected to.

Robots and other practical applications use these circuits for DC motor inversion control, speed control, stepper motor control, the majority of DC-AC converters (such as inverters) used in electrical energy conversion, some DC-DC converters (push-pull converters), and other power electronics devices.

H-bridge is a common DC motor control circuit named after its circuit shape, which resembles the letter H. The four vertical legs of H are formed by four transistors, and the horizontal bar in H is formed by a motor.

The H-bridge circuit, which is commonly used in inverters, can be built as discrete components or integrated into an integrated circuit (DC-AC conversion).

DC power (from batteries, for example) is inverted into AC power of a fixed or variable frequency, which is used to drive AC motors by opening and closing switches (asynchronous motors, etc.). (Utmel Electronics, 2021)

Using the direction of the current, the H-bridge transfers current from one direction to the other and has the ability to reverse it to enable the motor to move in a clockwise or anti-clockwise direction.

Due to the power ratings of the motor, the H-bridge was built with two relays. When current flows through one of the relays, the other relay is turned off. The relays were mounted side by side and functioned as follows:

When the relay on the right is turned off, current flows from the left relay into the right and vice versa. The relay circuit can not be connected directly to the microcontroller so a transistor is employed.

A MOSFET and BJT (D882) are used. The D882 is switched so that it can switch the relays in turn. The signal from the microcontroller is connected to the base of the transistor and the connected base is responsible for switching the relay.

The MOSFET (IRFZ44) is a power MOSFET and it functions as a PWM to vary the speed of the motor.

#### The Battery Pack

The battery pack is a 12V battery connected in series with the BMS to prevent the battery from going into deep sleep and for easily awakening the battery.

A picture of the battery pack is given below:



Fig 3.2.2. Battery pack

#### 

#### The Battery Management System(BMS)

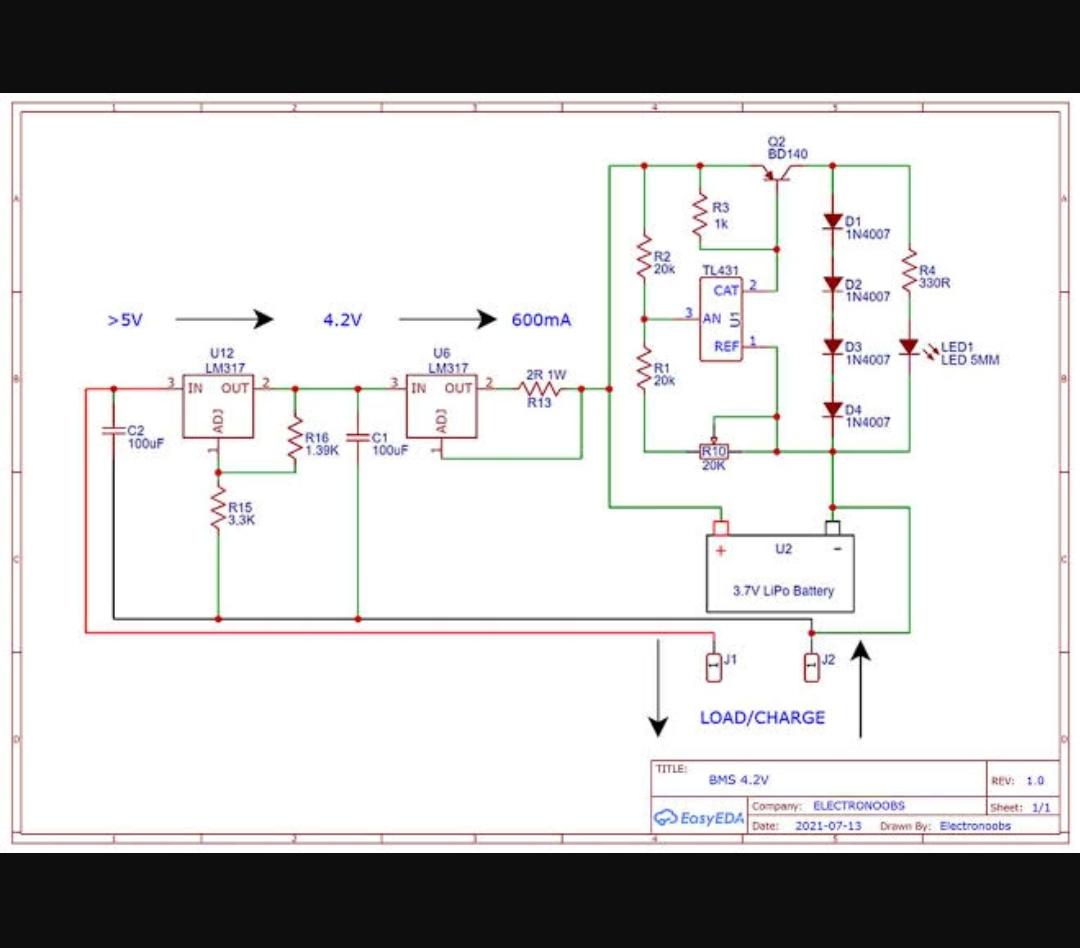


Fig 3.2.3 Circuit Diagram Of The BMS

We made use of a battery management system (BMS) to prevent overcharging and complete drainage to avoid making the battery sleep.

The BMS is simply a computing device connected to several sensors that monitor voltage, current and temperature of each battery cell. These sensors send the result of their monitoring process to the Battery Management System.

After this, the BMS evaluates the data to ascertain that every cell is operating within the prescribed limits. If any of the cells is discovered to not be working within the stated limits. The BMS tries to rectify the issue.

This could happen in many forms, for example: The cells in the battery pack might be too hot, so the BMS would work with the cooling system to regulate and bring down the temperature of the battery pack.

In other cases, it could be from the variation of cell voltage. Here, the Battery Management System would perform cell balancing. To achieve this, the BMS transfers energy from one cell to another to make sure every cell performs at the same level of voltage.

#### THE Microcontroller Unit (MCU) And The Buck Converter

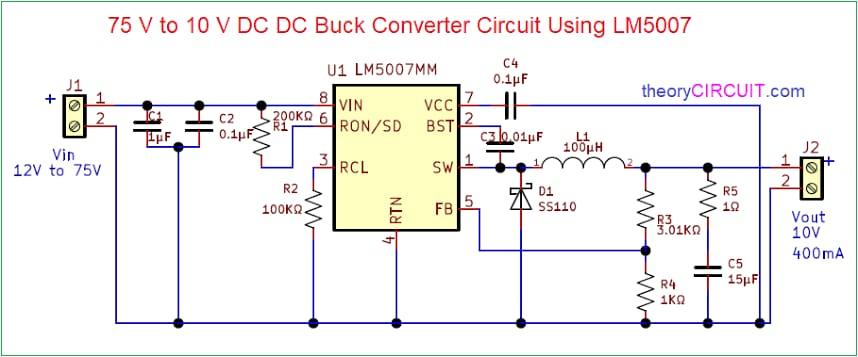


Fig 3.2.4a The Buck Converter Circuit

The Microcontroller used is the ATMega because it has more I/O pins and PWM pins compared to the UNO. The MCU consists of the buck converter.

The board contains two sets of voltage - 12V and 5V. The motors function with 12V but the microcontroller needs a minimum of 5.5V for operation.

The buck converter in the MCU unit helps us to split the voltage resulting in a direct line of 12V and a different line of 5V. The 5V is from the output of the bulk converter which is connected as power to the microcontroller.

All the GPR pins on the H-bridge connect to the ribbon cable on the microcontroller board which in turn switches the H-bridge. The design for the microcontroller and buck converter circuit is shown below:

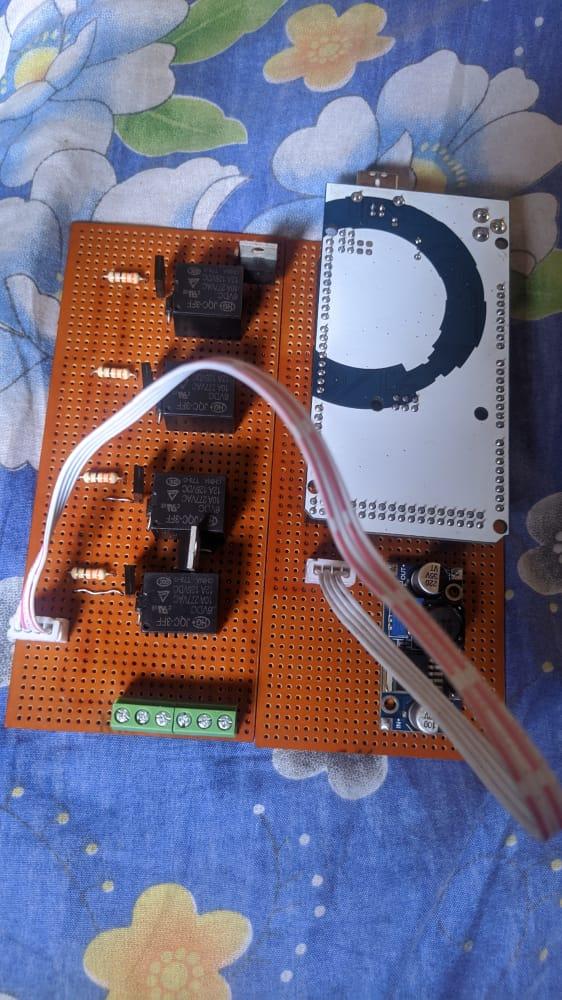


Fig 3.2.4b Design Of The MCU showing Buck converter and microcontroller

The Arduino Mega on this circuit will receive commands from the Raspberry Pi containing the Machine Learning model.

1. The Raspberry Pi

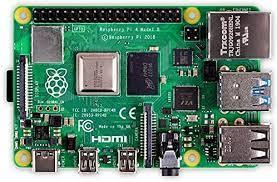


Fig 3.2.5. The Raspberry Pi 4

The Raspberry Pi 4 running on the Raspberrian OS acts as a central controller and receives input from the camera in the form of an array of numbers in a matrix to process the images and determine the position of the plastic of interest.

After it detects the position of the plastic of interest in the image, it searches for the position of the pixels in the image that represents the plastic.

The Raspberry Pi is connected to the Arduino through a serial connection using the USB serial ports on both the Raspberry Pi and the Arduino.

The Camera is also connected via a serial USB port to the Raspberry Pi and the Arduino Mega.

### **3.1.3. COMPONENTS USED FOR THE ELECTRONIC CIRCUIT**

The following are the components of the electronic circuit used for this project.

1. **Terminal Screw:**



Fig 3.1.1 A Terminal Screw

Terminal screw commonly known as screw terminal is used to connect bare and unterminated wire to a circuit.

They make it possible to connect multiple different connecting devices, however they easily come undone and this can cause a bare wire to loose in your circuit.

A small dab of hot glue can address this without being too difficult to remove later.

They are typically designed for narrow wire gauges as easily small wires are as bothersome as big wires.

1. **Vero Board:**

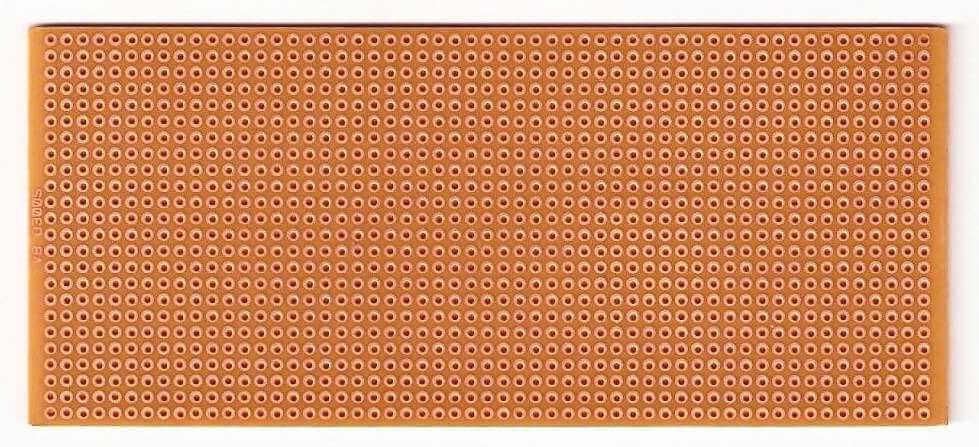


Fig 3.1.2. Veroboard

A veroboard is a printed circuit board (PCB) designed with rows of copper tracks with holes drilled in them for soldering electronic components for electronic circuits.

While it is cheap and easily available, there are a few disadvantages of using a vero board. Circuits on the veroboard are tricky to build because there’s no specific place to put components.

This makes it easy to put components in the wrong place. The larger the circuit the more complex it becomes to build on a veroboard.

On the bright side, besides being cheap and easily available, it takes a short time to create a functioning circuit. It is also more reliable and more permanent than using a breadboard.

1. **1 Panel of Acko Board:**

This is the material that was used to make the chassis of the plastic retrieval robot in this experiment. The measurement of the Acko board for the robot was 12X24. We chose an ACKO for the chassis because it’s lightweight and low cost.

Picking a lightweight Acko board is to reduce time-lag with steady-state response due to the weight of the robot.

1. **Ribbon Cables:**



Fig 3.1.3 A ribbon cable

A ribbon cable, also called a multiplanar cable, is a flat, thin cable made up of multiple small-grade cables placed parallel to each other.

They form a wide-flat cable that looks like a piece of ribbon, and that is why it’s called a ribbon cable.

They are mostly used in electronic systems that require multiple data buses for linking internal peripherals to their respective drive controllers.

1. **High torque motor:**

The motors used for the construction of the waste retrieving robots are high torque motors.

1. **Raspberry Pi**

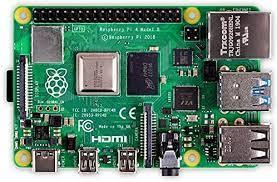


Fig 3.2.11. The Raspberry Pi 4

Raspberry Pi is a minicomputer with an equivalent size to that of a credit card and it works with any input and output device as a fully functional computer. It is a programmable device that has all the important features of a motherboard of a typical computer system without peripherals or internal storage.

An SD card with the Raspberrian OS installed has to be inserted in the provided space to set up the Raspberry computer. The installed operating system is required for the Raspberry Pi to boot.

Raspberry computers are compatible with Linux OS and this decreases the memory space needed and creates an environment for diversity. After setting up the OS, the Raspberry Pi can be connected to output devices. (BasuMallick, 2022)

The Raspberry Pi usually has the following features

* The CPU
* An HDMI port
* General Purpose I/O pins
* SD card slot
* RAM
* GPU
* Ethernet Port
* LEDs
* USB ports
* Power source

1. **Arduino Mega**



Fig 3.2.12. The Arduino Mega

Arduino is an open-source electronics platform that makes use of simple hardware and software. Arduino boards read inputs from sensors as in light or motor activation. The Arduino Programming Language is a native language supported by Arduinoand is a framework built on C++ programming language. Its Integrated Development Environment is based on processing, and this programming is often based on wiring. (Kumar, 2022)

The Arduino board used for this project was the ATMega 2560 and it depends on the ATmega2560 microcontroller. It includes 54 digital I/O pins, out of which 16 pins are analog inputs, 14 of these pins are used as PWM outputs hardware serial ports (UARTs) – 4.

A crystal oscillator of 16 MHz, an ICSP header, a USB connection, an RST button in addition to a power jack. This board mainly includes everything which is essential for supporting the microcontroller.

The pin configuration for the Arduino Mega is given below:

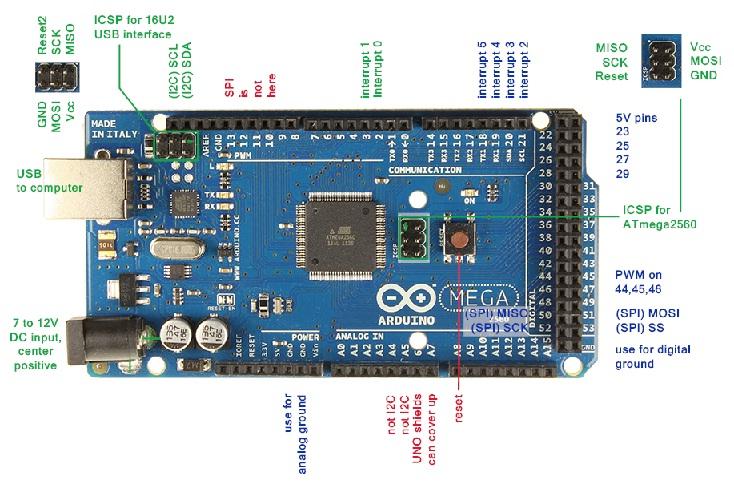


Fig 3.2.12b Pin Configuration of the Arduino Mega

All the pins on this board have a predetermined function allied with it. All the analog pins of the ATMega 2560 can be used as digital Input Output pins.

The ATMega 2560 has a flexible work memory space and its processing power that permits it to work with different types of sensors without delay. The ATMega boards are superior physically compared to other Arduino boards. (Elprocus, 2020)

1. **Transistors**

The transistors used for this project were the MOSFET (IRFZ44) and the D882 BJT

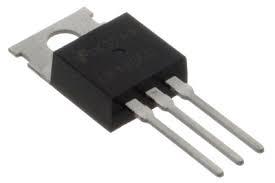


Fig 3.2.13. A MOSFET

The IRFZ44N is a N-channel MOSFET that has a high drain current of 49A and its Rds value is low at 17.5 mΩ. The IRFZ44N has a low threshold voltage of 4V and the MOSFET starts conducting at this voltage. Due to this, it is common to use microcontrollers to drive it with 5V.

However, for the MOSFET to switch completely, it needs a driver circuit. While the MOSFET will come on when it has a small gate voltage of 4V, the drain current will only reach a maximum when 10V is the gate voltage.

Since the MOSFET has to be driven from a microcontroller, the logic level version IRLZ44N MOSFET was used.

The difference between the IRLZ44N and IRFZ44N MOSFETs is that the IRLZ44N MOSFET operates at a logic level with a very low gate threshold voltage of 5V. As a result, the MOSFET can be fully turned on with just 5V on its gate pin. This characteristic makes a driver circuit irrelevant.

On the other hand, the IRFZ44N needs a gate driver circuit for the MOSFET to be turned on completely. And it uses a microcontroller like Arduino to achieve this.

In spite of this, the IRFZ44N turns on partially with direct 5V form a I/O pin, but the output drain current will be limited. (Components 101, 2019)



Fig 3.2.13b. The BJT D882

The D882 is an NPN transistor capable of handling voltages up to 30V and currents up to 3. The peak current it can handle is 6A for pulses of less than 5ms.

It is used as a **switch**. In this mode, the emitter is grounded, and the collector is linked to one end of the load with the power supply connected to the other end of the load. TO prevent voltage spikes damaging the transistor, a flyback diode is positioned anti-parallel to the load when there is inductive load.

A base resistor is used to limit the base current and a tiny capacitor is added to it to aid the fast turn-off. (Utmel Electronics, 2022)

1. **Ultrasonic sensor**

The ultrasonic sensor measures the distance of an object by giving off ultrasonic sound waves. It then converts the sound that is reflected into an electrical signal. Ultrasonic waves travel at a faster speed than audible sound (audible sound in this context means sounds humans can hear). (Jost, n.d.)

It makes use of a transducer to send and receive ultrasonic signals that inform it about an object’s proximity. Ultrasonic sensing is one of the best ways to sense proximity and detect levels with high reliability.

The sensor’s transducer acts as a microphone for receiving and sending the ultrasonic sound. Most ultrasonic sensors use a single transducer to send a pulse and to receive the echo, but in some cases they are separate.

By measuring the time lapses between sending and receiving the ultrasonic pulse, the sensor is able to determine the distance to a target. (Burnett, 2020)

The formula for this calculation is D = ½ T x C (where D is the distance, T is the time, and C is the speed of sound ~ 343 m/s). (Jost, n.d.)



Fig 3.2.14. Ultrasonic sensor

1. **Belt**



Fig 3.2.15. A mechanical/conveyor belt

The belt is a flexible element of a mechanical system. It is used to transfer the power from one System to another System. (Dey, n.d.)

The belt: A long stretch of thick, durable material upon which materials are transported from one place to another.

A conveyor belt works by using two motorized pulleys that loop over a long stretch of thick, durable material. When motors in the pulleys operate at the same speed and spin in the same direction, the belt moves between the two.

If objects are particularly heavy or bulky — or if the conveyor belt is carrying them for a long distance or duration — rollers may be placed on the sides of the conveyor belt for support. (Semcor, n.d.)

1. **Bearing and shaft**



Fig 3.2.16. Bearing



Fig 3.2.16b Shaft

Bearings support the shaft that rotates inside the machinery. Including gears, turbines, rotors, etc. in said machines.This enables them to rotate without a hitch.

This in turn helps to reduce the friction and drastically reduce the amount of energy the motor or rotor will consume. This is the most basic importance of bearings.

This helps to prevent damage due to the force of rotation on the parts that aids rotations and it also helps the shaft that is rotated to maintain the correct position.

It is due to this function of bearing that machines can be used repeatedly for an extended period of time.(*What Are Bearings? Let's Learn About the Basic Functions of Bearings*, 2019)

To summarise it, the function of a bearing in an electric motor is to support and locate the rotor, maintain the air gap and move load from the shaft to the motor. (AHR international, n.d.)

## 3.1.4. TRAINING AND TESTING DATA ACQUISITION

Images that were used for training and testing the machine learning classifier were collected from Electrical and Electronics final year class (Lecture Theatre 1) in the faculty of Engineering, University of Benin. Due to the long period of time it takes to test and train with a large dataset, we decided to stick to one type of plastic (Fearless Energy drink bottles) for faster training and ease of testing. The images contained bottles in varied positions namely; bottles kept vertical to the floor and bottles kept horizontal to the floor at varied angles. The distance of the bottles from the camera ranged from 0 to 10 metres.

|  |  |  |  |
| --- | --- | --- | --- |
| Image Description | Total Images Captured | Number used for training | Number used for testing |
| Images with a fearless bottle | 500 | 300 | 200 |
| Images without any bottle | 1000 | 600 | 400 |

**Positive Images**



**Negative Images**







## 3.1.5. TRAINING THE MACHINE LEARNING MODEL

We considered using Neural Network and template matching for object detection. We decided against these two because they were too computation-intensive for the Raspberry Pi.

The training algorithm we used for the project was the Haar’s Cascade Classifier. This is because it is easy to train and the scope of the project work was retrieving plastic bottles in a particular environment, it was faster to run on the Raspberry Pi.

Google Co-lab notebook was used to train the classifier in the cloud because it provides better hardware resources than our laptop.

A Haar’s classifier looks for features of an image in different layers. At the top layer, it looks at features large enough to cover the entire window of the image. And at the bottom layer, it scans for finer details.

Because of this, the model is fast enough to detect the plastic in real time at the end and it is quick to reject images whose areas do not match with the features in the top layer. After this, it spends more time to analyse other areas of the image that match with the features of the top-most layer and scrutinises its finer details at the bottom layer.

The accuracy and quality of the model depends more on the data that was used to train the model than the code. So, you need as much quality data as possible to get good results.

The classifier is fed two different types of data which are:

1. Positive Image: This will contain the exact image of the plastic bottle we are trying to detect.
2. Negative Image: These will be images that are not the plastic bottle we are searching for.

The Machine Learning algorithm has to see a variety of the right and the wrong objects in order for it to learn properly. We got different pictures of our plastic bottle in different conditions from lighting and positions.

The first step we employed was getting our positive and negative images, and after this we wrote some code to make it easy to determine which is which. A screenshot of the code is given below:

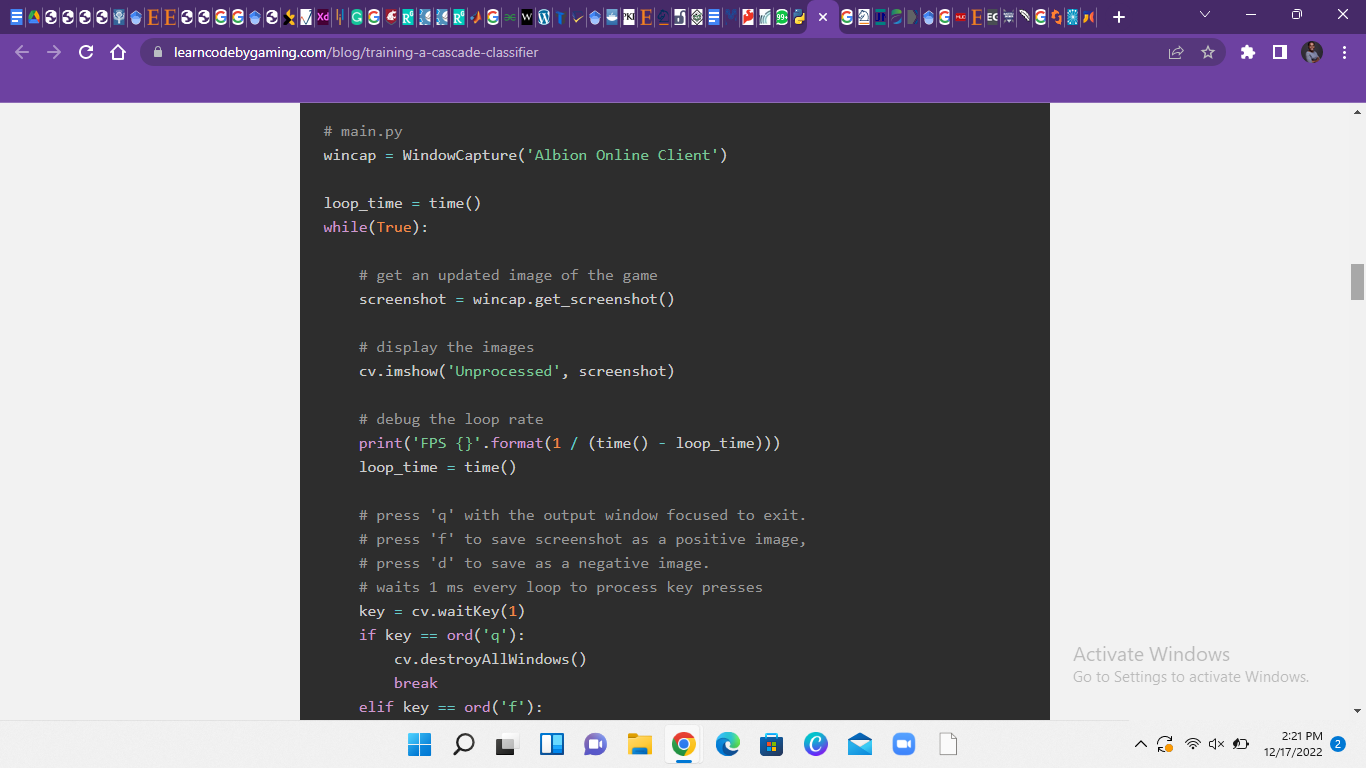




Fig 3.2.3. Code for training the classifier with negative and positive images.

It is advised to manually create folders for your negative and positive images each. With the OpenCV output window focused we press 'f' to capture a screenshot and save it to the positive folder, or 'd' to save a screenshot to the negative folder.

To prepare the negative samples to be used for training, we need to create a text file that lists where all our negative samples can be found and the code below provides that:

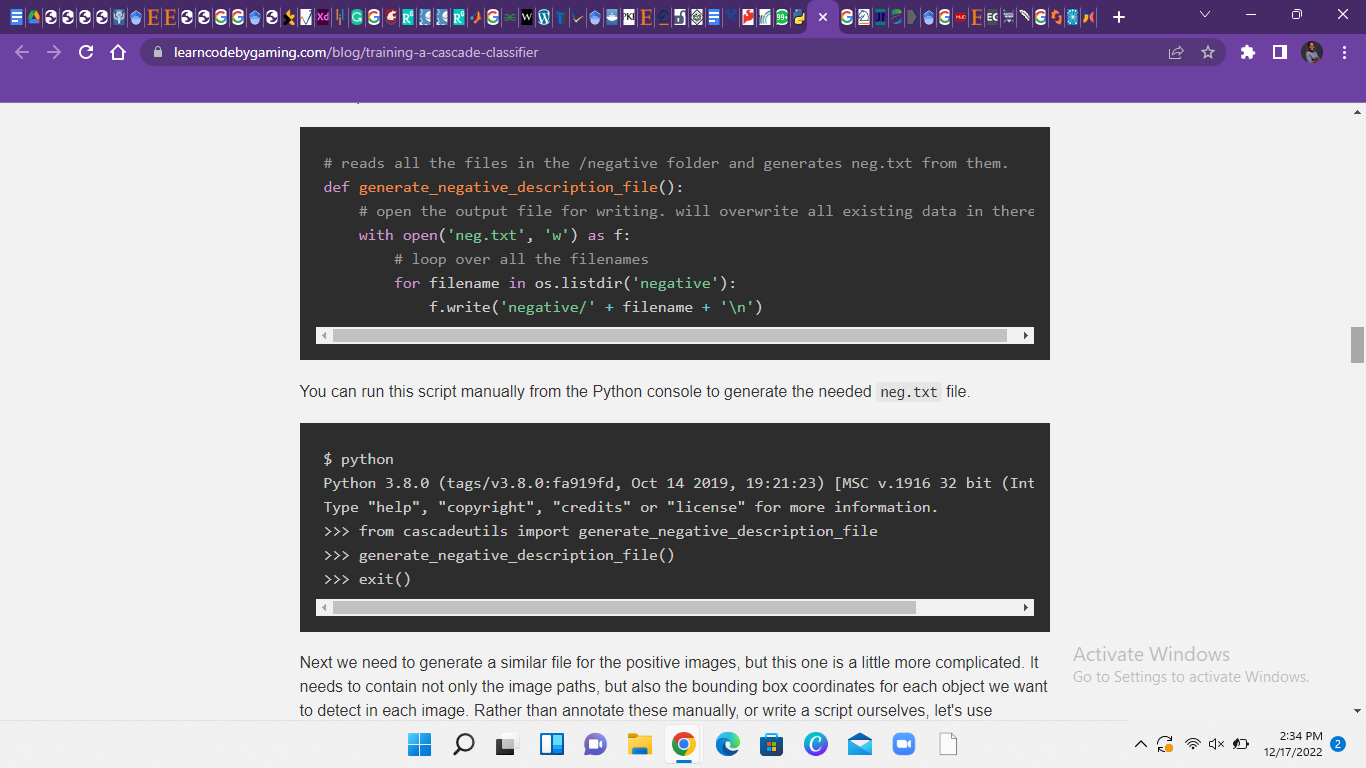


Fig 3.2.4. Code to generate a negative text file.

This script can be run manually from the Python console to generate the needed neg.txt file.

After doing this, we do the same thing to generate a similar file for the positive images. However this process is a little complicated compared to the first as it needs to contain additional information like the bounding box coordinates for each object we want to detect in each image.

We used the OpenCV annotation program since it is specifically designed specifically for creating this file.

But this simple *opencv\_annotation.exe* command line program is only available on the OpenCV version 3.4. So we installed the newest version of 3.4 to get access to it and other Cascade Classifier programs we might need.

After extracting the ZIP file, we found the executable codes we need in */opencv/build/x64/vc15/bin/* in the location we chose to save the extracted file.

All the programs we used were all in that folder: *opencv\_annotation.exe, opencv\_createsamples.exe,* and *opencv\_traincascade.exe*.

Even though we prepared our samples and trained our model with OpenCV 3.4, the classifier will still be usable in newer versions of OpenCV.

Now we can run the annotation program.

The annotation program opened each image in our positive folder one at a time in an OpenCV window. In each image we drew a box around the objects within it that we wanted to detect.

Next we created a vector file from all of our positive annotations.

The *-numPos* has to be an amount lower than the number of samples created by *createsamples*.

We get the error message: Can not get new positive sample when the *-numPOS* value is higher than the samples created. In this case, it is advised to either lower your *-numPos* or lower the *-minHitRate* ( the default value is 0.995).

A popular suggestion for *-numNeg* is to use half of *-numPos*. Using numbers higher than twice the number of negative to positive sometimes lead to better results.

The *-w* and *-h* must match what the values of the createsamples step.

The more *-numStages* the longer it will take to train the classifier.

We trained the classifier for 10 to 19. As the stages increased, the time it took to run became longer.

## 3.1.6. ROBOT PROGRAMMING METHODOLOGY

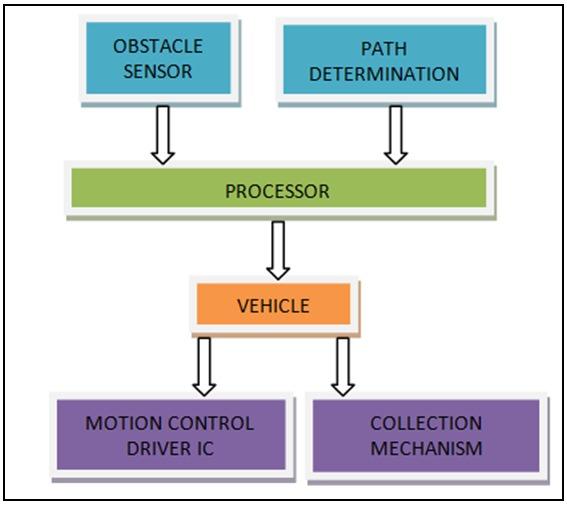


Fig 3.2.1 Workflow for Robot Programming

The Programming Language that was used for machine learning was Python on the Raspberry Pi and C on the Arduino Mega. We decided to use Python for the following reasons

1. We were familiar with Python Programming
2. Python has a lot library
3. It is very easy to set up Python on the Raspberry Pi.

Although the difference in performance between C++ and Python was negligible, we forfeited the former due to familiarity with the latter.

The code to initiate communication between the Raspberry pi and the Arduino is:

#!/usr/bin/env python

import sys

import serial

import threading

import time

import socket

class CommThread(threading.Thread):

def \_\_init\_\_(self, port, baudrate):

# script crashes without this

super(CommThread, self).\_\_init\_\_()

# save handle

self.comm = serial.Serial(port=port, baudrate=baudrate, timeout=1)

# an event that can be used for anything, here used to stop the thread

self.running = threading.Event()

def write(self, data):

# send data to the device

try:

return self.comm.write(data)

except:

return False

def setSpeed(self, speedLeft, speedRight):

# must be unsigned value

if speedLeft < 0:

speedLeft += 255

if speedRight < 0:

speedRight += 255

return self.write('s' + chr(speedLeft) + chr(speedRight))

def run(self):

print "Communication thread started"

# make sure we're still running

while not self.running.isSet():

# see if there is something in the incoming buffer

if self.comm.inWaiting():

self.process(self.comm.readline())

# wait for a bit or it will consume all the CPU

time.sleep(0.2)

def process(self, data):

if data == 'p':

s = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

# create a socket to an external website

s.connect(('google.com', 0))

ip = s.getsockname()[0].split(".")

# transform list of strings to list of integers

ip = map(int, ip)

# send the external ip address via serial to the arduino board

self.write(chr(ip[0]) + chr(ip[1]) + chr(ip[2]) + chr(ip[3]))

def stop(self):

print "Stopping communication thread"

self.running.set()

A code on the Raspberry Pi sends commands of direction to the Arduino Mega when searching for Pixels on an image and also gives directions for searching for bottles in the vicinity.

The code containing the commands for direction is given below:

#!/usr/bin/env python

import time

import sys

from cv2 import waitKey

import comm

import camera

# types of elements to detect

class RobotControl:

def \_\_init\_\_(self, port, baudrate):

# configuration for the serial connection to the robot

self.port = port

self.baudrate = baudrate

def connect(self):

# start serial communication thread

try:

print "Connecting to robot"

self.commThread = comm.CommThread(self.port, self.baudrate)

self.commThread.start()

except:

print "Could not connect to robot"

return

preview = False

# run the code with or without the preview, by default it's disabled

if len(sys.argv) > 1:

if sys.argv[1]:

preview = True

# start the camera thread

try:

print "Starting camera"

self.cameraThread = camera.ImageProcessing(256, 128, preview)

self.cameraThread.start()

except:

print "Could not start camera"

return

# start the control loop

self.control()

def control(self):

# wait for the camera and communication threads to start

while not self.commThread.isAlive() and not self.cameraThread.isAlive():

time.sleep(0.1)

pass

print "Starting robot control loop"

# play 880hz sound to indicate scropt start when working without a screen

print "Beep"

self.commThread.write('t' + chr(880 >> 8) + chr(880 & 0xff))

# turn on the led lights on the robot

print "Turning light on"

self.commThread.write('l' + chr(1))

print "Enable bottle detection"

self.commThread.write('0')

while True:

try:

self.stateSearching()

# check for keyboard input

if self.checkInput():

break

time.sleep(0.1)

except (KeyboardInterrupt, SystemExit):

print "User forced exit"

break

except Exception as e:

# when an error occurs make sure to end the dependant threads

print e

self.stopThreads()

break

# turn off the led lights on the robot

print "Turning light off"

self.commThread.write('l' + chr(0))

print "Disable bottle detection"

self.commThread.write('0')

print "Control loop exited"

self.stopThreads()

def stopThreads(self):

print "Stopping threads"

self.commThread.stop()

self.commThread.join()

self.cameraThread.stop()

self.cameraThread.join()

def checkInput(self):

# check for user input

key = waitKey(10) & 0xff

if key == 27:

print "Manually stopped"

return True

elif key == ord('w'):

self.commThread.write('s' + chr(self.maxSpeed) + chr(self.maxSpeed))

elif key == ord('a'):

self.commThread.write('s' + chr(-self.maxSpeed + 255) + chr(self.maxSpeed))

elif key == ord('d'):

self.commThread.write('s' + chr(self.maxSpeed) + chr(-self.maxSpeed + 255))

elif key == ord('s'):

self.commThread.write('s' + chr(-self.maxSpeed + 255) + chr(-self.maxSpeed + 255))

elif key == ord('1'):

self.commThread.write('u')

elif key == ord('2'):

self.commThread.write('d')

elif key == ord('q') or key == 32:

self.commThread.write('s' + chr(0) + chr(0))

elif key != 255:

print key

return False

def stateSearching(self):

data = self.cameraThread.checkDataQueue()

if data:

self.commThread.write("!" + chr(data["position"][0] + chr(data["position"][1]))

print "Bottle detected", data["position"]

robot = RobotControl("/dev/ttyACM0", 9600)

robot.connect()

The robot’s default state is to search for plastic bottles and the Arduino monitors this search process by command from the Raspberry Pi. When the Raspberry Pi on the robot detects a bottle, it focuses on it in the picture and sends commands to the Arduino Mega to move the robot either left or right to centre the picture of the plastic bottle it focused on.

The code to access the camera and run the machine learning model to detect the bottles is given below:

#!/usr/bin/env python

import io

import picamera

import threading

import cv2

import numpy as np

import Queue

# types of elements to detect

OBSTACLE, BOTTLE = range(2)

class ImageProcessing(threading.Thread):

def \_\_init\_\_(self, width, height, preview):

super(ImageProcessing, self).\_\_init\_\_()

self.obstacleThreshold = 100000

self.preview = preview

self.width = width

self.height = height

self.camera = picamera.PiCamera()

self.camera.resolution = (self.width, self.height)

# boost colors

self.camera.saturation = 100

#self.camera.shutter\_speed = 10000

#self.camera.awb\_mode = u'off'

#self.camera.exposure\_mode = u'fixedfps'

#self.camera.meter\_mode = u'spot'

#self.camera.exposure\_compensation = 10

#self.camera.framerate = 2

#self.camera.sharpness = 0

#self.camera.video\_stabilization = True

#self.configureCamera()

# load the xml file for

self.cascadeXml = cv2.CascadeClassifier('bottle.xml')

# load the mask that hides the non important parts of the image

self.mask = cv2.imread("mask.png", 0)

self.stream = io.BytesIO()

self.dataQueue = Queue.Queue()

# if enabled show the preview window

if self.preview:

cv2.namedWindow("Preview", flags=cv2.cv.CV\_WINDOW\_AUTOSIZE)

# event to stop the thread

self.running = threading.Event()

def run(self):

print "Image processing thread started"

# make sure we're still running

while not self.running.isSet():

# take the picture

self.camera.capture(self.stream, format='jpeg', use\_video\_port=True)

# construct a numpy array from the stream

data = np.fromstring(self.stream.getvalue(), dtype=np.uint8)

self.stream.truncate()

self.stream.seek(0)

# "decode" the image from the array, preserving colour in BGR format

self.img = cv2.imdecode(data, cv2.CV\_LOAD\_IMAGE\_COLOR)

# image that can be modified by the image processing for preview purposes

if self.preview:

self.previewImage = self.img

#self.detectObstacles()

self.detectBottles()

if self.preview:

try:

cv2.imshow("Preview", self.previewImage)

except:

pass

# destroy the preview window

if self.preview:

cv2.destroyAllWindows()

def stop(self):

print "Stopping image processing thread"

self.running.set()

def detectBottles(self):

# get a gray picture

gry = cv2.cvtColor(self.img, cv2.COLOR\_BGR2GRAY)

# run the haar cascade algorithm (slow)

haar = self.cascadeXml.detectMultiScale(gry, 1.1, 6)

# update preview

if self.preview:

for (x,y,w,h) in haar:

cv2.rectangle(self.previewImage, (x,y), (x+w, y+h), (255, 255, 255), 1)

# normalised value between 0 and 1 of the first bottle position

if len(haar):

bottleX = (haar[0][0] + haar[0][2] / 2);

bottleY = self.height - (haar[0][1] + haar[0][3] / 2) ;

self.dataQueue.put({"type": BOTTLE, "position": (bottleX, bottleY)})

def detectObstacles(self):

colorFloorL = np.array([8, 30, 30], np.uint8)

colorFloorH = np.array([28, 255, 255], np.uint8)

# convert to hue, saturation, value format

hsv = cv2.cvtColor(self.img, cv2.COLOR\_BGR2HSV)

colorThreshold = cv2.inRange(hsv, colorFloorL, colorFloorH)

colorThreshold = np.invert(colorThreshold)

cv2.erode(colorThreshold, cv2.getStructuringElement(cv2.MORPH\_ELLIPSE, (7, 7)), colorThreshold, (-1, -1), 1)

np.bitwise\_and(colorThreshold, self.mask, colorThreshold)

# average values vertically and split screen in 4 horizontally

sumVertical = np.sum(colorThreshold, axis = 0)

splitHorizontal = np.array([np.sum(sumVertical[0:63]), np.sum(sumVertical[64:127]), np.sum(sumVertical[128:191]), np.sum(sumVertical[192:255])])

# print "H", splitHorizontal

if self.preview:

# show obstacle position with the overlay on the image

mask = np.invert(colorThreshold)

np.bitwise\_and(self.previewImage[:,:,0], mask, self.previewImage[:,:,0])

np.bitwise\_and(self.previewImage[:,:,1], mask, self.previewImage[:,:,1])

np.bitwise\_and(self.previewImage[:,:,2], mask, self.previewImage[:,:,2])

pass

peakX = splitHorizontal.argmax()

# obstacle threshold has been reached, obstacle has been detected

if splitHorizontal[peakX] > self.obstacleThreshold:

# average values horizontally and split screen in 4 vertically

sumHorizontal = np.sum(colorThreshold, axis = 1)

splitVertical = np.array([np.sum(sumHorizontal[0:31]), np.sum(sumHorizontal[32:63]), np.sum(sumHorizontal[64:95]), np.sum(sumHorizontal[96:127])])

#print "V", splitVertical

peakY = splitVertical.argmax()

self.dataQueue.put({"type": OBSTACLE, "position": (peakX, peakY)})

def checkDataQueue(self):

if not self.dataQueue.empty():

return self.dataQueue.get()

else:

return False

When the robot centres the plastic bottle in the image detected, it moves in a straight line towards the bottle until it longer sees the image of the bottle. When this happens, it means the plastic bottle is in front of the robot.

When the image of the bottle disappears, the robot proceeds to retrieve the plastic bottle into the chassis, after which the cycle repeats starting from searching for bottles.

When the robot detects multiple bottles, it centres the first and gives it priority in retrieval before moving on the other bottles. The communication is between the Raspberry Pi from running the Algorithm of the machine learning model.

The Arduino accepts commands and processes the direction.

To enable the Arduino and the Raspberry Pi To communicate serially using the json is given below:

#include <ArduinoJson.h>

#define BAUD 115200

#define DEBUG true

int leftMotorSpeed = 0;

int rightMotorSpeed = 0;

void setup() {

Serial.begin(BAUD);//Arduino Serial port

}

void loop() {

// use data format like {"left":-255,"right":100}

StaticJsonDocument<50> data;

DeserializationError error = deserializeJson(data, Serial);

if (!error)

{

leftMotorSpeed = data["left"];

rightMotorSpeed = data["right"];

}

if (DEBUG) {

Serial.print("left: ");

Serial.print(leftMotorSpeed);

Serial.print(" | right: ");

Serial.println(rightMotorSpeed);

}

}

OTHER OBSTACLE DETECTION

This is achieved by using ultrasonic and infrared sensors to detect any obstacle that is not plastic bottles on the path of the robot. This is monitored by the Arduino Mega and when it notices an obstacle that is not plastic, it sends a signal to the Raspberry Pi.

The Raspberry Pi sends a command to disregard retrieval of plastic bottles and focus on obstacle avoidance. After the obstacle has been successfully avoided, the robot proceeds with searching for plastic bottles for retrieval.

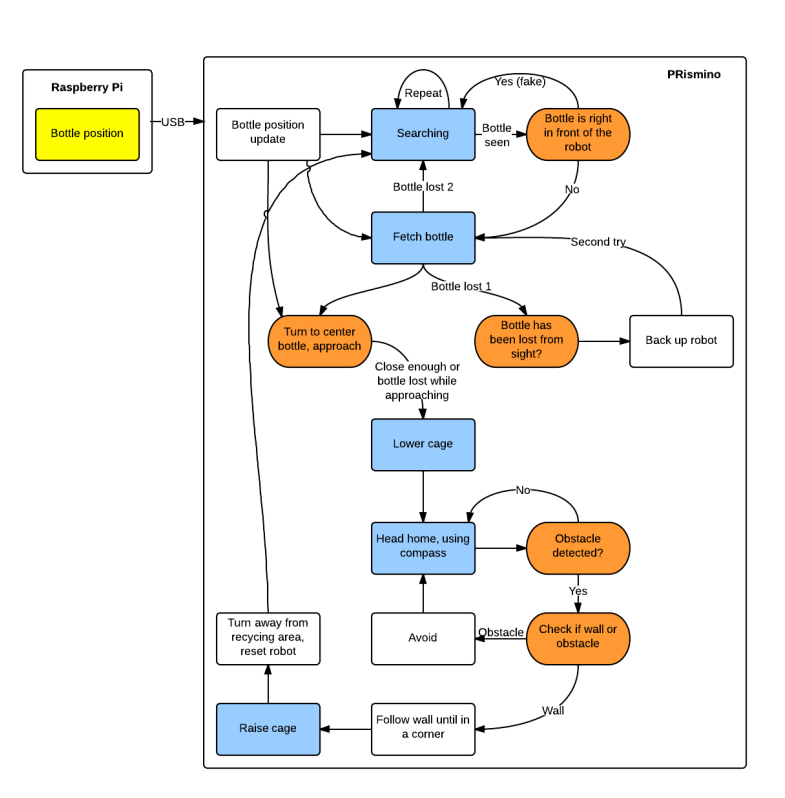


Fig 3.2.2 Visual representation of Robot working

# CHAPTER FOUR

# TEST RESULTS AND ANALYSIS

### 4.1. TESTING RESULTS

The table below gives the result of testing each classifier trained for different stages

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number of Stages | Bottles Detected | Actual Bottles | Number of False Positives | Number of True Positives |
| 10 | 540 | 200 | 343 | 197 |
| 11 | 432 | 200 | 237 | 195 |
| 12 | 329 | 200 | 139 | 190 |
| 13 | 299 | 200 | 110 | 189 |
| 14 | 203 | 200 | 14 | 189 |
| 15 | 129 | 200 | 9 | 120 |
| 16 | 119 | 200 | 4 | 115 |
| 17 | 98 | 200 | 2 | 96 |
| 18 | 75 | 200 | 0 | 75 |
| 19 | 50 | 200 | 0 | 50 |

### 4.2. OBSERVATION

We noticed that the lesser levels we trained for the the classifier detected more bottles but more false positives i.e under-training

While the more trained, the classifier detected less bottles( more misses) i.e over-training

We found balance in training for 14 levels which detected 203 bottles (189 true positive and 14 false positives)

The machine learning algorithm that was trained for the Haar’s cascade has an efficiency of 93.10%.

# CHAPTER FIVE

# CONCLUSIONS AND RECOMMENDATIONS

## 5.1. CONCLUSION

We were able to effectively train a classifier to detect plastic bottles from images with a 93% accuracy when it comes to positive results.

## 

## 

## 5.2. LIMITATIONS

The project only covered detecting one type of plastic bottle and also didn’t cover plastic bottle retrieval on any other surface other than a flat and smooth surface of the ground of LT1.

## 5.3. RECOMMENDATIONS

For future work, the programming language C++ can be used for programming the machine learning model. Also, the classifier can be trained to detect more than one type of plastic bottle.

Future work can also take into account plastic waste retrieval on a rocky terrain or a grass field for better efficiency.

## References

1. AHR international. (n.d.). *Electric motor bearings and their application*. AHR International. Retrieved December 12, 2022, from https://www.ahrinternational.com/bearings\_electric\_motor\_applications.htm
2. BasuMallick, C. (2022, August 26). *What Is Raspberry Pi? Models, Features, and Uses*. Spice Works. Retrieved 11 12, 2022, from https://www.spiceworks.com/tech/networking/articles/what-is-raspberry-pi/#:~:text=Raspberry%20Pi%20is%20defined%20as,PC%20at%20a%20low%20cost.
3. Britannica. (n.d.). *Robotics | Definition, Applications, & Facts | Britannica*. Encyclopedia Britannica. Retrieved December 5, 2022, from https://www.britannica.com/technology/robotics
4. Burnett, R. (2020, March 24). *Understanding How Ultrasonic Sensors Work*. MaxBotix Inc. Retrieved December 18, 2022, from https://www.maxbotix.com/articles/how-ultrasonic-sensors-work.htm
5. Components 101. (2019, April 18). *IRFZ44N MOSFET Pinout, Features, Equivalents & Datasheet*. Components101. Retrieved December 17, 2022, from https://components101.com/mosfets/irfz44n-datasheet-pinout-features
6. Dey, A. K. (n.d.). *Belt Drive: Types, Material, Applications, Advantages, Disadvantages [PDF] – Learn Mechanical*. Learn Mechanical. Retrieved December 18, 2022, from https://learnmechanical.com/belt-drive/
7. Ejimofor, M. I., Aniagor, C. O., Oba, S. N., Menkiti, M. C., & Ugonabo, V. I. (2022). *Artificial intelligence in the reduction and management of land pollution*. sciencedirect.com. Retrieved December 6, 2022, from https://www.sciencedirect.com/science/article/pii/B9780323855976000094
8. Elprocus. (2020). *Arduino Mega 2560 Board: Specifications, and Pin Configuration*. ElProCus. Retrieved November 17, 2022, from https://www.elprocus.com/arduino-mega-2560-board/
9. Jost, D. (n.d.). *What is an Ultrasonic Sensor?* Fierce Electronics. Retrieved December 18, 2022, from https://www.fierceelectronics.com/sensors/what-ultrasonic-sensor
10. Klearstack. (2021, August 17). *Image Processing and Machine Learning: Redefining the Future*. KlearStack. Retrieved December 10, 2022, from https://www.klearstack.com/image-processing-and-machine-learning/
11. Kumar, S. E. (2022, April). Arduino Working Principle and It’s Use in Education. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, *10*(IV).
12. Semcor. (n.d.). *How Do Conveyor Belts Work? | Belt Functions, Uses & Applications*. SEMCOR. Retrieved November 18, 2022, from https://www.semcor.net/blog/how-do-conveyor-belts-work/
13. Tripathi, M. (2021, June 21). *Image Processing using CNN | Beginner's Guide to Image Processing*. Analytics Vidhya. Retrieved December 9, 2022, from https://www.analyticsvidhya.com/blog/2021/06/image-processing-using-cnn-a-beginners-guide/
14. Utmel Electronics. (2021, December 3). *H-bridge: Working, Circuits and Applications*. Utmel Electronics. Retrieved December 5, 2022, from https://www.utmel.com/blog/categories/motors/h-bridge-working-circuits-and-applications
15. Utmel Electronics. (2022, January 13). *D882 Transistor: NPN Medium Power Transistor, Pinout, Equivalent, Uses*. Utmel. Retrieved December 18, 2022, from https://www.utmel.com/components/d882-transistor-npn-medium-power-transistor-pinout-equivalent-uses?id=1157
16. *What Are Bearings? Let's learn about the basic functions of bearings*. (2019, June 13). Koyo Bearings(JTEKT). Retrieved December 11, 2022, from https://koyo.jtekt.co.jp/en/2019/06/column01-01.html
17. The World Bank. (2019). What a Waste: A Global Review of Solid Waste Management. *International Journal For Environmental Research And Public Health*, (1060), 15-30. Pub Med Central. Retrieved November 28, 2022, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6466021/#B4-ijerph-16-01060