HAND GESTURE BASED ON MEDIA PLAYER CONTROL SYSTEM

MAJOR PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF COMPUTER APPLICATIONS, BHARATHIAR UNIVERSITY IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

MASTER OF COMPUTER APPLICATIONS

Submitted by

M.NAGARAJAN (22CSEA59)

Under the guidance of

Dr. S.GAVASKAR, MCA.,Ph.D.,

DEPARTMENT OF COMPUTER APPLICATIONS

BHARATHIAR UNIVERSITY

COIMBATORE – 641 046

APRIL – 2024

DECLARATION

I hereby declare that this project titled **"HAND GESTURE BASED ON MEDIA PLAYER CONTROL SYSTEM"** submitted to the Department of Computer Applications, Bharathiar University, Coimbatore is a record of original project work done by **M.NAGARAJAN (22CSEA59)** under the supervision and guidance of **Dr. S. GAVASKAR, MCA., Ph.D., Assistant Professor** Department of Computer Applications, Bharathiar University, Coimbatore and that this project work has not previously formed the basis of the award of the Degree / Diploma / Associate Ship / Fellowship or similar title to any candidate of any university.

Place: Coimbatore Signature of Candidate

Date: M.NAGARAJAN

CERTIFICATE

This is to certify that the project work title **"HAND GESTURE BASED ON MEDIA PLAYER CONTROL SYSTEM"** submitted to Department of Computer Applications, Bharathiar University in partial fulfilment of the requirement for the award of Degree in **Master of Computer Applications,** is a record of the original work done by **M.NAGARAJAN (22CSEA59)** under my supervision and guidance and this project work has not formed the basis of the award of any Degree / Diploma / Associate Ship / Fellowship or similar title to any candidate of any university.

Place: Coimbatore

Date:

Project Guide **Head of the Department**

Submitted for the University Viva-Voce Examination held on————————

Internal Examiner **External Examiner**

COMPANY CERTIFICATE

NEXUS INFO

INTERNSHIP COMPLETION CERTIFICATE

March 2024,
Coimbatore, Tamil Nadu

TO WHOM IT MAY CONCERN

This is to certify that Mr./Ms. Nagarajan M has successfully completed a 3-month internship at Nexus Info in the field of Web development. The internship tenure spanned from December 25, 2023, to March 25, 2024. Throughout the internship, Nagarajan M actively engaged in diverse tasks within the Web development domain. Notably, he/she played a pivotal role in a client demo project, showcasing exceptional skills and dedication.

Nagarajan M demonstrated commendable hard work, enthusiasm, and a strong commitment to excellence. His/Her proactive approach and ability to tackle challenges reflect a promising foundation for a successful career in Web development. Throughout the internship, Nagarajan M consistently displayed dedication and a proactive attitude towards learning. His/Her enthusiasm for taking on new tasks and overcoming obstacles was evident in every project undertaken. These qualities underscore his/her potential to excel in Web development and contribute meaningfully to future endeavors.

Naveen

Naveen Kumar,
Founder, Nexus Info

NEXUS INFO APPROVED

contact@nexusinfo.in +91 8778993293 Coimbatore, Tamil Nadu
www.nexusinfo.in

ACKNOWLEDGEMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I express my sincere gratitude to our Professor & Head of the Department **Dr. M. Punithavalli M.Sc., M.Phil., Ph.D.,** Department of Computer Applications, Bharathiar University, Coimbatore for acknowledging our request and permitting me to undertake this project work.

I am forever indebted to **Dr. S. GAVASKAR, MCA.,Ph.D., Assistant Professor** Department of Computer Applications, Bharathiar University for providing valuable suggestions and always been enthusiastically guiding us throughout the project.

Finally, I also extend my special thanks to my family, friends and who have kindly provided necessary support for successful completion of the project and for this moral support.

ABSTRACT

Touchless interaction is an active area of research due to its application in the area of medical system to gaming. Sign langauge could benefit a people with hearing difficulty. It is important to realize sign language recognition on smartphones. Hand gestures are an effective touchless way to interact with computer systems. Several methods have been proposed for hand gestures. We are presenting a real-time on-device hand tracking solution which predicts a hand skeleton of a human using a single RGB camera. Here we are making use of Mediapipe library provided by google which is open-source and it is a well trained model to achieve high performance. Gestures of a hand can be determined using Mediapipe library using different technologies. In this Mediapipe hands library will use two models.1) a palm detector, that is providing a bounding box of a hand to, 2) a hand landmark model, that is predicting the hand skeleton. User can easily interact with the computers having RGB cameras using gestures.

TABLE OF CONTENTS

INTRODUCTION

In recent years we have seen so many people with hearing difficulty. Face-to-face communication is an very important channel that can convey message, feeling, and personal connection. People who have hearing difficulty use two types of communication, which are namely reading writing and a sign language. However, the inability to communicate has become an obstacle to the advancement of the hearing impaired it should be solved. Therefore, we focus on sign language as the most used means of communication. Hand tracking is a requisite component to provide a natural way for interaction and communication in AR/VR, and it has been an active research topic in the industry. Currently, there are many devices that use voice commands, especially on smartphones that we often use, voice commands themselves are based on speech recognition algorithms. Due to some of the problem like noisy background voice command itself has several shortcomings. An alternative to this problem is hand gestures. And also communication using hands is risky due to covid pandemic which requires maintaining distance. Different countries have their own sign languages. A sign language is not a universally common language, the U.S. has American Sign Language (ASL), the U.K. has British Sign Language (BSL) and China has its own sign language named Chinese Sign Language (CSL), Thailand has Thailand sign language TSL. Basically this sign language employs two signing modes: they are semantic and spelling signings. One of the communication media for deaf friends is to use sign language. In this paper, we propose a innovative approach that does not require any additional hardware and performs in real-time on mobile devices. Our main contributions are: • we are doing an efficient two-stage hand tracking pipeline which can track multiple hands in real-time on mobile devices. • A hand pose estimation model that is capable of predicting 2.5 Dimensional hand pose with only RGB input. • And open source hand tracking pipeline named media pipe as a ready-to go solution on a variety of platforms, including Android, iOS, Web and desktop PC's. Therefore, we created a project about hand gesture detection that can detect the movement or pose of the hand to be implemented in sign language recognition. The method used to perform hand gesture detection, of course, is to use computer vision which goes through several adjustments and filters on the hand. Then, the data obtained will be processed to be able to provide the appropriate output. In this paper, we attempt to simplify fingerspelling recognition and data processing using MediaPipe Hand proposed by Google, which can be used with typical digital camera.

LITERATURE SURVEY

We will get the dataset (collection of images) of different sign language's from the google and then our mediapipe models will be work on them. Then the collected data will be in the form CSV for the collected dataset Data cleaning , Data normalization and training of data will be carried out then the machine learning algorithms will be work on those dataset then finally predictions will be done. Different algorithms and their accuracy for the dataset will be calculated. Mediapipe is the best library for the gesture recognition.

Previous contribution

Fan Zhang Valentin Bazarevsky Andrey Vakunov Andrei Tkachenka George Sung Chuo-Ling Chang Matthias Grundman he had done a On-device hand tracking using mediapipe an end to end hand tracking is possible. Arsheldy Alvin1 , Nabila Husna Shabrina2 , Aurelius Ryo3 , Edgar Christian4 Fakultas Teknik dan Informatika, Universitas Multimedia Nusantara, Teknik Kompute Hand Gesture Detection for American Sign Language using K-Nearest Neighbor with Mediapipe was achieved

METHODOLOGY

Mediapipe is an open source cross-platform framework provided by Google to build a pipeline for processing perceptual data from different modalities such as video and audio. The solutions used in MediaPipe include multiple items such as posture estimation and face recognition. In this paper, we will use MediaPipe Hands for hand tracking. A similar skeletal estimation model, OpenPose , has been proposed. OpenPose can estimate 2D joint point coordinates from the input image, along with their reliability. In contrast, MediaPipe uses regression analysis to calculate the finger coordinates from the detected palm of the hand. Compared to OpenPose, our MediaPipe library reduces the amount of computation by using a smaller detection range. Also, by reducing the detection range, we were able to improve the accuracy of coordinate estimation for finger shapes, which OpenPose was not good at. Along with that from a single frame captured by a monocular camera, 21 3D coordinates in X, Y, and Z can be inferred and obtained. Of the output 3D coordinates obtained, the X and Y coordinates are normalized by the width and height of the bounding box. The Z coordinate represents the depth information, and its value becomes smaller as it gets closer to the camera and larger as it gets further away from the camera, using the wrist location as the origin. By using this method, we can perform advanced hand and finger tracking with less processing, and it can work in low performance environments such as mobile environments.

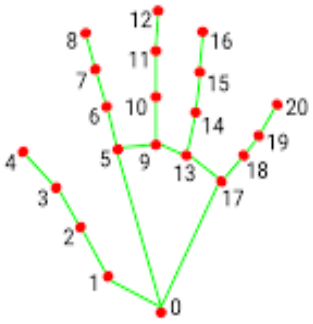
a)Palm detection Model

To detect initial hand locations, we employ a single shot detector model optimized for mobile real-time application similar to Blaze Face, which is also available in MediaPipe. Detecting hands is a not so easy it's a complex task: In this our model has to work across a variety of hand sizes with a large scale span and it should be able to detect occluded and self-occluded hands. Whereas our faces have high contrast patterns, e.g., around the eye and mouth region, the lack of such features in hands makes it comparatively difficult to detect them reliably from their visual features alone. Our solution is an alternative to the above challenges using different strategies. First, we train a palm detector instead of training a hand detector, since estimating bounding boxes of rigid objects like palms and fists is significantly simpler than detecting hands with articulated fingers. In addition, as palms are smaller objects, the non-maximum suppression algorithm(NMS) works well for the twohand self-occlusion

cases, like handshakes. Moreover, palms can be modelled using only square bounding boxes and ignoring other aspect ratios, and therefore reducing the number of anchors by a factor of 3∼5. Second, we are also using encoder-decoder feature extractor similar to FPN for a larger scene-context awareness even for small objects. Lastly, we will minimize the focal loss during training to support a large amount of anchors(rigid boxes) resulting from the high scale variance.

b) Hand Landmark Model

After the palm detection is over the whole image our subsequent hand landmark model performs precise key point localization of 21 3D hand-knuckle coordinates inside the detected hand regions via regression, that is direct coordinate prediction. The model learns a consistent internal hand pose representation and our Hand e-ISSN: 2582-5208 International Research Journal of Modernization in Engineering Technology and Science (Peer-Reviewed, Open Access, Fully Refereed International Journal) Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com www.irjmets.com @International Research Journal of Modernization in Engineering, Technology and Science [4151] landmark model is robust even to partially visible hands which re blurry sometimes and self-occlusions. To obtain ground truth data in our model, we have manually using-30K real-world images with 21 3D coordinates, as shown below (we take Z-value from image depth map, if it exists per corresponding coordinate). To cover the possible hand poses and also to provide additional supervision on the nature of hand geometry, we also render a high-quality synthetic hand model over various backgrounds and map it to the corresponding 3D coordinates.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 2.1 palm identitfication

Dataset Table 1: Details of different sign language fingerspelling datasets used in this work e-ISSN: 2582-5208 International Research Journal of Modernization in Engineering Technology and Science (Peer-Reviewed, Open Access, Fully Refereed International Journal) Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com www.irjmets.com @International Research Journal of Modernization in Engineering, Technology and Science [4152]

Architecture to detect hand gestures

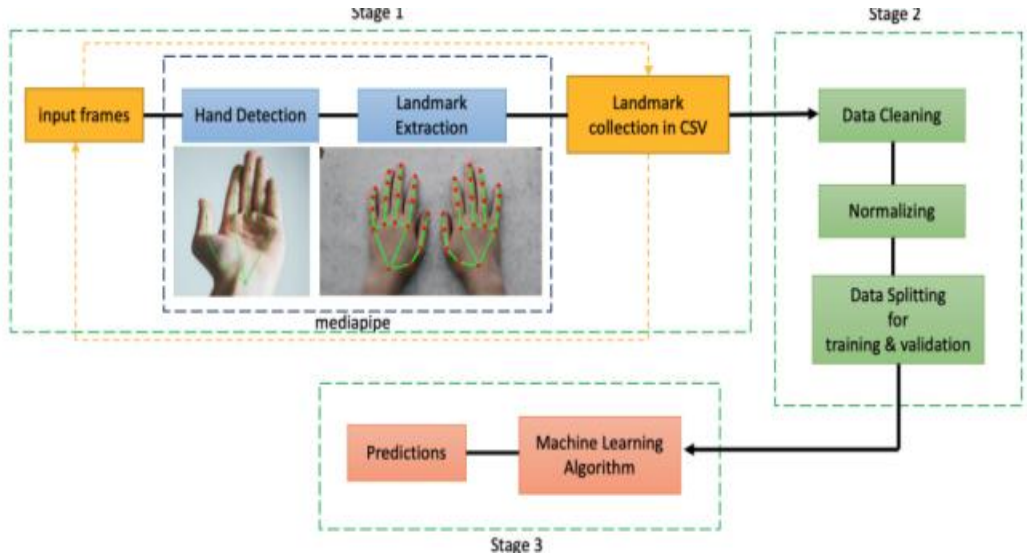


Figure 2.2 Hand landmark model

Stage 1: Pre-Processing of Images to get Multi-hand Landmarks using MediaPipe

In this step we will be making hands landmarks detection model with the profound library called as mediapipe as base library and for other computer vision pre-processing CV2 library. There are many use cases in the market for this problem statement whether it's for business related virtual reality or in the gaming section for the real-time experience. MediaPipe is a library like we already knows that framework that enables developers for building multi-modal like video, audio, any times series data cross-platform applied ML pipelines. MediaPipe library has a large collection of human body detection and also tracking models which are trained on a most diverse dataset of Google. As the skeleton of nodes and edges or landmarks, they track key points on different parts of the body. All co-ordinate points are three-dimension normalized. Models build by Google developers using Tensorflow lite facilitates the flow of information easily adaptable and modifiable using graphs.

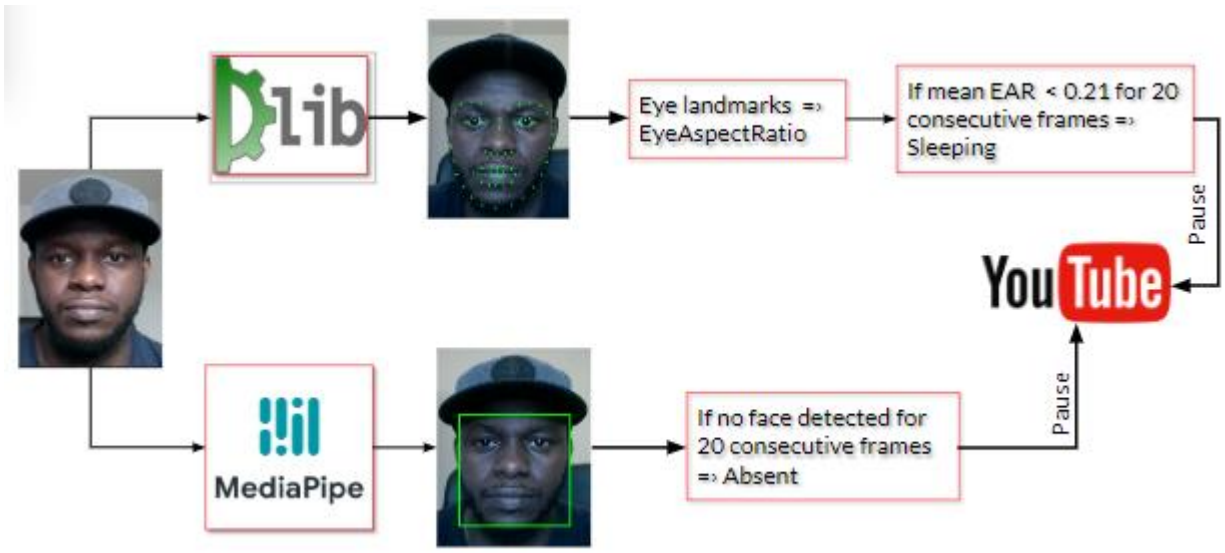


Figure 2.3 face and eye detection

Detecting hands is a complex procedure as you have to perform image processing and thresholding and work with a variety of hand sizes which leads to consumption of time. Instead of directly detecting hand from the current frame, first, the Palm detector is trained which estimates bounding boxes around the rigid objects like palm and fists which is simpler than detecting hands with coupled fingers. Secondly, an encoder-decoder is used as an extractor for bigger scene context. e-ISSN: 2582-5208 International Research Journal of Modernization in Engineering Technology and Science (Peer-Reviewed, Open Access, Fully Refereed International Journal) Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com www.irjmets.com @International Research Journal of Modernization in Engineering, Technology and Science [4153] After the palm detection is skimmed over the whole image frame, subsequent Hand Landmark models comes into the picture. This model precisely localize 21 3D hand-knuckle coordinates (i.e., x, y, z-axis) inside the detected hand regions. The model is so well trained and robust in hand detection that it even maps coordinates to partially visible hand. Figure 2 shows the 21 landmark points detection by the Hand Landmark model. Now that we have is a functional Palm and Hand detection model running, this model is passed over our dataset of various languages. Considering the American Sign Language dataset, we have a to z alphabets. So, we pass our detection model over every alphabet folder containing images and perform Hand detection which yields us the 21 landmark points as shown in Figure 2. The obtained landmark points are then stored in a file of CSV format. A simultaneous, elimination task is performed while extracting the landmark points. Here, only the x and y coordinates detected by the Hand Landmark model is considered for training the ML model. Depending upon the size of the dataset around 10-15 minutes is required for Landmark extraction.

Stage 2: Data cleaning and normalization

As in stage 1, here we are considering only x and y coordinates from the detector, in this stage each image in the dataset is passed through stage 1 to collect all the data points under one file. This file is then scraped through the pandas' library function to check for any nulls entries. Sometimes due to blurry image, the detector cannot detect the hand which leads to null entry into the dataset. we need to remove those entries in the dataset. Hence, it is necessary to clean these points or null entries otherwise it will lead to biasness while making the predictive model. And rows containing these null entries are searched and using their indexes removed from the table. After the removal of unwanted points, we normalized x and y coordinates to fit into our system. Then the data file is then prepared for splitting into two namely training and validation set. 80% of the data is retained for training our model with various optimization and loss function, whereas 20% of data is reserved for validating the model.

Stage 3: Prediction using Machine Learning Algorithm

Predictive analysis of different sign languages are performed using machine learning algorithms and Support Vector Machine (SVM) outperformed other algorithms. The details of the analysis are discussed in the result section. SVM is effective in high dimensional spaces. In the case where the number of samples are greater than the number of dimensions, SVM performs effectively. SVM is a cluster of supervised learning methods capable of classification, regression and outliers detection. This following formula shows the optimization problem tackled by In equation (1) and equation (2), di denotes the distances to the correct margin with di>= 0, I = 1, …, n, C denotes a regularization parameter, $wTw = w$ denotes the normal vector, $\emptyset\ xi$ denotes the transformed input space vector, b denotes a bias parameter,yi denotes the i-th target value. The objective is to classify as many data points correctly as possible by maximizing the margin from the Support Vectors to the hyperplane while minimizing the term wTw. The kernel function used is RBF (radical basis function) that turns the input space into a higher-dimensional space, so that not every data point is explicitly mapped. Support Vector Machine works relatively well when there is a clear margin of separation between classes. Hence, we used SVM to classify multiple classes of sign language alphabets and numeric.

Stage 4 : Quantitative Analysis

In this step to analyze the results for each of this datasets, we have used performance matrix such as accuracy, precision, recall and F1 score. Accuracy is the number of correctly predicted data points

out of all the data e-ISSN: 2582-5208 International Research Journal of Modernization in Engineering Technology and Science (Peer-Reviewed, Open Access, Fully Refereed International Journal) Volume:04/Issue:06/June-2022 Impact Factor- 6.752 www.irjmets.com www.irjmets.com @International Research Journal of Modernization in Engineering, Technology and Science [4154] points.$ASLR$ can be calculated as the number of all correct predictions to the total number of items in the data measures, shown in equation (3) P describes how accurate our model is out of those predicted positives, how many of them are actual positive. $PSLR$ is a good measure to determine, when the cost of False positive is high. $RSLR$ calculates how many of the actual positives our model capture by labelling them as positive. $RSLR$ represent the model metric we will select when there is a high cost associated with False Negatives. The mathematical formulation of Precision and Recall are given in equation (4) and (5) respectively. F-Measure in equation 6 provides us a way to combine both the precision and recall into a single measure that captures both the properties. It is used to handle imbalanced classification. And Confusion matrix was also analyzed and used to have a better understanding of the types of errors being made by our classifier. The key to confusion matrix is number of correct and incorrect predictions are summarized with count values and broken down by each class.

REQUIREMENTS TO RUN THE APPLICATION

Supportive Operating Systems: The supported Operating Systems for client include: Windows 2011

4.1 Software Requirements:

The Software Requirements in this project include:

a. Python

b. OpenCV framework

c. Sqlite

d. MS Excel

Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed. OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 7 million. The library is used extensively in companies, research groups and by governmental bodies. As an asynchronous event driven framework

4.2 Hardware Requirements:

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibilitylist (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. Face detection in a fixed image without special hypothesis is a difficult problem due to the high variability of the shape to detect. Many techniques of detection and face recognition have been developed in recent years and many of which are very efficient. Among these methods, we find the method of Viola and Jones were studied in this work. The aim of our work is to study this method to implement on the CPU with C / C + +, then acceleration was presented with OpenCV. Subsequently, a second implementation in FPGA was presented Keywords— Face detection, method of Viola/Jones, C/C++, OpenCV, FPGA. I. INTRODUCTION

Face detection is a new computer technology that determines the locations and sizes of human faces in images.

Components	**Minimum**	**Recommended**
Processor	Intel Core i3-2100 2nd generation	Intel Core i7 5th generation
RAM	4GB	8GB
Camera	HD 720p Webcam	Full HD 1080p Webcam
Disk	128Gb	512Gb

FLOW DIAGRAM

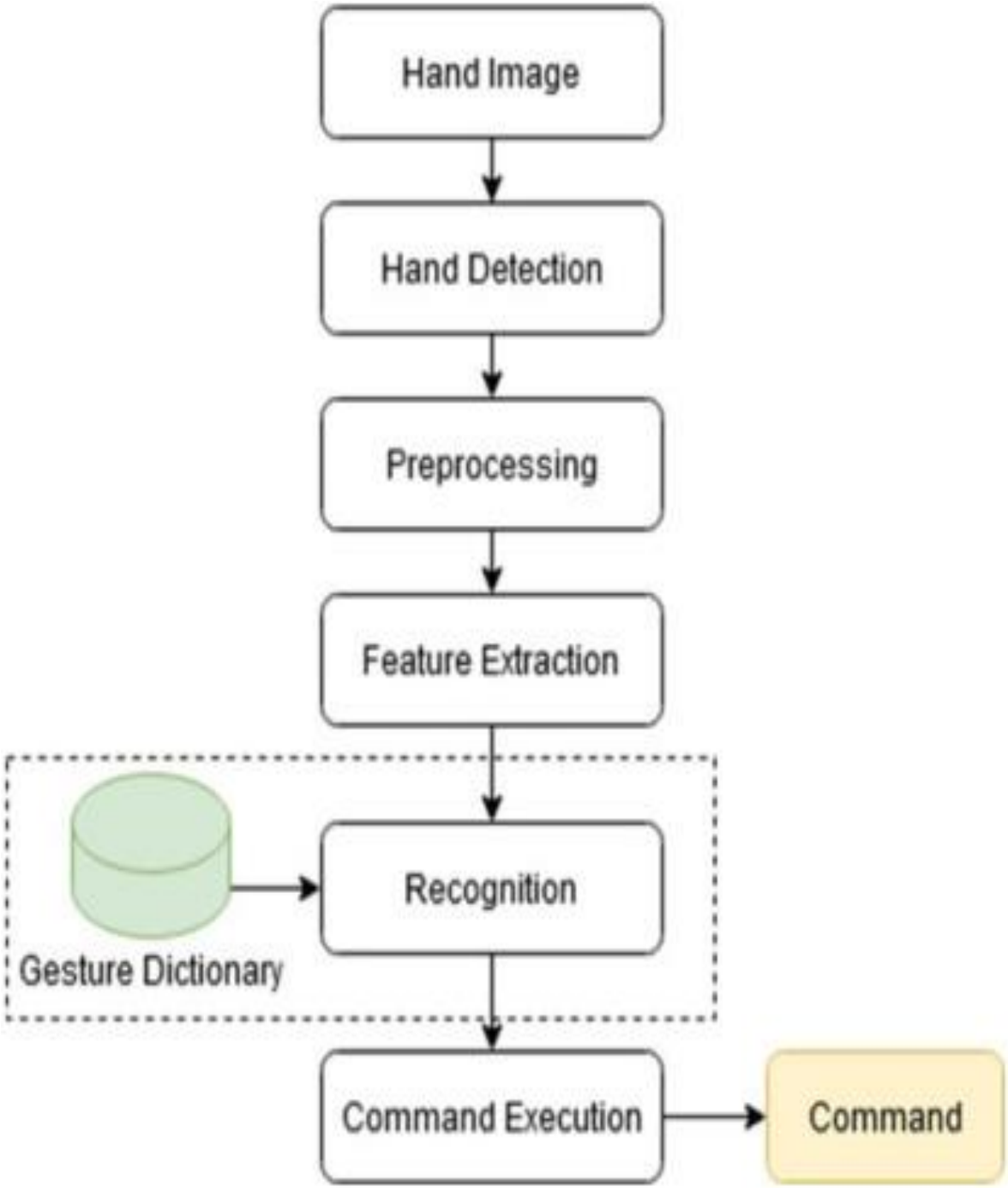


FIGURE 4.0 HAND GESTURE MODEL

SYSTEM TESTING

System testing is a comprehensive evaluation process conducted on a software system to ensure that it meets specified requirements and performs as expected in its intended environment. This phase of testing examines the entire system as a whole, including its integration with external dependencies, such as databases, servers, and third-party services. System testing involves executing test cases that validate various aspects of the system's functionality, performance, security, and usability, covering both functional and non-functional requirements. It aims to identify defects, inconsistencies, or deviations from expected behavior across different system components and interfaces. System testing encompasses a wide range of activities, including regression testing, integration testing, acceptance testing, and stress testing, to verify the system's readiness for deployment and operation in a production environment. By rigorously evaluating the system's behavior under various conditions, system testing helps mitigate risks, ensure quality, and enhance the overall reliability and effectiveness of the software solution.

5.1 DATABASE CONNECTIVITY TEST

Database connectivity test is a crucial process in software development and deployment, ensuring that applications can establish and maintain connections with databases successfully. This test verifies the integrity and functionality of database connection settings, network configurations, and authentication credentials. By simulating real-world scenarios, such as querying data or executing transactions, the connectivity test validates the ability of the application to interact with the database seamlessly. Additionally, it helps identify and troubleshoot connectivity issues, such as firewall restrictions, network latency, or misconfigured database drivers, before deploying the application to production environments. Through rigorous testing and validation of database connectivity, developers can ensure the reliability, performance, and security of their applications, ultimately delivering a seamless and uninterrupted user experience.

5.2 DATABASES

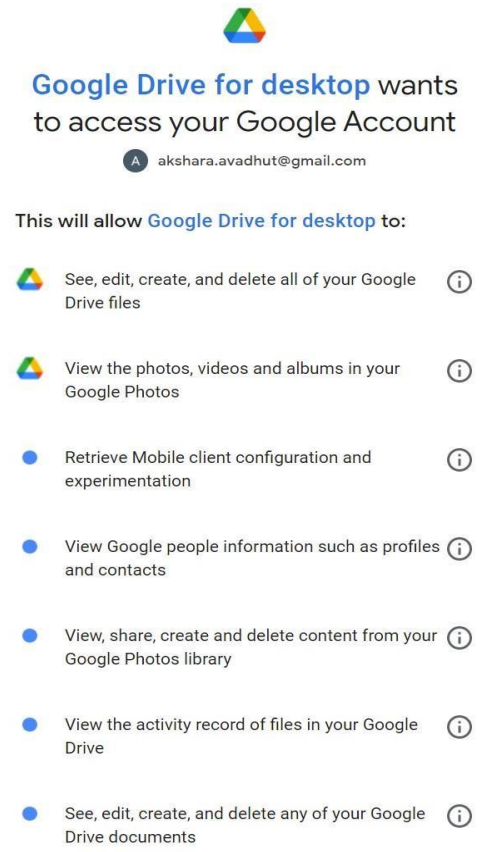
A database is a structured collection of organized data that is designed to be easily accessed, managed, and updated. It serves as a centralized repository for storing and retrieving information efficiently, enabling users to store, manipulate, and query data in a systematic manner. Databases typically consist of one or more tables, each containing rows and columns representing individual records and attributes, respectively. These tables are interconnected through relationships, facilitating the retrieval of related data across multiple tables. Databases are essential components of modern software systems, supporting a wide range of applications such as e-commerce platforms, customer relationship management (CRM) systems, and financial management software.

```
from google.colab import drive
drive.mount('/content/drive/')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleuser

Enter your authorization code:
```

Allow access through the link.



Drive mounted for data access.

```
from google.colab import drive
drive.mount('/content/drive/')
```

```
Mounted at /content/drive/
```

Files are accessible.

PROJECT STRUCTURE

```
├── data
│   ├── check_data.ipynb
│   ├── gestures.csv
│   ├── label.csv
│   └── player_state.json
├── flask_app
│   ├── static
│   │   ├── css
│   │   │   └── styles.css
│   │   └── icons
│   │       └── favicon-32x32.png
│   ├── templates
│   │   └── demo.html
│   ├── app.py
│   └── video_feed.py
├── models
│   ├── model.pth
│   ├── model_architecture.py
│   └── shape_predictor_68_face_landmarks.dat
├── main.py
├── requirements.txt
├── train.ipynb
└── utils.py
```

5.3 IMAGE PROCEESING

In [29]:

```
plt.figure(figsize=[16,16])
for i in range(36):
    img = test_images[i,:,:,0]
    p_dist = cnn.predict(img.reshape(1,48,48,1))
    k = np.argmax(p_dist)
    p = np.max(p_dist)

    cam = GradCAM(cnn, k)
    heatmap = cam.compute_heatmap(img.reshape(1,48,48,1))

    plt.subplot(6,6,i+1)
    plt.imshow(img, cmap='binary_r')
    plt.imshow(heatmap, alpha=0.5, cmap='coolwarm')
    plt.title(f'{emotions[test_labels[i]]} - ({emotions[k]} - {p:.4f})')
    plt.axis('off')
plt.tight_layout()
plt.show()
```

Figure 5.3 IMAGE PROCESSING

LIMITATIONS

➢ System faces recognition issues if the light conditions are poor

If lighting conditions are poor than it affects the recognition as it recognizes wrong and sometimes don't recognize in poor lighting conditions, so while creating dataset and recognizing it needs proper lighting

➢ Webcam and processor with high specification is required as system consumes a lot of resources.

As this project is doing image processing, so to run it smoothly high specification is required and graphic card is must to run it smoothly otherwise it will be too slow and may hang during its execution.

CONCLUSION AND FUTURE IMPLEMENTATION

The provided code appears to be a Python script for controlling YouTube playback using hand gestures captured from a webcam feed. Here's a summary of its functionalities and components:

1. **Imports**: The script imports necessary libraries such as OpenCV, MediaPipe, dlib, and pandas for various functionalities like video processing, hand detection, facial landmark detection, and data manipulation.
2. **Variables Initialization**: The script initializes various parameters and variables, including camera settings, hand detector, model paths, confidence thresholds, gesture history tracking, mouse movement stabilization, and facial detection parameters.
3. **Main Loop**: The script enters a loop where it continuously captures frames from the webcam feed, processes them, detects hand gestures, and controls the YouTube playback accordingly.
4. **Gesture Detection and YouTube Control**: Within the loop, the script detects hand gestures using MediaPipe, predicts gestures using a trained model, and maps them to specific YouTube control actions such as play/pause, volume adjustment, seeking forward/backward, and toggling fullscreen mode. It also implements mouse control functionalities such as moving the cursor, left-clicking, and right-clicking.
5. **Sleepiness Detection**: The script also includes functionality for detecting sleepiness by analyzing facial landmarks, particularly the eyes' aspect ratio. If the eyes are closed for a certain duration, it pauses the video playback.
6. **Absence Detection**: Additionally, the script detects the absence of a face using MediaPipe's face detection model. If no face is detected for a certain duration, it pauses the video playback.
7. **User Interface**: The script displays the webcam feed with overlays indicating detected gestures, eye aspect ratio, and face bounding boxes for debugging purposes.
8. **Loop Termination**: The script terminates when the user presses the 'q' key.

In conclusion, this script offers a comprehensive solution for controlling YouTube playback using hand gestures captured from a webcam feed. It combines computer vision techniques, machine learning for gesture recognition, and API integration for controlling the YouTube player. Additionally, it incorporates features for detecting sleepiness and absence of the user's face to enhance the system's robustness and usability.

FUTURE SCOPE

Certainly! Here are some potential areas for future improvement and expansion of the hand gesture-based YouTube control system:

Enhanced Gesture Recognition:

- Explore more advanced machine learning techniques or deep learning architectures to improve the accuracy and robustness of gesture recognition.
- Experiment with different hand pose estimation models or fine-tune existing models to better capture subtle gestures and variations in hand movements
- Allow users to define and customize their own hand gestures for controlling YouTube playback, providing a more personalized and intuitive interaction experience.
- Implement a calibration process where users can train the system to recognize their unique hand gestures and adapt to their preferences.
- Integrate additional input modalities such as voice commands or facial expressions alongside hand gestures to provide users with more flexibility and control options.
- Develop algorithms to seamlessly combine and interpret inputs from multiple modalities, enabling more natural and context-aware interactions.
- Implement real-time visual feedback mechanisms to provide users with immediate confirmation and guidance on recognized gestures and their corresponding actions.
- Explore techniques for visualizing hand gestures and their trajectories in the video feed, enhancing the user's understanding of the system's interpretation.
- Incorporate machine learning algorithms for adaptive system behavior, where the system learns and adapts its gesture recognition models over time based on user feedback and usage patterns.

Develop algorithms to dynamically adjust gesture recognition thresholds, parameters, and mapping strategies to optimize performance and user satisfaction.

- Support complex gesture sequences or gestural shortcuts for executing multi-step actions or navigating through YouTube content more efficiently.
- Design intuitive gestural shortcuts for common tasks such as bookmarking, sharing, or accessing settings, enhancing the system's usability and productivity.

- Extend the system's capabilities to control YouTube playback across multiple devices and platforms, such as smart TVs, gaming consoles, or mobile devices, using standardized protocols or APIs.

- Ensure seamless interoperability with popular YouTube clients and applications, allowing users to enjoy gesture-based control across their preferred devices and environments.

- Explore ways to make the system more accessible and inclusive for users with diverse abilities and preferences, such as providing alternative input methods or customizable interaction modes.
- Conduct user studies and usability tests with diverse user groups to identify and address potential accessibility barriers and optimize the system's usability for all.

By pursuing these avenues of development, the hand gesture-based YouTube control system can evolve into a more sophisticated, adaptive, and user-friendly interface for interacting with digital content, offering users a seamless and immersive multimedia experience.

APPENDIX

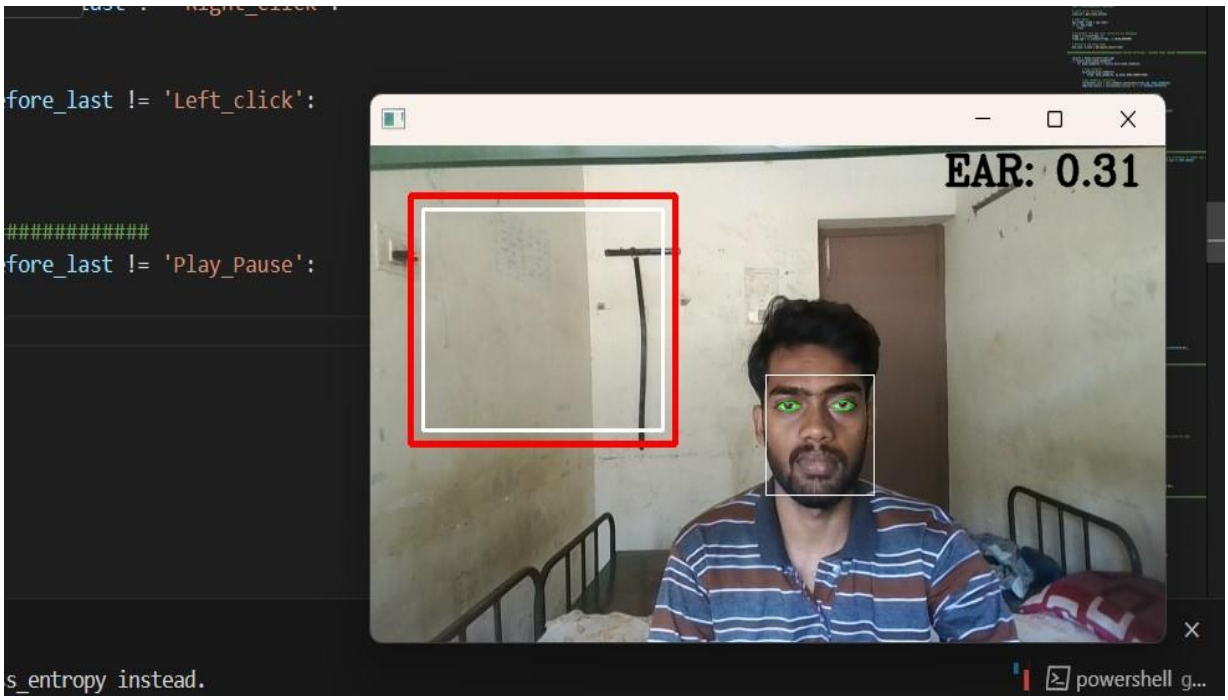


Figure 8.1 face,eye and hand detection

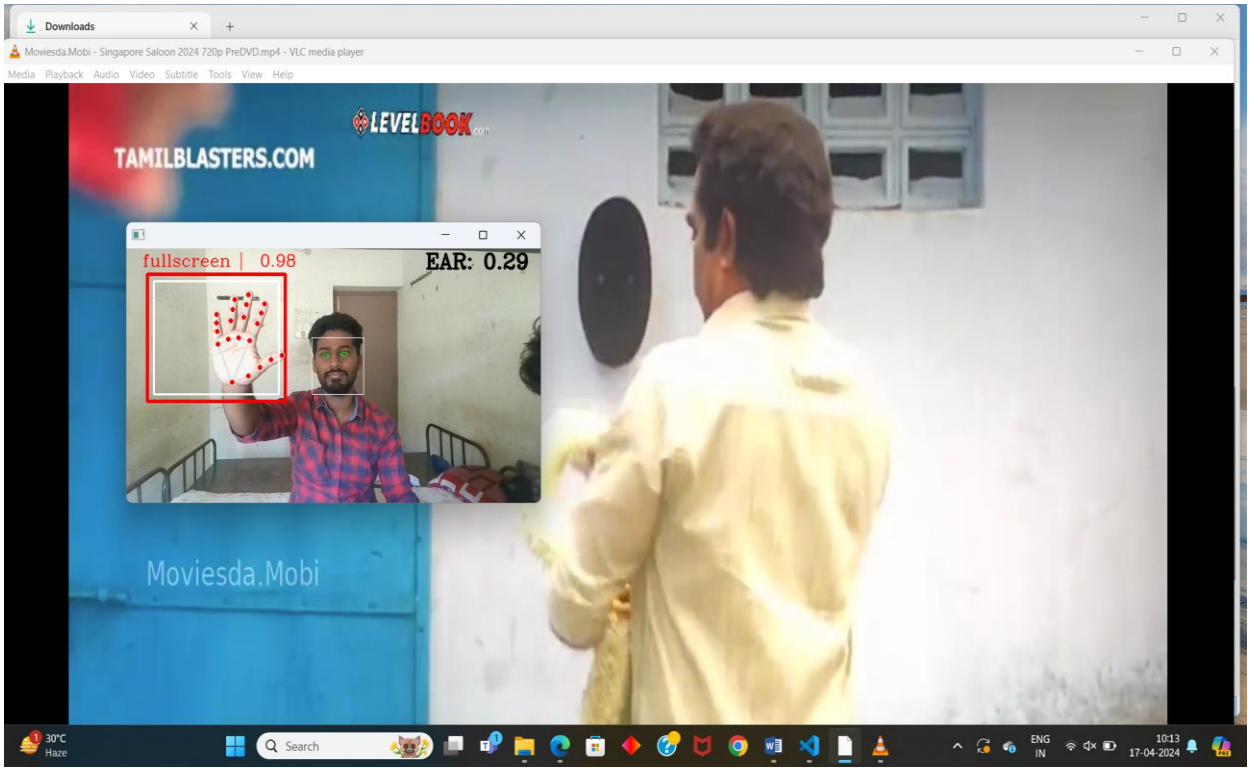


Figure 8.2 Full screen gesture

Figure 8.3 Fullscreen

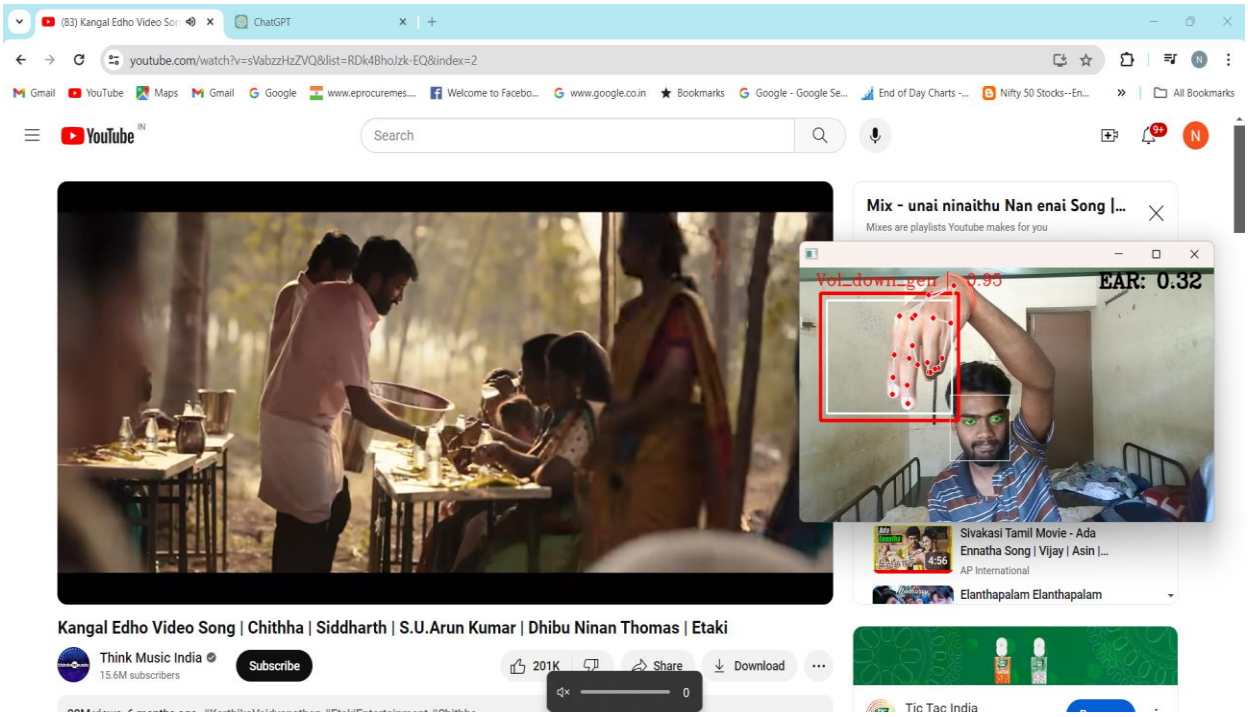


Figure 8.4 Volume down

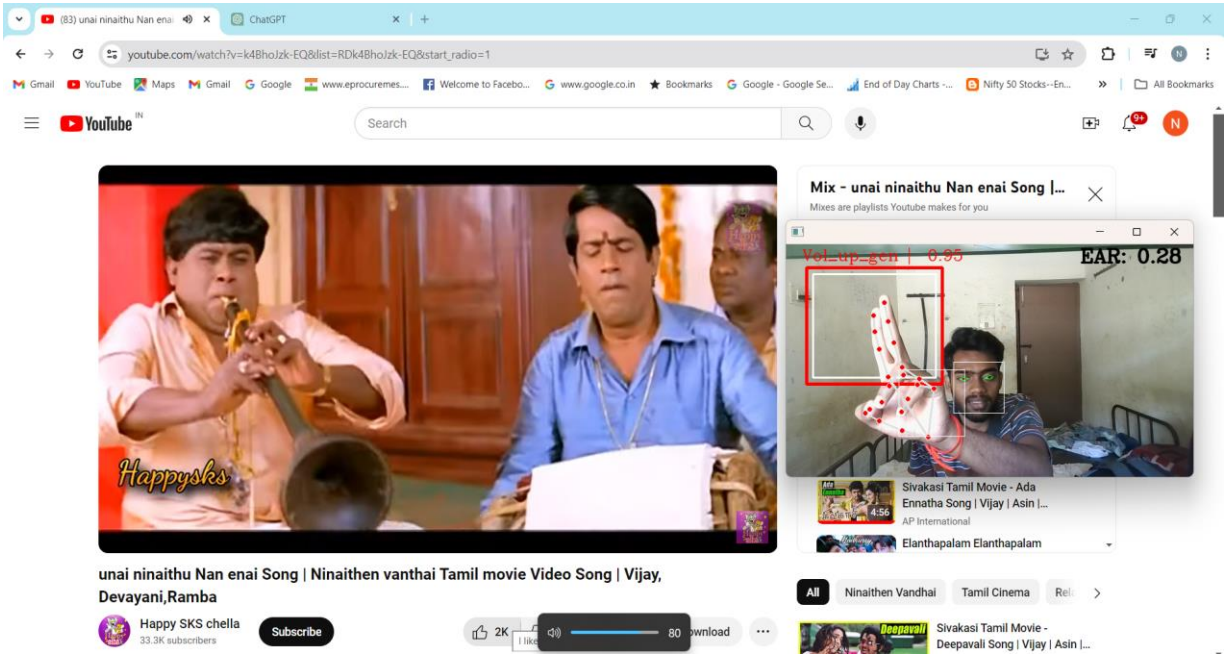


Figure 8.5 Volume up

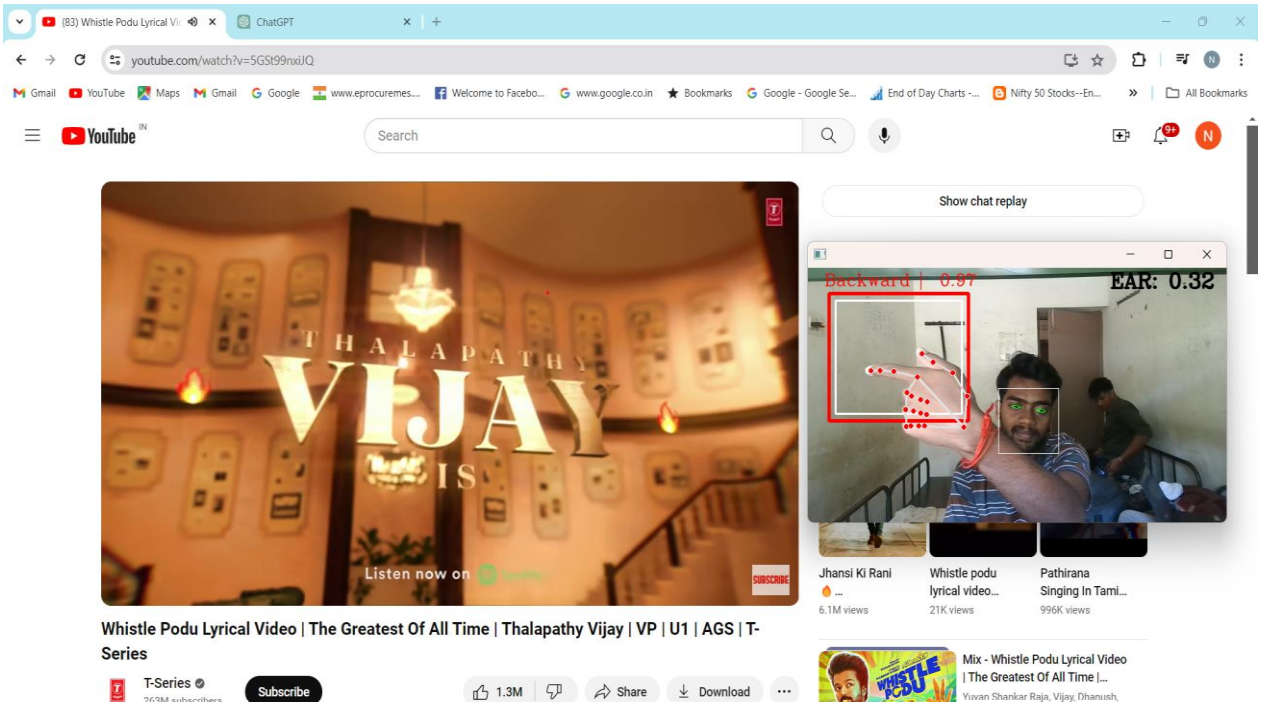


Figure 8.6 Backward

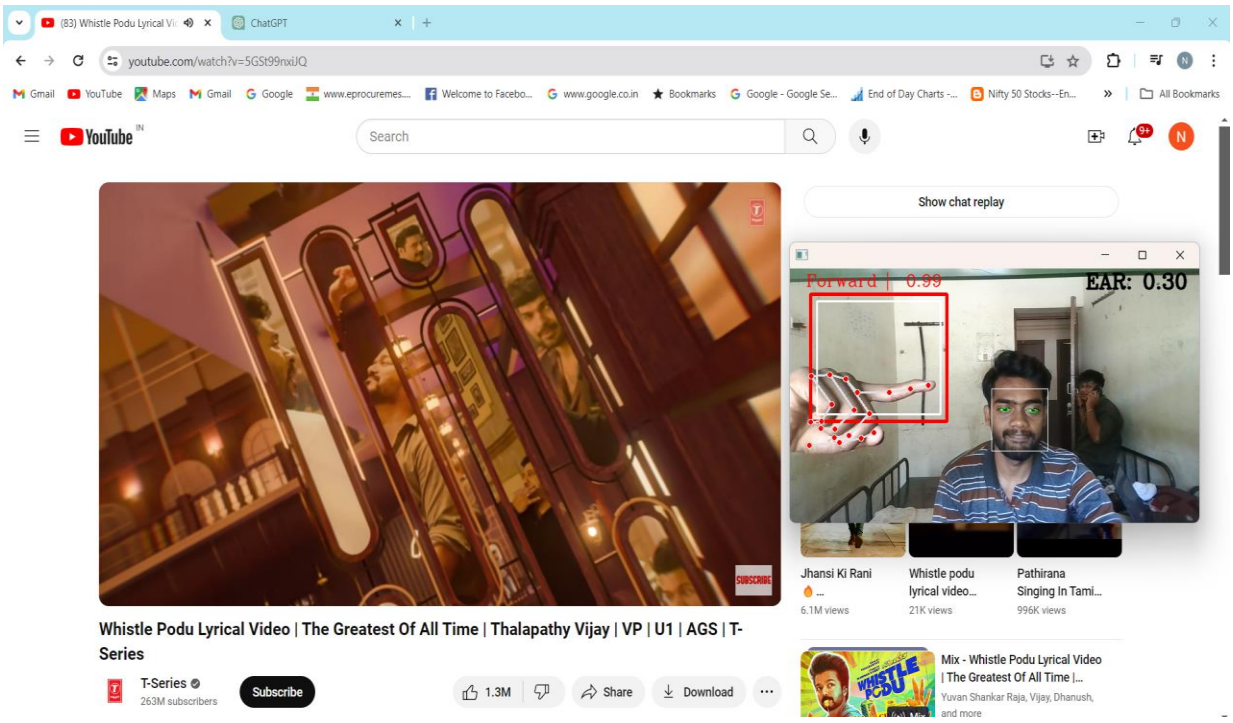


Figure 8.7 Forward

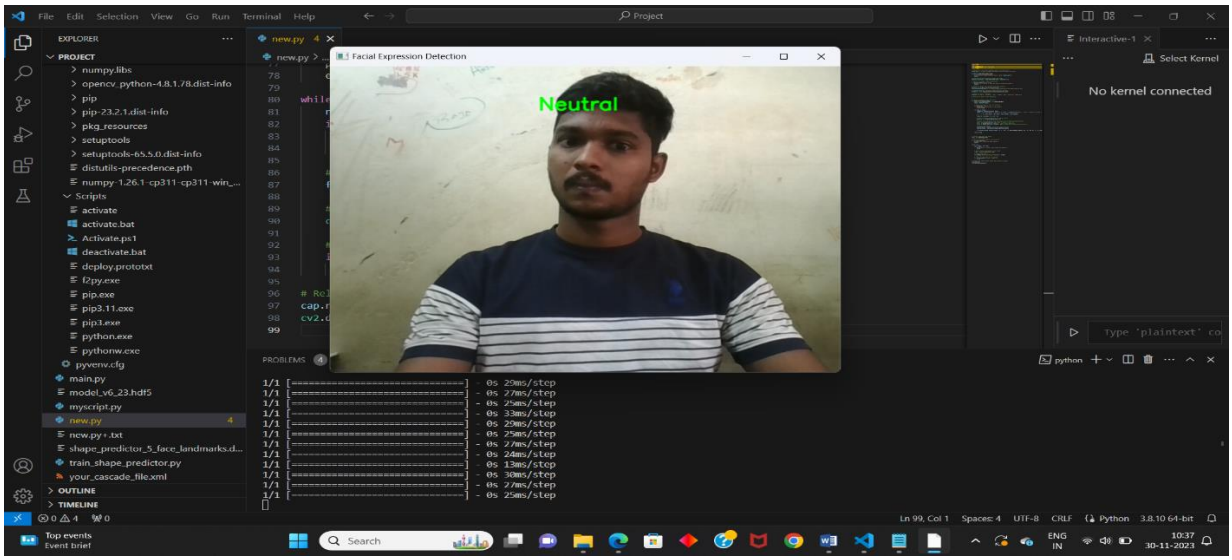


Figure 8.8 Neutral

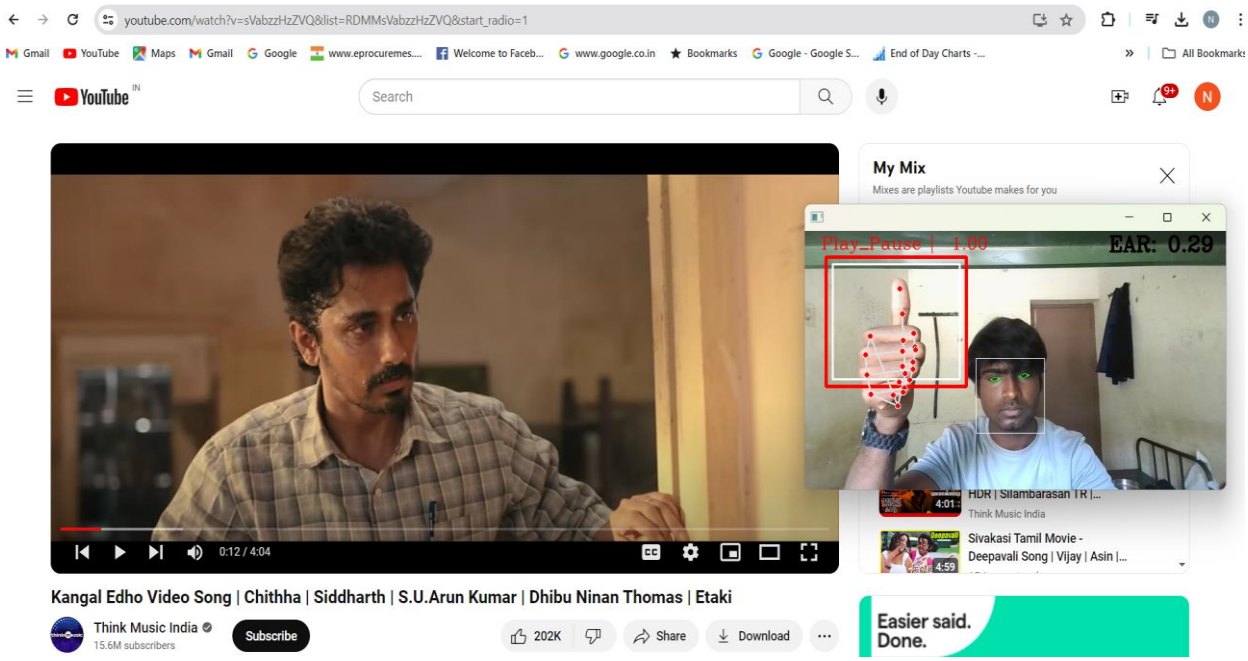


Figure 8.9 Pause

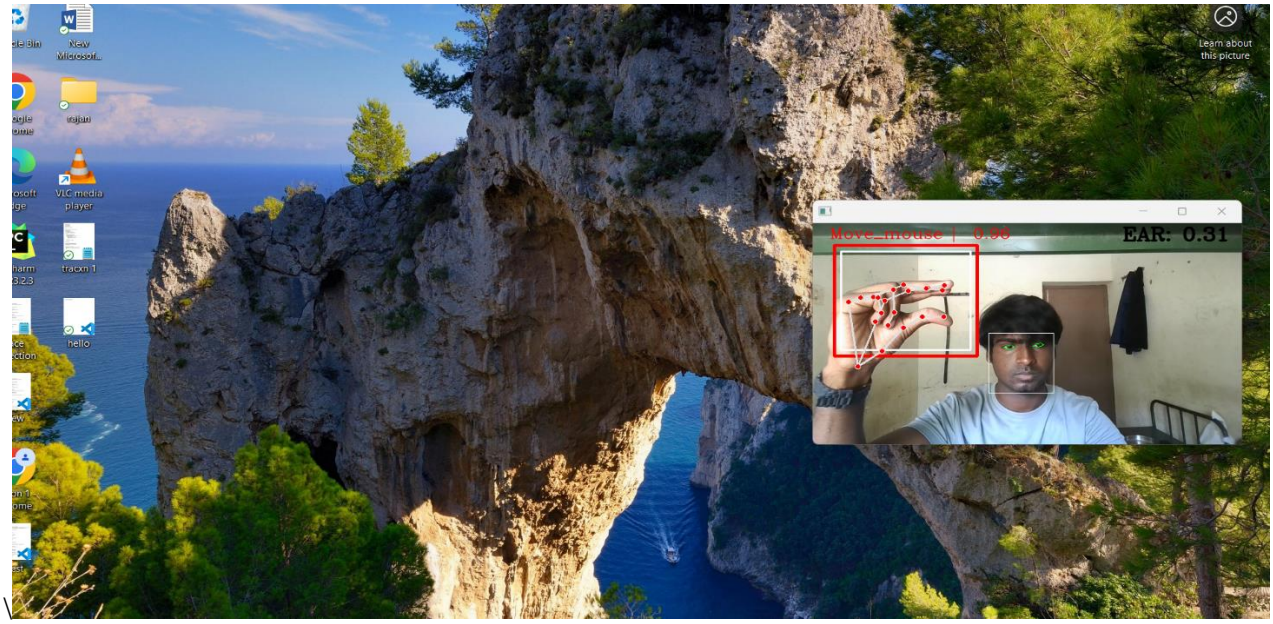


Figure 8.10 Move mouse

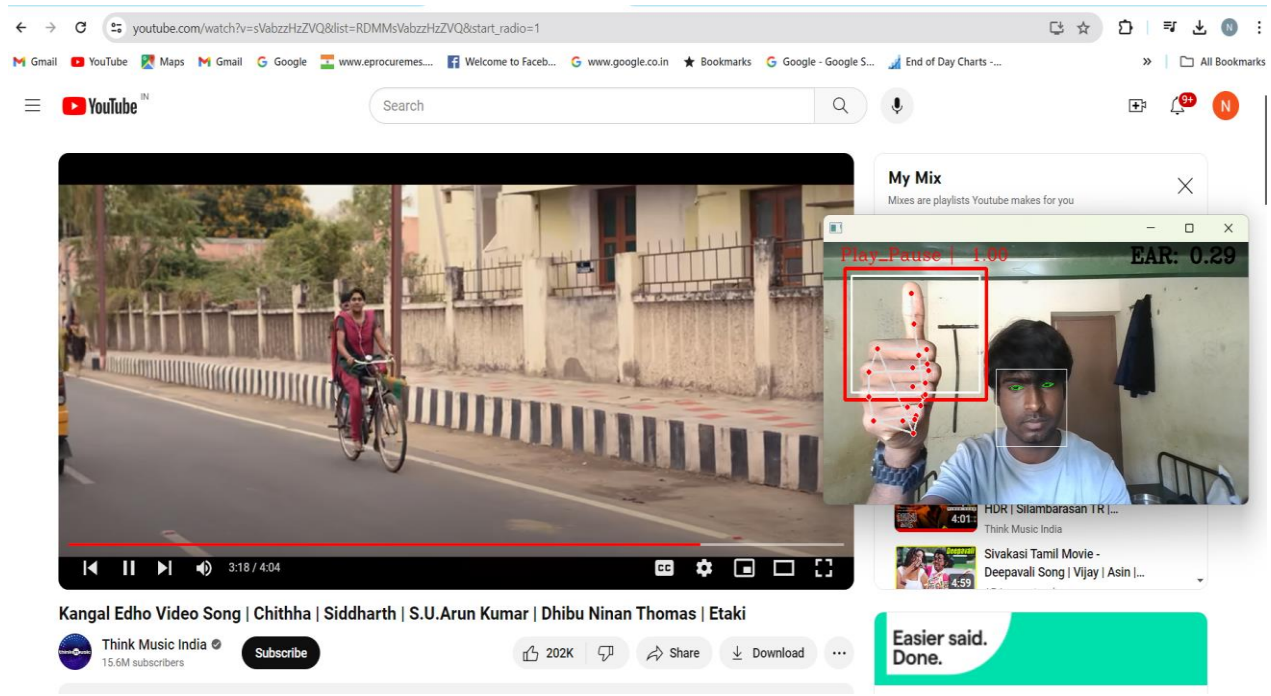


Figure 8.11 Play

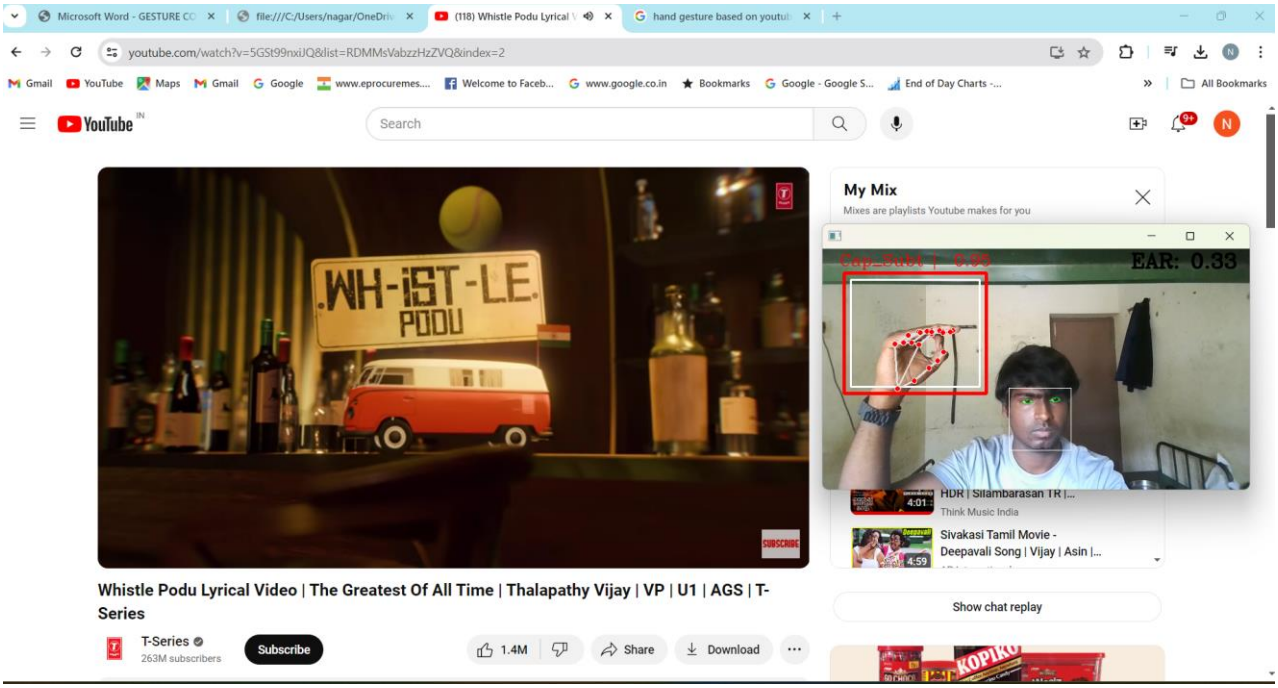


Figure 8.12 Caption subtitle

Figure 8.13 Left click

Figure 8.14 Right click

BIBLIOGRAPHY

WEBSITE REFERENCES:

- https://developers.google.com/mediapipe
- http://dlib.net/
- https://pyimagesearch.com/author/adrian/
- https://emerj.com/ai-sector-overviews/artificial-intelligence-in-gestural-interfaces/
- http://programmingcomputervision.com/
- https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/
- https://opencv24-python-tutorials.readthedocs.io/_/downloads/en/stable/pdf/
- https://pyimagesearch.com/deep-learning-computer-vision-python-book/
- https://ehmatthes.github.io/pcc/

BOOK REFERENCES:

- 1.Building a Hand Tracking System using OpenCV Analytics Vidhya https://www.analyticsvidhya.com/blog/2021/07/building-a-hand-trackingsystem-using-opencv/
- Hand Gesture Detection & Recognition System by Muhammad Inayat Ullah Khan Master Thesis Computer Engineering Nr: E4210D 3. Video Sources: -
- Hand Tracking 30 FPS using CPU | OpenCV Python (2021)| Computer Vision https://www.youtube.com/watch?v=NZde8X178lw
- Python Expert Full Course | Python Expert Tutorial | Python Expert Programming Simplilearn https://www.youtube.com/watch?v=OnhFe34iTJQ
- Graduation Project Seminar: - Hand Gesture Based Application “Virtual Mouse, Virtual Piano, Integration with interactive game”. Supervised by: Dr.LuaiMalhis. Prepared by SuadSeirafy, Fatima Zubaidi.
- Garg, P., Aggarwal, N., & Sofat, S. (2009). Vision based hand gesture recognition. World academy of science, engineering and technology, 49(1), 972-977
- Hasan, M. M., & Mishra, P. K. (2012). Hand gesture modeling and recognition using geometric features: a review. Canadian journal on image processing and computer vision, 3(1), 12-26.
- Ahuja, M. K., & Singh, A. (2015). Static vision based Hand Gesture recognition using principal component analysis. Paper presented at the 2015 IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE).