Gesture Controlled Home Automation System

Overview

The **Gesture Controlled Home Automation System** enables touch-free control of home appliances using hand gestures. Using a **Raspberry Pi** along with a **Pi Camera**, this system recognizes hand gestures and controls devices such as LEDs accordingly. The project is an innovative application of **computer vision** and **gesture recognition**, which makes it easy and hygienic to operate appliances without physical touch.

Objective

The primary objective of this project is to create an intuitive, hands-free solution for home automation, allowing users to control devices with simple hand gestures. The system detects the number of fingers raised and interprets these gestures to turn on or off connected appliances like LEDs.

Libraries Used

- **cv2**: For image processing and displaying the video feed.
- **cvzone.HandTrackingModule**: For detecting and analyzing hand gestures.
- **picamera2**: To capture images from the Raspberry Pi camera.
- **gpiozero.LED**: To control LEDs connected to GPIO pins.
- **time.sleep**: For creating delays when needed.

Hardware Requirements

- **Raspberry Pi**: A small, affordable computer to run the system.
- **Pi Camera**: For capturing live video of the user's hand gestures.
- **LEDs**: Three LEDs connected to GPIO pins (17, 18, 27) with appropriate resistors to visualize the system's output.
- **Resistors**: To limit the current flowing through the LEDs and protect them from damage.
- **Breadboard & Jumper Wires**: For easy connections.

Software Requirements

- **Python 3**: The primary programming language.
- **cv2**: For image and video processing.
- **cvzone**: For hand gesture detection.
- **gpiozero**: To control GPIO pins for LEDs.
- **picamera2**: For interfacing with the Raspberry Pi camera module.

Install the required libraries:

bash
Copy code
pip install opencv-python
pip install cvzone
pip install gpiozero
pip install picamera2

Code

```
import cv2
from cvzone.HandTrackingModule import HandDetector
from picamera2 import Picamera2
from gpiozero import LED
from time import sleep

# Initialize Picamera2
picam2 = Picamera2()
picam2.preview_configuration.main.size = (720, 560)
picam2.preview_configuration.main.format = "RGB"
picam2.preview_configuration.align()
picam2.configure("preview")
picam2.start()

# Initialize HandDetector
detector = HandDetector(maxHands=2, detectionCon=0.5, minTrackCon=0.5)

# Initialize LEDs connected to GPIO pins
led1 = LED(17)  # GPIO pin 17
led2 = LED(18)  # GPIO pin 18
led3 = LED(27)  # GPIO pin 27

# Variables to track LED states
led1_state = False
led2_state = False
led3_state = False

while True:
    # Capture image from Picamera2
    im = picam2.capture_array()
    im = cv2.cvtColor(im, cv2.COLOR_RGB2BGR)

    # Detect hands and draw landmarks
    hands, im = detector.findHands(im, draw=True)
```

```
# Check if hands are detected
if hands:
    # Iterate over each hand detected
    for hand in hands:
        lmlist = hand['lmList']
        cx = lmlist[4][0]
        cy = lmlist[4][1]
        cv2.circle(im, (cx, cy), 13, (255, 0, 0), -1)

        fingers = detector.fingersUp(hand)
        count_1 = fingers.count(1)
        count_0 = fingers.count(0)

        print("Count of 1s:", count_1)
        print("Count of 0s:", count_0)

        # Gesture for turning on/off LED1
        if count_1 == 5:
            if not led1_state:
                led1.on()
                led1_state = True
                print("LED1 turned on")
        elif count_0 == 4:
            if led1_state:
                led1.off()
                led1_state = False
                print("LED1 turned off")

        # Gesture for turning on/off LED2
        if count_1 == 3:
            if not led2_state:
                led2.on()
                led2_state = True
                print("LED2 turned on")
        elif count_0 == 2:
            if led2_state:
                led2.off()
                led2_state = False
                print("LED2 turned off")

        # Gesture for turning on/off LED3
        if count_1 == 2:
            if not led3_state:
                led3.on()
```

```
            led3_state = True
            print("LED3 turned on")
        elif count_0 == 1:
            if led3_state:
                led3.off()
                led3_state = False
                print("LED3 turned off")

    cv2.imshow("im", im)
    key = cv2.waitKey(1) & 0xFF

    if key == ord("q"):
        cv2.destroyAllWindows()
        break
```

Working Principle

The **Gesture Controlled Home Automation System** works by capturing real-time video from the **Pi Camera** and analyzing the video feed using the **cvzone HandTrackingModule**. It detects hand gestures and determines the number of fingers held up. Each unique gesture corresponds to specific actions such as turning on or off LEDs.

Gesture Recognition Logic

- **5 Fingers Up** → Turn on LED1.
- **1 Finger Up** → Turn off LED1.
- **3 Fingers Up** → Turn on LED2.
- **3 Fingers Down** → Turn off LED2.
- **2 Fingers Up** → Turn on LED3.
- **1 Finger Down** → Turn off LED3.

Step-by-Step Process

1. Setting Up the Raspberry Pi

- Set up your Raspberry Pi with Raspbian OS installed.
- Connect the **Pi Camera** to the Raspberry Pi.
- Attach three LEDs to GPIO pins (17, 18, 27) and connect them to the breadboard with resistors.

2. Installing Dependencies

Install the required libraries for image processing and hand gesture detection:

bash

Copy code
pip install opencv-python
pip install cvzone
pip install gpiozero
pip install picamera2

3. Initializing the Camera

The **Picamera2** library is used to capture real-time video from the **Pi Camera**. It is initialized to capture video in **RGB format**.

python
Copy code
picam2 = Picamera2()
picam2.preview_configuration.main.size = (720, 560)
picam2.preview_configuration.main.format = "RGB"
picam2.configure("preview")
picam2.start()

4. Hand Gesture Detection

The **HandDetector** from **cvzone** is initialized to detect hands in the video feed. It can detect up to 2 hands, tracking the position of the fingers and palm.

python
Copy code
detector = HandDetector(maxHands=2, detectionCon=0.5, minTrackCon=0.5)

5. Controlling LEDs Based on Gestures

For each hand detected, the system counts the number of fingers held up. Based on this count, the system controls the state of the LEDs.

- **5 Fingers Up** turns on LED1.
- **1 Finger Up** turns off LED1.
- **3 Fingers Up** turns on LED2.
- **3 Fingers Down** turns off LED2.
- **2 Fingers Up** turns on LED3.
- **1 Finger Down** turns off LED3.

Example code to detect and control LED:

python
Copy code
if count_1 == 5:
    led1.on()elif count_0 == 4:
    led1.off()

6. Displaying the Video Feed

The processed video feed is displayed with hand landmarks and the thumb tip highlighted for visual feedback. The system provides real-time feedback about the detected gesture.

```
python
Copy code
cv2.imshow("im", im)
```

7. Exiting the Program

The program runs in a loop until the user presses the "q" key. At that point, the program stops and closes the video feed window.

```
python
Copy code
key = cv2.waitKey(1) & 0xFFif key == ord("q"):
    cv2.destroyAllWindows()
    break
```

Conclusion

This **Gesture Controlled Home Automation System** demonstrates how **Raspberry Pi** and **computer vision** can be used to create a hands-free, intuitive solution for controlling home appliances. By leveraging the power of **gesture recognition**, the project enhances interactivity, convenience, and accessibility for users in smart home environments.