Project Report

on

HAND GESTURES AND SIGN LANGUAGE RECOGNITION

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

Computer Science Engineering (AIML)

Submitted by:

Kriti Upadhyay(22BAI71229-22AML104-A)

Reeya Ottalwar(22BIA50028-22AML104-A)

Harshdev Singh(22BAI71311-22AML104-A)

Srijita Chatterjee(22BAI71230-22AML104-A)

Lithiga Jayaprakash(22BIA50004-22AML104-A)

Under the Guidance of

Dr. Ashish Sharma

INSTITUTE – UIE
ACADEMIC UNIT-I

CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

<u>Index</u>

1. Problem statement

There are many applications where hand gesture can be used for interactionwith systems like, videogames, controlling UAV's, medical equipment's, etc. These hand gestures can also be used by handicapped people to interactwith the systems. Communication is an important thing to share our ideas but if it comes to deaf and mute people, two-way communication is sometimes not possible. **One of the uses of hand gesture recognition is for sign language. It can be very beneficial for deaf and mute people for communicate with someone who is not familiar with sign language.So, this project is a step from us to help them to communicate in an easier way. It will recognize the hand gesture and convert it into text (tell its meaning) and convert it into speech.** Classical interactions tools like keyboard, mouse, touchscreen, etc. may limit the way we use thesystem. All these systems require physical contact, in order to interact withsystem. Gestures can interpret same functionality without physically interacting with the interfacing devices. The problem lies in understanding these gestures, as for different people, the same gesture may look differentfor performing the same task. This problem may be overthrown using Deep Learning approaches. Convolution neural networks (CNN/'s) are proving to be ultimate tool to process such recognition systems. The only problem with the deep learning approaches is that they may work poorly in real world recognition. High computing power is required in order to process gestures.

2. Methodology

The recognition of hand gestures and hand sign language involves the use of various machine learning techniques and algorithms, including supervised and unsupervised learning, and computer vision. One common approach is to use convolutional neural networks to analyze visual data and classify hand gestures or sign language symbols. Another approach is to use hidden Markov Models or other probabilistic models to model the temporal dynamics of hand movements and sign language prediction.
Data collection and annotation is a crucial step in the development of machine learning models for hand gestures and sign language recognition. There are several publicity available datasets that can be used for this purpose, including the American Sign Language alphabet dataset, the continuous sign language recognition dataset, and the sign language MNISR dataset. These datasets consist of videos or images of hand gestures or sign language symbols, along with annotations or transcripts of the corresponding language.

Recent advances in ML and computer vision have enabled significant progress in the recognition of hand gestures and sign language. For example, deep-learning based approaches have achieved high accuracy on various datasets, with some models achieving over 90% accuracy on the ASL alphabet dataset. However, there are still many challenges and limitations to be addressed in this area.
One challenge is the variability of hand gestures and sign language, which can vary across individuals, languages, and contexts. This variability can make it difficult to generalize machine learning models across different languages or users. Another challenge is the limited availability of annotated data, which can make it difficult to train and evaluate machine learning models. Finally, there is a need for more robust and efficient algorithms that can handle real-time processing and recognition of hand gestures and sign language.

The recognition of hand gestures and sign language is a complex and important task in natural language processing and human-computer interaction. Machine learning techniques and algorithms, particularly deep learning and computer vision, have made significant progress in this area, but there are still many challenges and limitations to be addressed. Further research and development in this area could have significant implications for assistive technology, human-computer interaction, and machine learning applications.

Libraries Used:

- **TensorFlow:** TensorFlow is an open-source library specially designed for the artificial intelligence and machine learning sector. It is used in training and interference of deep neural networks. It can build and train models with usual performance. It is the board library of pre-trained models which is easy to use in projects. The main advantage of this library is it provides the output with accuracy using a limited size dataset.

- **Time module:** Python time module allows to work with time in Python. It allows functionality like getting the current time, pausing the Program from executing, etc. So before starting with this module we need to import it.
  The time module comes with Python's standard utility module, so there is no need to install it externally. We can simply import it using the import statement.
- **Pyttsx3:** Pyttsx3 is a text-to-speech Library that us used in python.
  Unlike other Libraries, it is very compatible to use with python 2 &3 and can work offline, it is a very easy to use tool which converts the entered text into speech. In our project we use pyttsx3 library for hand tracking, in which the interface with recognize the gestures made by sign and then interpret the text to convert it into speech with the help of this library**.**
- **CV Zone:** This is a computer vision package that makes it easy to run Image processing and AI functions. At the core it uses OpenCV and Mediapipe libraries.
- **Teachable Machine:** Teachable Machine is a web tool that makes it fast and easy to create machine learning models for your projects, no coding required. Train a computer to recognize your images, sounds, & poses.
  In our report are using this tool for hand tracking so as to recognize our hand gestures for sign language for deaf people.
- **CV 2:** Cv2 is a Python library that is part of the open-source computer vision library, OpenCV. It provides functions for performing operations on arrays and matrices, image and video processing, and machine learning. Cv2 is used to read, write, and process images and videos in Python. Some of the common tasks that you can perform with cv2 include reading and writing images, drawing on images, resizing and cropping images, and applying image filters. It is a powerful tool for working with images and videos in Python, and is widely used in the field of computer vision.

3. Results

- We created this project for deaf and mute people by merging python and OpenCV. They will be able to converse the alphabets without any difficulty with this program.
- We also added some gestures used in a day-to-day basis.
- The result of sign detected was shown on the screen.
- Hand gesture recognition will have a very broad reach in the future as the number of users grows every day.
- In order to find an alternative to using a person to transmit their sentiments, they are willing to utilize software like this. As a result, this initiative will be of more assistance to them.
- It also converts the result into speech to make the communication easier.

4. Conclusion

Hand Gesture Recognition will provide two-way communication which helps to interact between the impaired people to normal people without any difficulties by recognizing the alphabets or number the person wants to say.Hence the implementation system can translate sign language and predict characters and numbers. When we’ll run the code, camera access will be given and it will track our hand. If you keep your hand in a certain way which will be recognized by the camera, the text version of that hand gesture (sign language) will be displayed on the screen. And a text-to- speech converter will convert that text into audio to make the communication easier. This project will be beneficial for deaf and mute people to communicate easily with others.

5. Source Code

Training the model

```
#Libraries
import cv2
from cvzone.HandTrackingModule import HandDetector #For detecting
import numpy as np
import math
import time
import pyttsx3 as tts #Text to speech

#Main
cap = cv2.VideoCapture(0)
detector = HandDetector(maxHands=2)
offset = 20
imgSize = 400
count = 0

#Text to Speech
eng = tts.init()
RateOS = eng.getProperty('rate')
eng.setProperty('rate', RateOS-25)
eng.say("Hello")
eng.runAndWait()
time.sleep(3)
print("\...You can show your hand sign and press \'s\' to save that image...")

#Data Storing File
folder = "venv\Data\A"

while True:
    success, img = cap.read()
    hands, img = detector.findHands(img)
    #Cropping the image
    if hands:
        hand = hands[0]
        x, y, w, h = hand['bbox']
        #x1, y1, w1, h1 = hand['vbox']
        imgCrop = img[y-offset:y + h+offset, x-offset:x + w+offset]
        imgWhite = np.ones((imgSize, imgSize, 3), np.uint8) * 255  # Box
        #Centralizing the image
        asp_ratio = h/w
        if asp_ratio > 1:
            i = imgSize/ h
            newWidth = math.ceil(i*w)
            ResizeImg = cv2.resize(imgCrop, (newWidth, imgSize))
            imgReShape = ResizeImg.shape
            WidthGap = math.ceil((imgSize-newWidth)/2)
            imgWhite[:, WidthGap:(newWidth+WidthGap)] = ResizeImg
        else:
            i = imgSize / w
            newHeight = math.ceil(i * h)
            ResizeImg = cv2.resize(imgCrop, (imgSize, newHeight))
            imgReShape = ResizeImg.shape
            HeightGap = math.ceil((imgSize - newHeight) / 2)
            imgWhite[HeightGap:newHeight + HeightGap, :] = ResizeImg

        cv2.imshow("ImageWhites", imgWhite)
        cv2.imshow("ImageCrop", imgCrop)

    cv2.imshow("Image", img)
    key = cv2.waitKey(1)

    #Saving the images
    if key == ord("s"):
        count += 1
        cv2.VideoWriter(f'{folder}/Img_{time.time()}.mp4', imgWhite)
        print("Counter: ", count)
```

Language Detection

```
#Libraries
import cv2
from cvzone.HandTrackingModule import HandDetector #For detecting
from cvzone.ClassificationModule import Classifier
import numpy as np
import math
#import tensorflow
import time
import pyttsx3 as tts

cap = cv2.VideoCapture(0)
detector = HandDetector(maxHands=1)

#Importing trained model
classifier = Classifier("Model/keras_model.h5", "Model/labels.txt")

offset = 20
imgSize = 400
count = 0
```

```
#Text to Speech
eng = tts.init()
speed = eng.getProperty('rate')
eng.setProperty('rate', 160)
eng.say("Hello! Nice to meet you. You can now start by showing your hand")
print("Show your hands")
eng.runAndWait()
time.sleep(3)
Labels = ["A", "B", "C", "D", "E", "DISLIKE", "F", "G", "H", "I",  "K", "L",
"M", "N", "O", "OKAY", "P", "Q", "R", "S", "ROCK", "T", "U", "V", "W", "X",
"Y"]

while True:
    time.sleep(1)
    success, img = cap.read()
    OutputImg = img.copy()
    hands, img = detector.findHands(img)

    #Cropping the image
    if hands:
        hand = hands[0]
        x, y, w, h = hand['bbox']
        imgCrop = img[y-offset:y + h+offset, x-offset:x + w+offset]
        imgWhite = np.ones((imgSize, imgSize, 3), np.uint8) * 255

        #Centralizing the image
        asp_ratio = h/w
        if asp_ratio > 1:
            i = imgSize/h
            newWidth = math.ceil(i*w)
            ResizeImg = cv2.resize(imgCrop, (newWidth, imgSize))
            imgReShape = ResizeImg.shape
            WidthGap = math.ceil((imgSize-newWidth)/2)
            imgWhite[:, WidthGap:(newWidth+WidthGap)] = ResizeImg
            predict, index = classifier.getPrediction(imgWhite, draw=False)
            print(predict, index)
        else:
            i = imgSize / w
            newHeight = math.ceil(i * h)
            ResizeImg = cv2.resize(imgCrop, (imgSize, newHeight))
            imgReShape = ResizeImg.shape
            HeightGap = math.ceil((imgSize - newHeight) / 2)
            imgWhite[HeightGap:newHeight + HeightGap, :] = ResizeImg
            predict, index = classifier.getPrediction(imgWhite, draw=False)
            print(predict, index)
        cv2.rectangle(OutputImg, (x-offset, y-offset-50), (x-offset+200, y-offset), (255, 155,
255), cv2.FILLED)
        cv2.putText(OutputImg, Labels[index], (x, y-24), cv2.FONT_HERSHEY_DUPLEX, 3,
(255, 1, 255), 2)
        cv2.rectangle(OutputImg, (x-offset, y-offset), (x+w+offset, y+w+offset), (255, 155,
255), 4)
        cv2.imshow("ImageWhites", imgWhite)
        cv2.imshow("ImageCrop", imgCrop)

    #Output-prediction
    cv2.imshow("Image", OutputImg)
    eng.say(Labels[index])
    eng.runAndWait()
    cv2.waitKey(1)
```

6. References

[1] K. -H. Kim, D. -U. Jung, S. -H. Lee and J. -S. Choi, "A hand tracking framework using the 3D active tracking volume," The 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision, 2013, pp. 159-163, doi:10.1109/FCV.2013.6485480.

[2] B. Sugandi, H. Toar and A. Alfianto, "Hand Tracking-based Motion Control for Robot Arm Using Stereo Camera," 2018 International Conference on Applied Engineering (ICAE), 2018, pp. 1-5, doi:10.1109/INCAE.2018.8579360.

[3] C. Liu, Z. Li, C. Zhang, Y. Yan and R. Zhang, "An Improved Hand Tracking Algorithm for Chinese Sign Language Recognition," 2019 IEEE3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2019, pp. 229-232, doi: 10.1109/ITNEC.2019.8729147.

[4] M. Alsoos and A. Joukhadar, "Posture independent model for hand detection and tracking," 2013 6th International Conference on Human System Interactions (HSI), 2013, pp. 175-179, doi: 10.1109/HSI.2013.6577819.

[5] C. -H. Chen, Y. H. Hu and R. G. Radwin, "A motion tracking system for hand activity assessment," 2014 IEEE China Summit & International Conference on Signal and Information Processing (ChinaSIP), 2014, pp. 320-324, doi: 10.1109/ChinaSIP.2014.6889256.

[6] T. R. D. Scott et al., "Quantitative evaluation of two methods of control bilateral stimulated hand grasps in persons with tetraplegia," in

EEE Transactions on Rehabilitation Engineering, vol. 8, no. 2, pp. 259-267, June 2000, doi: 10.1109/86.847827.

[7] J. -M. Guo and H. -S. Nguyen, "Hybrid hand tracking system," 2011 18th IEEE International Conference on Image Processing, 2011, pp. 549- 552, doi: 10.1109/ICIP.2011.6116404.

[8] W. Wei, L. Zhijun, C. Zhiping, M. Xiaomeng, Z. Rui and Y. Ruo, "The Application of Improved Camshift Algorithm in Hand Tracking," 2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), 2017, pp. 87-90, doi: 10.1109/ICIICII.2017.15.

[9] Y. Zheng and P. Zheng, "Hand tracking based on adaptive kernel bandwidth mean shift," 2016 IEEE Information Technology, Networking,Electronic and Automation Control Conference, 2016, pp. 548-552, doi: 10.1109/ITNEC.2016.7560420.

[10] P. S. Lengare and M. E. Rane, "Human hand tracking using MATLAB to control Arduino based robotic arm," 2015 InternationalConference on Pervasive Computing (ICPC), 2015, pp. 1-4, doi: 10.1109/PERVASIVE.2015.7087039.

[11] J. Xu, Y. Wu and A. Katsaggelos, "Part-based initialization for hand tracking," 2010 IEEE International Conference on Image Processing, 2010, pp. 3257-3260, doi: 10.1109/ICIP.2010.5651179.

[12] Qi Sumin and Huang Xianwu, "Hand tracking and gesture recognition by anisotropic kernel mean shift," 2008 International Conference on Neural Networks and Signal Processing, 2008, pp. 581- 585, doi: 10.1109/ICNNSP.2008.4590417.

[13] Y. Sun, X. Liang, H. Fan, M. Imran and H. Heidari, "Visual Hand Tracking on Depth Image using 2-D Matched Filter," 2019 UK/ China Emerging Technologies (UCET), 2019, pp. 1-4, doi: 10.1109/UCET.2019.8881866.

[14] Y. Yun, I. Y. -H. Gu, J. Provost and K. Åkesson, "multi-view hand tracking using epipolar geometry-based consistent labeling for an industrial application," 2013 Seventh International Conference on Distributed Smart Cameras (ICDSC), 2013, pp. 1-6, doi: 10.1109/ICDSC.2013.6778217.

[15] S. Mu and A. Sourin, "Virtual Assembling Using Hand Tracking with Leap Motion Controller," 2021 International Conference on Cyberworlds (CW), 2021, pp. 121-124, doi: 10.1109/CW

[16] J. Shukla and A. Dwivedi, "A Method for Hand Gesture Recognition," 2014 Fourth International Conference on Communication Systems and Network Technologies, 2014, pp. 919-923, doi:10.1109/CSNT.2014.189.52790.2021.00026.

[17] Z. Meng, J. -S. Pan, K. -K. Tseng and W. Zheng, "Dominant Points Based Hand Finger Counting for Recognition under Skin Color Extraction in Hand Gesture Control System," 2012 Sixth International Conference on Genetic and Evolutionary Computing, 2012, pp. 364-367, doi: 10.1109/ICGEC.2012.85.

[18] S. K. Shareef, I. V. S. L. Haritha, Y. L. Prasanna and G. K. Kumar, "Deep Learning Based Hand Gesture Translation System," 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI), 2021, pp. 1531-1534, doi: 10.1109/ICOEI51242.2021.9452947.

[19] A. S. Ghotkar, R. Khatal, S. Khupase, S. Asati and M. Hadap, "Handgesture recognition for Indian Sign Language," 2012 International Conference on Computer Communication and Informatics, 2012, pp. 1-4, doi: 10.1109/ICCCI.2012.6158807.