



# PRESIDENCY UNIVERSITY

Itgalpura, Rajanukunte, Bengaluru - 560064

## School of Engineering

A Project Report on

### “IOT- Based Color Sorting Machine”

Submitted in partial fulfillment of the requirement for the course  
**Innovative Project – Raspberry-pi using Python (CSE 1003)**

Submitted by  
Group: IPR-452

Student Name	Roll No
Elluri Bhavya Sree	20211CAI0035
Ronit Pathak	20211CAI0010
Sohith N R	20211CAI0006
Uppara Dhana Lakshmi	20211CSE0270
Vishnu Sathvik Reddy	20211CCS0085

Under the supervision of

**Guide name: Mrs.Annapurna**  
**Designation: Assistant Professor**  
**Department: ECE-Dept**

Dec-2022

## **Abstract:**

There is a wide usage of many products in our day-to-day life, and manufacturing of this products are done in many large and small-scale industries. Arranging makes quality consistency issue.

Nowadays the main difficulty that is faced after the production is of sorting Arranging of items in an industry is a dull modern process, which is by and large done physically.

Consistent manual the need of this type of machine in the industries will help in sorting the machine according to their weight, size, color, shape, etc.

IOT based colored products sorting machine widely used in candy industry, food industry (grain, fruit) and mining industry.

Using this in candy industry can differentiate the candies according to their color, whereas in grain industry using this can differentiate the grains based on their color.

In Diamond and mining industry, segregates the precious stones according to their color.

This machine arranges the items in particular order as required, so physical work is not needed, i.e., Improves automation and decreases the man work.

We are implementing this machine in a much effective way, using color sensor (TCS 3200), Raspberry Pi, Servo motors and LCD display. Here we are arranging the setup in a pro-type manner where we can arrange the Servo motors to rotate based on the presented-color.

## Hardware, Software and tools used:

### Hardware:

- Servo Motor – 2 motors
- Raspberry Pi Module – 4
- Pi camera



**Servo motors:** Servo motors or “servos”, as they are known, are electronic devices and rotary or linear actuators that rotate and push parts of a machine with precision. Servos are mainly used on angular or linear position and for specific velocity, and acceleration.

- Input 5 volt is connected to Pin 1.
- Ground is Connected to Pin 3.
- Pin 11 is GPIO is connected to 3<sup>rd</sup> terminal.

### Proto-Type Code for Servo Motor:

```
from gpiozero import Servo
from time import sleep

myGPIO=17

servo = Servo(myGPIO)
print("Rassberry Pi Servo");
while True:
    servo.min()
    print("min")
    sleep(0.5)
    servo.mid()
    print("mid")
    sleep(1)
    servo.max()
    print("max")

    sleep(1)
```

```
from gpiozero import Servo
from time import sleep

myGPIO=17

myCorrection=0
maxPW=(2.0+myCorrection)/1000
minPW=(1.0-myCorrection)/1000

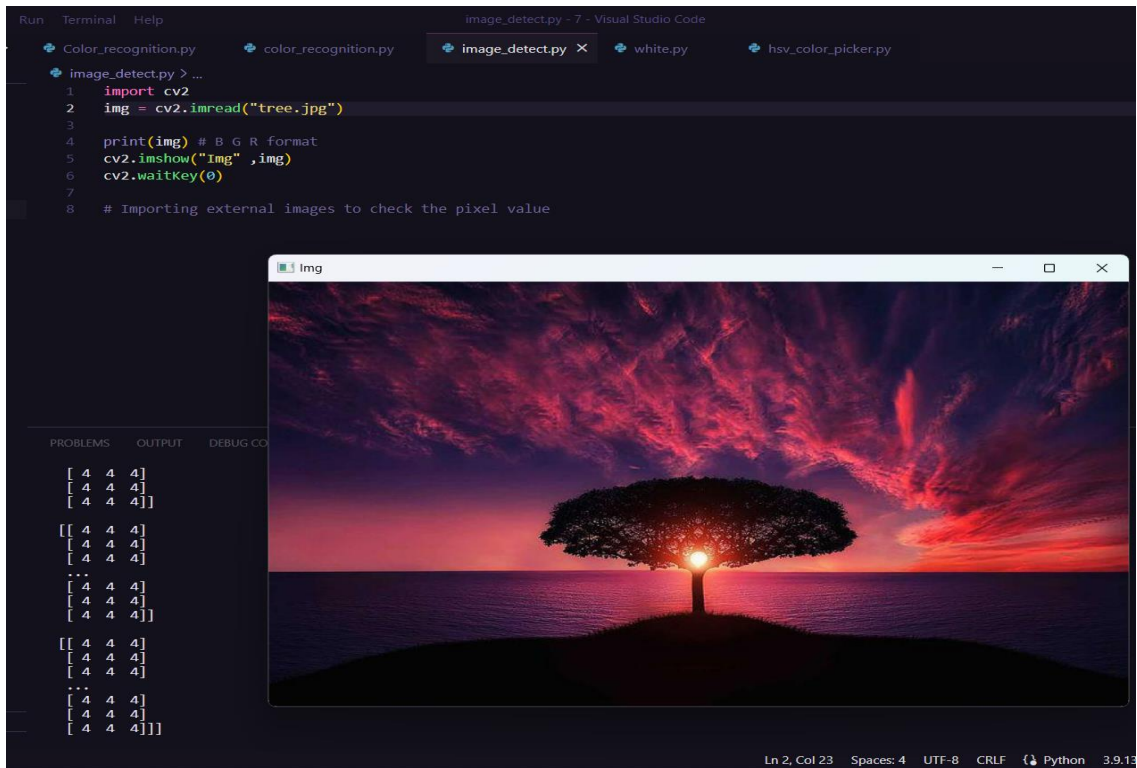
servo = Servo(myGPIO,min_pulse_width=minPW,max_pulse_width=maxPW)

while True:

    print("Set value range -1.0 to +1.0")
    for value in range(0,21):
        value2=(float(value)-10)/10
        servo.value=value2
        print(value2)
        sleep(0.5)

    print("Set value range +1.0 to -1.0")
    for value in range(20,-1,-1):
        value2=(float(value)-10)/10
        servo.value=value2
        print(value2)
        sleep(0.5)
```

## R G B Detection:



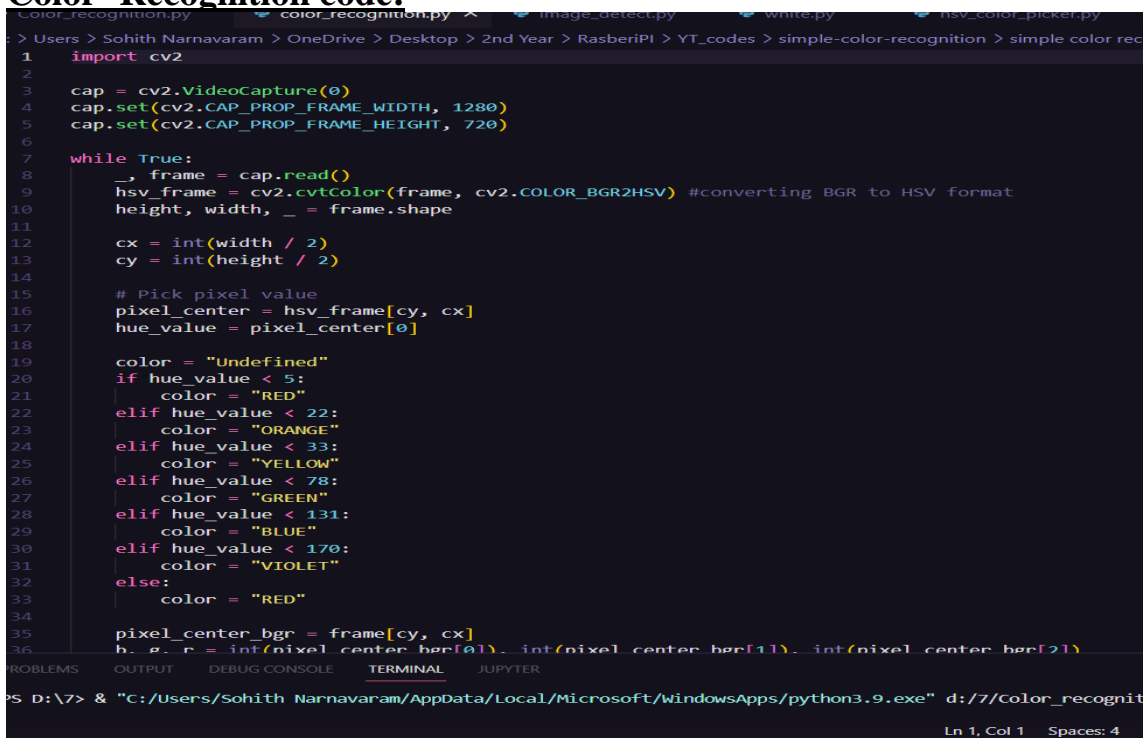
The screenshot shows the Visual Studio Code editor with the file `image_detect.py` open. The code imports `cv2` and reads an image named `tree.jpg`. It prints the image and displays it in a window titled `Img`. The image shows a tree on a hill with a bright sun setting behind it, creating a silhouette effect. The sky is filled with vibrant orange and red clouds. The console output shows the image data as a list of lists of integers, representing the BGR pixel values.

```
1 import cv2
2 img = cv2.imread("tree.jpg")
3
4 print(img) # B G R format
5 cv2.imshow("Img",img)
6 cv2.waitKey(0)
7
8 # Importing external images to check the pixel value
```

Output:

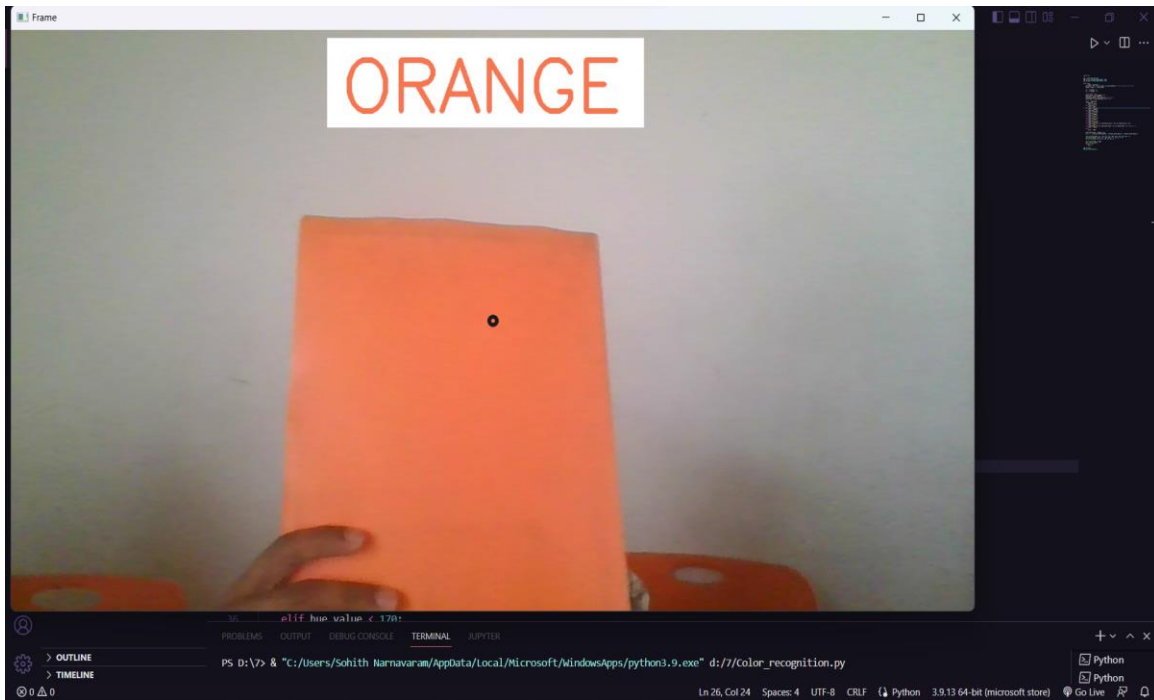
```
[ 4 4 4]
[ 4 4 4]
[ 4 4 4]
...
[[ 4 4 4]
 [ 4 4 4]
 [ 4 4 4]
 ...
 [ 4 4 4]
 [ 4 4 4]
 [ 4 4 4]
 ...
 [ 4 4 4]
 [ 4 4 4]
 [ 4 4 4]]]
```

## Color -Recognition code:



The screenshot shows the Visual Studio Code editor with the file `Color_recognition.py` open. The code uses `cv2.VideoCapture` to capture video from the default camera. It sets the frame width to 1280 and frame height to 720. It then enters a `while True` loop where it reads the frame, converts it to HSV format, and picks a pixel value from the center. Based on the hue value, it assigns a color name: RED, ORANGE, YELLOW, GREEN, BLUE, or VIOLET. If the hue value is outside the defined ranges, it defaults to RED. The code also prints the BGR values of the pixel center.

```
1 import cv2
2
3 cap = cv2.VideoCapture(0)
4 cap.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)
5 cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)
6
7 while True:
8     _, frame = cap.read()
9     hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV) #converting BGR to HSV format
10    height, width, _ = frame.shape
11
12    cx = int(width / 2)
13    cy = int(height / 2)
14
15    # Pick pixel value
16    pixel_center = hsv_frame[cy, cx]
17    hue_value = pixel_center[0]
18
19    color = "Undefined"
20    if hue_value < 5:
21        color = "RED"
22    elif hue_value < 22:
23        color = "ORANGE"
24    elif hue_value < 33:
25        color = "YELLOW"
26    elif hue_value < 78:
27        color = "GREEN"
28    elif hue_value < 131:
29        color = "BLUE"
30    elif hue_value < 170:
31        color = "VIOLET"
32    else:
33        color = "RED"
34
35    pixel_center_bgr = frame[cy, cx]
36    b, g, r = int(pixel_center_bgr[0]), int(pixel_center_bgr[1]), int(pixel_center_bgr[2])
```



### **Colorama:**

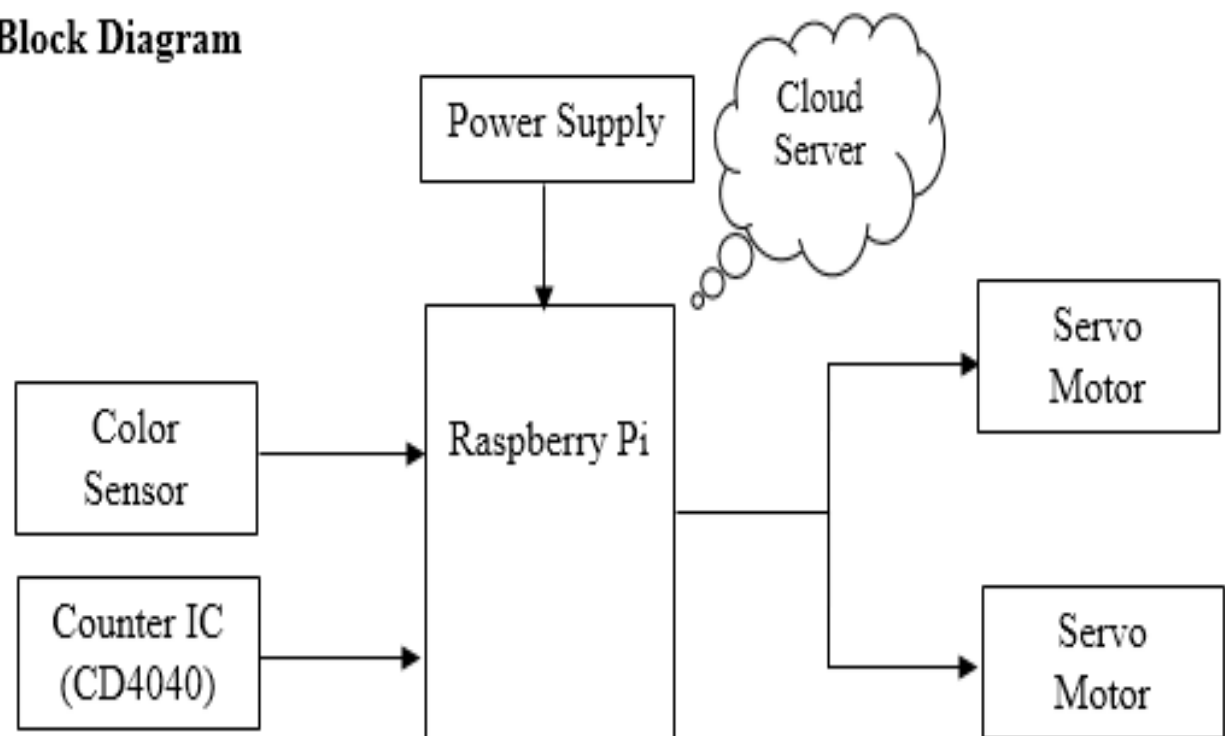
Colorama is a Python library for rendering colored terminal text.

### **Object's Color Recognition with OpenCV:**

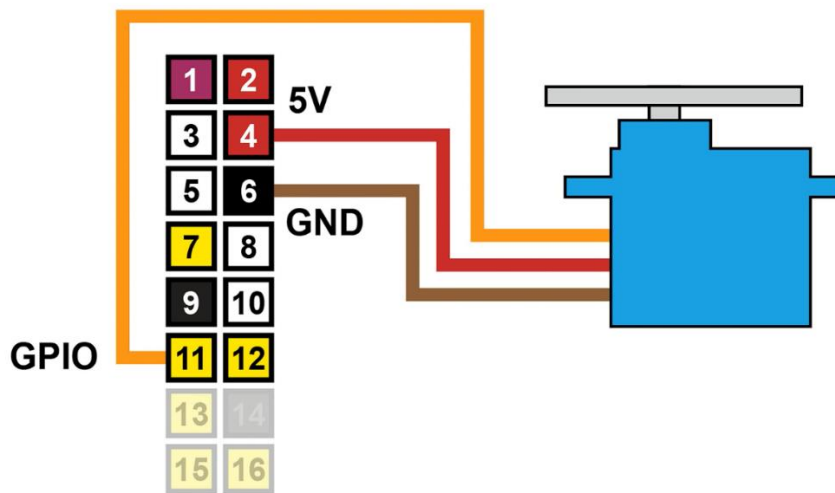
- Library used is OpenCV.
- Recognition of the color using a camera in Real Time.
- Defining color using H S V color picker
- Hue - Color itself
- Saturation – Intensity of the color
- Value – Brightness

**Block diagram:**

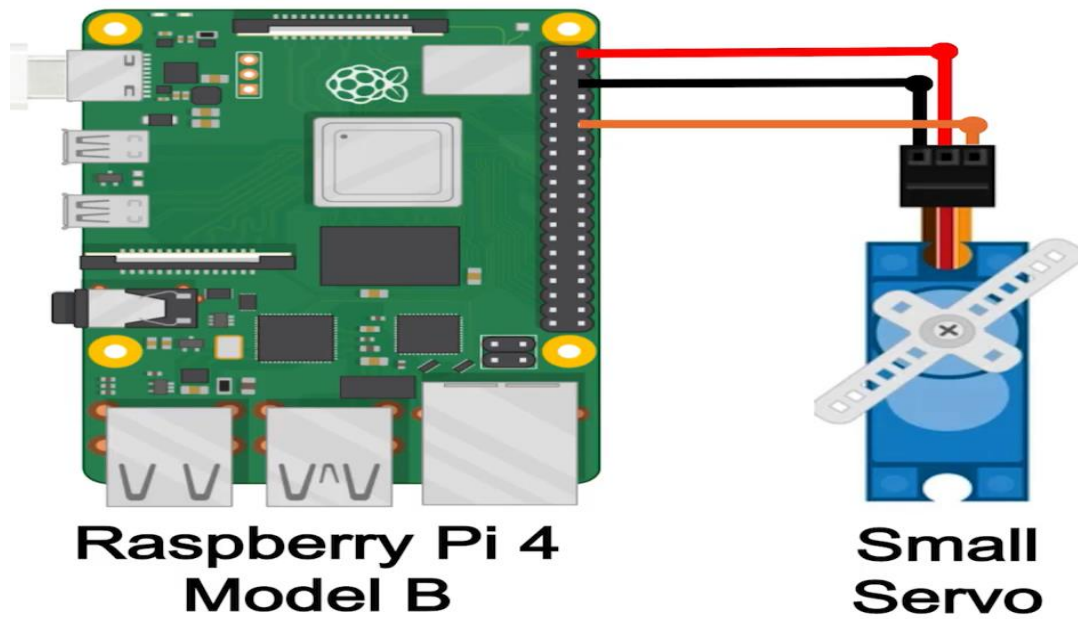
**Block Diagram**



### Circuit Diagram of Servo Motor:

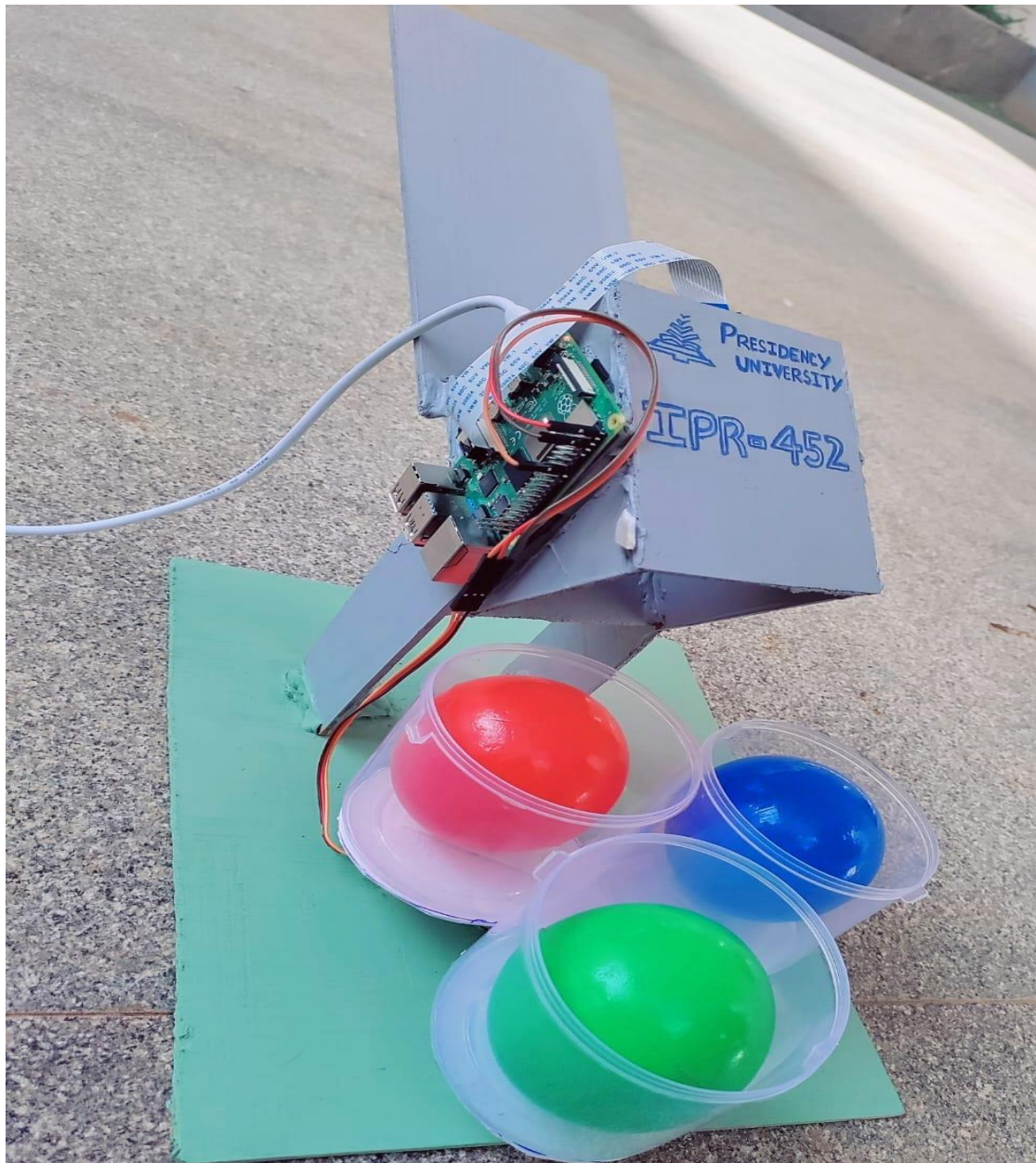


### Connections with Raspberry - Pi 4 :





## Results:



The objects are sorted with respect to their color and dropped into the respective box.

Its advantages are that it is-

- Accurate
- Good repeatability
- Reduce labor cost
- Requires Less human interference.



They are mainly used -

- In fruits and vegetable farming areas (rural areas) where installation of expensive sorters is very difficult.
- In food industry to identify rotted fruits and vegetables, in minor scale and big scale productions, to categorize the products established on the several factors.
- In production units to scan and identify the defects in raw materials.

### **Challenges faced:**

- Challenges faced in gaining more information about the hardware, and Software with limited libraries.
- Project time management.
- Challenge faced in choosing the appropriate tools and equipment for the project, be it the sensors or the Pi-cam.
- Learning and exploring the Raspberry-Pi as it is a new concept for everyone.
- Finalizing the working model's structure which had to be perfect in order to make it run.
- Finding the exact measurements/angles for the servo motors to rotate and function.

## **Conclusion:**

The suggested framework will be a demo rendition which gives expense effective, taking less time and technically the easiest way for differentiating objects. This framework utilizes Raspberry Pi-4 which makes this model simple to utilize which is more additional effective. The main failure will be caused if the sensing of object according to color is not done. Therefore, it is very important to have proper and checked Apparatus. Further, making desirable changes it can be used in small-scale and large-scale industries as well.

When any color from red, green or blue is kept for detection in front of the Pi-cam then the desired color as shown is detected on the screen and the output of the sensing of color is seen.