MECHATRONICS SYSTEM INTEGRATION

EXPERIMENT 9/9A: COLOUR DETECTION AND IMAGE IDENTIFICATION

NAME	MATRIC NUMBER
MUHAMAD IMRAN HAZIQ BIN HAIZAM	2216429
TANISHA ROWSHAN SALEH	2228290
MUHAMMAD YUSOF IRFAN BIN ABIDIN	2210915
ARIESHA DALIELA HANIM BINTI AMIR	2316834

Date of Submission: 21/5/2025

TABLE OF CONTENTS

Abstract

Part 1 :
This lab activity involves the design and implementation of a color detection system using Arduino, Python, and either a color sensor or a USB camera. The experiment demonstrates how digital systems can be used to sense and interpret color information from physical objects. RGB data is captured using a color sensor (e.g., TCS3200 or TCS34725) connected to an Arduino, which processes and transmits the data to a computer. A Python program, utilizing the pyserial library, receives and analyzes the color information to determine the detected color. The system is tested under various conditions to assess accuracy, performance, and response time. This setup provides an accessible and practical approach to understanding the integration of hardware and software in real-time color detection and sensor-based applications.

Part 2 :
This lab activity investigates color detection using a USB camera and Python programming with the OpenCV library. The experiment demonstrates how real-time video processing and computer vision techniques can be applied to identify colors from live video feeds. A USB camera captures video frames, which are analyzed using HSV color space to detect specific colors. The detected colors are highlighted and displayed on the screen, enabling a visual and interactive method for evaluating color recognition performance. The goal is to explore the implementation and accuracy of a software-based color detection system, and compare its performance with sensor-based approaches. This setup offers a flexible and scalable approach to computer vision applications, promoting a deeper understanding of digital image processing and color recognition.

Introduction

Part 1 :
Experiment 9 was conducted to explore the principles of color detection using a microcontroller-based system. Color detection involves identifying and interpreting colors from physical objects using sensors or cameras, and in this case, it was achieved using an Arduino microcontroller interfaced with either a color sensor or a USB camera. The Arduino served as the hardware platform for capturing RGB data from the sensor, while Python was used on the computer side for processing and interpreting the color information through serial communication.

Color detection systems have a wide range of applications, including quality control in manufacturing, object sorting in robotics, and assistive technologies for visually impaired individuals. Accurate color recognition can enhance automation and user interaction in numerous fields.

By combining Arduino and Python, this experiment aimed to demonstrate real-time communication between hardware and software, and to evaluate the system's performance in identifying various colors under different lighting conditions. It was expected that the system would successfully detect and differentiate colors, display the results via the Python terminal, and allow for analysis of accuracy, lighting sensitivity, and response time.

Part 2 :

Experiment 9A was conducted to explore computer vision-based color detection using a USB camera and Python. Unlike sensor-based systems that rely on physical components to capture color data, this approach utilizes image processing techniques to identify and classify colors from a video stream in real time. The USB camera acts as the primary input device, and Python, along with the OpenCV library, processes the video feed to detect colors based on HSV (Hue, Saturation, Value) values.

Color detection using cameras is widely used in industrial automation, object tracking, gesture recognition, and even augmented reality. Its ability to detect colors across a wide field of view, without requiring direct contact, makes it a powerful tool for dynamic and versatile applications.

By implementing this system, the experiment aimed to understand how color can be detected and interpreted from video frames, and how lighting and environmental factors affect accuracy. The results were then compared with those from the sensor-based system to evaluate the advantages, limitations, and potential improvements in future iterations.

Materials and equipment

Part 1:
1)Arduino board

2)Color sensor (e.g., TCS3200 or TCS34725)

3)Jumper wires

4)Breadboard

5)RGB LED (optional)

6)Computer with Arduino IDE and Python installed

7)USB cable for Arduino

Part 2 :

1)Arduino board

2)USB camera

3)Jumper wires

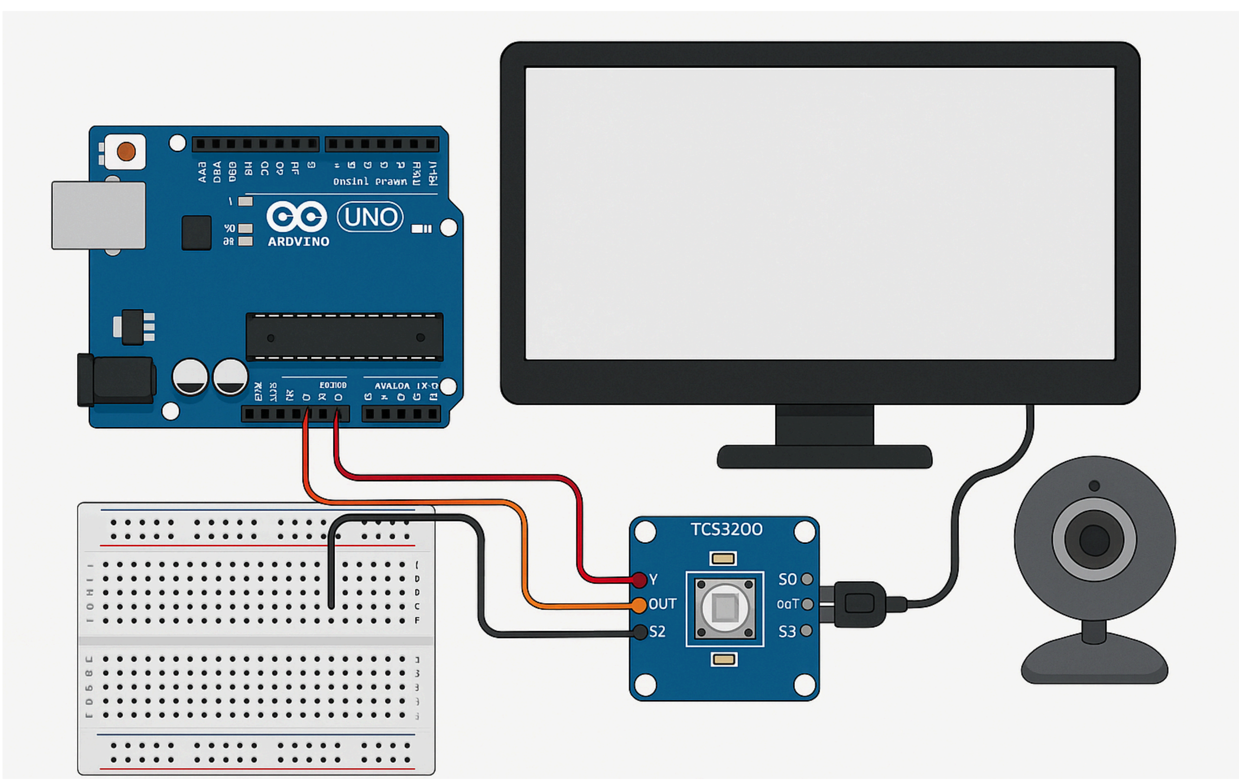
4)Breadboard

5)Computer with Python installed

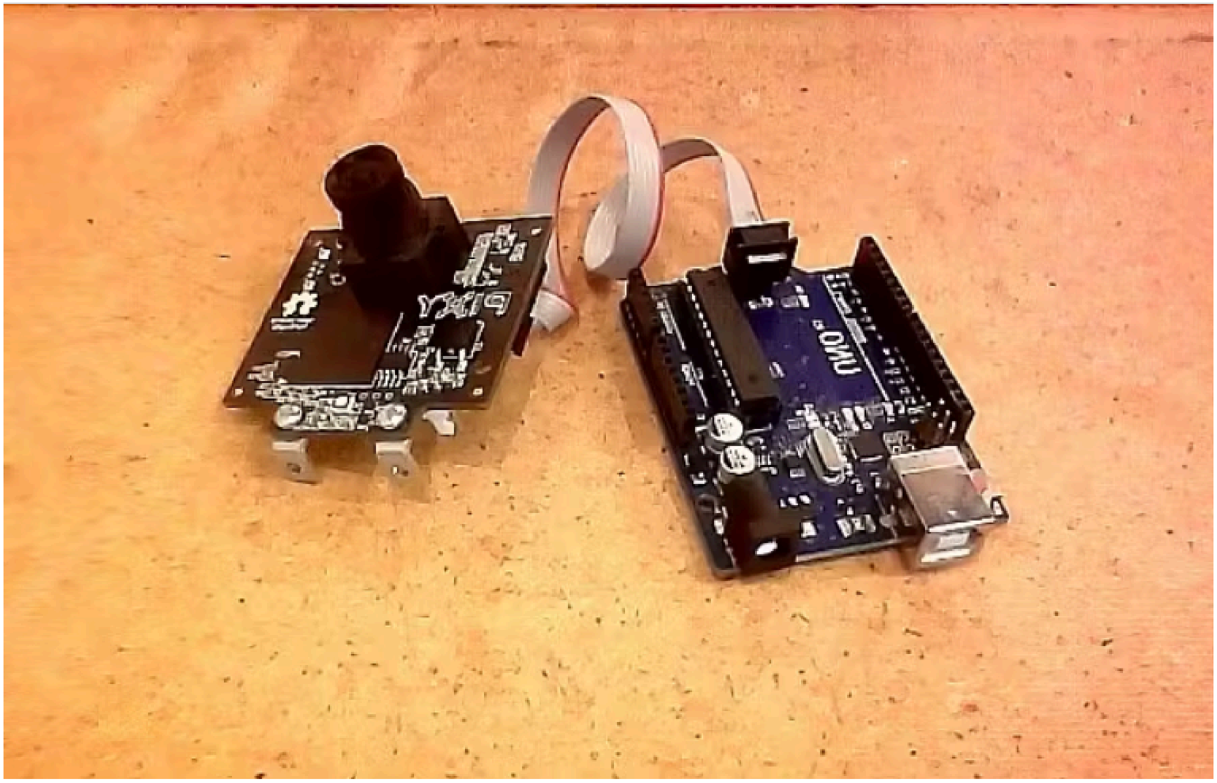
6)USB cable for Arduino

Experimental Set Up :

Part 1:



Part 2:



Credit: https://www.hackster.io/JustJ0e/pixy-cmucam-arduino-3ac141

<u>Methodology</u>

Part 1:

1) The Arduino was hooked to the colour sensor, TCS230
2) A set of code was written to digitally interface the sensor with the Arduino
3) Once linked, the sensor should have been able to sense the colours it was programmed to recognize and respond accordingly
4) A Python code was written to send data from the Arduino
5) After receiving the data from the Arduino, the goal was to identify the colour based on the sent data.

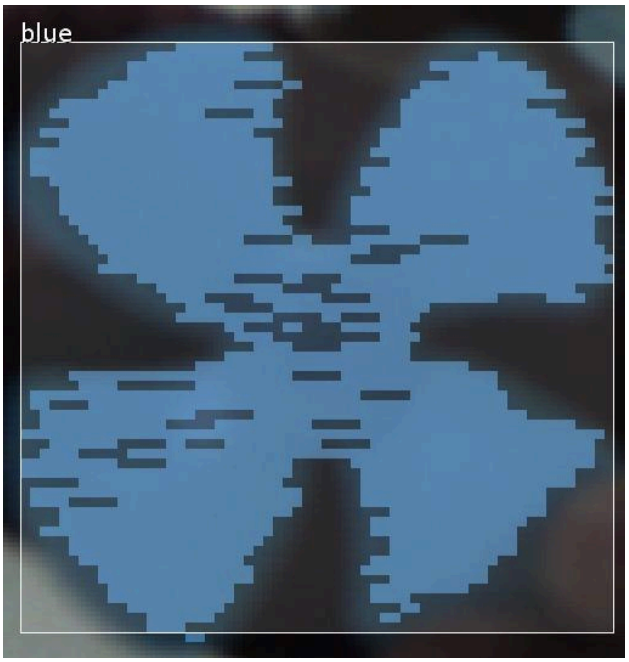
Part 2:

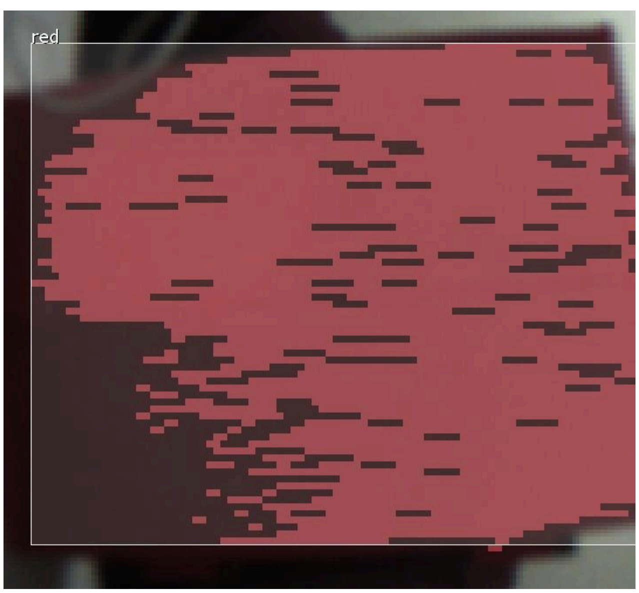
1) Pixy1 was connected to the Arduino board through its 6-pin cable
2) PixyMon for Pixy1 was installed and set up
3) Three Signatures were assigned to three different colours: red, green, and blue
4) A series of code was written to connect the Arduino to the Pixy1 camera

5) Whenever an object of red, green or blue was held in front of the camera, the serial monitor of the Arduino IDE would display the object's x-y coordinates as well as the text, '(colour) detected', where colour was defined as red, green or blue

Results

Part 2:






```
1 X:
160
 Y:
86
Red detected
Block
0
:
Signature:
1 X:
160
 Y:
87
Red detected
Block
0
:
Signature:
1 X:
160
 Y:
90
Red detected
Block
0
:
Signature:
1 X:
160
 Y:
92
```

<u>Discussion</u>

Part 1: First off, the TCS37425 provided couldn't connect to the Arduino, so another sensor was used in its stead - the TCS230. At the start, it did work, but an error caused it to malfunction, preventing it from communicating with the Arduino along with the Python IDE that it was supposed to be connected to as well. Therefore, even after multiple tries, both the Arduino and the Python IDE failed to exchange information with the colour sensor.

Part 2: Using PixyMon for Pixy1, three signatures were set: red, green and blue. Each colour was determined using a sample colour from different objects. When the same objects or objects of similar colours were held in front of the Pixy1 camera, it was able to identify each object's colour.

To further enhance this, a series of code was written to provide a written output in the Arduino IDE's serial monitor. Using the Pixy1 library, the code was written such that when the Pixy camera detected a blue object, the serial monitor would display the object's x and y coordinates, as well as a line of text, '(colour) detected', where colour is defined as red, green or blue.

Overall, there weren't any errors while doing this assignment, although the quality of the camera was subpar, with the images being blurry even at its highest possible resolution.

Conclusion

Part 1: In conclusion, experiment 9 failed to achieve its goal, which was to allow the colour sensor (TCS37425/TCS230) to communicate with the Arduino and the Python IDE. This could have been due to multiple sources of error, such as: incorrect wiring, incorrect coding, or faulty hardware.

Part 2: To summarize, experiment 9A was a success, with all results being obtained without any issues. The Pixy1 camera was interfaced with the Arduino board, and was able to detect the three different colours it was set to detect, displaying different objects' x-y coordinates as well as the line of text, '(colour) detected', where colour was red, green or blue. The only issue was the quality of the camera. The objects appeared to be blurry and unclear, despite the camera being at its highest possible resolution.

Recommendation

1) Using a camera of better quality for clearer images
2) Faulty hardware should be replaced as soon as possible

Reference

Picture for part 2's set-up: https://www.hackster.io/JustJ0e/pixy-cmucam-arduino-3ac141

Student Declaration Form

STUDENT'S DECLARATION

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 acknowledgement, and that the original work contained herein have 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 that no further improvement on the reports is needed from any of the individual contributors to the report.

We therefore, agree unanimously that this report shall be submitted for **marking** and this **final printed report** has been **verified by us**.

Signature:	**Read**	/
Name: Muhamad Yusuf Irfan bin Abidin	**Understand**	/
Matric No: 2210915	**Agree**	/
Contribution : Abstract, Introduction, Experimental Set-Up		

Signature: Tanisha	**Read**	/
Name: Tanisha Rowshan Saleh	**Understand**	/
Matric No: 2228290	**Agree**	/
Contribution : Discussion (Part 2), Conclusion (Part 2), Recommendation		

Signature:	**Read**	/
Name: Muhamad Imran Haziq bin Haizam	**Understand**	/
Matric No: 2216429	**Agree**	/

Contribution : Abstract, Introduction, Materials and Equipment		

Signature:	**Read**	/
Name: Ariesha Daliela Hanim Binti Amir	**Understand**	/
Matric No: 2316834	**Agree**	/
Contribution : Methodology		