A Project Report on

Design and Fabrication of Automatic Color Sorting Machine

By

Mr. Pawan.S.Sawant Exam Seat No: 71712751G

Ms. Nidhi.R.Bhattad Exam Seat No: 71711979D

Ms. Yugandhara.S.Kolte Exam Seat No: 71712359G

Mr. Pranav.P.Bhandari Exam Seat No: 71711970L

Guide

Prof G.B.Firame

Sinhgad Institutes

Department of Mechanical Engineering

Sinhgad Technical Education Society's

Smt. Kashibai Navale College of Engineering

[2019-20]

Sinhgad Technical Education Society's

Smt. Kashibai Navale College of Engineering

Sinhgad Institutes

C E R T I F I C A T E

This is to certify that

Mr. Pawan.S.Sawant	*Exam Seat No: 71712751G*
Ms. Nidhi.R.Bhattad	*Exam Seat No: 71711979D*
Ms. Yugandhara.S.Kolte	*Exam Seat No: 71712359G*
Mr. Pranav.P.Bhandari	*Exam Seat No: 71711970L,*

have successfully completed the Project Stage – I entitled "*Design and Fabrication of Automatic Color Sorting Machine*" under my supervision, in the partial fulfillment of *Bachelor of Engineering-Mechanical Engineering* of Savitribai Phule Pune *University.*

Date: - 30/04/20

Place: - Pune

Prof G.B.Firame
Guide

Prof……..
External Examiner

Dr. N.P. Sherje
Head of Department

Dr. A.V. Deshpande
Principal

ACKNOWLEDGEMENT

We take this opportunity to thank all those who have contributed in successful completion of this Project. We would like to express our sincere thanks to our guide**Prof G.B. Firame** who have encouraged us to work on this topic and provided valuable guidance wherever required. We also extend our gratitude to **Dr. N. P. Sherje (H.O.D Mechanical Department)** who has provided facilities to explore the subject with more enthusiasm.

We express our immense pleasure and thankfulness to all the teachers and staff of the **Department of Mechanical Engineering of Smt. Kashibai Navale College of Engineering**for their co-operation and support.

Mr. Pawan.S.Sawant

Ms. Nidhi.R.Bhattad

Ms. Yugandhara.S.Kolte

Mr. Pranav.P.Bhandari

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Nowadays, the main obstacle that is encountered after the production is of sorting. Arranging items in an industry is a dull modern process, which is by and large done physically. As a result, an automated system for sorting is greatly needed to replace a manual sorting system. Automation has and will continue to lead to the growth of industries. In this paper, we have recommended an automated system that sorts objects according to their color using TCS230 color sensor, Raspberry Pi 3B+(Raspbian Operating System), Conveyor belt, and servo motors. The identification of the color is based on the frequency analysis of the output of the color sensor. This machine is open-ended and can be further used to sort products based on their weights, size, material, etc. by using a different type of sensor and some changes in the code.

Keywords: Color Sorting, Raspberry Pi 3B+, TCS230 color sensor, Conveyor belt, Servos.

1. Introduction

1.1 Relevance

We're in the midst of a significant transformation regarding the way we produce products, thanks to the digitization of manufacturing. This transition is so compelling that it is being called Industry 4.0 to represent the fourth revolution that has occurred in manufacturing. From the first industrial revolution (mechanization through water and steam power) to the mass production and assembly lines using electricity in the second, the fourth industrial revolution will take what was started in the third with the adoption of computers and automation and enhance it with smart and autonomous systems fuelled by data and machine learning.

Industry 4.0 is all about automation and the Internet of things. Automation provides mechanical assistance by using a control system. This reduces manual efforts done by a human, time consumed, and also improves the time to market, and thus it has gained a lot of scope in industries. The research on automation and robotics has shown importance in:

- Industries
- Defense
- Surveillance and Security

Sorting of objects is necessary for industries where products are manufactured on a large scale. This process is simplified by automation. Objects are classified based on different characteristics like shape, color, and weight. The purpose of this model is to design and implement a system that automatically separates products based on their color. Color-based sorting is used in many industries like food-processing factories, agricultural machinery, etc.

This machine consists of three parts: conveyor belt, color sensor, and servos. The output and input of these parts were interfaced using Raspberry Pi. Raspberry Pi is a Linux based operating system which allows user to perform and develop task efficiently. With the help of the color sensor, the color of the object is detected. This detected color is used as an object sorting parameter by Raspberry Pi.

Some of the applications of the proposed machine are mentioned below:

1) Food processing factories: The machine can segregate between rotten and ripen fruits used for local and export business. Apples, dates, figs, plums, cherries are some fruits from the extensive list of fruits that can be sorted using the machine.

2) Agricultural: Scaling and grading of a diverse variety of cereals can be done according to their appearance using this machine. Rice, peanuts, beans, maize can be graded using sorting machines.

3) Recycle industries:Materials can be recycled and reused as raw material. Metals, plastic, glass, electronic, and electrical waste are some of the products that can be sorted based on their color.

The machine that we have designed is not only efficient but also cost-effective. The machine is built at 20% of the annual income paid to an individual operator with a nominal maintenance cost. Equipping such machines proves beneficial as it gives accurate results and saves time with no loss due to fatigue.

1.2 Problem Statement

The main problems with human labor being used for sorting or the current machines prototyped for sorting are:

- Worker fatigue on assembly lines can result in reduced performance and causes consistency issues as well.
- The machines that are prototyped till now use PLC or Arduino Uno. The problem with these is that PLC is very expensive, and Arduino UNO is not computational that powerful.
- The time taken by these systems is also very high.
- The accuracy and color range of these machines is also limited.
- The suggested cost of the current machines is very high because of using PLC or other expensive operating systems.

The proposed machine is aimed at tackling the above mentioned problems.

1.3 Research Gap

The referenced papers suggest that a number of studies have been carried out to develop an automatic sorting machine based on different properties. It is evident from the research that importance of automatic sorting has been increased than the regular manual segregation process that was carried out earlier. The objective of this section has been aimed to report the work carried out by various researchers in this field and report the areas untouched by them. After an elaborate scrutiny of the published work, the following gaps in the literature have been observed:

1) Systems designed using Arduino have a slow running time.

2) PLC system is only economical for large scale production.

3) The system designed using Raspberry Pi is small in size and can sort products of smaller dimensions only.

4) Previous systems built using Raspberry Pi use a camera module, which is expensive and takes greater time to detect and sort objects as well.

5) Systems designed using camera modules are not suitable for compact design, as for accurate results, the camera module needs to be placed minimum 20cm above the object

6) Also, the algorithm used with the camera module combines a variety of algorithms and thus is computational heavy on the operating system.

1.4 Proposed Work

Our proposed machine will solve all the above-mentioned problems by using some new systems. The solutions we have developed for the above-mentioned issues are mentioned below:

1) Our machine uses Raspberry Pi, which is highly optimized computationally and thus executes programs faster than Arduino.

2) The Raspberry Pi costs a fraction of the cost of PLC and thus helps reduce and bring down the cost of the entire machine.

3) Our machine uses a color sensor along with designed mechanisms that will help detect objects of larger sizes and segregating them by overcoming the difficulty of oldercolor sorting machines having a size restriction for only smaller objects.

4) The color sensor we are using is cheap and fast, thus solving the issue of the camera module.

5) The algorithm designed for the color sensor is highly optimized and thus requires very little computational time and power.

6) The machine is designed to be flexible to sort objects based on weight, material, or other parameters as well by changing the sensor and some minor changes in the code.

1.5 Methodology

In the ever-growing industrial sector, efficiency, quality, and reliability are significant for supplier-consumer relations. Understanding the market scenario is critical for every sector of the industry to enhance market performance. This need is seen even in the packaging and sorting industry. Arranging items in an industry is a tedious modern process, which is still done physically. Worker fatigue on assembly lines can result in reduced performance and can cause consistency issues as well. Therefore, we have designed a system that operates with the help of color sensor TCS230 to detect different colors. The color sensor is coded using the Python scripting language to recognize the color and feed it as input to the Raspberry Pi. Raspberry Pi is used as the controller to control hardware components like the servo motors. Raspberry Pi is a portable computer with the ARM11 architecture, which runs on a Linux Debian environment. The proposed system has an inclined passage through which the objects are placed in the system. The passage consists of a color sensor that detects the color and feeds this to the Raspberry Pi. The Raspberry Pi then instructs the servo motor to open the flap at the end of the inclined passage for the object to slide onto the conveyor belt. The conveyor belt, which is controlled by a motor which is powered by DC supply, then passes on the object to another inclined passage. In this passage, two servos are mounted, which segregate the objects into three containers depending on their color.

2. Literature Review

2.1 Paper 1: Automated Object Sorting Based on Color Detection

Sorting is one of the most vital industries today. This paper proposes an automatic sorting system using Raspberry Pi 2, USB camera, and OpenCV library in Python. Color Detection is done using the HSV model i.e. Hue (H), Saturation (S), and Value (V) of color. Systems proposed in older papers use RGB (Red Blue Green) color model. However, the RGB model defines a specific shade and the values of Red, Blue, and Green change with illumination resulting in erroneous operations. Thus, the main purpose of this paper is to implement the HSV model with OpenCV to increase the accuracy, efficiency, and optimize the system. Hue (H) defines the color itself, Saturation(S) is the amount of grey from 0 to 100%, and Value (V) is the amount of light gravitating on the object. The paper finds that the Hue value increases linearly from red color. The HSV values range from [0 0 0] to [179 255 255] in OpenCV. For a discretecolor, the Hue value lies in a range which is given in the paper. Amongst the colors given in the paper, the least is for Red (0-10) and maximum for Pink (160-179) with Orange, Yellow, Green, Sky Blue, Dark Blue, and Violet being in between the two. The database for the system is created by capturing an image of the entered object by using the camera model with Raspberry Pi. The captured image is in RGB form (which is taken as BRG in OpenCV). It is then converted to HSV form. The hue component of some pixels is extracted from the converted HSV image. Average Hue value is compared with the lower and upper range of each color, and then the value of each color is stored in the database. The system operates by capturing multiple images with the USB camera. The hue component is extracted from the image and compared with the color database. The paper concludes that the experiment performed using the HSV system was more accurate than the RGB system. The biggest advantage of the HSV system was that the Hue component remains in range even if the illumination changes.

2.2 Paper 2: Automatic Color Sorting Machine Using TCS 230 Color Sensor and PIC Microcontroller

This paper describes a working model of automatic color sorting machine using TCS230 color sensor and Programmable Interface Controller (PIC) Microcontroller. The main motivation for designing this machine was:

- To check the quality of the product
- Ease of sorting and packing
- Monitor waste products
- Assess the equality of products in storage

The system favoured using low cost and simple color sensor rather than more sophisticated solutions, which are more power-consuming and costly. In this paper, objects considered for sorting were bottle caps with a diameter of 2.5cm and a height of 1cm. To maintain the distance between the object and the sensor, a window of width 17cm and height 6cm is created using four aluminum rods, and a fiber plate. This structure is used to suspend the color sensor above the conveyor. If the object deviated even a bit from its ideal position, an error would be probable in the detection.

The identification of the color is based on the frequency analysis of the output of the TCS230 sensor. The machine is designed for automatic sorting of Red, Green, and Black colored objects. The prototype consists of two conveyors, each driven by separate DC motors. The second conveyor is placed at the end of the first conveyor and a certain height below. The first conveyor is where the object's color is detected. The second conveyor consists of separators wherein the sorted objects fall. The separator consists of three compartments for three different sorted objects. The working of the model starts when 3.4 volts DC current is supplied to the motor, which controls the movement of the conveyor on which the objects are placed. When the light falls on the product and it is reflected in the color sensor. The sensor converts light to a certain frequency depending upon the color. When there is no object in front of the sensor, it produces a frequency of 330Hz, whereas if there is an object, frequency in the range of 7-14 kHz is generated by the sensor. Then, the sensor sends the signal to the PIC microcontroller, which instructs the L293D hybrid IC to control the motion of the second DC motor. This motor will control the forward and backward motion of the second conveyor. In this way, the three colors are sorted. The object once placed on the first conveyor belt takes less than half a second to reach the sensor. It takes another 200ms for the sensor to detect the color. An additional 0.6secs is required if the color of the object is not black to position the correct compartment in the sorting container, which implies that an additional 0.6secs will be consumed to reposition the container back to the normal position on the second conveyor belt.

2.3 Paper 3: Object Sorting Using Image Processing

The prime focus of this paper is to sort objects based on their color with the help of Raspberry Pi 3 and a combination of object recognition algorithms. The intention of creating this prototype was to automate the object segregation process based on the properties of the object. OpenCV (Open Source Vision Library), a powerful image processing library along with Python as a scripting language is used for the identification of the objects. The algorithm is constructed by using four algorithms, namely;

- Bag of Features
- SIFT
- K-means clustering
- SVM classification.

Bag of features is a popular classification method. It is used to extract the features from the input images to build the codebook. The codebook is then used to compare the features of the unknown image with the features in the codebook. A database of N classes is contrived that are trained for classification. The images of an object can look divergent in different lighting conditions; they can have elevated levels of noise, etc which will retard the results. To avoid this, the SIFT method is used. Scale-Invariant Features Transform (SIFT) method is used to extract meaningful blob-like feature points of the object. These features are in the form of small arrows, pointing in various directions which removes the effect of lighting, noise, and other factors on the image, thus preventing the algorithm from making errors while predicting. Thereafter clustering is done to group all image features of different objects within the same class. K-means clustering is used for this purpose. In K-means, similar data is clustered together by assigning random centroids. The Euclidian distance is calculated to find the data points nearest to a particular centroid. After locating the points, the value of the centroid is recalculated, and the centroid is repositioned.

Whenever a new point is appended, it is placed in the cluster with the nearest centroid, and the value of the centroid is then recalculated. These steps are iterated until the centroid stops changing. K-means is used to provide labels for each data point which is then given as input to SVM classifier. Each data point clustered in this manner represents a vector that contains a weighted count of each visual word. The number of occurrences of each word is calculated, which indicates the probability of words in each class object. To classify the input data Support Vector Machine (SVM) classifier is used. SVM is a linear classifier which builds a model that can assign a class to a new data point by training it and thereby classifying the new image and recognizing it. In the proposed system CPU is used to process the image from webcam, and Raspberry Pi 3 as a controlling unit to drive various hardware devices. Socket communication is used whenever a new object is detected on the server computer. The server computer sends a code to the client Python script running in Raspberry Pi. Raspberry Pi further runs the corresponding thread and activates the actuators, thereby sorting the objects.

2.4 Paper 4: IOT Color Based Object Sorting Machine

The study includes tackling the main hindrance that is faced after production in many large scale and small scale industries, which is sorting. Within the analysis, an Arduino is used along with the Internet of Things (IoT) ideas to build a robotic arm to filter materials based on color. Arduino UNO is used to analyze the specifics of the object. To carry out the process color sensor, various servo motors, and a control board was used. The color sensor works by using a photodiode, which measures the power reflected by the protest for a red, green, and blue light source. The photodiode has an 8 * 8 matrix resulting in 16 of green filters, 16 of red filters, 16 of blue filters, and the remaining 16 of a transparent filter resulting in the absence of color. This will aid in the identification of the specified color. Color image processing is required for detection. Images are captured, transmitted, and positioned in the advanced frame in image processing. Computerized image processing is one of the electronic domain sectors where the image is converted to pixels, packed away in advanced stockpiling and prepared by the PC. As a result, the cost, processing speed, and adaptability are reduced. Image processing stimulates intrigue enhancement of picture highlights, and helpful details from the enhanced model may be found. Upon capturing the picture from the camera, the system shares a framework with a PC using distinguishing procedures. Image processing is achieved with the following steps-

- Taking the photograph using an optical scanner or electronic process
- Examining and adjusting the image that combines evidence load and portrait headway
- And detecting plans, not to mortal senses such as digital TV imaging.

The yield image can be modified representation based on the interpretation of the data. With the aid of a color sensor and multiple servo motors, a robotic arm is used to separate objects according to their color. A microcontroller is used to help guide the creation of an automatic arm to pick the articles on a transport track. In this project, the robotic arm groups the shaded objects that carry out transport by selecting and positioning the objects in their unique pre-customized habitat.

The machine's basic operation can be defined as follows:

- Objects to be separated are fed in the tube.
- A color sensor senses the objects coming into its sight, and code for the same is programmed in such a way that only the desired object colors are detected and retrieved at the end using servo motors in the bin.
- The program got its output when the desired color LED was turned on, and the output of the sensing of color was seen when any color from red, green, or blue was kept for detection in front of the color sensor.

This system's key advantage over other comparable systems was that it was simple, precise, reduced human errors, and improved accuracy. This framework found applications in the vegetable industry, malls, and other similar locations where color based segregation was required.

3. Components

3.1 Raspberry Pi 3b+

Numerous components are available in the market for image processing and controlling interfaced hardware. Raspberry Pi is a compact device with an open-source and is a flexible platform for experimentation. In addition to C and C++, it can also be programmed in Python and JAVA. It provides flexibility to append or make changes in the program easily. It is a Linux based board. Different software can be installed on Raspberry Pi as per the user's requirement. The precedence that Raspberry Pi has over other systems is that:

i. It is faster than Arduino
ii. It is cheaper than PLC
iii. It can detect and sort objects of smaller dimensions.

Raspberry Pi has a fast 64-bit 1.4GHz quad-core processor, 1GB of RAM, fast dual-band 802.11 b/g/n/ac wireless LAN, Bluetooth 4.2, and significantly faster 300Mbit/s ethernet.

- 4 x USB 2.0 ports
- 40 GPIO pins
- Full-size HDMI 1.3a port
- Combined 3.5mm analog audio and composite video jack
- Camera interface (CSI)
- Display interface (DSI)
- MicroSD slot
- VideoCore IV multimedia/3D graphics core @ 400MHz/300MHz

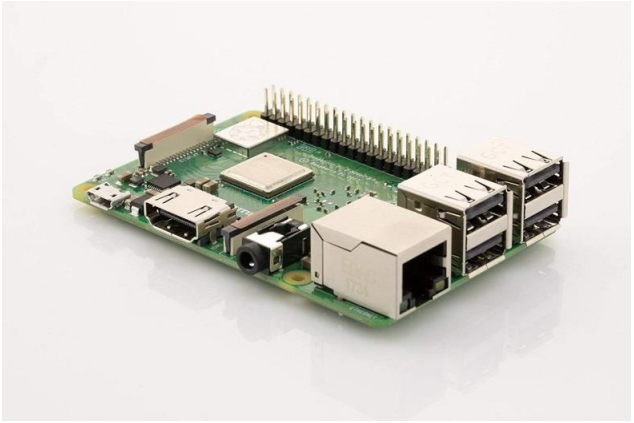

Fig 1: Raspberry Pi 3b+

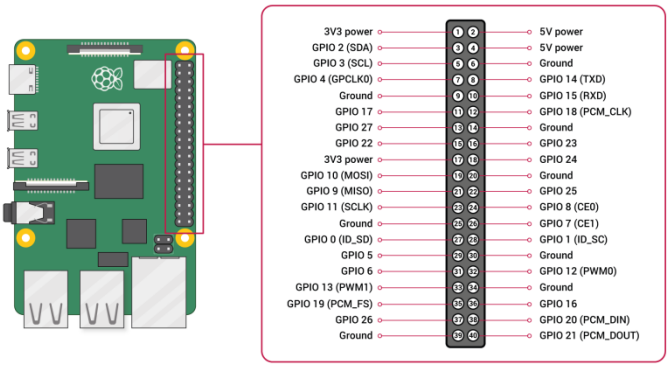

Fig 2: Raspberry Pi 3b+ Pin Configuration

3.2 Color sensor TCS 230:

The choice of TCS 230 was an unambiguous one because of its fast detection time, cheap cost, and small size. It incorporates a TAOS TCS230 RGB sensor chip and 4 white LEDs. The main part of the module is the TCS230 chip, which is a Color Light-to-Frequency Converter. The white LEDs are employed for rendering suitable lighting for the sensor to detect the object color correctly. This chip can sense a comprehensive variety of colors, and it gives the output in the form of corresponding frequency. This module can be used for making color sorting robots, test strip reading, color-matching tests, etc.

Some of the features of TCS 230 are:

- Input voltage: (3.3V to 5.5V)
- Programmable color and full-scale output frequency
- No need of ADC
- Working temperature: -150C to 600C

Pin Name	Description
GND(4)	Power supply ground
OE(3)	Enable for output frequency(active low)
OUT(6)	Output frequency
S0, S1(1,2)	Output frequency scaling selection inputs
S2, S3(7,8)	Photodiode type selection inputs
VDD(5)	Voltage supply

Table 1: Pin Configuration

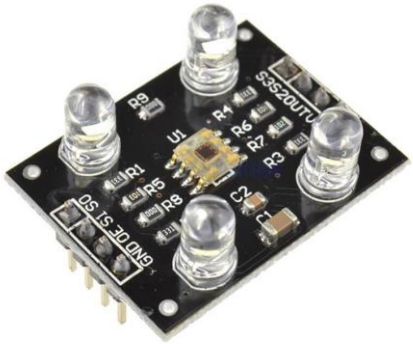

Fig 3: TCS 230 Color Sensor

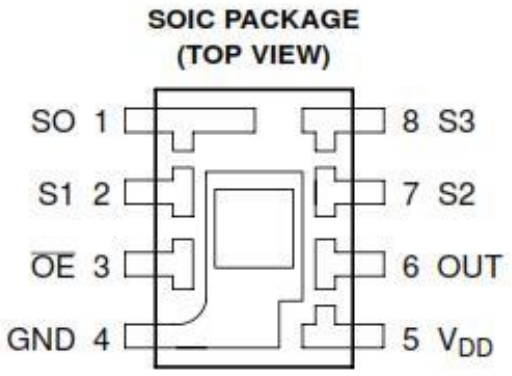

Fig 4: TCS 230 Color Sensor Pin Configuration

3.3 Servo motor

A servomotor is an elementary electric motor, governed with the help of servomechanism. It is compact and lightweight, with high output power. They are used for special applications where the rotation of the motor is required for just a certain angle. This needs some special arrangement which makes the motor rotate a certain angle for a given electrical input (signal), which is known as servomechanism. This servo can rotate approximately 180 degrees (90 in each direction). Any servo code, hardware, or library can be used to control these servos. It comes with 3 horns (arms) and hardware.
Characteristics of servomotor include

- Motion accuracy: Closed-loop mechanism, with an encoder or another device giving accurate feedback on the actual motor path.
- Torque and speed: Consistent torque output over a wide speed range

FUTABA S3003 servo was chosen having specifications as listed below:

Control System:	+ Pulse Width Control 1520usec Neutral	Current Drain (4.8V):	7.2mA/idle
Required Pulse	3-5V Peak to Peak Square Wave	Current Drain (6.0V):	8mA/idle
Operating Voltage:	4.8-6.0V	Direction:	Counter Clockwise/Pulse Travelling 1520-1900usec
Operating Temperature Range	-20 to +60 Degree C	Motor Type:	3 Pole Ferrite
Operating Speed (4.8V):	0.23sec/60degree at no load	Potentiometer Drive:	Indirect Drive
Operation Speed (6.0V):	0.19sec/60degree at no load	Bearing Type:	Plastic Bearing
Stall Torque (4.8V):	44oz/in. (3.2kg.cm)	Gear Type:	All Nylon Gears
Stall Torque (6.0V):	56.8oz/in. (4.1kg.cm)	Connector Wire Length:	12”
Operating Angle:	45 Deg. One side pulse travelling 400usec	Dimensions:	1.6” x 0.8” x 1.4” (41 x 20 x 36mm)
360 Modifiable:	Yes	Weight:	1.3oz. (37.2g)

Table 2: Servo Motor Specifications

Fig. 5: Servo Motor

3.4 DC Motor

A Brushless DC Motor (BLDC) is utilized as a driving mechanism to the conveyor belt. It has diverse advantages over its brushed counterparts. These advantages include:

- Higher efficiency
- Better performance
- Lower Susceptibility to mechanical wear

Brushless motors offer several other advantages, including:

- Higher torque to weight ratio.
- Increased torque per watt of power input which leads to increased efficiency.

Specifications:

- Robodo REL_9 Johnson Geared Motor 12V DC Gear
- Speed 300 RPM

Fig 6: Brushless DC Motor

3.5 Adapter

The proposed system uses an AC power supply for motor actuation. As DC supply motor was finalised because of its advantages, an adapter is used for signal conversion from AC to DC supply.
Specifications:

- Make: Sun-Power
- Input: 90-270V-0.8A, 50-60Hz
- Output: 5V

Fig 7: AC Adaptor

4. Designing of Components

4.1 Main Frame

The main intent behind designing the frame was to deal with the static and the dynamic loads that will be arising while creating a structure on which various sub-components can be mounted. In this project, we have manufactured two frames:

- Main Frame
- Sorter

The Main Frame incorporates a conveyor belt, an inclined box with a color sensor and servo, and all the electronic components. The Sorter is employed to sort objects based on the instructions it receives from the Raspberry Pi. Remembering that the machine will not be stationary and might need to be relocated between various departments in the industry, two separate frame structures were created forthe mobility of the machine.

Before designing the frame, we made some assumptions:

1) As the object insertion is manual, the height ofthe object insertion cavity was decided based on average human height (Indian).

2) The length of the conveyor depends on the distance at which the object needs to be transported. As we are creating a prototype, we assumed a length of 750mm, and accordingly, the main frame was designed.

3) For the sorter frame, we iterated various slope angles manually before designing. To achieve optimum velocity for segregation, we found that a minimum slope of 25^0 was required.

4) The size of the object the machine will be sorting is in the range of 60x20x10mm to 200x130x80mm. Hence, a width of 150mm was taken for the belt. This was used as the basis for deciding the width of the frame.

5) Since we were creating a prototype, Mild Steel was used for manufacturing the frame.

The entire CAD model of the frame is designed on SOLIDWORKS-2017 Version.

Dimensions of Main Frame:

- Length : 1000mm
- Width : 500mm
- Height : 1400mm

Following are the images of dimensions of the frame structures. (All the dimensions are in mm)

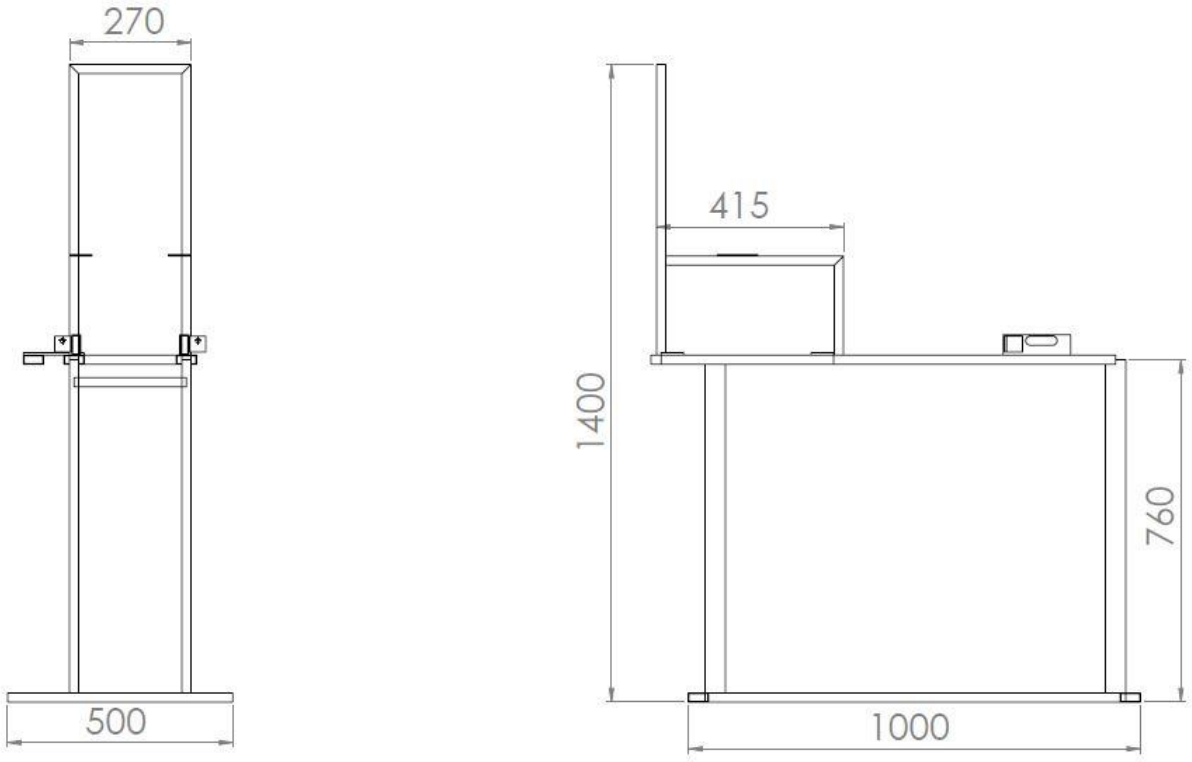

Fig 8: Front view and Side view of the Main Frame.

4.2 Conveyor System

In the processing or manufacturing industry, raw materials and products need to be transported from one manufacturing stage to another. Material handling equipment is designed such that they facilitate easy, cheap, fast, and safe loading and unloading with least human interference. The conveyor system is a type of mechanical handling system that moves material from one location to another. The path of flow of material is either horizontal or inclined, depending upon the requirement of the transportation. The length of the conveyor depends on the distance at which the object needs to be transported.

For our machine, we have used a belt conveyor. A belt conveyor consists of an endless and flexible belt of high strength with two end pulleys (driver and driven) at fixed positions.The material of the belt used is PVC.

The load handled by conveyor systems is of two types:

1) Bulk load
2) Unit load

In this project, we will be sorting the single boxes at a time, and hence load on the conveyor is Unit Load.

Considering the load, we decided to use a flat belt pulley along with our conveyor because:

1) Low power required for operation due to low frictional characteristics.
2) They require minimum maintenance.

Dimensions of Conveyor:

- Length: As we are creating a prototype, we assumed a length of 750mm on each side and a total length of 1800 mm.
- Width: The machine will be sorting objects in the range of 60x20x10mm to 200x130x80mm. Hence, a width of 150mm was taken for the belt.

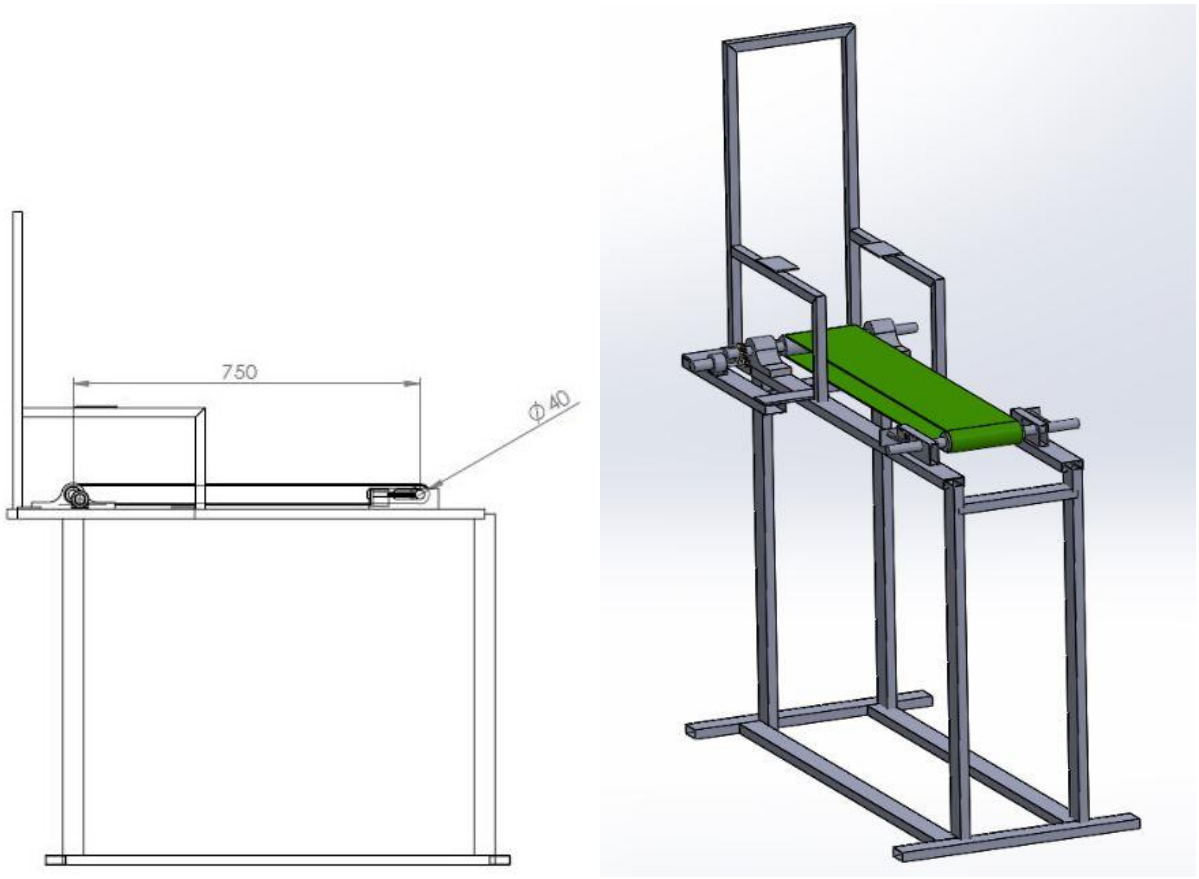

Fig 9: Conveyor system

4.3 Design of Gear

Initial design considerations-

1) A 20^0Pressure Angle (α) system was adopted as it reduces the risk of undercutting and interference.
2) The minimum number of teeth to avoid interference and undercutting for a 20° full depth system is 17 teeth. Hence, the Pinion is designed having 17 teeth.
3) Modules of 1mm, 1.25mm, 1.5mm, etc. were available with the manufacturer. Thus, a module of 1mm was chosen for the gear system.
4) The standard speed for most unit handling conveyors is 65 FPM (feet per minute) which is 0.3m/s. Also considering pulley diameter as 40mm. Therefore, the speed of driven gear is approximately 150RPM.

Calculations:

v=0.3m/s

Radius of pulley (r) = 20mm

Therefore, Ng= 150RPM (approx)

Assumed Data: Module m=1mm

Number of teeth on Pinion (Z_p) = 17

Pitch Circle Diameter of Pinion (d_p) = 17mm

Gear ratio (i) =2

I. Number of teeth on gear (N):

$i=\frac{N}{n}$

Therefore, N=i*n

N=34

II. Pitch circle diameter of gear(d_g)

$m=\frac{dg}{Zg}$

Therefore, $d_g = m*Z_g$

d_g=34mm

Pitch circle diameter of pinion (d_g) = 34mm

III. Speed of pinion(n)

$i=\frac{n}{N}$

n=i*N

Therefore,n=300RPM

IV. Center distance (a)= $\frac{m*(Zg+Zp)}{2}$

a = 25.5mm

V. Addendum and Dedendum

Addendum (h_a) =m= 1mm

Dedendum (h_f) =1.25m=1.25mm

VI. Tooth Thickness = 1.5708m

= 1.5708mm≅ 1.6mm

VII. Whole depth (h) = 2.25m

= 2.25mm

Material used for manufacturing of Gear is EN19.

The Gear and Pinion are manufactured using the same material, thus the Pinion will be weaker. Hence, all the calculations are done considering Pinion.

Torque provided by the motor (M_t) = 8kg.cm

= 800 N.mm

Force Analysis:

Tangential Component of force on the pinion is given by,

$$P_t = \frac{2M_t}{d_p}$$

$$P_t = \frac{2 * 800}{17}$$

$$P_t = 94.117N$$

Radial Component of force on the pinion is given by,

$$P_r = P_t \tan(\alpha)$$
$$P_r = 94.117 * \tan(20^0)$$
$$P_r = 34.255N$$

Resultant force is given by

$$P_N = \frac{P_t}{\cos(\alpha)}$$

$$P_N = \frac{94.117}{\cos(20^0)}$$

$$P_N = 100.157N$$

Beam Strength Calculation:

Bending Moment (M_b) = P_t*h

= 94.117*2.25

M_b = 211.763 N.mm

Moment of Inertia of tooth (I) = $\frac{b*t^2}{12}$

$=\frac{15*1.6^2}{12}$

I = 5.12 mm^4

Bending stress (σ_b) = $\frac{M_b*y}{I}$ = $\frac{M_b*(\frac{t}{2})}{I}$

$= \frac{211.763*(\frac{1.6}{2})}{5.12}$

σ_b = 33.088 N/mm^2

For 20^0, full depth involute system and for 17 teeth the Lewis Form Factor(Y) is 0.302.

Therefore, Beam Strength (S_b) = m.b.σ_b.Y

= 1*15*33.088*0.302

S_b = 149.888 N

Wear Strength Calculation:

Q = $\frac{2Z_g}{Z_p+Z_g} = \frac{2*34}{17+17}$

Q = 1

K= 0.16 $(BHN/100)^2$ = 0.16* $(250/100)^2$

K= 1

Therefore, wear strength (S_w) = b.Q.d_p.K

= 15*1*17*1

= 255 N

Effective Load Calculations:

$C_v = \frac{3}{3+v} = \frac{3}{3+0.3}$

C_v= 0.909

Service factor C_s for given electric motor can be considered as 1.

Therefore, the effective load (P_{eff}) = $\frac{C_s}{Cv} P_{eff} = \frac{1}{0.909}$*94.117

P_{eff}= 103.539N

Here, beam strength is less than wear strength. Therefore, wear strength is criterion for design.

Factor of Safety (FOS) = $\frac{S_w}{P_{eff}} = \frac{255}{103.539}$

FOS = 2.46

Since, the factor of safety is greater than 1, the design is safe.

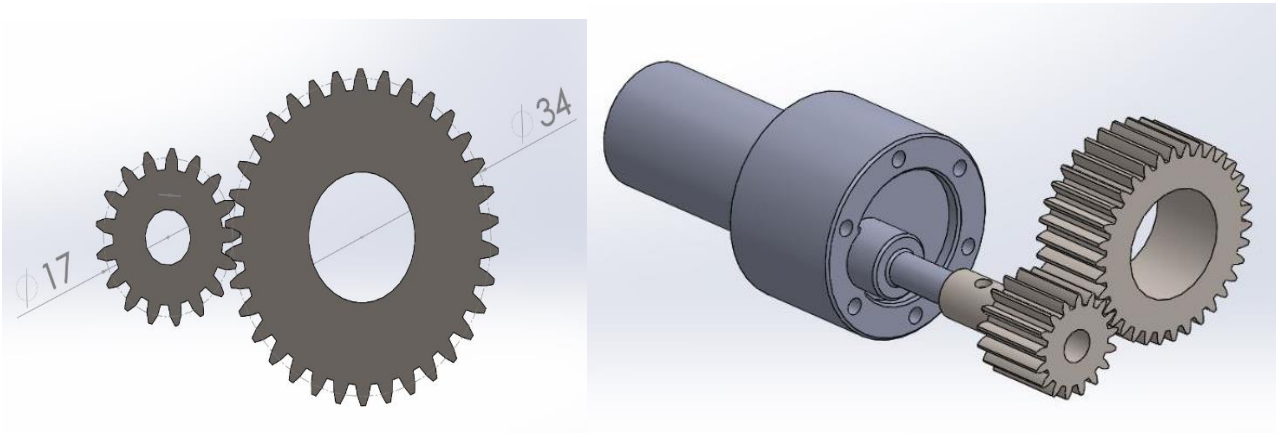

Fig 10: Gear Pair

4.4 Servo

In motion control applications, stipulating the right motor is pivotal for efficiency and productivity. The choice between servo motors and stepper motors is a vital one. The main benefit of servo motors is they yield high levels of torque at high speed – something stepper motors can't do. They also operate at 80 – 90% efficiency. Servo motors can work in AC or DC drive and do not agonize from vibration or resonance issues. Thus, a servo motor for motion control of sorting flaps was finalized.

Servo sizing

Proper sizing and selection of a motor for this application is the key to ensuring the performance, reliability, and cost of the setup. The first step was to determine the drive mechanism for the setup. Along with the type of drive mechanism, the dimensions, mass and friction coefficient, etc. that are essential for the load calculation were also determined. In order to deduce the performance needed from the motor, there are three factors to calculate; Moment of Inertia, Torque, and Speed.

- Data from the manufactured setup:
  - Drive Mechanism- Direct DC supply
  - Arm dimensions-
    - Length (l) =0.250 m
    - Width (w) =0.015 m
    - Height (h) =0.005m
    - Mass (m) =0.03kg
    - Distance between Center of Gravity and Center of Rotation (L) = 0.115 m

- Moment of Inertia

  Moment of inertia is the measure of an object's resistance to changes in its rotation rate, and is given by

$$J = mL^2$$

  By using the above formula, moment of inertia calculated to be 34.50kg-cm^2 (188.62 oz-in^2).

- Torque

  Torque is the tendency of a force to rotate an object about an axis. It can be calculated by using the formula

$$T = F * r$$

  Where F = Force on inclined plane

  r = Distance between the center of rotation. (5mm)

  - Inclined plane force calculation for an external force of 300g, mass of 30g and a friction coefficient of 0.05 is done by the following formula-

$$F = F_A + mg(\sin\theta + \mu\cos\theta)$$

$$F = 300 + 30 * 9.81(\sin 25 + 0.05\cos 25)$$

  Thus, the force is determined to be4.377 kg and Torque comes out to be 2.18 kg-cm.

- Speed

  Speed is the ratio of distance upon time. For servo motors, acceleration time must also be taken into consideration.

$$Speed = \frac{\text{Distance}}{(Time - AccelerationTime)}$$

  To optimize the sorting time, a speed of at least 0.2sec/60º was required.

Thus, by considering output characteristics, market availabilityand costing, FUTABA S3003 servo motor was finalized having Torque of 44oz/in (at 4.8V) and Operating Speed of 0.23 sec/60 degree (at 4.8V).

4.5 Sorter

This is the second frame of the machine. The object from the conveyor slides onto the sorter, and then under the effect of gravity, slides down the sorter. Flaps are provided on the sorter to guides the object into the separator box. The sorter flaps are actuated using Servo Motors, and they rotate with an angle of 30^0. To achieve optimum velocity for segregation, we found that a minimum of 25^0 of the slope was required.

Dimensions of Sorter:

- Length: 1088mm
- Width: 300mm
- Height: 800mm

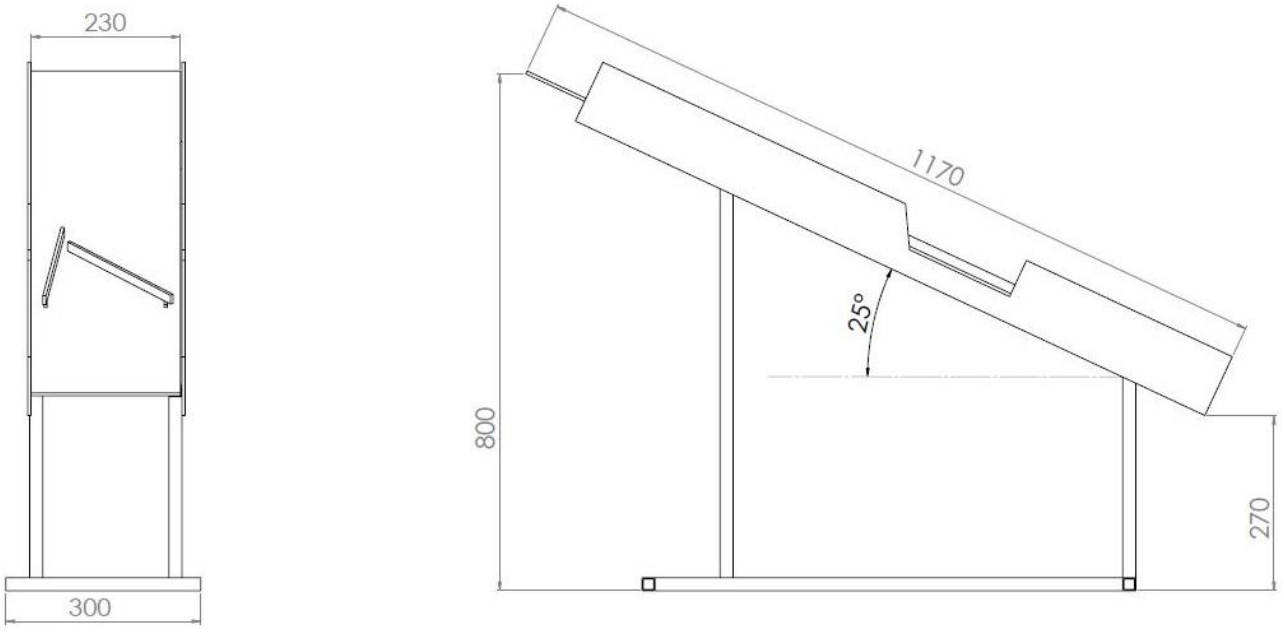

Fig 11: Front and Side view of Sorter

Fig 12: Manufactured Sorter

4.6 Inclined Box

This box is used to detect the color of the object. The color sensor is mounted inside this box. A servo is mounted at the top of the incline box. This servo, along with a flap, blocks the object from falling onto the conveyor before the color is detected. This box is made using an Acrylic Sheet of 3mm thickness.

The angle of the box was iterated before being mounted on the frame. We found that at 25^0, the object slides down and is stopped by the Servo actuated flap. If the angle is greater than 25^0, the object hits the flap at great velocity, thus inducing the risk of breaking the flap. Whereas at 25^0, the object does not slide down and stays in between due to friction.

The dimensions of the Object Detection Box are:

- Cross-section: 170mm x 140mm.
- Length: 350mm

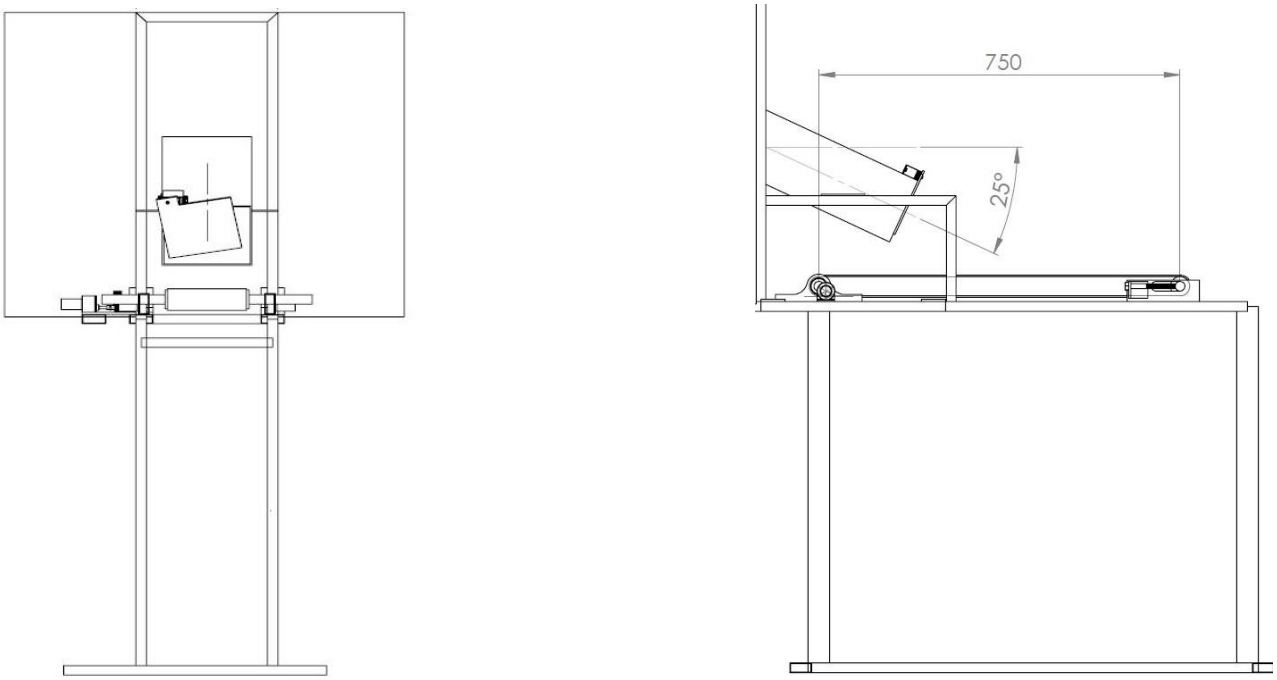

Fig 13: Front and Side View of Inclined Box

4.7Side Fenders

After the object sides from the inclined box onto the conveyor, the object might get disoriented. In order to avoid the object from tipping out of the conveyor and for guiding its path, Side Fenders are provided along the length of the conveyor. They are made using an Acrylic Sheet of 3mm thickness.

Dimensions of Side Fenders:

- Length: 650mm
- Height: 100mm
- Thickness: 3mm

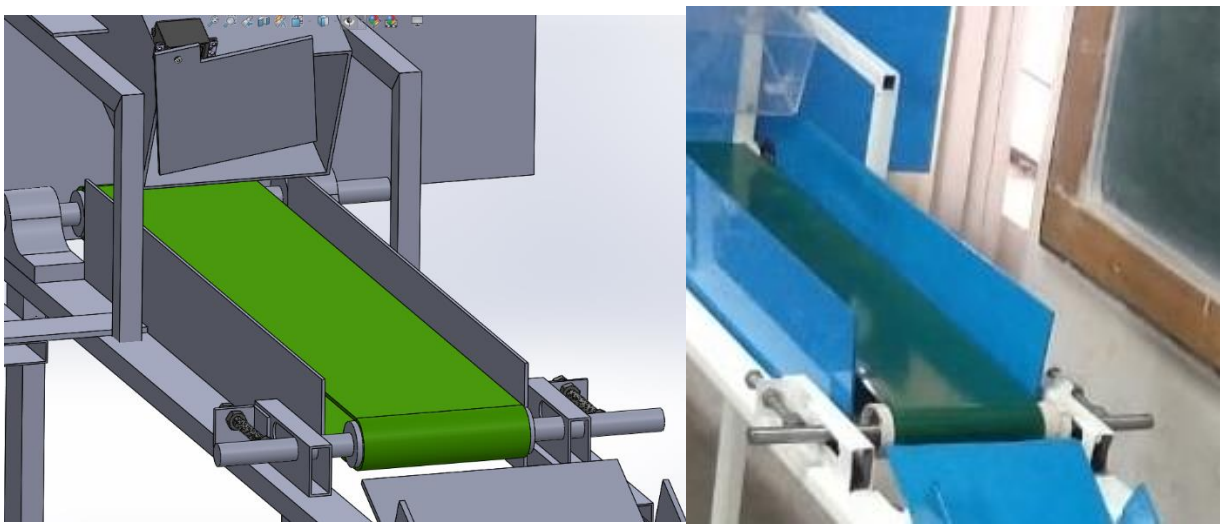

Fig 14: Actual and CAD model of Side Fenders

4.7 Front Panel

This is a panel made of an Acrylic Sheet. It is mounted behind the mainframe. Raspberry-Pi, Breadboard, and all the electronic components are mounted on it.

Dimensions of Front Panel:

- Length: 760mm
- Height: 650mm
- Thickness: 3mm

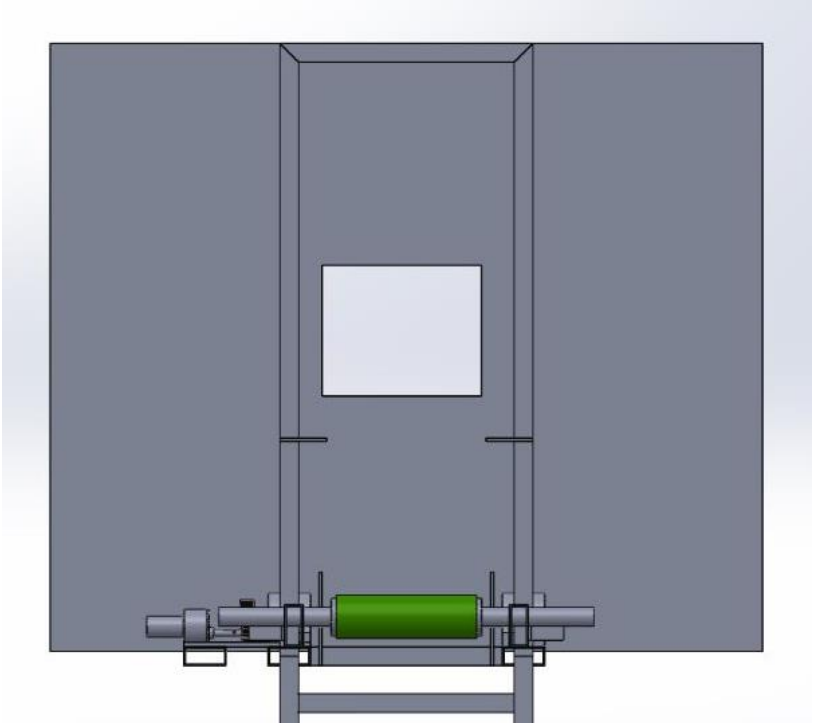

Fig 15: Electronics Mounting Panel.

4.8 Belt Take – Up Device

Belt take-up devices are used to maintain adequate tension in the belt conveyors. This helps to:

1) Transmit the mechanical power from drive pulley to belt without slip.

2) Limit the belt slag.

3) To compensate for changes in belt length due to elongation.

There are four types of Take-up devices:

- Screw Type Take-up.
- Vertical Gravity Type Take-up.
- Horizontal Gravity Type Take-up.
- Winch Operated Take-up.

From the above type, we decided to use Screw Type Take-up as:

- They are used in short type conveyors
- They are compact in arrangement
- Implementation was easy.

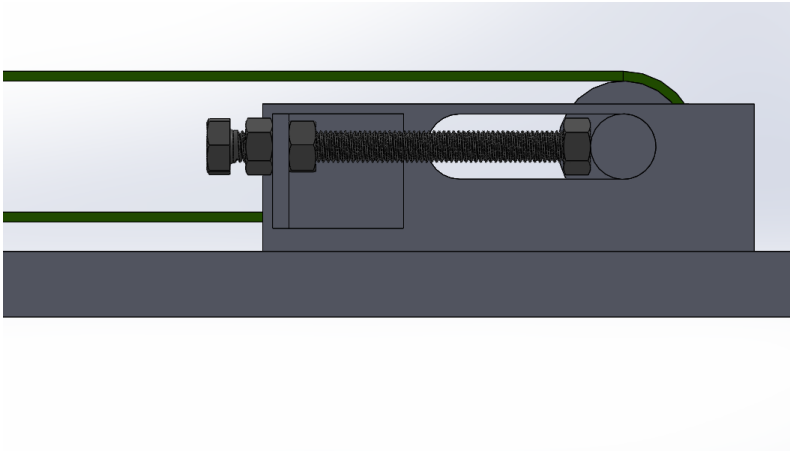

Fig 16: Front View of Screw Type Take-up Device

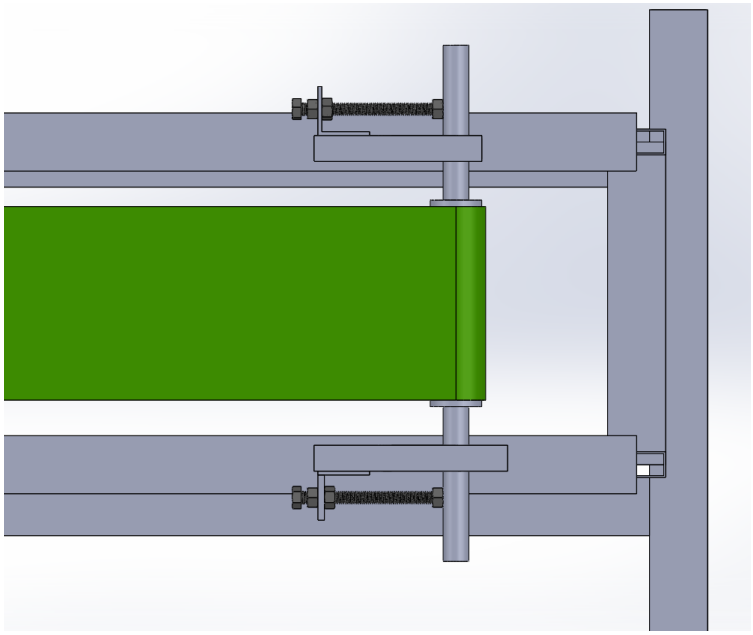

Fig 17: Top view of Screw Type Take-up Device

5. Circuit

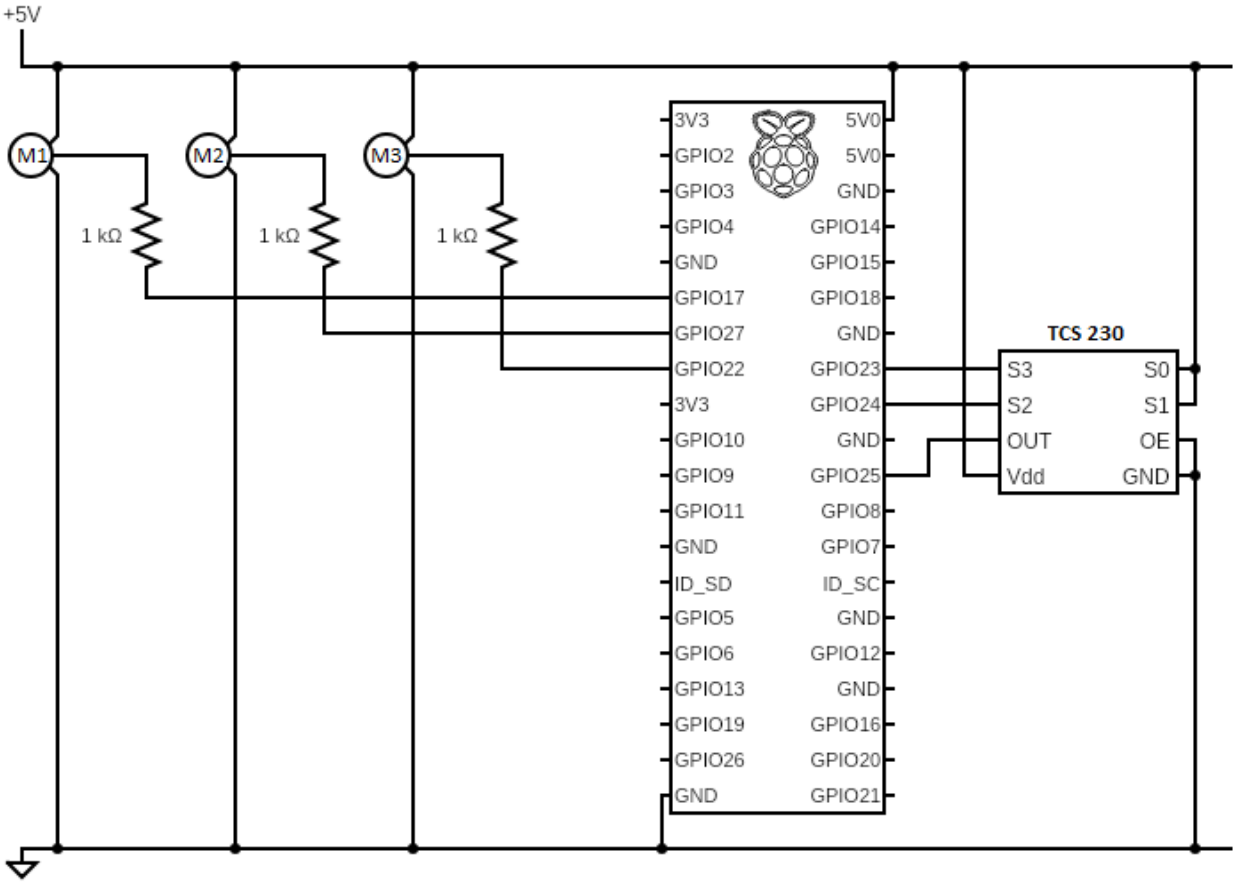

Fig 18: Circuit Board

The circuit diagram shows the connections of three Servo Motors, Color Sensor, and Raspberry Pi 3B+ board. A servo motor is a species of DC motor which, upon receiving a signal of a certain frequency, can rotate itself to any angle from 0-180 degrees. It has three pins- Power, Ground, and Pulse Width Modulation Pin. Here, we connect the power and ground pins to the +5V and GND pins of the Raspberry Pi 3B+ board. The PWM input is connected to one of the Raspberry Pi's digital output pins. Servo motor M1 is connected to GPIO 17, M2 to GPIO 27, and M3 to GPIO 22. A 1kΩ resistor is connected between the signal output of servos and the board to avoid damage because of fluctuations in voltage.

Along with the three servo motors, a color sensor is used to detect the color of objects. The chosen color sensor- TCS 230 has 8 pins. S0 and S1 are Output Signal Frequency Scaling inputs. Using these pins, you can scale the output frequency to one of the three pre-set values. This pin is connected to +5V. OE is the Output Enable Pin. It is an active LOW pin, and thus, connected to the ground. GND is a power supply Ground Pin. V_{dd} is Power Supply Pin and is connected to +5V. OUT is the Output Pin, connected to GPIO 25 of Raspberry Pi, which produces a square ware of 50% Duty cycle, and the frequency of the square wave is proportional to the intensity of the light. S2 and S3 are the Photo Diode Selection Pins and are connected to GPIO 23 and GPIO 24, respectively.

6. Code

6.1 Initialization of Raspberry Pi

The course of action that we followed to initialize Raspberry Pi is as follows:

1. The opening step was to determine which OS (Operating System) we wanted to use. There were two choices: Noobs and Raspbian. We choose to go with Raspbian.
2. After deciding the OS, we downloaded it from raspberrypi.org. Raspbian system has three versions; Raspbian Buster for desktop with recommended software, Raspbian Buster with desktop, and Raspbian Buster Lite. We downloaded the Raspbian Buster for desktop with recommended software as it had all the packages we needed preinstalled on it.
3. After downloading, we flashed the software on a memory card using Etcher. It is advocated to employ a minimum 8 GB memory card with Raspberry Pi.
4. After the flash was concluded, we inserted the memory card in Raspberry Pi and started it. For display, we connected our Raspberry Pi to a monitor using an HDMI cable, and power supply to Raspberry Pi was originally given using a power bank. Raspberry Pi needs 5V, 2.5 amp power supply through a micro USB port.
5. After the Raspberry Pi got powered up and was exhibiting its welcome screen, we opened its terminal and first updated the software by using the command “sudo apt update”.
6. After the update was completed, the next step was to install a VNC server on Raspberry Pi and our laptops so that for future times the Raspberry Pi can be seamlessly connected to our laptops by using just a WIFI, helping neglect the use of, HDMI cable and Monitor.
7. After installing the VNC server on our laptop and Raspberry Pi, we created an account on the VNC server website to login and connect with our Raspberry Pi. In the VNC server on our Raspberry Pi, we set a password for security purposes.
8. After all the above steps, we opened the VNC server on our laptop and then typed the IP address and password of our Raspberry Pi and thus connected our laptop to Raspberry Pi remotely.

6.2 Initialization of Color Sensor

The color sensor was initialized using the Raspberry Pi and Python. After adhering the color sensor to Raspberry Pi, using the breadboard, the Raspberry Pi was started. In Python, the pins to which the color sensor is connected were first illustrated. After describing this, various functions were defined to give the values for RBG when the color sensor saw a color. These values of RGB are proclaimed as Raw Values. Initially, Red color was displayed to the sensor, and the values for RGB for Red were taken. The same proceeding was replicated for Blue and Green colors. The values of RGB attained are exercised for creating loops for segregating the objects.

6.3 Working of Code

Preliminary to using the color sensor, it is essential to set it up. For setting up the color sensor, Raspberry Pi and Python are utilized. In the cardinal step, we designate the pins which we will be excersing, i.e. pin 16, 18, and 22. The remaining connections are explained in the circuit diagram. The color sensor incorporates four LEDs that aids the color sensor to detect the color properly. It has two photodiodes, which assist in acquiring the frequency of the color.

In the code, after designating the pins, a loop is created to retrieve the value the sensor gives for red, blue, and green color for a particular color. In the color sensor, we get values for all three colors, i.e. red, blue, and green, for every color we use. For example, when we put the red color object in front of the sensor, it gave us red>12000, blue<7000, and green<7000. The table below shows us the value given by the color sensor for each color.

Color	Red	Blue	Green
Red	>12000	<7000	<7000
Blue	<7000	>12000	<7000
Green	<10000	<10000	>12000

Table 3: Color Sensor Values.

Keeping the above table in mind, the code is drafted for the machine. After commencing the code using Python, in the initial step, the object is detected by the color sensor. Using the associations mentioned in the above table, the color sensor bestows the color of the object. An if statement is implemented for this purpose. If the frequency of object displays red>12000, blue<7000, and green<7000, then the color sensor gives the output as “Red”. Similar conditions apply for blue and green color. The output from the color sensor is taken as input for the loops used to control the servo motors. Three if statements are developed for the three colors. All the servos are controlled using Pulse Width Modulation. If the color is red, the first if statement gets activated. It first moves the servo attached to the inclined box by 90°, waits for two seconds, and then comes back to the initial position. As we do not actuate any servos on the sorter for red color, the object slides down to the container. For blue color, the second if statement gets activated. If the output of the color sensor is blue, the second if loop gets activated and rotates the inclined box servo by 90°, waits for two seconds, and gets it back to the initial position. It then rotates servo 2 by 30°, waits for 6seconds, and then gets the servo back to its initial position. For green color, the third if statement is activated. Its working is the same as if statement for blue color. The only difference is, instead of servo 2, servo 3 actuates for green color. The program is designed to terminate when no output is received from the color sensor for 2minutes

6.4 Code to get values from color sensor

```
#Importing libraries
importRPi.GPIOas GPIO
import time

#Designating Pins for configurations
s2 = 16
s3 = 18
signal = 22
NUM_CYCLES = 10
#Defining function
defsetup():
        GPIO.setmode(GPIO.BOARD)
        GPIO.setup(signal,GPIO.IN, pull_up_down=GPIO.PUD_UP)
        GPIO.setup(s2,GPIO.OUT)
        GPIO.setup(s3,GPIO.OUT)
        print("\n")
```

```
#Defining loop to get the values from color sensor for Red color
defloop():
        temp = 1
        while(1):
                GPIO.output(s2,GPIO.LOW)
                GPIO.output(s3,GPIO.LOW)
                time.sleep(0.3)
                start = time.time()
                forimpulse_countinrange(NUM_CYCLES):
                        GPIO.wait_for_edge(signal, GPIO.FALLING)
                duration = time.time() - start #seconds to run for loop
                red  = NUM_CYCLES / duration   #in Hz
                print("red value - ",red)

#For blue color
                GPIO.output(s2,GPIO.LOW)
                GPIO.output(s3,GPIO.HIGH)
                time.sleep(0.3)
                start = time.time()
                forimpulse_countinrange(NUM_CYCLES):
                        GPIO.wait_for_edge(signal, GPIO.FALLING)
                duration = time.time() – start
                blue = NUM_CYCLES / duration
                print("blue value - ",blue)

#For green color
                GPIO.output(s2,GPIO.HIGH)
                GPIO.output(s3,GPIO.HIGH)
                time.sleep(0.3)
                start = time.time()
                forimpulse_countinrange(NUM_CYCLES):
                        GPIO.wait_for_edge(signal, GPIO.FALLING)
                duration = time.time() – start
                green = NUM_CYCLES / duration
                print("green value - ",green)
                time.sleep(2)
#Function to end program
defendprogram():
        GPIO.cleanup()
if __name__=='__main__':
#To run above code
        setup()
        try:
                loop()
#To stop code if anything on the keyboard is pressed
        exceptKeyboardInterrupt:
                endprogram()
```

6.5 Entire Machine Code

```
#Importing libraries
importRPi.GPIOas GPIO
import time

#Designating Pins for configurations
s2 = 16
s3 = 18
signal = 22
NUM_CYCLES = 10

#Defining function
defsetup():
        GPIO.setmode(GPIO.BOARD)
        GPIO.setup(signal,GPIO.IN, pull_up_down=GPIO.PUD_UP)
        GPIO.setup(s2,GPIO.OUT)
        GPIO.setup(s3,GPIO.OUT)
        GPIO.setup(11,GPIO.OUT)
        GPIO.setup(13,GPIO.OUT)
        GPIO.setup(15,GPIO.OUT)
        p = GPIO.PWM(11,50)
        q = GPIO.PWM(13,50)
        r = GPIO.PWM(15,50)
        p.start(4)
        sleep(1)
        q.start(8)
        sleep(1)
        r.start(3)
        sleep(1)
        print("\n")

#Defining loop to get the values from color sensor for Red color
defloop():
        temp = 1
        while(1):
                GPIO.output(s2,GPIO.LOW)
                GPIO.output(s3,GPIO.LOW)
                time.sleep(0.3)
                start = time.time()
                forimpulse_countinrange(NUM_CYCLES):
                        GPIO.wait_for_edge(signal, GPIO.FALLING)
                duration = time.time() - start #seconds to run for loop
                red  = NUM_CYCLES / duration   #in Hz
#For blue color
                GPIO.output(s2,GPIO.LOW)
                GPIO.output(s3,GPIO.HIGH)
                time.sleep(0.3)
                start = time.time()
                forimpulse_countinrange(NUM_CYCLES):
                        GPIO.wait_for_edge(signal, GPIO.FALLING)
                duration = time.time() – start
```

```
                blue = NUM_CYCLES / duration
#For green color
                GPIO.output(s2,GPIO.HIGH)
                GPIO.output(s3,GPIO.HIGH)
                time.sleep(0.3)
                start = time.time()
                forimpulse_countinrange(NUM_CYCLES):
                        GPIO.wait_for_edge(signal, GPIO.FALLING)
                duration = time.time() – start
                green = NUM_CYCLES / duration

#Creating loops for color detection
                if green<7000and blue<7000and red>12000:
                        p.ChangeDutyCycle(9)
                        sleep(2)
                        p.ChangeDutyCycle(4)
                        sleep(1)
                        p.ChangeDutyCycle(8)
                        sleep(6)
                elif red<12000and  blue<12000 and green>12000:
                        p.ChangeDutyCycle(9)
                        sleep(2)
                        p.ChangeDutyCycle(4)
                        sleep(1)
                        r.ChangeDutyCycle(4.75)
                        sleep(6)
                        r.ChangeDutyCycle(3)
                        sleep(1)
                elif green<7000and red<7000and blue>12000:
                        p.ChangeDutyCycle(9)
                        sleep(2)
                        p.ChangeDutyCycle(4)
                        sleep(1)
                        q.ChangeDutyCycle(5.5)
                        sleep(6)
                        q.ChangeDutyCycle(8)
                        sleep(1)

#Function to end program
defendprogram():
        GPIO.cleanup()
if __name__=='__main__':

#To run above code
        setup()
        try:
                loop()
#To stop code if anything on the keyboard is pressed
        exceptKeyboardInterrupt:
                endprogram()
```

7. Manufactured Machine Prototype

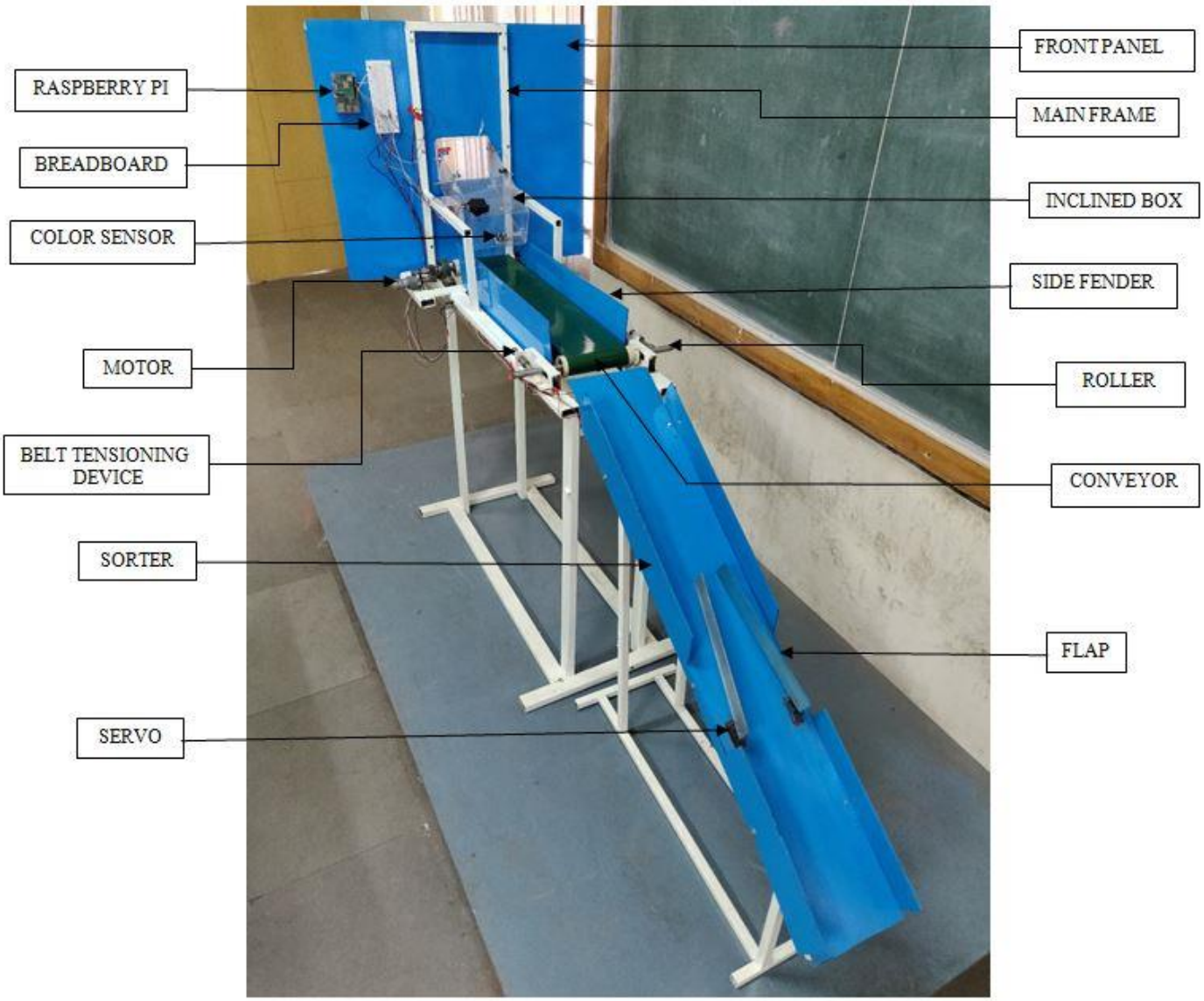

Fig 19: Manufacture Prototype Machine

8. Workflow

8.1 Flowchart

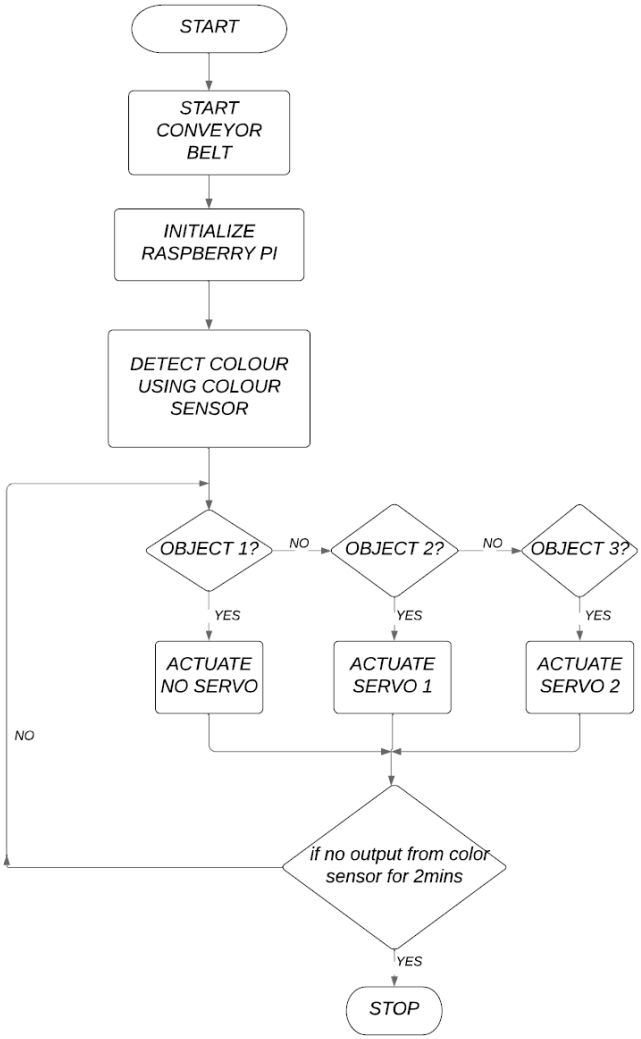

Fig 20: Workflow Flowchart

8.2 Explanation

The flow chart outlines the operation of the machine. The preliminary step is to start the conveyor belt, actuated by a DC motor. Then the Raspberry Pi is initialized by using a laptop (a power bank or a 5V adaptor can be used as well). After initializing the Raspberry Pi, start the Command Prompt and run the code using Python. The code initiates the color sensor and the servos.

After the color sensor and servos have been initiated, objects can be wedged into the machine. When an object is inserted, it gets blocked into the inclined box until the color sensor detects its color. Once the colorsensor gives an output i.e. the color, the first servo connected to the inclined box is actuated by 90^0, which lets one object slide onto the conveyor. The conveyor then passes on the object to the sorter.

The sorter consists of two servo motors. If the object is Red, no servos on the sorter actuate, and the object slides right down the slope into the container on the front side.

If the object is Blue, the left side servo actuates by 30º, thus directing the object through the first cavity on the left side into the second container.The servo returns to its original position after the object has fallen into the container.

And if the object is of Green color, the right side servo actuates by 30º, thus directing the object into the third container. The program terminates when no output is given by the color sensor for 2 minutes.

9. Object Size Range

9.1 Minimum Size

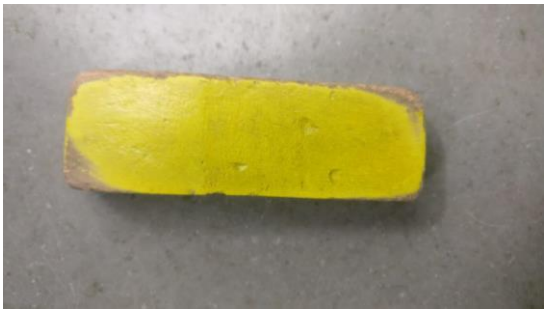

Fig 21: Smallest Object Size

Minimum Dimension: 60x20x10 mm (dimensions as l*b*h)

9.2 Maximum Size

Fig 22: Largest Object Size

Maximum Dimensions: 200x130x80 mm (dimensions as l*b*h)

9.3 Maximum Weight

The machine can sort objects up to a weight of 300gm.

10. Costing

10.1 Cost per unit:

Components	Cost
Raspberry Pi	3200
Color Sensor	300
Servos	1500
Motor	1000
Pi Case	450
Breadboard and Wires	200
AC Adapter	150
Manufacturing Process	5000
Total	11800

Table 4: Inventory Cost

10.2 Maintenance Cost:

It is advocated that the machine should undergo maintenance once per year. Considering that the servos need to be changed every year due to extreme working conditions, the maintenance cost will come out to be Rs 2000 per year. If the machine is operated properly, the maintenance cost will come around Rs 500 per year, with the servos needing to be changed every 4years.

10.3 Electricity Cost:

By running the main for 21hours per day for 30 days a month, the electricity cost comes out to be Rs 175.

11. Conclusions

The suggested framework will be a demo rendition that gives expense effective, taking less time, and technically the easiest way for differentiating objects. This framework utilizes Raspberry Pi, which makes this model simple to utilize, which is more additional effective. The entire system was developed in a budget of Rs 11,650 and took 10 months for completion.

11.1 Dimensional Analysis

The prototype is designed for sorting objects of any shape in the size array of 60x20x10mm to 200x130x80mm (dimensions as l*b*h) and having a weight limit of 300g.We can increase the dimension even further by making a longer inclined box, increasing the width of the cavities on the sorter, and using more powerful servos. The color palate of the prototype can be built further, but the prototype will get more complicated as we increase the number of colors to be detected.

11.2 Time Cost

The object once placed, in the inclined box takes, less than a second to reach the sensor. It takes another 200ms for the sensor to detect the color. An additional 4 to 6secs is required for the object to go from the conveyor to the sorter. And another 2 to 3secs will be taken by the object to reach from the sorter to the collection container.

12. Future Scope:

Packaging is one of the most important sections in the industries. The automatic sorting machine assists in improving the standards of packaging by enhancing efficiency. It guarantees ease of packaging with improved performance and elimination of human-made errors. The model can be improved by making some changes in the program and components. Some suggestions are given below.

- The system can be used as a quality controller by adding more sensors.
- A counter can be added for counting the number of products.
- The sensor can be changed according to the type of product.
- The DC motor can be replaced with a stepper motor.
- The Raspberry Pi can be replaced with PLC or PIC.
- A camera module can be used instead of a color sensor to segregate smaller dimensional objects.
- Time can be further optimized by using faster sensors.

13. References:

[1] Kunhimohammed C. K, Muhammed Saifudeen K. K, Sahna S, Gokul M. S and Shaeez Usman Abdulla, "Automatic Color Sorting Machine Using TCS230 Sensor And PIC Microcontroller", International Journal of Research and Innovations in Science and Technology, Volume 2 : Issue 2 : 2015

[2] Dr LeninaSvb, Anagha Kulkarni, Pranjali Sanjay Jaisingpure, "Automated Object Sorting Based On Colour Detection", 2nd International conference on Science, Technology & Management (ICSTM-2017), ISBN: 978-81-934288-7-0

[3] Rahul Vijay Soans, Pradyumna G.R, YoheiFukumizu, "Object Sorting using Image Processing", 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT-2018)

[4] Himanshu Patel, Riya Joy, Selin Macwan, Hardik Modi, "IOTColor Based Object Sorting Machine", International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 10 (2018) pp. 7383-7387

[5] Bhandari V.B., "Design of Machine Elements", Tata McGraw Hill Publication Co. Ltd.

[6] Vella Roderick, Micallef Lucienne, Abela Andrea, Tonna Clive, Xuereb Owen, Magro Brandon, " A comparative study of using colour sensor and Raspberry Pi camera to track colour detection", ECER 2018, Paper_OP-0021