Real Time Project Report

on

"GESTURE CONTROLLED LED GLOW"

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

by

B.INDHU PRIYA - 22WH1A0472

M.SAI VARSHINI -22WH1A0476

M.SRAVANI - 22WH1A0477

A.SARAYU -22WH1A04B4

under the guidance of

Dr. V. Hindumathi

Associate Professor

VISHNU

UNIVERSAL LEARNING

BVRIT H

Estd. 2012

Department of Electronics and Communication Engineering

BVRIT HYDERABAD College of Engineering for Women

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with A Grade

Bachupally, Hyderabad – 500090

2023-24

DECLARATION

We hereby declare that the work described in this report, entitled **"GESTURE CONTROLLED LED GLOW"** which is being submitted by us in partial fulfillment for the award of the degree of **Bachelor of Technology** in the department of **Electronics and Communication Engineering** at **BVRIT HYDERABAD College of Engineering for Women,** affiliated to **Jawaharlal Nehru Technological University Hyderabad**, Kukatpally, Hyderabad – 500085 is the result of original work carried out by us under the guidance of **Dr. V.Hindumathi ,**Associate Professor

This work has not been submitted for any Degree/Diploma of this or any other institute/university to the best of our knowledge and belief.

Place: Hyderabad

Date: 03-07-2024

Names and signatures of the students

22WH1A0472- B.INDHU PRIYA

22WH1A0476- M.SAI VARSHINI

22WH1A0477 -M.SRAVANI

22WH1A04B4 - A.SARAYU

Department of Electronics and Communication Engineering
BVRIT HYDERABAD College of Engineering for Women
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with A Grade
Bachupally, Hyderabad – 500090

Certificate

This is to certify that the Real Time project report, entitled **"GESTURE CONTROLLED LED GLOW"** is a record of bonafide work carried out by B.Indhu priya(22WH1A0472), M.Sai Varshini(22WH1A0476), M.Sravani(22WH1A0477), A.Sarayu (22WH1A04B4) in partial fulfillment for the award of the degree of **Bachelor of Technology** in the department of **Electronics and Communication Engineering** at **BVRIT HYDERABAD College of Engineering for Women,** affiliated to **Jawaharlal Nehru Technological University Hyderabad**, Kukatpally, Hyderabad – 500085.

Supervisor Head of the Department

ACKNOWLEDGMENT

The satisfaction that accompanies in successful completion of the task would be incomplete without the mention of the people who made it possible.

We wish to express our deep sense of gratitude to our guide **Dr. V.Hindumathi,** Associate Professor, Department of Electronics and Communication Engineering, BVRIT HYDERABAD College of Engineering for Women, for his/her able guidance and suggestions, which helped us in completing this project work on time.

We would like to thank **Dr. Nagesh Deevi,** Associate Professor and Head, Department of Electronics and Communication Engineering for his guidance, support and encouragement.

We express our gratitude towards our honorable Principal, **Dr. K V N Sunitha** and the **management** for providing all the facilities.

We also thank all the **faculty** and **non-teaching staff members** of Electronics and Communication Engineering department, who supported us directly or indirectly in successful completion of this project work.

Finally, we thank all our **friends** and **family members** for their continuous support and help.

22WH1A0472- B.INDHU PRIYA

22WH1A0476- M.SAI VARSHINI

22WH1A0477 -M.SRAVANI

22WH1A04B4 - A.SARAYU

ABSTRACT

Gesture-controlled systems are becoming increasingly popular due to their intuitive interaction methods. This project focuses on designing and implementing a gesture-controlled LED system using finger count detection. The main objective is to create a user-friendly interface that allows users to control LED lights by simply showing a specific number of fingers to a camera.

The system employs a computer vision approach using a webcam to capture real-time hand images. These images are processed to detect and count the number of raised fingers. The finger count is then translated into commands to control the LED lights. For example, showing one finger might turn on one LED, while showing two fingers could turn on two LEDs, and so forth.

The core components of the system include a webcam, a microcontroller (such as Arduino), LEDs, and a computer running image processing software. OpenCV, a powerful library for computer vision, is used for image processing tasks such as background subtraction, contour detection, and finger count determination. The microcontroller receives the processed data from the computer and controls the LEDs accordingly.

CONTENTS

LIST OF FIGURES AND TABLES

1.Introduction

Gesture-controlled systems are becoming increasingly popular due to their intuitive interaction methods. This project focuses on designing and implementing a gesture-controlled LED system using finger count detection. The primary objective is to create a seamless and user-friendly interface that allows users to control LED lights by merely showing a specific number of fingers to a camera.

1.1 Background

Traditional control mechanisms for LED lights, such as switches and remote controls, require physical contact and can sometimes be cumbersome, especially for individuals with physical disabilities. Gesture control offers a more engaging and interactive approach, eliminating the need for physical contact. By employing computer vision techniques, real-time images of the user's hand can be captured and processed to detect and count the number of raised fingers. This finger count is then translated into commands to control the LED lights. For example, showing one finger could turn on one LED, while showing two fingers might turn on two LEDs, and so on. This approach simplifies the control process, making it more intuitive and accessible.

1.3 Objective

The main objective of this project is to develop a highly responsive and accurate gesture-controlled LED system that effectively detects finger counts to control LED lights.

2. Components Used

2.1 Hardware

2.1.1 Arduino UNO

The Arduino Uno is a popular microcontroller board designed for building digital devices and interactive objects that can sense and control physical devices. It is based on the ATmega328P microcontroller and features 14 digital input/output pins (six of which can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header, and a reset button. The board can be powered via a USB connection or with an external power supply, and it is compatible with a wide range of shields and sensors, making it highly versatile for various projects. The Arduino Uno is programmed using the Arduino IDE, a simple yet powerful development environment that supports C/C++ programming languages. The IDE includes a comprehensive library of functions for controlling the microcontroller's hardware features, allowing users to easily interface with sensors, motors, LEDs, and other components.

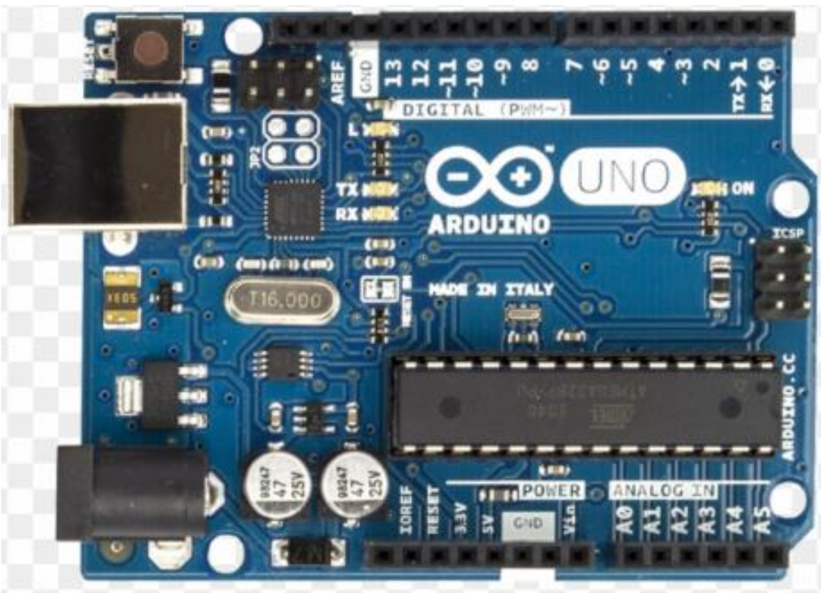


Fig.1 : Arduino UNO

Technical Specifications:

1. Operating Voltage: 5V
2. Input Voltage: 7-20 V
3. DC Current per I/O pin: 20mA
4. Flash Memory: 32KB
5. Clock Frequency: 16 MHz
6. No. of Digital Pins: 14
7. No. of Analog Pins: 6

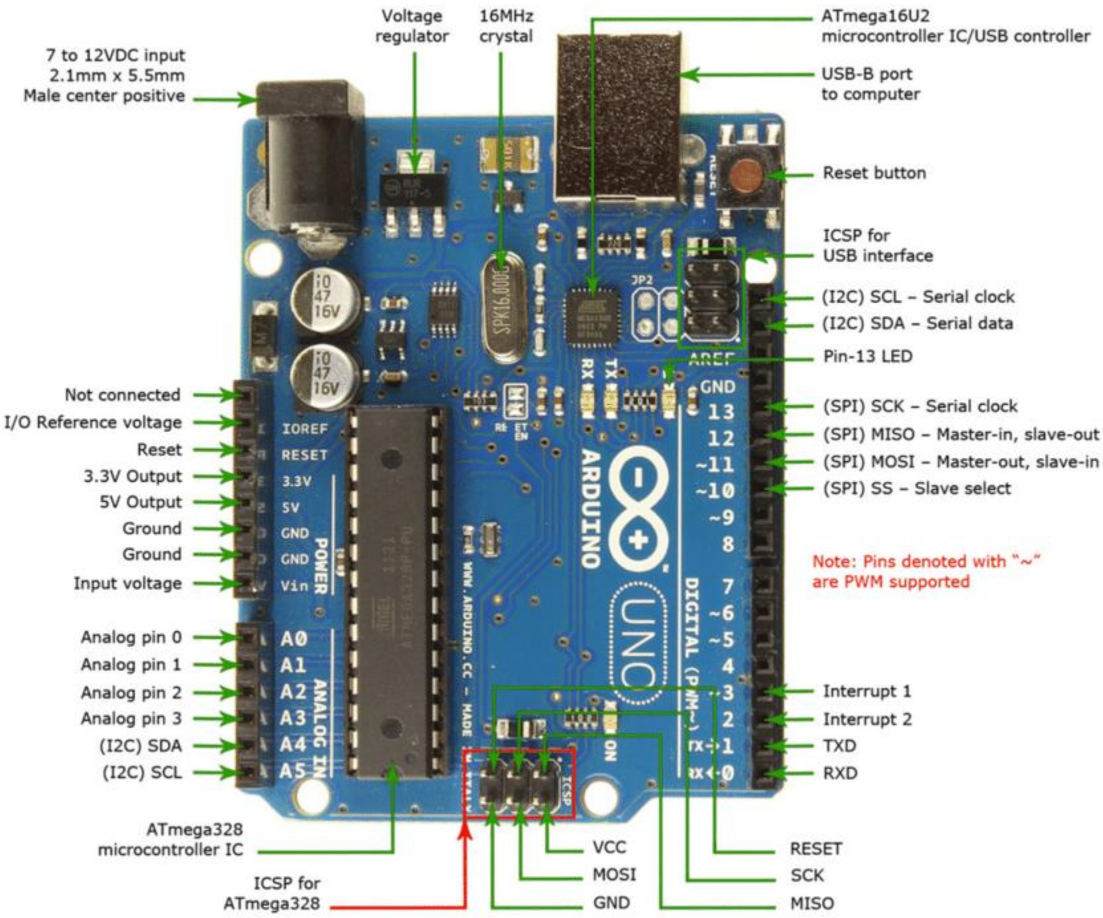


Fig.2 Detailed view of Arduino UNO

. The other components we used are Breadboard , leds and jumper wires and USB Cable.

3.Software

3.1 Arduino IDE

The Arduino Integrated Development Environment (IDE) is a software platform for writing, compiling, and uploading code to Arduino boards. It provides a user-friendly interface and extensive libraries to facilitate coding. The IDE supports C/C++ and offers various functions for hardware control and interaction.

void setup(): The void setup() function is called once when the program starts. It is used to initialize variables, pin modes, start using libraries, and set up serial communication. This function prepares the Arduino board to run the main code.

pinMode(): The pinMode(pin, mode) function configures a specified pin to behave either as an input or an output. The \`pin\` parameter specifies the pin number, and the \`mode\` parameter sets the mode (e.g., \`INPUT\`, \`OUTPUT\`, \`INPUT_PULLUP\`). This function is essential for defining how pins will be used in the project.

Serial.begin(): The Serial.begin(baud rate) function initializes serial communication at the specified baud rate. This allows the Arduino to communicate with a computer or other serial devices. It's commonly used for debugging and sending/receiving data.

void loop(): The void loop() function runs continuously after the \`setup()\` function has completed. This is where the main code resides and repeats indefinitely. Any instructions within this function are executed in a cycle.

digitalWrite(): The digitalWrite(pin, value) function sets a specified digital pin to either HIGH or LOW. The \`pin\` parameter specifies the pin number, and the \`value\` parameter sets the output (e.g., \`HIGH\`, \`LOW\`). This function controls the voltage level of digital pins.

digitalRead(): The digitalRead(pin) function reads the value from a specified digital pin. It returns either HIGH or LOW depending on the voltage level at the pin. This function is used to detect the state of digital inputs.

analogRead(): The analogRead(pin) function reads the value from a specified analog pin and returns a value between 0 and 1023. This function is used for reading the analog voltage level from sensors and other analog devices.

analogWrite(): The analogWrite(pin, value) function writes an analog value (PWM wave) to a specified pin. The `value` parameter sets the duty cycle between 0 (off) and 255 (fully on). This function is used to control devices like LEDs and motors that require variable output levels.

3.2 Visual Studio

Visual Studio is an integrated development environment (IDE) from Microsoft used for developing software applications. It supports a wide range of programming languages including C++, C#, and Python, and provides comprehensive tools for code editing, debugging, and version control. Visual Studio offers features such as IntelliSense for code completion, a robust debugger, and a suite of testing tools, making it a powerful choice for professional and enterprise-level development.

3.3 OpenCV

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. It provides a comprehensive set of tools and algorithms for image processing, video analysis, and real-time computer vision applications. OpenCV supports various programming languages such as C++, Python, and Java, and includes functionality for tasks such as object detection, facial recognition, image segmentation, and motion tracking. Its extensive documentation and large community make it accessible for both beginners and advanced users.

4.System Architecture

4.1 Hardware Integration:

The hardware integration for the finger count gesture-controlled LED glow system typically includes components like a webcam for image capture, an Arduino Uno microcontroller for processing data and controlling LEDs, LEDs themselves for illumination, and a computer running image processing software such as OpenCV. The webcam captures real-time images of the user's hand, which are processed using computer vision techniques to detect and count raised fingers. This count is then translated into commands sent to the Arduino Uno, which adjusts the states of the LEDs accordingly, enabling intuitive and contactless control of lighting based on gestures.

4.2 Software Development:

The software development for the finger count gesture-controlled LED glow system involves capturing hand movements with a webcam. OpenCV processes these images to detect and count raised fingers using techniques like contour analysis. The finger count data is then communicated to an Arduino Uno via serial communication. The Arduino controls LEDs based on the finger count, providing a responsive and intuitive way to interact with lighting environments. Testing ensures accuracy and reliability across different conditions.

4.3 Block Diagram

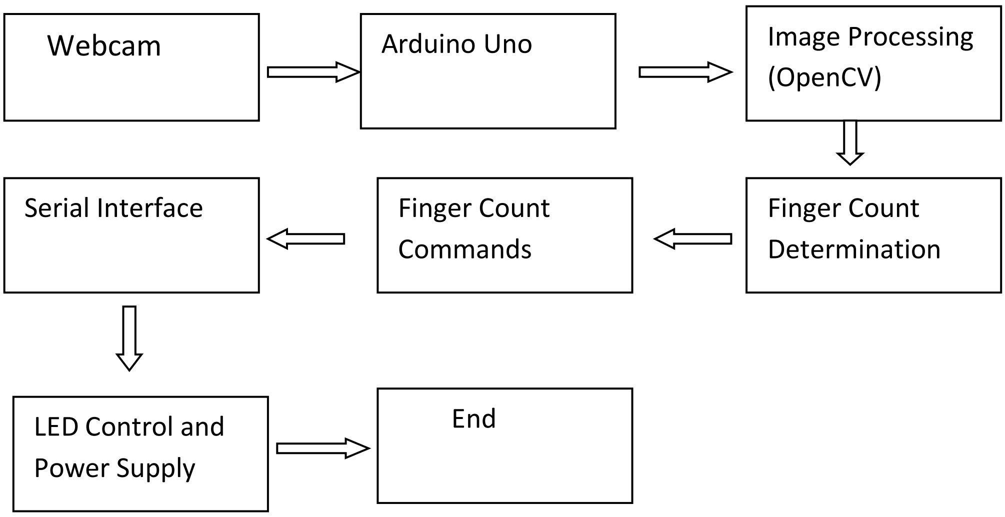
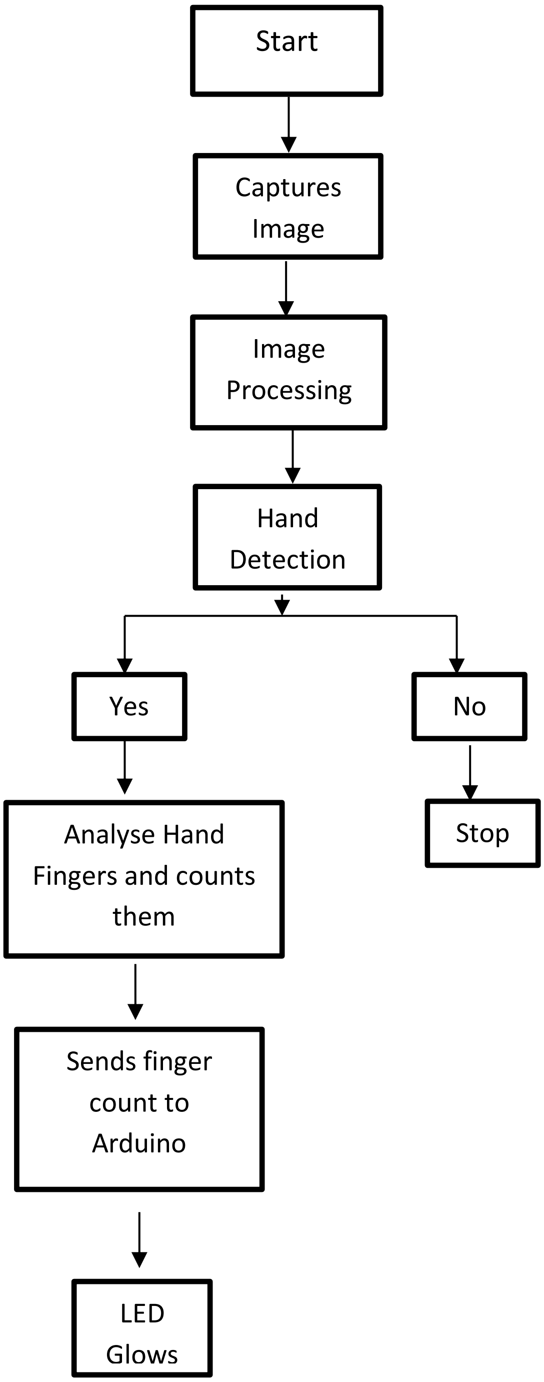


Fig.3 Block Diagram

- **Webcam**: Captures real-time images of the user's hand gestures.
- **Arduino Uno**: Receives finger count data through serial communication from the computer running OpenCV.
- **Image Processing (OpenCV)**: Processes the webcam images to detect and count the number of raised fingers using computer vision techniques.
- **Finger Count Determination**: Determines the finger count based on the processed image data.
- **Finger Count Commands**: Commands indicating the number of fingers detected are sent to the Arduino Uno.
- **Serial Interface**: Handles the communication protocol between the computer and Arduino Uno.
- **LED Control and Power Supply**: Arduino Uno controls and supplies power to the LEDs based on the received commands.
- **End**: The LED array provides the illuminated output based on the detected finger count, completing the system's operation.

5. Proposed System

5.1 Flow Chart



6.IMPLEMENTATION

6.1 Arduino Code

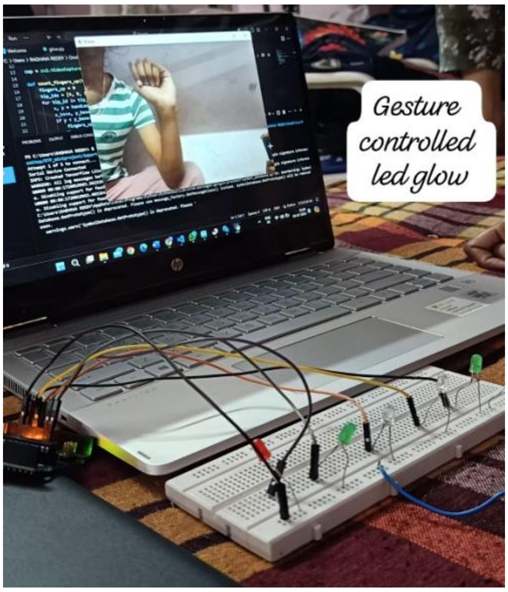
```
const int ledPins[] = { 8, 9, 10,11,12};  // Replace with your LED pin numbers
const int numLeds = sizeof(ledPins) / sizeof(ledPins[0]);
void setup() {
  Serial.begin(9600);
  for (int i = 0; i < numLeds; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
}
void loop() {
  if (Serial.available() > 0) {
    int fingers_up = Serial.parseInt();
    for (int i = 0; i < numLeds; i++) {
      if (i < fingers_up) {
        digitalWrite(ledPins[i], HIGH);
      } else {
        digitalWrite(ledPins[i], LOW);
      }
    }
  }
}
```

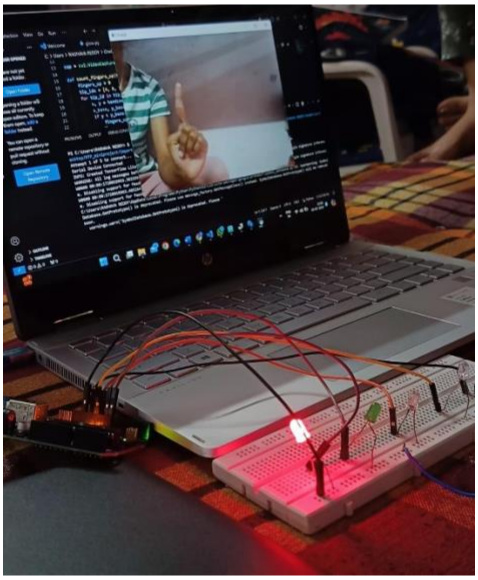
6.2 Python Code(Visual Studio)

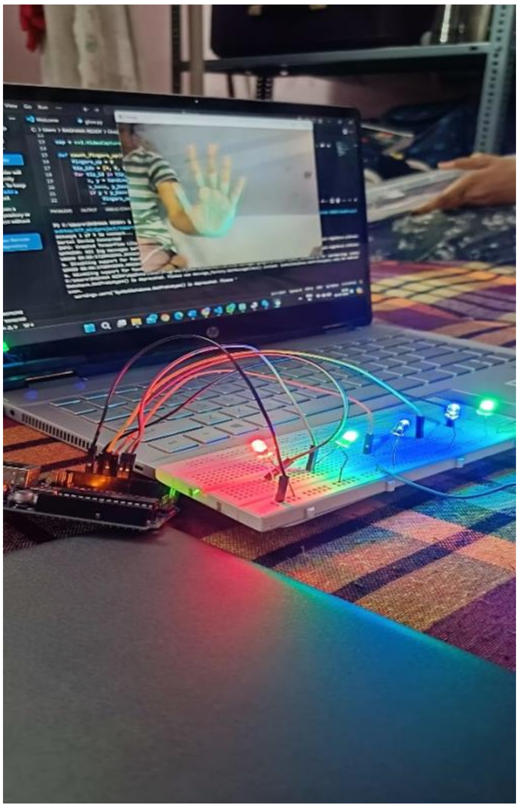
```
import cv2

import mediapipe as mp

from cvzone.SerialModule import SerialObject

arduino = SerialObject('COM3')

mpHands = mp.solutions.hands
```

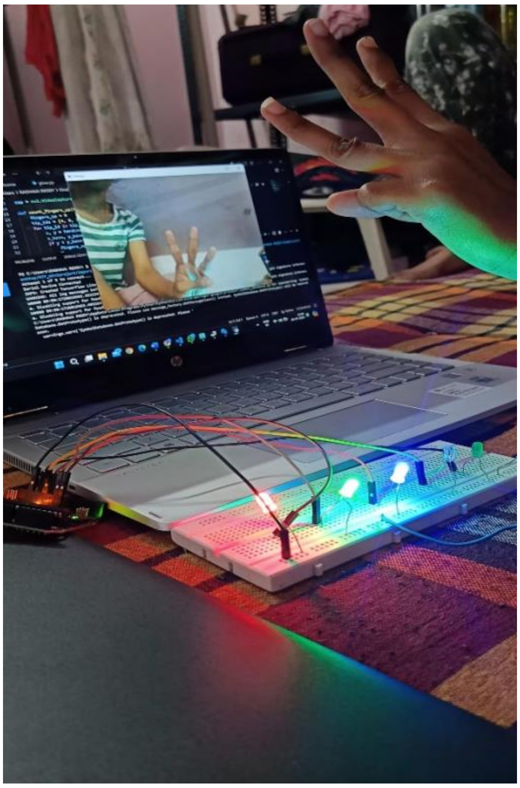
```
hands = mpHands.Hands(max_num_hands=1, min_detection_confidence=0.7)

cap = cv2.VideoCapture(0)

def count_fingers_up(handLms):

    fingers_up = 0

    tip_ids = [4, 8, 12, 16, 20]

    for tip_id in tip_ids:

        x, y = handLms.landmark[tip_id].x, handLms.landmark[tip_id].y

        x_base, y_base = handLms.landmark[tip_id - 2].x, handLms.landmark[tip_id -
2].y

        if y < y_base:

            fingers_up += 1

    return fingers_up

while True:

    success, img = cap.read()

    imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    results = hands.process(imgRGB)

   if results.multi_hand_landmarks:

        for handLms in results.multi_hand_landmarks:

            fingers_up = count_fingers_up(handLms)

            arduino.sendData([fingers_up])

 cv2.imshow("Image", img)

 cv2.waitKey(1)
```

7.RESULT









8.CONCLUSION

In conclusion, the gesture-controlled LED glow project using Arduino successfully demonstrates the integration of hardware and software to create an interactive and user-friendly system. By utilizing sensors such as the APDS-9960, along with an Arduino microcontroller, this project showcases the potential for developing intuitive control systems without the need for physical contact.

Key achievements of this project include:

1. Effective Gesture Recognition: The system can accurately recognize a variety of gestures, including swipes and proximity-based commands, allowing for seamless user interaction.

2. Responsive LED Control: The LEDs respond promptly to the gestures, providing immediate visual feedback and enhancing the user experience.

3. Scalability and Flexibility: The project can be easily expanded or modified to include additional features, such as controlling multiple LEDs, integrating different sensors, or applying the concept to other appliances.

4. Educational Value: This project serves as an excellent educational tool for learning about sensor integration, signal processing, and microcontroller programming.

Overall, the gesture-controlled LED glow project highlights the versatility and capability of Arduino-based systems in creating innovative and interactive electronic projects. It opens up opportunities for further exploration in the fields of home automation, user interface design, and assistive technologies.

8.1Future Scope

1. Enhanced Gesture Recognition: Implementing more advanced algorithms and machine learning for a wider range of gestures.

2. Multi-Device Integration: Expanding control to various household and smart devices for a comprehensive smart home environment.

3. Wireless Connectivity: Adding Bluetooth or Wi-Fi for remote control via smartphones and other devices.

4. Energy Efficiency: Developing low-power versions for portable or battery-operated applications.

5. Advanced Interfaces: Combining gesture control with voice commands or touchscreens for versatile user interfaces.

6. Commercial Applications: Adapting the technology for use in retail displays, automotive controls, and healthcare environments.

7. Community Contributions: Leveraging open-source contributions for new features and improvements.

8. Educational Kits: Creating kits to teach electronics, programming, and human-computer interaction.

9. REFERENCES

1. Electronics For You Magazine
2. C. -H. Hung, Y. -W. Bai and H. -Y. Wu, "Home outlet and LED array lamp controlled by a smartphone with a hand gesture recognition," 2016 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 2016, pp. 5-6, doi: 10.1109/ICCE.2016.7430502
3. P. Lin, R. Zhuo, S. Wang, Z. Wu and J. Huangfu, "LED Screen-Based Intelligent Hand Gesture Recognition System," in IEEE Sensors Journal, vol. 22, no. 24, pp. 24439-24448, 15 Dec.15, 2022, doi: 10.1109/JSEN.2022.3219645.
4. C. -H. Hung, Y. -W. Bai and H. -Y. Wu, "Home appliance control by a hand gesture recognition belt in LED array lamp case," 2015 IEEE 4th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 2015, pp. 599-600, doi: 10.1109/GCCE.2015.7398611