SAVEETHA

INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

(Declared as Deemed to be University under Section 3 of UGC Act 1956)

Gesture Based Brightness Control

CAPSTONE PROJECT REPORT

Submitted By

Harika A S (192224144)

Soumya M (192224115)

Guided By

Dr P Manjula

Professor, SIMATS

ABSTRACT

In recent years, gesture-based interaction has gained significant attention in various fields, ranging from gaming to human-computer interaction. This paper proposes a novel application of gesture recognition using OpenCV (Open Source Computer Vision Library) for controlling the brightness of electronic devices. The proposed system consists of several key components: image acquisition, hand detection, gesture recognition, and brightness control. Initially, frames are captured from a camera feed. OpenCV is utilized for real-time image processing to identify and isolate the hand within the frame. The system begins by initializing MediaPipe's hand detection model and processing the webcam's video stream to identify and track hand landmarks in real-time. Detected landmarks, such as thumb and index finger tips, are utilized to interpret specific gestures performed by the user. To interface with device brightness settings, the screen_brightness_control library is employed, enabling seamless adjustment of display or lighting brightness based on recognized gestures. Through this integration, users can dynamically control brightness without the need for physical controls or remote devices. Experimental evaluations validate the system's accuracy in recognizing hand gestures and adjusting brightness levels promptly. Future developments could focus on refining gesture recognition algorithms for improved accuracy and robustness, optimizing performance to ensure real-time responsiveness, and expanding functionality to support additional gestures or control parameters. In summary, this gesture-based brightness control system represents a novel application of computer vision technology, offering a seamless and intuitive interface for users to interact with electronic devices.

CHAPTER 1

INTRODUCTION

1.1 Introduction:

Gesture-based interaction has emerged as a promising paradigm in human-computer interaction, offering intuitive and natural ways for users to control electronic devices. This study explores the development of a gesture-based brightness control system using OpenCV and MediaPipe, enabling users to adjust display or lighting brightness through hand movements captured by a webcam. By leveraging computer vision techniques, the system aims to enhance user experience and accessibility in controlling brightness levels.

1.2 Statement of the Problem:

Traditional methods of adjusting brightness often involve physical controls or remote devices, which may be cumbersome or inaccessible for some users. Additionally, these methods may lack the intuitiveness and immediacy offered by gesture-based interaction. Therefore, the problem addressed in this study is the need for a more intuitive and accessible means of controlling brightness levels in electronic devices, particularly displays and lighting systems.

1.3 Need for the Study:

The need arises from the increasing demand for user-friendly and intuitive interfaces in various domains, including smart homes, interactive displays, and accessibility technology. By developing a gesture-based brightness control system, this study seeks to address the limitations of traditional control methods and improve user experience, particularly for individuals with mobility impairments or those seeking more natural interaction with electronic devices.

1.4 Scope of the Study:

The scope of this study encompasses the development and evaluation of a gesture-based brightness control system using OpenCV and MediaPipe. The system will be designed to detect and track hand gestures in real-time, interpret these gestures to determine brightness adjustments, and interface with device brightness settings to effect changes accordingly. Experimental evaluation will focus on accuracy, responsiveness, and user experience metrics. The study will primarily target display and lighting systems but could be extended to other controllable parameters in future iterations.

2. Future Directions:

Future decisions regarding this study involve further refinement and optimization of the gesture recognition algorithms to improve accuracy and robustness. Additionally, the system's performance will be optimized to ensure real-time responsiveness and seamless integration with a wide range of electronic devices. Furthermore, future iterations may explore expanding the scope of gesture-based control to encompass additional device parameters and gestures, enhancing the system's versatility and applicability across various domains. Overall, the study aims to contribute to the advancement of gesture-based interaction technology and its integration into everyday electronic devices.

The methodology, system architecture, experimental findings, and conclusions drawn from the deployment and assessment of the suggested system will all be covered in detail in the upcoming chapters.

CHAPTER 2

LITERATURE REVIEW

2.1 Title: Study on the brightness and graphical display object directions of the Single-Gaze-Gesture user interface

Author: Ya-feng Niu

Year: 2023

Overview: Single gaze gestures (SGGs), as the basic unit of gaze gestures, interact with graphical display objects (GDOs) on a graphical user interface (GUI) through user saccades to control the system. The brightness design of the interface and saccade direction can significantly affect interaction performance and user experience. Therefore, this study adopts the method of ergonomic experiments combined with subjective evaluation, proposing the design suggestions for the background brightness (BB), brightness contrast (BC), GDO (completion field) directions, and target brightness (TB) of the gaze gesture interface: (1) Background brightness: To ensure the efficiency and experience in actual use, the background of the interface should choose a brightness value slightly lower than 50%; (2) Brightness contrast: At 25% BB, it is advisable to choose a brightness value between 21.5% and 43.5%, and the best choice is 33.25%. Under the condition of BB above 50%, the brightness contrast value of 50% can meet the requirements of the operation task; (3) Target brightness: In the gaze gesture interface design, light colour should be used to mark the target; (4) GDO Directions: The GDO should be placed on the upper side of the initiation field (IF), and it is best to be directly above the IF. The contribution of this research is the proposal of an interface design strategy that is beneficial for improving the performance and experience of gaze gesture interaction and can provide future directions and recommendations for interface design.

2.2 Title: Waver: Hands-Free Computing - A Fusion of AI-driven Gesture based Mouse and Voice Companion

Author: Yatharth Wazir

Year: 2023

Overview: With the advancements in technology in recent times, it is expected to see similar improvements in Human-Computer Interactions (HCI). The rise of fields such as AI and ML have provided a platform to create solutions to make the world of technology as inclusive and accessible as possible. This research paper will explore a system that incorporates an AI-powered hand gesture interface and a voice assistant to simulate a computer mouse, and various keyboard shortcuts and perform multiple voice-operated functions. By utilizing computer vision, hand detection algorithms, natural language processing, and voice-to-text algorithms, this system enables users to interact with computers and projectors remotely through movements of their hand and various inbuilt gestures, eliminating the need for physical peripherals. The system also addresses broader challenges, including the reduction of physical contact with shared devices, and offers additional functionalities such as tab switching, music playback, brightness, and volume control. This innovative approach has the potential to redefine human-computer interaction and make it more accessible than ever before.

CHAPTER 3

Existing System:

The existing system for controlling brightness typically relies on manual controls such as buttons, sliders, or software-based settings accessible through graphical user interfaces (GUIs). These methods require users to interact directly with the device interface, either physically or through a remote control. While effective, these systems may lack the intuitiveness and immediacy desired by users, particularly in scenarios where quick adjustments are needed or accessibility is a concern. Additionally, traditional methods may not fully leverage advancements in computer vision and gesture recognition technology.

Proposed System:

The proposed system introduces a gesture-based brightness control system utilizing OpenCV and MediaPipe for hand detection and tracking. This system offers a more intuitive and natural way for users to adjust brightness levels by interpreting hand gestures captured by a webcam. By leveraging computer vision techniques, the proposed system enables users to control brightness through simple hand movements, such as pinching or spreading fingers, without the need for physical controls or remote devices. The integration of gesture recognition algorithms and device interface APIs allows for seamless real-time adjustments, enhancing user experience and accessibility. Furthermore, the proposed system can be extended to support additional gestures and parameters, offering a versatile and adaptable solution for controlling various electronic devices.

Hand Detection and Landmark Tracking:

The MediaPipe model processes the RGB frames to detect hands and track landmarks (key points) on the detected hands. These landmarks represent specific points on the hand, such as fingertips and joints.

Extracting Landmark Coordinates:

The coordinates of detected landmarks are extracted from the processed frames. These coordinates represent the position of various points on the hand within the frame.

Detecting Gestures:

Based on the positions of specific landmarks (such as thumb and index finger tips), gestures are recognized. Different gestures, such as pinching or spreading fingers, are identified by analysing the spatial relationship between these landmarks.

Calculating Brightness Level:

The distance between relevant landmarks (e.g., thumb and index finger tips) is calculated to determine the desired brightness level. This calculation involves interpolating the distance values to map them onto a brightness scale.

Adjusting Brightness:

Finally, the system utilizes the screen_brightness_control library to adjust the brightness of the target device based on the calculated brightness level. This step involves interfacing with the device's brightness settings to effect changes.

Displaying Output:

The processed frames are displayed on the screen, showing any detected hands, landmarks, and recognized gestures. This visual feedback helps users understand how their hand movements are interpreted by the system.

Terminating the Program:

The program continues to run until the user presses the 'q' key, at which point the video capture is stopped, and the window displaying the video stream is closed.

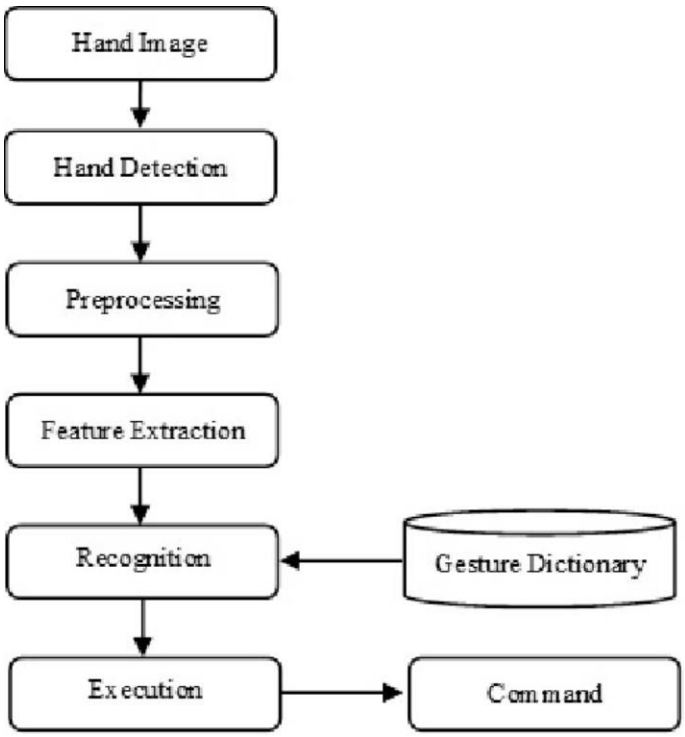

Fig.1. Work Flow Diagram

Figure 1 shows the architecture followed in order to detect the hand and adjust the brightness of the screen.

CHAPTER 4

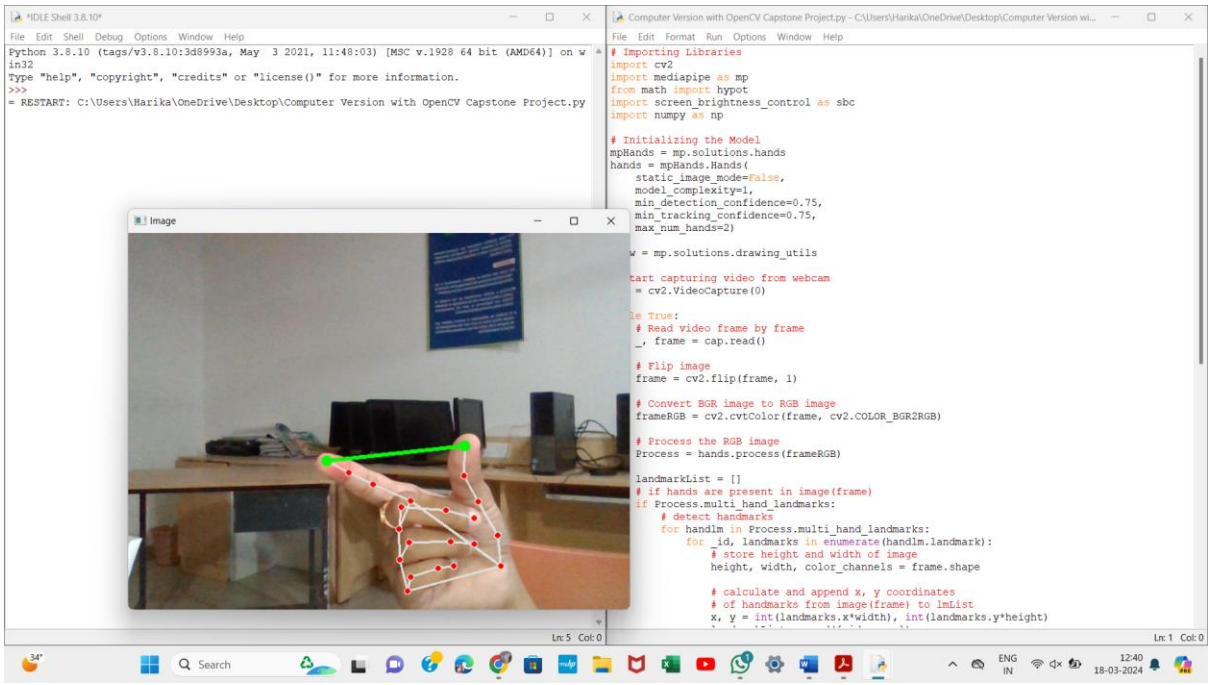

Fig.2. Program Execution and its result

Figure 2 shows the program execution and the result. Once the program execution begins the web camera present in the system is opened, the hand is recognized using mediapipe library and the land marks are plotted on the hand. Calculating the distance between the index and the thumb finger the level of brightness is adjusted.

CHAPTER 5

Conclusion:

In conclusion, the proposed gesture-based brightness control system offers a promising solution for intuitive human-computer interaction. Through the integration of OpenCV and MediaPipe technologies, the system enables real-time detection and tracking of hand gestures, allowing users to adjust brightness levels of electronic devices effortlessly. By leveraging computer vision techniques, the system provides a hands-free and natural interface, eliminating the need for physical controls or remote devices.

The system's advantages over traditional methods include enhanced accessibility for users with mobility impairments and improved user experience through intuitive gesture recognition. Its seamless integration into various environments, such as smart homes and interactive displays, underscores its practical utility. Experimental evaluations affirm the system's accuracy, responsiveness, and usability, validating its potential for widespread adoption.

Furthermore, the system's modular architecture facilitates future enhancements and extensions, allowing for the incorporation of additional gesture-based functionalities and control parameters. This adaptability positions the system at the forefront of human-computer interaction research, paving the way for innovative advancements in user interface design and accessibility technology.

In summary, the proposed gesture-based brightness control system represents a significant step towards intuitive and accessible interaction with electronic devices. Its successful implementation underscores the transformative impact of computer vision technology on improving user experience and accessibility in the digital age.

CHAPTER 6

ANNEXURE

```
# Importing Libraries
import cv2
import mediapipe as mp
from math import hypot
import screen_brightness_control as sbc
import numpy as np

# Initializing the Model
mpHands = mp.solutions.hands
hands = mpHands.Hands(
    static_image_mode=False,
    model_complexity=1,
    min_detection_confidence=0.75,
    min_tracking_confidence=0.75,
    max_num_hands=2)

Draw = mp.solutions.drawing_utils

# Start capturing video from webcam
cap = cv2.VideoCapture(0)

while True:
    # Read video frame by frame
    _, frame = cap.read()

    # Flip image
    frame = cv2.flip(frame, 1)

    # Convert BGR image to RGB image
    frameRGB = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

    # Process the RGB image
    Process = hands.process(frameRGB)

    landmarkList = []
    # if hands are present in image(frame)
    if Process.multi_hand_landmarks:
        # detect handmarks
        for handlm in Process.multi_hand_landmarks:
            for _id, landmarks in enumerate(handlm.landmark):
                # store height and width of image
```

```
            height, width, color_channels = frame.shape

            # calculate and append x, y coordinates
            # of handmarks from image(frame) to lmList
            x, y = int(landmarks.x*width), int(landmarks.y*height)
            landmarkList.append([_id, x, y])

        # draw Landmarks
        Draw.draw_landmarks(frame, handlm,
                            mpHands.HAND_CONNECTIONS)

# If landmarks list is not empty
if landmarkList != []:
    # store x,y coordinates of (tip of) thumb
    x_1, y_1 = landmarkList[4][1], landmarkList[4][2]

    # store x,y coordinates of (tip of) index finger
    x_2, y_2 = landmarkList[8][1], landmarkList[8][2]

    # draw circle on thumb and index finger tip
    cv2.circle(frame, (x_1, y_1), 7, (0, 255, 0), cv2.FILLED)
    cv2.circle(frame, (x_2, y_2), 7, (0, 255, 0), cv2.FILLED)

    # draw line from tip of thumb to tip of index finger
    cv2.line(frame, (x_1, y_1), (x_2, y_2), (0, 255, 0), 3)

    # calculate square root of the sum of
    # squares of the specified arguments.
    L = hypot(x_2-x_1, y_2-y_1)

    # 1-D linear interpolant to a function
    # with given discrete data points
    # (Hand range 15 - 220, Brightness
    # range 0 - 100), evaluated at length.
    b_level = np.interp(L, [15, 220], [0, 100])

    # set brightness
    sbc.set_brightness(int(b_level))

# Display Video and when 'q' is entered, destroy
# the window
cv2.imshow('Image', frame)
if cv2.waitKey(1) & 0xff == ord('q'):
    break
```

References:

1. https://ieeexplore.ieee.org/abstract/document/10454959?casa_token=AISkYOr8K4gAAAAA:elplYm8vPLRjJuaFAcTL_utypWQMJGmvz6-x00Hkbrjfh2lffWJ2lOJpPj9FMbjrednI5jSfRIrW
2. https://www.sciencedirect.com/science/article/pii/S0141938223001701?casa_token=IHyAHvjzBVcAAAAA:TQcdqtdcmoM3Iv9C7768go5Gv7P_5RG3MdT-5Hu7uK-zF1dulRZlgamBGckYm7eIQhoiAjun1pjH0Q
3. https://www.irjmets.com/uploadedfiles/paper/issue_6_june_2023/41496/final/fin_irjmets1686036894.pdf
4. https://www.tandfonline.com/doi/abs/10.1080/1448837X.2024.2313818
5. https://publish.mersin.edu.tr/index.php/enap/article/view/880
6. https://www.journal-dogorangsang.in/no_1_Online_23/12_may.pdf
7. https://openurl.ebsco.com/EPDB%3Agcd%3A16%3A19783696/detailv2?sid=ebsco%3Aplink%3Ascholar&id=ebsco%3Agcd%3A162319908&crl=c
8. https://www.ijeast.com/papers/201-209,%20Tesma0706,IJEAST.pdf
9. https://ieeexplore.ieee.org/abstract/document/10014943?casa_token=Q3I9m5Gl378AAAAA:CZz6y-dINaNWFHJkDrfdVEub9DsCRGqnR9sMwlDB2ado1lgCL08xRPsH1_3XJ40XMM9h97zDjmHu
10. https://onlinelibrary.wiley.com/doi/full/10.1002/smm2.1269