NAGARJUNA COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autunomous College under VTU, Belagavi, Accredited by NAAC with “A+” Grade)

A Mini Project Report
On

“Efficient Volume Control using Hand Gesture”

Submitted in partial fulfilment for the course 21IST67 of the degree

BACHELOR OF ENGINEERING

INFORMATION SCIENCE AND ENGINEERING

Submitted by

Rajmouli K V (1NC21IS047)

Under the Guidance of
Mrs. Nethra B R
Assistant Professor
Dept. of ISE, NCET

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
NAGARJUNA COLLEGE OF ENGINEERING AND TECHNOLOGY
(An Autonomous Institution under VTU, Accredited by NAAC with “A+” Grade)
BENGALURU– 562164

2023-2024

NAGARJUNA COLLEGE OF ENGINEERIGN AND TECHNOLOGY

Autonomous College under VTU, Accredited by NAAC with "A" Grade, Bengaluru-562164, Karnataka, India

Department of Information Science and Engineering

CERTIFICATE

This is to certify that the Mini project work entitled "Efficient Volume Control using Hand Gesture" by **Rajmouli K V (1NC21IS047)** of **Nagarjuna College of Engineering andTechnology**, an autonomous Institution under Visvesvaraya Technological University, Belagavi in partial fulfilment for the course (21IST67) Of **Bachelor of Engineering in Information Science and Engineering** during the academic year 2023-2024. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report.

Name and Signature of the Guide
Mrs. Nethra B R
Assistant Professor,
Dept. of NCET

Name and Signature of the HOD
Dr. Sanjeeva Kumar Hatture
HOD,
Dept. of NCET

Name and Signature of the Principal
Dr. B V Ravishankar
Principal,
NCET

Name of the Examiner

1. ....

2. ....

Signature with date

1. ....

2. ....

ACKNOWLEDGEMENT

It is my proud privilege and duty to acknowledge the kind of help and guidance received from several people in preparations of this project. It would not have been possible to prepare this project, in this form without their valuable help, cooperation and guidance.

I would like to thank **NGI Management** for constant support and facilities provided in the carrying out of project in college premises.

I wish to record my sincere gratitude **Dr. B V Ravishankar, Principal**, Nagarjuna college of Engineering and Technology for his constant support and encouragement in preparation of this project and for providing library and laboratory facilities needed to prepare this project.

I thank my project coordinator **Dr. Sanjeevakumar M Hatture**, **HOD**, Department of ISE, Nagarjuna college of Engineering and Technology for your valuable suggestions and all the encouragement that leads towards completion of our project.

I would like to thank my project guide **Mrs. Nethra B R**, Assistant Professor, Department of ISE, Nagarjuna College of Engineering and Technology for your valuable guidance and all the encouragement that leads towards completion of our project.

Last but not least, we would like to thank our parents, friends, teaching and non-teaching staff of NCET.

Rajmouli K V (1NC21IS047)

INDEX

CHAPTER 1

INTRODUCTION

Gesture recognition helps computers to understand human body language. This helps to build a more potent link between humans and machines, rather than just the basic text user interfaces or graphical user interfaces (GUIs). In this project for gesture recognition, the human body's motions are read by computer camera. Thecomputer then makes use of this data as input to handle applications. The objective of this project is to develop an interface which will capture human hand gesture dynamically and will control the volume level.

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

Pycaw : Pycaw (Python Audio Control Library) is a Python library designed to provide an interface for controlling the audio settings on a Windows machine. It allows developers to interact programmatically with the system's audio endpoint devices, such as speakers or headphones, making it possible to manage the audio output directly from within a Python script. Pycaw leverages the Windows Core Audio API to access and manipulate various audio parameters, such as volume levels, mute status, and device selection. In the context of this project, Pycaw is used to map the detected hand gestures to volume control commands, enabling real-time, intuitive audio adjustments based on user input.

Mediapipe: is an open-source machine learning library of Google, which has some solutions for face recognition and gesture recognition, and provides encapsulation of python, js and other languages. MediaPipe Hands is a high-fidelity hand and finger tracking solution. It uses machine learning (ML) to infer 21 key 3D hand information from just one frame. We can use it to extract the coordinates of the key points of the hand.

.

CHAPTER 2

LITERATURE SURVEY

Hand gesture recognition has emerged as a vital area of research in human-computer interaction, offering an intuitive and contactless way of controlling devices. Among the various applications of hand gesture recognition, controlling the volume of devices is gaining popularity due to its convenience and ease of use. Modern hand gesture recognition techniques have evolved significantly, with models such as Convolutional Neural Networks (CNNs) and machine learning algorithms playing a pivotal role in enhancing both accuracy and efficiency.

Early approaches to gesture recognition, like template matching and rule-based methods, laid the groundwork for detecting and interpreting hand movements. However, these methods were often limited by their dependency on predefined templates and their inability to handle the variability in hand shapes and gestures. The advent of deep learning, particularly CNNs, revolutionized gesture recognition by allowing for the automatic extraction of features and the classification of complex hand gestures.

In this context, CNN-based models are particularly suited for real-time gesture recognition due to their ability to process and classify images quickly and accurately. Recent advancements have introduced optimized architectures like MobileNet and EfficientNet, which offer a balance between speed and accuracy, making them ideal for applications in volume control where real-time responsiveness is essential.

Gesture recognition for volume control leverages these advancements by detecting specific hand movements and translating them into volume adjustment commands. Research indicates that these models can achieve high accuracy rates in diverse environments, handling variations in lighting conditions, background noise, and hand orientations effectively. Comparative studies highlight the superior performance of CNN-based models in real-time gesture recognition tasks, underscoring their potential for practical applications in smart devices, multimedia systems, and automotive controls.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

A System Requirements a statement that identifies the functionality that is needed by a system in order to satisfy the customer’s requirement.

3.1. Software Requirements:

1. Python 3
2. GIT BASH

3.2. Hardware Requirements:

1. Operating System: Windows 7(minimum)
2. Hard Disk:500Mb (minimum)
3. RAM 4GB or above
4. Processor: i3 (minimum)

CHAPTER 4
SYSTEM ANALYSIS

4.3 Existing System

Existing systems for volume control typically rely on conventional methods such as physical buttons, remote controls, or voice commands. While these methods are functional, they often fall short in providing an intuitive, seamless, or user-friendly experience. Physical buttons, for instance, require direct interaction, which may not be ideal in situations where users are occupied or have limited mobility. Remote controls, though offering some level of convenience, can easily be misplaced, and their small buttons may not be easy to use for everyone. Voice commands, on the other hand, can be effective, but they depend heavily on accurate speech recognition, which can be influenced by accents, background noise, or the clarity of the user's voice.

Traditional gesture recognition systems, which have started to appear in some smart devices, attempt to address these issues by allowing users to control volume through hand movements. However, these systems often rely on older computer vision algorithms or less sophisticated deep learning models that are not well-suited for the complex task of real-time gesture recognition. Such systems frequently struggle with accuracy and responsiveness, particularly under varying lighting conditions or with different hand gestures, leading to a frustrating user experience. Furthermore, these traditional systems often require significant computational power to process gestures, which can be a major limitation for devices with limited resources. Consequently, they fail to fully leverage the latest advancements in deep learning and computer vision, resulting in performance that is inadequate for modern, real-time applications where speed and precision are critical. As a result, there is a growing need for more advanced, efficient, and user-friendly solutions in gesture-based volume control systems.

4.2 Proposed System

The proposed system leverages advanced deep learning techniques, focusing specifically on convolutional neural network (CNN)-based models that have been optimized for real-time performance. These models are designed to process video input from a camera, enabling the system to detect and interpret specific hand movements with a high degree of precision. The hand gestures, such as pinching or swiping, are then translated into corresponding volume adjustment commands, providing a seamless and natural way for users to interact with their devices.

One of the key strengths of this system lies in its use of modern, lightweight architectures. Unlike traditional models that may require significant computational resources, the proposed system is designed to be efficient, allowing it to operate effectively even on devices with limited processing power. This efficiency does not come at the cost of accuracy or speed; instead, the system maintains a high level of responsiveness, ensuring that volume adjustments are made in real-time as the user performs the gesture.

Moreover, the system is built to perform reliably across a range of environmental conditions. Whether in low-light settings, varying backgrounds, or with different hand sizes and shapes, the system is capable of consistently recognizing gestures and executing the appropriate volume controls. This robustness is crucial for real-world applications, where users might use the system in a variety of scenarios.

The result is a highly reliable and intuitive volume control solution that significantly enhances user interaction with smart devices. By moving beyond traditional input methods and embracing gesture-based control, the system offers a more immersive and user-friendly experience, paving the way for more advanced and accessible human-computer interactions in the future.

CHAPTER 5
SYSTEM DESIGN

The design of the system involves the integration of computer vision techniques with audio control functionalities to create a user-friendly interface for adjusting the volume using hand gestures. The system design can be broken down into the following key components:

1. Input Capture:

➢ The system utilizes a webcam or any camera connected to the computer to capture real-time video feed. The camera is essential for detecting and tracking hand movements and gestures.
➢ The captured video feed is processed frame by frame to ensure accurate detection of the hand's position and movements.

2. Hand Detection and Tracking:

➢ The MediaPipe Hands library is employed for detecting and tracking hand landmarks. This involves identifying specific points on the hand, such as the tips of the thumb and index finger.
➢ The system processes each frame to detect the presence of a hand and track the landmarks on the detected hand. The position of the thumb and index finger is of particular interest for this application.

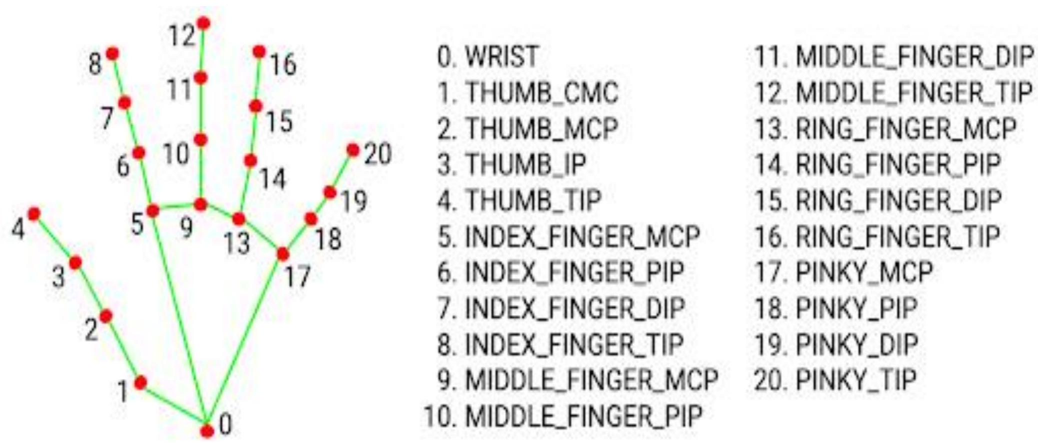


Fig 5.1 Hand Landmarks Mapping Figure

3. Gesture Recognition:

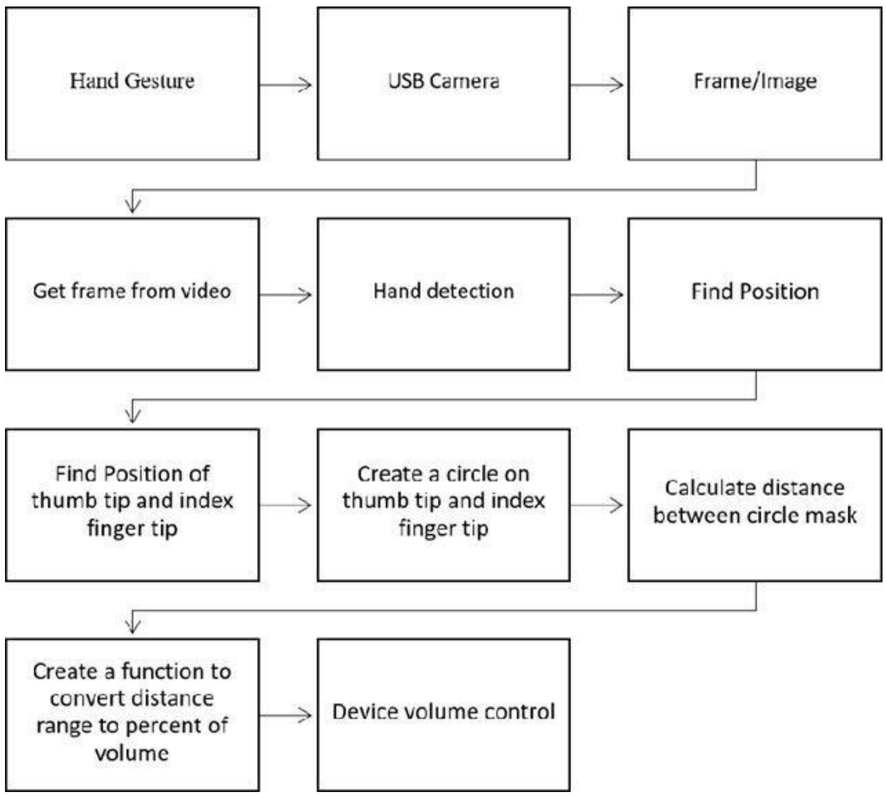
- The primary gesture used for volume control is the pinching motion between the thumb and index finger. The system calculates the distance between these two landmarks to recognize the gesture.
- The distance between the thumb and index finger is directly correlated with the volume level. A smaller distance indicates a lower volume, while a larger distance indicates a higher volume.

4. Volume Control Interface:

- The system interfaces with the computer's audio hardware using the PyCaw library, which provides access to the audio endpoint volume control.
- The calculated distance between the thumb and index finger is mapped to a corresponding volume level using linear interpolation. This mapped volume level is then applied to the system's audio output.
- The system continuously adjusts the volume in real-time based on the detected hand gestures, providing an intuitive and responsive user experience.

Flow Chart Visualization:

Fig 5.2 Flow diagram of System Design



CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Source Code

```
from ctypes import cast, POINTER

from comtypes import CLSCTX_ALL, GUID

from pycaw.pycaw import AudioUtilities, IAudioEndpointVolume

import cv2

import numpy as np

import mediapipe as mp

import math

IID_IAudioEndpointVolume = GUID("{5CDF2C82-841E-4546-9722-0CF74078229A}")

devices = AudioUtilities.GetSpeakers()

interface = devices.Activate(IID_IAudioEndpointVolume, CLSCTX_ALL, None)

volume = cast(interface, POINTER(IAudioEndpointVolume))

volumeinformation = volume.GetVolumeRange()

minVolume = volumeinformation[0]

maxVolume = volumeinformation[1]

cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)

mpHands = mp.solutions.hands

hands = mpHands.Hands()

mpDraw = mp.solutions.drawing_utils

while True:

    status, img = cap.read()
```

```
img = cv2.flip(img, 1)
imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
results = hands.process(imgRGB)
multiLandMarks = results.multi_hand_landmarks

if multiLandMarks:
    indexPoint = ()
    thumbPoint = ()
    for handLms in multiLandMarks:
        for idx, lm in enumerate(handLms.landmark):
            h, w, c = img.shape
            cx, cy = int(lm.x * w), int(lm.y * h)
            if idx == 4:
                thumbPoint = (cx, cy)
            if idx == 8:
                indexPoint = (cx, cy)
    # Calculate distance
    length_squared = ((indexPoint[0] - thumbPoint[0]) ** 2) + ((indexPoint[1] - thumbPoint[1]) **
    length = math.sqrt(length_squared) if length_squared >= 0 else 0
    vol = np.interp(length, [20, 200], [minVolume, maxVolume])
    print(vol)
    volume.SetMasterVolumeLevel(vol, None)
cv2.imshow("HandGesture", img)
cv2.waitKey(1)
```

6.2 Results

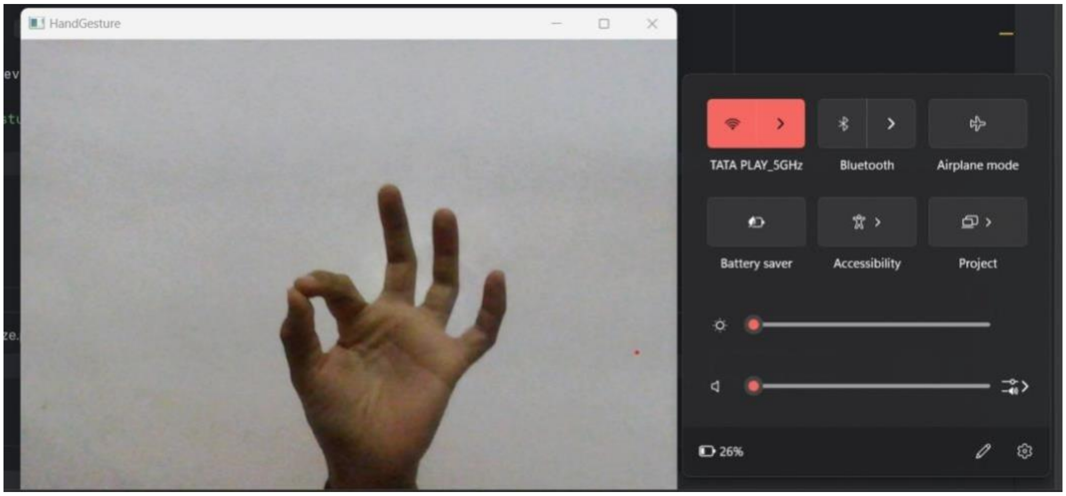


Fig 6.2.1 Gesture for Minimum Volume

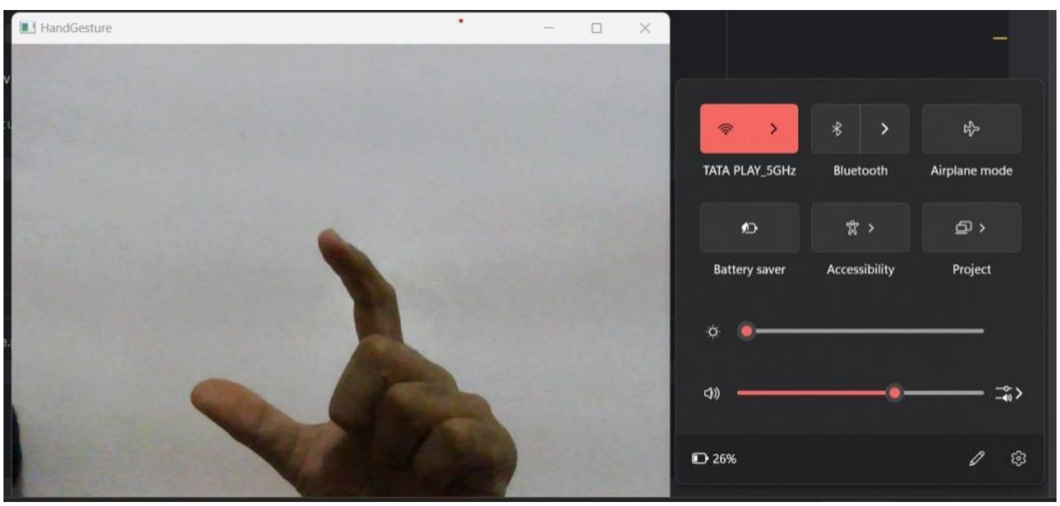


Fig 6.2.2 Gesture for Adjusting Volume

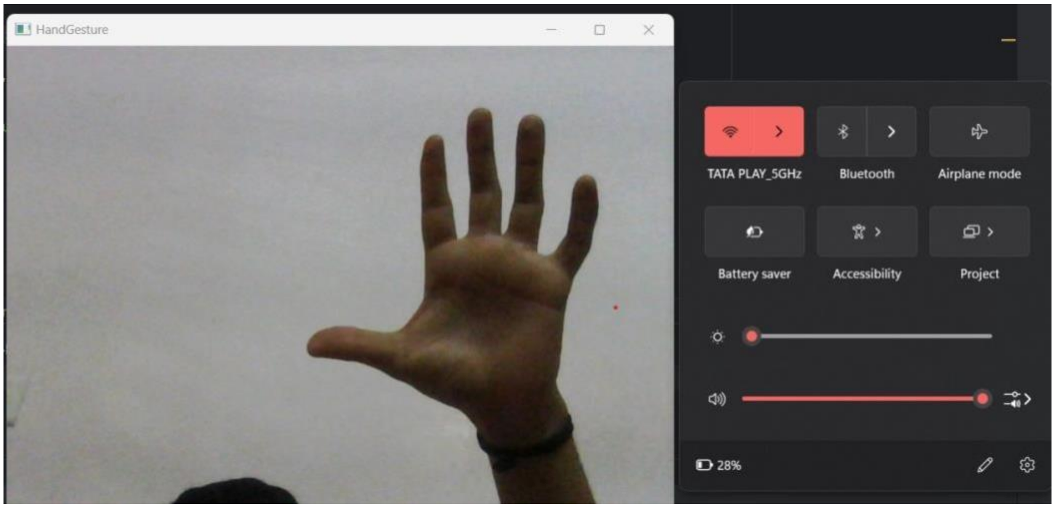


Fig 6.2.3 Gesture for Maximum Volume

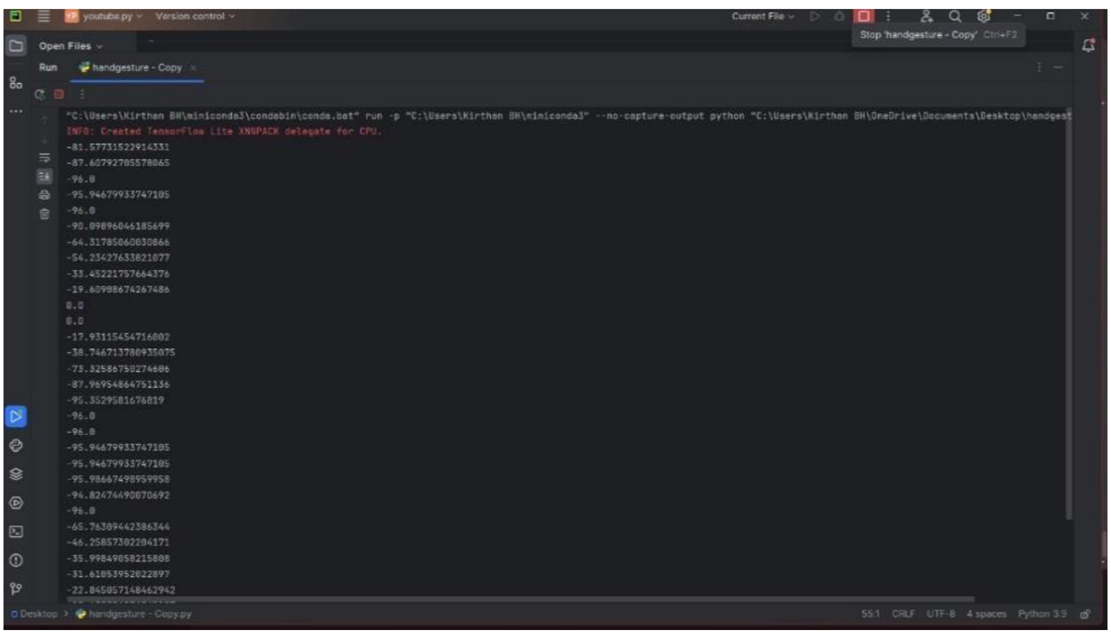


Fig 6.2.4 Readings in db

CONCLUSION

The project on "Volume Control using Hand Gestures" has successfully demonstrated the practical application of computer vision techniques in creating an intuitive and contactless user interface for audio control. By utilizing MediaPipe for real-time hand tracking and PyCaw for audio manipulation, the system has effectively shown how simple gestures, such as the pinching motion between the thumb and index finger, can be translated into precise volume adjustments.

This approach not only enhances user convenience by eliminating the need for physical interaction with volume controls but also opens up new possibilities for gesture-based interfaces in various applications. The real-time responsiveness and accuracy of the system highlight the potential for integrating such technology into everyday devices, making them more accessible and user-friendly.

The project’s success in achieving seamless volume control through hand gestures underscores the potential of combining computer vision with audio management systems. It paves the way for further exploration into gesture-based controls, which could significantly influence the development of future human-computer interaction systems.

REFERENCES

- https://www.hackster.io/as4527/volume-control-using-hand-gesture-using-python-and-opencv-7aab9f
- https://pyimagesearch.com/2023/05/15/hand-gesture-recognition-with-yolov8-on-oak-d-in-near-real-time/
- https://www.researchgate.net/publication/329039460_Research_on_Gesture_Recognition_Method_Based_on_Computer_Vision
- https://www.geeksforgeeks.org/introduction-to-opencv/