Mechatronics System Integration (MCTA3203) Semester 2, 2023/2024

Report: Experiment; week 9a - Image And Vedeo

Lecturers:

Dr/ WAHJU SEDIONO

Dr/ ZULKIFLI BIN ZAINAL ABIDIN

Dr/ ALI SOPHIAN

Section 1 Group 3

Names	**Matric**
Abdallah Mahamat Abacar	2117777
Almasri Suhail Jihad	2128771
Ibrahim Bin Nasrum	2116467
Khan Toqi Tahmid	2025209

TABLE OF CONTENTS

ABSTRACT

This report outlines the development of a color detection system utilizing an Arduino board in conjunction with either a color sensor or a USB camera. The experiment includes hardware setup, programming in Arduino and Python, and performance analysis of color detection in various scenarios. The findings highlight the accuracy, response time, and environmental robustness of both approaches.

INTRODUCTION

The creation of a color detecting system using an Arduino board and either a color sensor or a USB camera is described in this study. The hardware setup, Python and Arduino programming, and performance measurement of color identification in various settings are all included in this experiment. The results demonstrate how accurate, quick to react, and resilient to changing conditions both methods are.

MATERIAL AND EQUIPMENT

For Color Sensor:

- Arduino board
- Color sensor (TCS3200 or TCS34725)
- Jumper wires
- Breadboard
- RGB LED (optional)
- Computer with Arduino IDE and Python installed

- USB cable for Arduino

For USB Camera:

- Arduino board (for additional functionalities, optional)
- USB camera
- Jumper wires
- Breadboard
- Computer with Python and OpenCV library installed
- USB cable for Arduino

EXPERIMENT SETUP

1. Hardware Setup:
    - Connect the color sensor to the Arduino using jumper wires. Refer to the sensor's datasheet and Arduino's pinout diagrams for guidance1.
    - If using an RGB LED, connect it to the Arduino for color display.
2. Arduino Programming:
    - Write an Arduino sketch1 to interface with the color sensor.
    - Read RGB color data from the sensor and convert it to a format that can be sent to the computer.
    - Calibrate the sensor for accurate color readings.
3. Python Programming:
    - Write a Python program to communicate with the Arduino over the serial connection using the pyserial library.
    - Receive RGB color data from the Arduino.

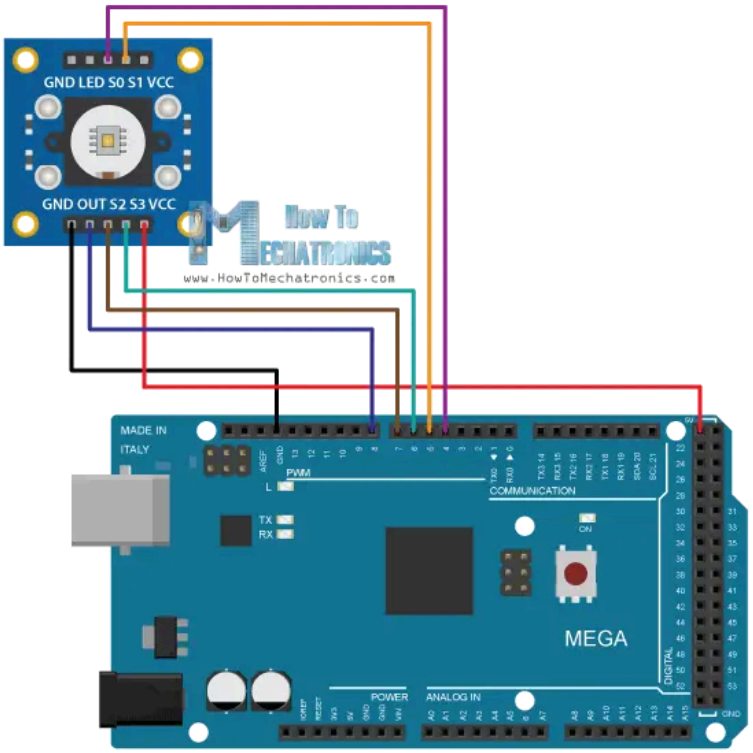
- Interpret the received data to determine the detected color. You can modify the code shown below for your own purposes2.

4. Testing and Data Collection:
    - Test the system with different colored objects.
    - Collect data on the detected colors, their accuracy, and how the system performs in various lighting conditions.
    - Analyze the response time of the system when detecting colors.
5. Analysis:
    - Evaluate the accuracy of color detection by comparing detected colors with actual colors.
    - Analyze how the system performs in different lighting conditions.
    - Calculate the average response time for color detection.

METHODOLOGY

Part 1: Color Detection with Color Sensor

An Arduino script was created to read RGB data from the color sensor and format it for serial communication with a computer in order to make communication with the sensor easier. To achieve accurate color readings, the sensor was carefully calibrated to take into account potential fluctuations in the lighting and environment. Using the pyserial package, a Python application was written to communicate with the Arduino, receive RGB data, and process it in order to identify the detected color. Using a range of colored items, the system was put through a rigorous testing process to evaluate its accuracy and performance in various lighting scenarios. Data on color recognition accuracy and system performance were gathered, and the system's reaction time was computed to assess its effectiveness.



Part 2: Color Detection with USB Camera

A Python program was developed to capture video from a USB camera utilizing the OpenCV library. This program implemented color detection using the HSV color space, which is better suited for handling variations in lighting conditions compared to the RGB color space. The video feed and detected colors were displayed on the screen in real time. The system was tested with various colored objects to evaluate its performance and accuracy under different lighting scenarios. Data were collected to assess how accurately the system identified colors and its overall performance, which was then compared to the color sensor-based system. This comparison provided insights into the advantages and limitations of using a USB camera for color detection in terms of accuracy, response time, and adaptability to different environmental conditions.

ARDUINO CODE For Part 1: Color Detection with Color Sensor

```
#define S0 4

#define S1 5

#define S2 6

#define S3 7

#define sensorOut 8

int frequency;

void setup(){

 pinMode (S0, OUTPUT);

 pinMode (S1, OUTPUT);

 pinMode (S2, OUTPUT);

 pinMode (S3, OUTPUT);

 pinMode (sensorOut, INPUT);

 digitalWrite(S0, HIGH);

 digitalWrite (S1, LOW);

Serial.begin(9600);}

void loop(){

 digitalWrite(S2, LOW);
```

```
digitalWrite (S3, LOW);

frequency = pulseIn(sensorOut, LOW);

Serial.print("R= ");

Serial.print (frequency);

Serial.print(" ");

delay(100);

digitalWrite (S2, HIGH);

digitalWrite(S3, HIGH);

frequency = pulseIn (sensorOut, LOW);

Serial.print("G= ");

Serial.print (frequency);

Serial.print(" ");

delay(100);

digitalWrite(S2, LOW);

digitalWrite (S3, HIGH);

frequency = pulseIn (sensorOut, LOW);

Serial.print("B= ");

Serial.print (frequency);
```

```
 Serial.println(" ");

 delay(100);}
```

Python Code For Part 1: Color Detection with Color Sensor

```
# --

# Simple code for RGB color detection

# --

import matplotlib.pyplot as plt

import numpy as np

def create_channel(color_val):

return np.ones((10, 10, 3))*color_val

def display_color(r, g, b):

color = [r/255, g/255, b/255] # Normalize RGB values

# Create a block for each channel with linear gradients

r1 = create_channel([r/255, 0, 0])

g1 = create_channel([0, g/255, 0])

b1 = create_channel([0, 0, b/255])

rgbchannel = [r1, g1, b1]

# Combine the color channel horizontally
```

```
# channels_combined = np.hstack(rgbchannel)

# Create the final RGB values for the specified color

true_color = np.ones((10, 30, 3))*color

# Combine all vertically

# final_display = np.vstack([channels_combined, true_color])

final_display = np.vstack([np.hstack(rgbchannel), true_color])

# Display the final block

plt.imshow(final_display)

plt.axis('off')

plt.show()

# -- Example: Display color for random RGB values

random_r = np.random.randint(0, 256)

random_g = np.random.randint(0, 256)

random_b = np.random.randint(0, 256)

display_color(random_r, random_g, random_b)
```

Python Code For Part 2: Color Detection with Color Sensor

```
# --
```

```
# Basic code for checking connection to webcam (or USB camera)

# --

import cv2

print("Press ESC to exit.")

capture = cv2.VideoCapture(0) # adjust as needed

if capture.isOpened():

print('Camera is opened')

while True:

ret, frame = capture.read()

if ret:

cv2.imshow('Vid1', frame)

t = cv2.waitKey(1) # wait for 1 second

# Check if the window is closed

if (cv2.getWindowProperty('Vid1', cv2.WND_PROP_VISIBLE) < 1):

break

# Check if ESC or 'x' is pressed

if (t == 27) or (t & 0XFF == ord('x')):

break
```

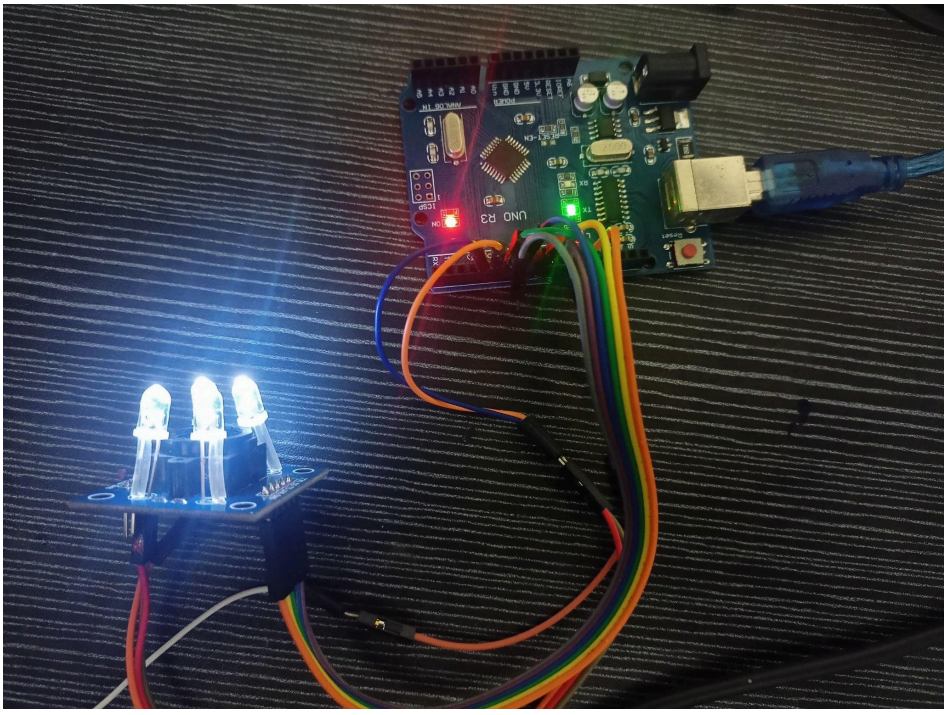
```
capture.release()

cv2.destroyAllWindows()

print('Camera is closed')
```

RESULT

Data on the accuracy of color identification, response speed, and environmental robustness were gathered during the testing of the color detection systems with a variety of objects under varying lighting situations. Due to direct color sensing, the color sensor showed excellent accuracy in controlled lighting circumstances and quick response times; nevertheless, under variable illumination conditions, its accuracy drastically dropped. In contrast, the USB camera required image collection and processing, which resulted in longer response times, middling accuracy, and a heavy reliance on image processing algorithms. However, because the HSV color space is more resilient to handling changes in illumination, the USB camera showed higher adaptation to various lighting condition



DISCUSSION

When subjected to consistent lighting circumstances, the color sensor performed exceptionally well in terms of precise and quick color recognition; nevertheless, it had difficulties in other lighting environments. On the other hand, because of its sophisticated image processing features, especially the usage of the HSV color space, the USB camera fared better in changing lighting circumstances while being slower and less accurate. The effectiveness of both systems was significantly influenced by calibration and ambient conditions, underscoring the necessity of meticulous modification and optimization to get dependable color identification in various contexts.

CONCLUSIONS

For color detection, the requirements of the particular application will determine whether to use a USB camera or a color sensor. Applications needing great speed and precision in controlled environments are better suited for the color sensor. On the other hand, even though it is slower and less accurate, the USB camera is better suited for applications that require stability in a variety of lighting situations.

RECOMMENDATION

Incorporating machine learning algorithms and sophisticated image processing techniques could increase USB camera-based systems' accuracy and flexibility in the future. Using dynamic calibration techniques may also help color sensors function better under changing illumination conditions. A better balance between resilience, speed, and accuracy in various conditions might be possible by looking into the usage of other sensors and cameras with better specifications.

REFERENCE

- How To Mechatronics. Arduino Color Sensing Tutorial - TCS230 TCS3200 Color Sensor. Link
- YouTube Video on Arduino Color Sensing. Link
- YouTube Video on Connecting to a Webcam Using OpenCV. Link

CERTIFICATE OF ORIGINALITY AND AUTHENTICITY

This is to certify that we are responsible for the work submitted in this report, that the original work is our own except as specified in the references and acknowledgment, and that the original work contained herein has not been untaken or done by unspecified sources or persons.

We hereby certify that this report has not been done by only one individual and all of us have contributed to the report. The length of contribution to the reports by each individual is noted within this certificate.

We also hereby certify that we have read and understand the content of the total report and no further improvement on the reports is needed from any of the individual contributors to the report. We therefore, agreed unanimously that this report shall be submitted for marking and this final printed report has been verified by us.

Signature: Ibrahim

Name: Ibrahim Bin Nasrum

Matric Number: 2116467

Contribution: Introduction, Abstract

☑ Read
☑ Understand
☑ Agree

Signature: Mahamat
Name: Abdallah Mahamat Abacar

Matric Number: 2117777

Contribution: Discussion and Conclusion

☑ Read
☑ Understand
☑ Agree

Signature: Suhail

Name: Almasri Suhail Jihad

Matric Number: 2128771

Contribution: Methodology

☑ Read
☑ Understand
☑ Agree

Signature: Toqi

Name:Khan Toqi Tahmid

Matric Number:2025209
Contribution: Result and discussion

☑ Read
☑ Understand
☑ Agree