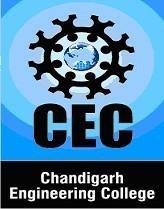
**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**(BTECH CSE)**



**CHANDIGARH ENGINEERING COLLEGE**

**LANDRAN, MOHALI, 140307**

**(PROJECT FILE)**

**Name: Tahmoor Ali Yezdani**

**Team Member: Vansh Pratap Singh Guleria**

**Roll no: 2003267, 2003285**

**Address: Shantikunj Hostel CGC Landran**

**E-mail:** [**tahmooryezdani@gmail.com**](mailto:tahmooryezdani@gmail.com) **,vanshguleria12@gmail.com**

**Mobile: +91 9796123254 , +91 8427440334**

**Branch: CSE**

**Batch: 2024**

**Topic: Volume Control Using Hand Gestures.**

**ABSTRACT**

A hand gesture recognition system provides a natural, innovative and modern

way of non-verbal communication. It has a wide area of application in human

computer interaction and sign language. The intention of this implementation

is to discuss a novel approach of hand gesture recognition based on detection

of some shape-based features. The setup consists of a single camera to capture

the gesture formed by the user and take this as input to the system, A primary

goal of gesture recognition is to create a system which can identify specific

human gestures and use them to convey information for device control and by

implementing real time gestures recognition, a user can control a computer by

doing a specific gesture in front of a video camera linked to a computer. In this

project, we will develop a hand gesture volume control system with the help of

OpenCV module. Here the system can be operated using hand gestures without

using keyboard and mouse.

**Keywords – hand gesture recognition, shape-based features, volume control**

**system, OpenCV.**

**INTRODUCTION**

Hand gestures are spontaneous and powerful communication mode for Human

Computer Interaction (HCI). Traditional input devices are available for interaction

with computer, such as keyboard, mouse, joystick as well as touch screen; however

they do not provide natural interface. The proposed system will consist of desktop

or laptop interface, the hand gesture may be used by the users may need to wear

any data glove, or may use the web camera for capturing the hand image. The initial

step towards any hand gesture recognition is hand tracking and segmentation. Sensor

devices are used in Data-Glove based methods for digitizing hand and finger motions

into multi parametric data. The other sensors will collect hand configuration and hand movements. In contrast; the Vision Based methods require only a camera, thus realizing

a natural interaction between humans and computers without the use of any extra

devices. These systems tend to complement biological vision by describing artificial

vision systems that are implemented in software and/or hardware. The challenging

problems of these systems need to be background invariant, lighting insensitive, person

and camera independent to achieve real time performance. Various algorithms used in

hand posture and gesture recognition and discuss the advantages and disadvantages of

each. Various algorithmic techniques for recognizing hand postures and gestures are discussed. The Segmentation is the process of finding a connected region within the

image with a specific property such as colour or intensity, or a relationship between pixels, that is, a pattern and the algorithms should be adaptable.

**EXISTING SYSTEM**

**ANN for Gesture Recognition using Accelerometer Data:** The authorsintroduced

an Artificial Neuralnetwork application used for theclassification and gesture

recognition.The gesture recognition is donethrough the Wifi remote, this remote

will rotate in X, Y, Z directions. To reduce the computational cost and memory consumption the gesture recognition is processed in two levels. In first level User Authentication is

done for gesture recognition. Accelerometer- Based gesture recognition method is used. In second level without any kind of signal processing for gesture recognition Fuzzy automata algorithm has been proposed. After recognizing the data of the gestures, the data was

normalized and filtered by k-means and Fast Fourier transform algorithm. Using this Dynamic Bayesian Network, the recognition accuracy has increased up to 95%.

**Combining multiple depth-based descriptors for hand gesture recognition:** Based on the depthinformation of the image taken by thedepth cameras the authors haveintroduced a scheme known as novelhand gesture recognition scheme. Toproperly recognize complex gesturesby using 3-D information they used aset of 3-Dimensional features. Theproposed hand gesture recognitionsystem consists of three main steps**.** The first step based on colour anddepth information the hand samplesare segmented from the background.Wrist samples, palm and the fingersare subparts of the segmented handsamples. The proposed hand gesturerecognition consists of four types of features. The second step is to extract

these features for the segmentation. The first two set of features are based on the distance from elevation of finger tips to palm centre. The third feature set concentrate on computed

curvature features of hand contour. The fourth set is on constructed geometry of the palm region. The SVM classifier is used with constructed feature vectors to identify the hand

gesture which is performed in front of the camera. The accuracy of 95% is achieved with the combination of feature set and the SVM classifier.

**A Real-Time Hand Gesture Recognition System for Daily Information Retrieval from Internet:** In this paper, a system isproposed in such a way that with thehand movements the dailyinformation is retrieved from theinternet. Principal componentanalysis is used for the identifying thehand. Using YcbCr colour spaces skincolour detection and CAMSHIFT

algorithm is used to detect and track the hand gestures. The position and the region of the hand is detected from the skin detection. It keeps on detecting the skin region until the

condition of tracking trigger is enough? The CAMSHIFT algorithm is used when the tracking trigger condition is enough. Segmentation and normalization are done through the PCA. The experimental proves that the 93.1% of accuracy rate is achieved for hand gesture recognition. For processing a single frame, the total time taken was in between 0.1sec to

0.3 sec.

**PROPOSED WORK & ANALYSIS**

Hand Gesture is an active area of research in the vision community, mainly for the

purposes of sign language recognition and human computer interaction. One of the

original tracking systems to focus on articulated hand motion was present. In their system,

a 27 degree-of freedom hand could be tracked at 10Hz by extracting point and line

features from grayscale images. However, it has difficulty tracking in the presence of occlusions and complicated backgrounds, and it requires a manual initialization step

before tracking can begin. From an interaction perspective, most of the hand tracking

work to date has focused on 2D interfaces. In [Zhang01], a finger was tracked across a planar region using low-cost web cameras in order to manipulate a traditional graphical

interface without a mouse or keyboard. Fingertip detection was accomplished by fitting a conic to rounded features, and local tracking of the tip was performed using Kalman filtering. Similarly, in infrared cameras were used to segment skin regions from background pixels in order to track two hands for interaction on a 2D tabletop display. Their method then used a template matching approach in order to recognize a small set of gestures that could be interpreted as interface commands. However, no precise fingertip position

information was obtained using their technique.

**SYSTEM ARCHITECTURE**



This project is developed using python technology, the code is arranged and designed in python using OpenCV and NumPy modules OpenCV supports a wide variety of programming languages like C++, Python, Java etc and is available on different platforms including

Windows, Linux, OS X, Android, iOS etc. Also, interfaces based on CUDA and OpenCL are

also under active development for high-speed GPU operations. OpenCV-Python is the

Python API of OpenCV. It combines the best qualities of OpenCV C++ API and Python language.

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

 a powerful N-dimensional array object.

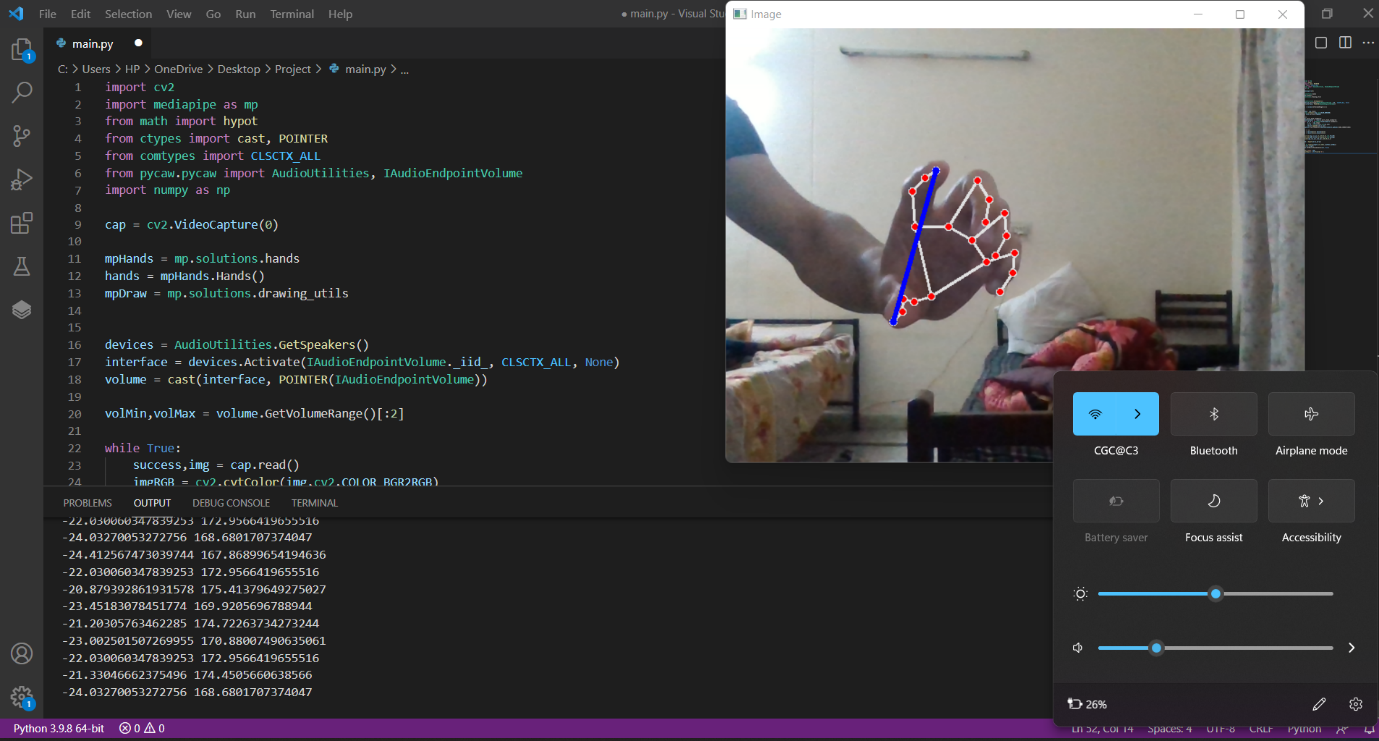
 sophisticated (broadcasting) functions.

 tools for integrating C/C++ and Fortran code.

 useful linear algebra, Fourier transform, and random number capabilities.

Here we have to perform the Hand Gestures for quality output, webcam should be enabled while running the program, static gesture recognition is used here, static gesture recognition is the recognition of hand shape, and gives the result. In this project the volume will be controlled based on the posture of thumb and index finger. The system will capture the object and detects it after that hand gesture recognition will be performed.

**RESULT:**



**CONCLUSION:**

The project presented a program that allowed user to perform hand gestures for easy software control. A vision-based hand Gesture system that does not require any special markers or gloves and can operate in real-time on a commodity PC with low-cost cameras.

Specifically, the system can track the tip positions of the counters and index finger for each hand. The motivation for this hand Gesture was a desktop-based volume control system

in which a user can control volume and cursor navigation in real time using natural thumb and index finger motions. For the sake of reliability, we, furthermore, propose a simple probabilistic model to effectively prevent the developed system from responding to invalid gestures.

**PROGRAM:**

import cv2

import mediapipe as mp

from math import hypot

from ctypes import cast, POINTER

from comtypes import CLSCTX\_ALL

from pycaw.pycaw import AudioUtilities, IAudioEndpointVolume

import numpy as np

cap = cv2.VideoCapture(0)

mpHands = mp.solutions.hands

hands = mpHands.Hands()

mpDraw = mp.solutions.drawing\_utils

devices = AudioUtilities.GetSpeakers()

interface = devices.Activate(IAudioEndpointVolume.\_iid\_, CLSCTX\_ALL, None)

volume = cast(interface, POINTER(IAudioEndpointVolume))

volMin,volMax = volume.GetVolumeRange()[:2]

while True:

success,img = cap.read()

imgRGB = cv2.cvtColor(img,cv2.COLOR\_BGR2RGB)

results = hands.process(imgRGB)

lmList = []

if results.multi\_hand\_landmarks:

for handlandmark in results.multi\_hand\_landmarks:

for id,lm in enumerate(handlandmark.landmark):

h,w,\_ = img.shape

cx,cy = int(lm.x\*w),int(lm.y\*h)

lmList.append([id,cx,cy])

mpDraw.draw\_landmarks(img,handlandmark,mpHands.HAND\_CONNECTIONS)

if lmList != []:

x1,y1 = lmList[4][1],lmList[4][2]

x2,y2 = lmList[8][1],lmList[8][2]

cv2.circle(img,(x1,y1),4,(255,0,0),cv2.FILLED)

cv2.circle(img,(x2,y2),4,(255,0,0),cv2.FILLED)

cv2.line(img,(x1,y1),(x2,y2),(255,0,0),3)

length = hypot(x2-x1,y2-y1)

vol = np.interp(length,[15,220],[volMin,volMax])

print(vol,length)

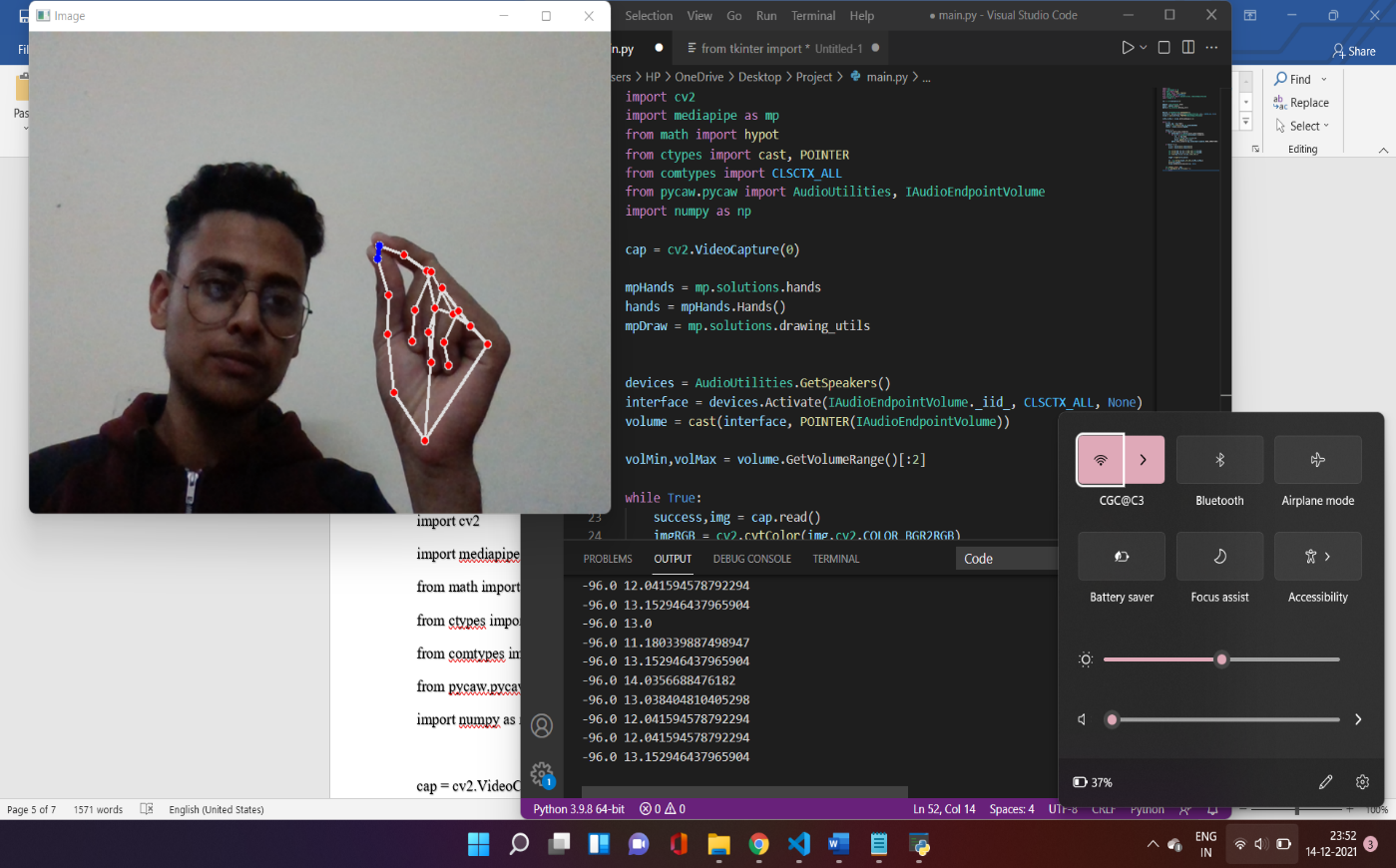
volume.SetMasterVolumeLevel(vol, None)

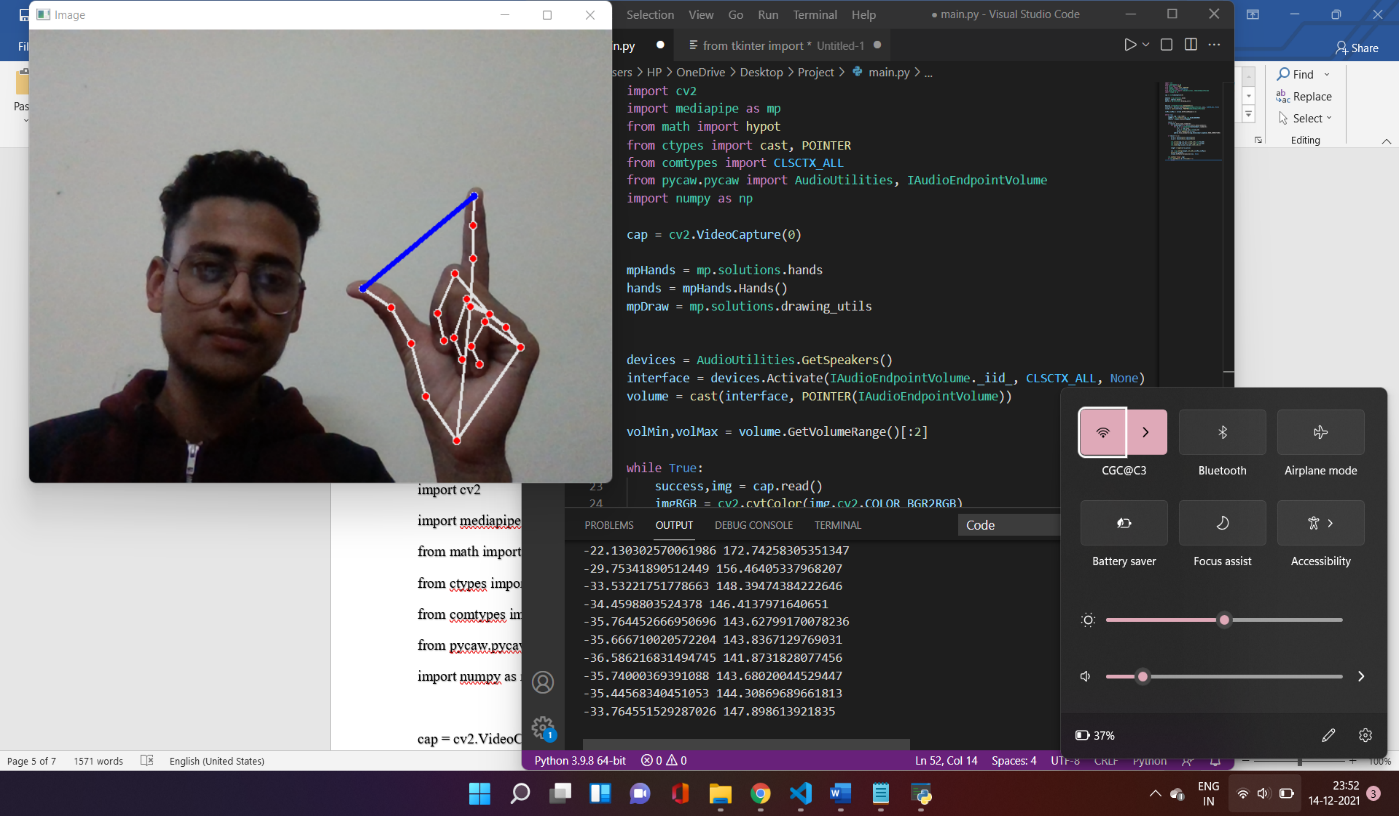
cv2.imshow('Image',img)

if cv2.waitKey(1) & 0xff==ord('E'):

break

**OUTPUT:**

****

****