REVA UNIVERSITY
Bengaluru, India

SCHOOL OF COMPUTER SCIENCE AND APPLICATIONS

A PROJECT REPORT

on

Real-time Translation of Indian Sign Language to Text and Speech

Submitted in partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF COMPUTER APPLICATIONS

Submitted by

Amisha R21DA009

Under the Guidance of

Dr. S Senthil

Professor and Director

School of Computer Science and Applications

Reva University

June 2024

Rukmini Knowledge Park, Kattigenahalli, Yelahanka, Bengaluru -560064

www.reva.edu.in

REVA
UNIVERSITY
Bengaluru, India

SCHOOL OF COMPUTER SCIENCE AND APPLICATIONS

CERTIFICATE

The project work titled **"REAL-TIME TRANSLATION OF INDIAN SIGN LANGUAGE TO TEXT AND SPEECH"**, is being carried out under my guidance by **Amisha (R21DA009)** a bonafide students at REVA University, and are submitting the project report in partial fulfillment, for the award of **Bachelor of Computer Applications** during the academic year **2023–24**. The project report has been approved, as it satisfies the academic requirements with respect to the Project Work prescribed for the aforementioned Degree.

Signature with Date

[illegible] 8/6/2024

Dr. S Senthil
Internal Guide

Signature with Date

[illegible] 8.6.24

Prof. Lokesh C. K
Head of the Department

Signature with Date

[illegible]

Dr. S. Senthil
Professor and Director

DIRECTOR
School of Computer Science & Applications
REVA University, Kattigenahalli,
Bangalore - 560 064

Name of the Examiner with Affiliation	**Signature with Date**
1. Dr Mrinmayee Murthy	[illegible] 08/06/24.
2. Dr. Suhaas KP	[illegible] 08/6/24

DECLARATION

I, **Amisha (R21DA009)** pursuing my **Bachelor of Computer Applications,** offered by **School of Computer Science and Applications,** REVA University, declare that this Project titled **"REAL-TIME TRANSLATION OF INDIAN SIGN LANGUAGE TO TEXT AND SPEECH",** is the result of the Project Work done by me under the supervision of Dr. S Senthil.

I'm submitting this Project Work in partial fulfillment of the requirements for the award of the degree of Bachelor of Computer Applications by REVA University, Bengaluru, during the Academic Year 2023-24.

I further declare that this Project Report or any part of it has not been submitted for the award of any other Degree / Diploma of this University or any other University/ Institution.

Amisha

(Signature of the candidates)

Signed by me on: 07-06-2024

*Certified that this project work submitted by **Amisha** has been carried out under our guidance and the declaration made by the candidates is true to the best of my knowledge.*

[illegible] 8/6/2024

Signature of Internal Guide
Date:

Signature of External Guide
Date:

[illegible] 8/6

Signature of Director of the School
Date:
Official Seal of the School

DIRECTOR
School of Computer Science & Applications
REVA University, Kattigenahalli,
Bangalore - 560 064.

ACKNOWLEDGEMENT

I hereby acknowledge all those, under whose support and encouragement, I have been able to fulfil all my academic commitments successfully. In this regard, I take this opportunity to express my deep sense of gratitude and sincere thanks to School of Computer Science and Applications which has always been a tremendous source of guidance.

I express my sincere gratitude to **Dr. P. SHYAMA RAJU**, Honorable Chancellor, REVA University, and Bengaluru for providing us the state-of-the-art facilities.

I am thankful to **Dr. N RAMESH**, Vice Chancellor (i/c), REVA University, **Dr. K S NARAYANASWAMY**, Registrar (i/c), REVA University and **Dr. SANJAY CHITNIS**, Pro-Vice Chancellor, REVA University, Bengaluru for their support and encouragement.

I take this opportunity to express my heartfelt thanks to **Dr. S. SENTHIL**, Professor & Director, School of CSA, REVA University and my sincere thanks to **Prof. LOKESH C. K** Assistant Professor, Head of the Department for BCA Program, School of CSA, REVA University, whose encouragement and best wishes provided impetus for the Project Work carried out.
Also, my sincere gratitude goes to my internal guide, Dr. S Senthil, Professor and Director, School of CSA for the valuable suggestion and constant encouragement towards the completion of this project.

Last, but not the least, I thank my parents for their incredible support and encouragement throughout.

ABSTRACT

This Project report details the research and development efforts in creating a real-time translation system for Indian Sign Language (ISL) into text and speech, aiming to enhance communication and inclusivity for ISL users. The primary objective is to bridge the communication gap between individuals using ISL and those unfamiliar with the language.

The report outlines the findings and proposed solutions implemented in the project, providing practical insights and suggesting potential enhancements. The system's core functionality involves capturing ISL gestures via a webcam and translating them into corresponding text and speech outputs. Emphasis is placed on ensuring accuracy, efficiency, and user-friendly interaction.

Sign language translation is crucial for facilitating communication, especially for the deaf community. By developing a prototype translation system, this project aims to address communication barriers faced by ISL users. The designed system operates as a non-intrusive, real-time translation platform, prioritizing user safety and accessibility.

Machine learning techniques are employed to recognize and interpret ISL gestures accurately. The translated text is then synthesized into speech using advanced speech synthesis algorithms. The web interface, developed using modern technologies such as HTML, CSS, and JavaScript React.js, ensures a seamless and intuitive user experience.

This project aims to contribute to the advancement of accessibility and inclusivity by providing a practical solution for real-time ISL translation. Through ongoing optimization and refinement, the system endeavors to enhance communication effectiveness and promote a safer and more inclusive environment for all users. By prioritizing user safety, accessibility, and ease of use, the project offers a significant step towards eliminating communication barriers for ISL users. The integration of advanced machine learning and speech synthesis technologies demonstrates the potential for innovative solutions in the field of assistive communication. This report underscores the importance of continued research and development to further improve the system's utility and effectiveness, ultimately fostering greater inclusivity and accessibility for individuals relying on Indian Sign Language.

TABLE OF CONTENTS

CHAPTER-1

INTRODUCTION

1.1 INTRODUCTION TO PROJECT

PROJECT OVERVIEW:

Our project focuses on developing a real-time translation system for Indian Sign Language (ISL) to text and speech. This system leverages advanced technologies to capture ISL gestures through a webcam and translate them into written and spoken language in real-time. The primary goal is to bridge the communication gap between individuals using ISL and those who do not understand sign language. By providing an efficient and accurate translation system, we aim to enhance communication and understanding in various social, educational, and professional settings.

SIGNIFICANCE:

Addressing communication barriers faced by the deaf and hard of hearing community is crucial for fostering inclusivity and accessibility. Effective communication is a fundamental human right, and this project seeks to uphold that by providing ISL users with a tool that facilitates their interaction with the wider community. The significance of this project lies in its potential to transform how ISL users communicate, enabling them to participate more fully in society. By reducing misunderstandings and promoting clear communication, the system can improve the quality of life for ISL users, helping them access services, information, and opportunities that were previously challenging.

MOTIVATION:

Individuals using ISL often encounter significant challenges in everyday interactions. These challenges can affect their ability to access education, secure employment, and engage in social activities. The lack of effective communication tools often leads to isolation and limited opportunities for personal and professional growth. Our motivation stems from the desire to empower ISL users by enabling them to communicate effectively and participate fully in society. We aim to create a tool that not only aids in communication but also promotes understanding and empathy among non-ISL users, fostering a more inclusive community.

OBJECTIVES:

To develop a prototype system capable of accurately translating ISL gestures into textual representations and spoken language. To improve communication accessibility and promote inclusivity for ISL users in diverse contexts. By providing a reliable and user-friendly interface, we aim to make the system accessible to individuals of all ages and technical proficiencies. To conduct rigorous testing and refinement of the system to ensure high accuracy and responsiveness, thereby making it a practical tool for real-world use.

1.1.1 STATEMENT OF THE PROBLEM

Communication barriers between individuals who use Indian Sign Language (ISL) and those who do not understand it create significant challenges in daily interactions. These barriers manifest in various aspects of life, including access to essential services, education, employment opportunities, and social integration.

For instance, deaf and hard of hearing individuals often face difficulties in accessing healthcare services due to the lack of ISL interpreters, leading to misunderstandings and potential misdiagnoses. In educational settings, students using ISL may struggle to keep up with lessons and participate in classroom discussions, resulting in unequal learning opportunities.

In the workplace, communication challenges can hinder job performance, career advancement, and interaction with colleagues, ultimately affecting job retention and satisfaction. Socially, these barriers can lead to feelings of isolation and exclusion, as ISL users may find it challenging to engage in everyday conversations and activities with non-signing individuals.

The lack of effective communication tools to bridge this gap underscores the urgent need for an innovative solution that promotes inclusivity and accessibility. Current methods, such as relying on human interpreters or written communication, are not always feasible or efficient. Human interpreters are not always available, especially in spontaneous or informal situations, and written communication can be slow and cumbersome.

An innovative solution that can provide real-time translation of ISL into text and speech would significantly alleviate these communication barriers. Such a solution would enable ISL users to interact more freely and naturally with the wider community, ensuring they have equal opportunities to access services, education, and employment. It would also foster greater social integration and understanding, creating a more inclusive and accessible environment for all.

In summary, addressing these communication barriers is essential for enhancing the quality of life for ISL users and promoting a more inclusive society. The development of a real-time ISL translation system represents a critical step towards achieving these goals, empowering the deaf and hard of hearing community to participate fully and equally in all aspects of life.

1.1.2 BRIEF DESCRIPTION OF THE PROJECT

Our project aims to develop a real-time translation system that converts Indian Sign Language (ISL) gestures into text and speech. This system captures ISL gestures using a webcam and processes these gestures through machine learning algorithms to accurately recognize and interpret the signs. The recognized gestures are then translated into textual representations and synthesized into speech using advanced speech synthesis algorithms.

The web-based interface of the system is developed using modern technologies such as HTML, CSS, and JavaScript, to ensure an intuitive and user-friendly experience. The primary goal is to facilitate seamless communication between ISL users and those who do not understand sign language, thereby promoting inclusivity and accessibility. The system prioritizes user safety, accuracy, and efficiency, ensuring that it operates non-intrusively and effectively in real-time. By addressing communication barriers, this project aims to empower ISL users, enabling them to participate fully in various aspects of society, including education, employment, and social interactions.

1.1.3 SOFTWARE AND HARDWARE SPECIFICATION

1.1.3.1 SOFTWARE REQUIREMENTS

Concerning the Software Requirements of this project, it was done in its entirety using the programming language Python 3.10 and with the help of the following libraries:
Operating System : WindowsXP/7/8/10or Linux
Programming Language : Python 3.10
IDE : jupyter notebook, Visual Studio Code

Webcam : To get an image, a webcam is required. The camera is used for real-time images at any time the pc is on. The system will pick frames for prediction.
TensorFlow : TensorFlow is a free and open-source software libraryfor machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.Tensorflow is a symbolic math library based on dataflow and differentiable programming.
Keras: Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

1.1.3.2 HARDWARE REQUIREMENTS

RAM: 8 GB (minimum)
HDD: 1 TB or more
Processor: Intel® i3 or faster (2–4 GHz)
GPU: 2 GHz or more

1.2 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

1.2.1 FUNCTIONAL REQUIREMENTS

➢ Gesture Recognition:

The system must capture ISL gestures using a webcam. It must accurately recognize and interpret a predefined set of ISL gestures.

➢ Translation:

The system must translate recognized ISL gestures into corresponding textual representations. It must translate the recognized ISL gestures into spoken language using speech synthesis.

➢ User Interface:

The system must provide a user-friendly web interface for interaction. It must display the translated text in real-time on the interface. The interface must include audio output for the translated speech.

➢ Machine Learning Integration:

The system must use machine learning algorithms to improve the accuracy of gesture recognition. It must support the training and updating of the gesture recognition model with new data.

➢ Real-Time Processing:

The system must process and translate ISL gestures in real-time to ensure immediate communication.

➢ User Input:

The system must allow users to manually correct translations if necessary. It must support user feedback to continuously improve translation accuracy.

1.2.2 NON-FUNCTIONAL REQUIREMENTS

- Performance:

The system must have low latency, ensuring real-time translation without significant delays. It must handle high volumes of data processing efficiently.

- Accuracy:

The system must achieve high accuracy in gesture recognition and translation to be reliable and useful.

- Usability:

The interface must be intuitive and easy to use for all users, including those with limited technical skills. It must be accessible to users with different levels of ability and proficiency.

- Scalability:

The system must be scalable to accommodate an increasing number of users and larger datasets for training.

- Reliability:

The system must be reliable and consistently available, with minimal downtime. It must provide robust error handling and recovery mechanisms.

- Maintainability

The system must be easy to maintain, with well-documented code and architecture. It must support regular updates and improvements without disrupting existing functionalities..

- Accessibility:

The system must comply with accessibility standards to ensure it is usable by individuals with disabilities. It must provide features such as adjustable text size, high-contrast modes, and screen reader compatibility.

CHAPTER-2

LITERATURE SURVEY

Sign language is not a new computer vision problem. Over the past two decades, researchers have addressed it in various ways. These can be broadly classified into glove-based or sensor-based methods and appearance based or vision-based methods. Glove or sensor-based approaches provide good accuracy as these specific devices collect data or features directly from the signer but, it will have an overhead of carrying and signing with the external device. Vision-based methods, on the other hand, do the recognition task from images or videos based on the features calculated using various image or video processing techniques. This can be again categorized into 2-D Vision-based techniques and 3-D Vision-based techniques. Researchers have used classifiers from a variety of categories that we can group roughly into linear classifiers, neural networks and Bayesian networks. These two main approaches are used for acquiring gestures from the user. The first approach is vision-based, which provides more freedom to the user to perform natural actions and the other is a sensor-based approach that has simpler data acquisition and processing requirements.
Accorting to paper, T. Pan, L. Lo, C. Yeh, J. Li, H. Liu and M. Hu, "Real-Time Sign Language Recognition in Complex Background Scene Based on a HierarchicalClustering Classification Method," 2016, a real-time sign language recognition system is implemented that can overcome a complex background scene using a hierarchical clustering classification method. It uses 26 hand gestures in the training phase to develop a robust hand segmentation method to generate contour information. Three types of features are then extracted to represent the gesture and a dimension reduction step is applied to find the most discriminative features for the final 7 classification step. These features are combined to describe the contours and the salient points of hand gestures. Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and Support Vector Machine (SVM) are integrated to construct a novel hierarchical classification scheme. The procedures in the testing phase are similar to those in the training phase; however, the input in the testing phase is a video sequence composed of continuous hand gestures. Considering the real-time issue, a keyframe selection method filters out the redundant frames in the video, and only important/stable frames are kept to be recognized. To extract hand features with less noisy information even though the background is complex, the hand region is first located by skin colour detection. A method to adaptively update the skin colour model for different users and various lighting conditions is also designed. This is

evaluated over the CSL and public ASL datasets. The method achieves the accuracies of 99.8% and 94%, respectively, which outperforms the existing works.

According to paper, M. Kim, L. Chau and W. Siu, "Keyframe selection for motion capture using motion activity analysis," 2012 a keyframing approach is proposed to reduce the motion databy extracting keyframes using a motion analysis approach in sampling windows. Motion changes in sampling windows for original motion without frame skipping and with frame skipping are computed. The difference in the motion changes is the main aspect in deciding whether the frames in sampling windows are possible candidates for keyframe selection. Significant motion changes shall be defined as motion activity that holds above a threshold parameter. Five types of motion sequence were tested based on different dynamics of the motion, with low dynamic motion examples such as walking, jumping and stretching motion; and high dynamic motion examples such as dancing and playing basketball motion. The proposed motion analysis method was tested to evaluate the performance of keyframe extraction. The proposed method which uses 8 different sampling window size at the same threshold value to extract keyframes performs better than the curve simplification method when comparing the same number of keyframes extracted. For low dynamic motion such as the walking sequence, 58 frames were extracted out of a total of 316 frames and showed an improvement of 52.2% in performance. For high dynamic motion such as basketball sequence, the proposed method can reduce 70% of the total frames with an MSE error of 0.0463.

Accorting to paper, S. C.J. and L. A., "Signet: A Deep Learning based Indian Sign Language Recognition System," 2019 Indian sign language static alphabet recognition problems with a vision-based approach are addressed. A Convolutional Neural Network (CNN) which is a deep learning technology is used to create a model named signet, which can recognize signs, based on supervised learning on data. The whole process can be divided into CNN training and model testing. In this work, a dataset containing binary hand region silhouettes of the signer images is used. These images were extracted and saved after the preprocessing stage. This preprocessing includes Viola-Jones face detection algorithm to detect the face of the signer and then it is eliminated by replacing it with black pixels. This image is then processed with a skin colour segmentation algorithm followed by the largest connected component algorithm for hand region segmentation. These images are used in this work to train and test the signet architecture. Images along with their class labels are given to the developed CNN architecture to learn the classification model. The learned classification model can be tested and then saved for recognizing ISL static alphabets. The proposed method produced a remarkable result compared to current state of art methods. Training accuracy of 99.93% and validation accuracy of 98.64% was achieved.

Accorting to paper, Thad Starner and Alex Pentland, "Real-Time American Sign Language Recognition from Video using Hidden Markov Models" 1995 an extensible system is discussed which uses one colour camera to track hands in real-time and interprets American Sign Language (ASL) using Hidden Markov Models (HMM's). In this implementation, the tracking process produces only a coarse description of handshape, orientation, and trajectory. The hands are tracked by their colour: in the first experiment via solidly coloured gloves and the second, via their natural skin tone. In both cases, the resultant shape, orientation, and trajectory information is input to an HMM for recognition of the signed words. By combining the relative motion of the hand between frames and the absolute position of the hand with the angle, eccentricity, area, and length of the major eigenvector, the highest fair test accuracy, 91.9%, was reached for natural skin tracking.

Accorting to paper, Washef Ahmed, Kunal Chanda, Soma Mitra, "Vision-based Hand Gesture Recognition using Dynamic Time Warping for Indian Sign Language", 2016, an algorithm of Hand Gesture Recognition by using Dynamic Time Warping methodology is presented. The system consists of three modules: real-time detection of face region and two hand regions, tracking the hands' trajectory both in terms of direction among consecutive frames as well as the distance from the centre of the frame and gesture recognition based on analyzing variations in the hand locations along with the centre of the face. The proposed technique is used to perform similarity measurement efficiently and increases the adaptability of a hand gesture recognition system. The technique works well under different degrees of scene background complexity and illumination conditions. Experimental results showed the recognition rate to be around 90%.

CHAPTER-3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In the realm of real-time translation systems for Indian Sign Language (ISL) into text and speech, the current landscape presents several significant limitations and challenges. Existing solutions are often sparse and lack comprehensive coverage, leaving many communication needs of ISL users unaddressed. Many of these systems are narrowly focused, addressing only specific aspects of ISL translation without offering an integrated end-to-end solution. This fragmentation leads to gaps in functionality and usability, limiting the practical application of these systems in real-world scenarios. A primary issue is accuracy; many systems struggle to recognize and interpret the nuanced gestures inherent in ISL accurately. ISL involves complex hand movements, facial expressions, and body postures that are context-dependent and highly variable, making accurate recognition a challenging task. The limited availability of annotated ISL datasets further exacerbates this problem, as machine learning models trained on small or poorly annotated datasets fail to generalize well to the diversity of ISL gestures, leading to frequent misinterpretations.

Moreover, inefficiencies in processing speed and resource utilization pose significant barriers to the effectiveness of these systems. Real-time translation requires rapid processing to minimize latency, ensuring seamless communication. However, many existing systems experience delays due to inefficient algorithms and high computational demands. These delays disrupt the natural flow of conversation, rendering the systems less effective for real-time use. Additionally, resource-intensive solutions can be impractical for deployment on standard consumer hardware or mobile devices, further limiting their accessibility and usability.

User-friendliness and accessibility are also critical areas of concern. Many existing systems feature complex interfaces that are difficult for users, particularly those who are not technologically adept, to navigate. The lack of customization options means these systems cannot be easily adapted to individual user preferences, such as adjusting font sizes, color

schemes, or interaction methods, which is essential for accommodating diverse needs. Furthermore, compliance with web accessibility standards is often inadequate, making these systems less usable for individuals with additional disabilities, such as visual impairments. This lack of accessibility and ease of use hinders effective communication between ISL users and non-proficient individuals, failing to bridge the communication gap fully.

Additionally, the isolation of these systems from other commonly used communication tools, such as video conferencing platforms, messaging apps, and educational software, limits their practical utility. Integration with these tools is essential for facilitating seamless communication across different contexts, but many existing solutions operate in silos, reducing their effectiveness. Finally, the lack of mechanisms for continuous learning and adaptation is a significant drawback. Sign languages, including ISL, are dynamic and evolve over time. Effective translation systems must support ongoing updates and improvements to stay current with new gestures and expressions. However, many current solutions employ static models that do not learn from new data, resulting in outdated and less accurate translations over time. Without incorporating user feedback and continuous learning, these systems fail to adapt, limiting their long-term relevance and utility. In summary, while existing real-time ISL translation systems represent important steps toward inclusivity, they are plagued by significant limitations in accuracy, efficiency, usability, accessibility, integration, and adaptability, highlighting the need for more advanced and comprehensive solutions.

3.2 LIMITATIONS OF THE EXISTING SYSTEM

Limited Availability: Existing real-time translation systems for Indian Sign Language (ISL) into text and speech are few in numbers and may not be readily accessible to all users.

Accuracy Challenges: Many existing solutions struggle with accurately interpreting complex gestures and expressions inherent in ISL, leading to errors and misunderstandings in translation.

Inefficiency: Some systems may exhibit inefficiencies in processing speed and resource utilization, resulting in delays and suboptimal performance during translation.

Accessibility Issues: User-friendliness and accessibility are often compromised in existing systems, as they may not cater to the diverse needs and preferences of ISL users, hindering effective communication with non-proficient individuals.

3.3 PROPOSED SYSTEM

In response to the shortcomings of existing solutions, our project proposes the development of a real-time translation platform specifically tailored for ISL. This proposed system represents a significant advancement in accuracy, efficiency, and user-friendliness, aiming to revolutionize the communication experience for ISL users and non-proficient individuals alike. By leveraging state-of-the-art technologies, including machine learning for gesture recognition and advanced speech synthesis algorithms, the proposed system seeks to deliver unparalleled accuracy and speed in translating ISL gestures into textual representations and spoken language. Moreover, the platform is designed with a strong emphasis on user centricity, ensuring accessibility and inclusivity for ISL users across various settings and contexts.

3.4 ADVANTAGES OF THE PROPOSED SYSTEM

Comprehensive Coverage: The proposed system aims to address the limitations of existing solutions by offering a comprehensive real-time translation platform tailored specifically for ISL.

Enhanced Accuracy: Leveraging advanced machine learning techniques, the proposed system strives to achieve higher accuracy in recognizing and interpreting ISL gestures, ensuring more reliable translation outcomes.
Improved Efficiency: By optimizing processing algorithms and utilizing efficient computing resources, the proposed system aims to minimize latency and enhance overall performance during translation.

User-Centric Design: Accessibility and user-friendliness are central to the design philosophy of the proposed system, with features and functionalities tailored to accommodate the diverse communication needs and preferences of ISL users.

Inclusivity: The proposed system is designed to promote inclusivity by facilitating seamless communication between ISL users and non-proficient individuals, thereby breaking down barriers and fostering mutual understanding.

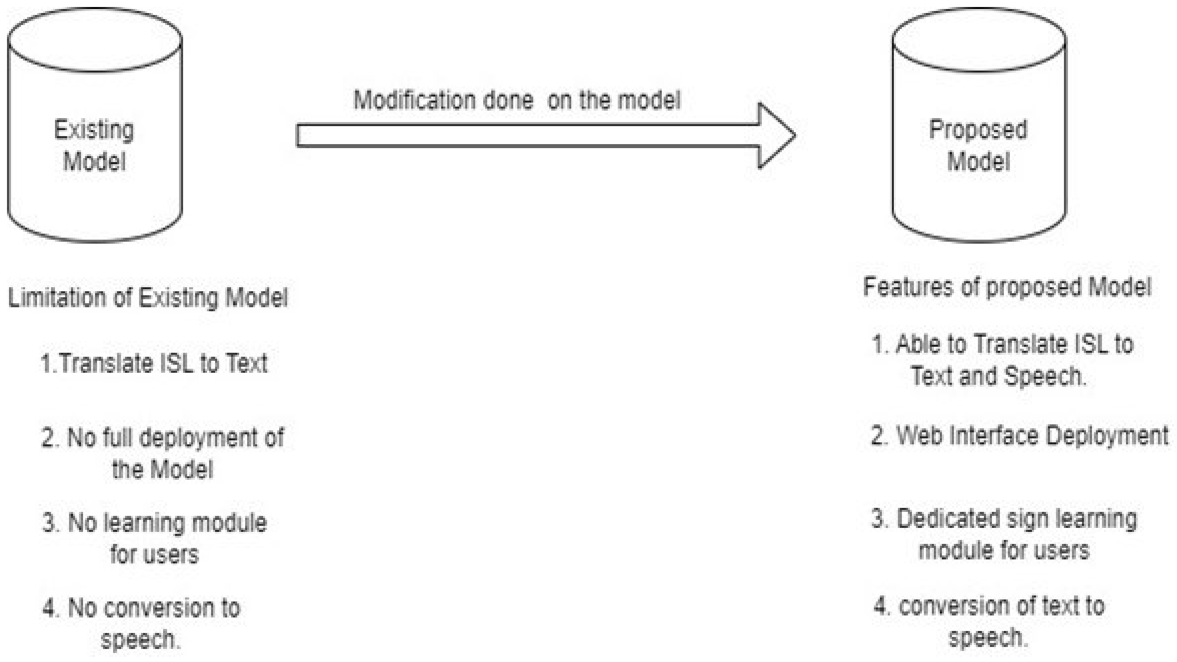

Existing Model
Modification done on the model
Proposed Model
Limitation of Existing Model
1.Translate ISL to Text
2. No full deployment of the Model
3. No learning module for users
4. No conversion to speech.
Features of proposed Model
1. Able to Translate ISL to Text and Speech.
2. Web Interface Deployment
3. Dedicated sign learning module for users
4. conversion of text to speech.

FIGURE 1: OVERCOME LIMITATION OF EXISTING MODEL

3.5 FEASIBILITY STUDY

3.5.1 TECHNICAL FEASIBILITY

Technical feasibility assesses the capability to design, develop, and implement the proposed real-time translation system for Indian Sign Language (ISL) into text and speech using current technologies and resources.

Gesture Recognition Technology: The system will utilize advanced machine learning algorithms, particularly deep learning models such as convolutional neural networks (CNNs) for spatial feature extraction and long short-term memory networks (LSTMs) for handling temporal dependencies in gesture sequences. These models are proven effective for visual recognition tasks. The availability of open-source machine learning frameworks like TensorFlow, PyTorch, and Keras simplifies the implementation process. Moreover, existing research in sign language recognition can be leveraged to fine-tune models specifically for ISL gestures.

Hardware Requirements: The system's hardware requirements are minimal, primarily needing a standard webcam for capturing gestures. Most modern consumer devices, including laptops, desktops, and smartphones, are equipped with webcams and sufficient processing power to run the required software. This ensures that the system can be widely accessible without necessitating specialized hardware investments.

Software and Tools: The development of the system's web interface using HTML, CSS, and JavaScript is technically feasible and cost-effective. Frameworks like React.js provide robust tools for building dynamic, user-friendly interfaces. For gesture recognition and translation, Python-based libraries and tools can be utilized for model training and deployment. The text-to-speech (TTS) synthesis can be integrated using high-quality APIs like Google Text-to-Speech or open-source alternatives like Festival and eSpeak, ensuring natural and accurate speech output.

Data Availability: Although annotated ISL datasets are limited, recent initiatives and collaborations with institutions specializing in ISL can provide the necessary training data. Data augmentation techniques can expand these datasets by generating variations of existing gestures, enhancing the model's robustness. Additionally, transfer learning from models trained on other sign languages can improve recognition accuracy for ISL.

Scalability and Adaptability: The system is designed to be scalable and adaptable, allowing for continuous updates and improvements. Cloud-based solutions can handle large-scale data processing and storage, ensuring that the system remains responsive and efficient as the user base grows.

3.5.2 ECONOMICAL FEASIBILITY

Economical feasibility evaluates the cost-effectiveness of the project, ensuring that the benefits justify the investment.

Development Costs: The primary costs involve hiring software developers, machine learning experts, and UI/UX designers. Utilizing open-source tools and libraries can significantly reduce software development costs. Collaboration with academic institutions and leveraging existing research can also minimize expenses related to model development and training.

Hardware Costs: The system's reliance on standard consumer hardware, such as webcams and personal computers, means that additional hardware costs are minimal. Most users will not need to purchase new devices, making the system economically accessible.

Operational Costs: Once developed, the system incurs minimal operational costs. Cloud-based platforms like AWS, Google Cloud, or Microsoft Azure offer scalable processing and storage solutions on a pay-as-you-go basis, ensuring cost efficiency. Maintenance costs will include regular updates and potential server fees, which can be managed within a reasonable budget.

Benefit Analysis: The system's benefits include enhanced communication for ISL users, increased accessibility, and inclusivity in various settings, including education, employment, and social interactions. By reducing communication barriers, the system can lead to improved social integration and opportunities for ISL users, thereby justifying the investment.
Funding Opportunities: Various grants and funding sources are available from governmental and non-governmental organizations dedicated to disability inclusion, technological innovation, and education. Securing such funding can offset development and operational costs, making the project more economically feasible.

3.5.3 OPERATIONAL FEASIBILITY

Operational feasibility examines the practicality of implementing and operating the system in real-world scenarios, considering user acceptance and support.

User Acceptance: Ensuring user acceptance involves creating an intuitive and user-friendly interface. By involving end-users in the development process through surveys, focus groups, and beta testing, the system can be tailored to meet their needs and preferences. Incorporating user feedback will help refine the interface, making it accessible and easy to navigate for ISL users of varying technical proficiency.

Training and Support: The system will require minimal training for users, thanks to its intuitive design. Comprehensive user guides, video tutorials, and an online helpdesk can support users in learning how to use the system effectively. Regular training sessions and workshops can also be organized for institutions and organizations adopting the system.

Integration with Existing Systems: The system can be integrated with widely used communication tools such as Zoom, Microsoft Teams, Google Meet, and messaging platforms like WhatsApp and Slack. APIs and plugins can be developed to facilitate this integration, ensuring that ISL users can seamlessly communicate across different platforms and contexts.

Maintenance and Updates: Regular maintenance is crucial for the system's ongoing effectiveness and relevance. A dedicated team will manage updates, incorporating new gestures, improving recognition accuracy, and enhancing the user interface based on feedback and technological advancements. This will ensure the system remains up-to-date and continues to meet user needs.

Scalability: The system is designed to handle a growing number of users without compromising performance. Cloud-based infrastructure will ensure efficient scaling, allowing the system to manage increased loads and user data effectively. This scalability ensures that the system can expand its reach and impact over time.

In summary, the feasibility study indicates that the proposed real-time ISL translation system is technically viable, economically justified, and operationally practical. By leveraging advanced technologies, minimizing costs, and ensuring user-centric design, the system promises to enhance communication and inclusivity for ISL users, addressing the limitations of existing solutions.

CHAPTER-4

SYSTEM DESIGN AND DEVELOPMENT

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. It encompasses both high-level architectural decisions and low-level detailed designs. In the context of the Indian Sign Language (ISL) translation system, system design involves creating a framework that can accurately interpret sign language gestures, translate them into text, and then convert the text into audible speech. Here's how this process is approached in the project:

Conceptual Framework:

Understanding User Needs: The first step involves understanding the requirements and needs of the users, who are individuals communicating through sign language.Defining System Objectives: Clarify the objectives of the ISL translation system, which may include real-time translation, accuracy, and ease of use.Researching Existing Solutions: Investigate existing solutions and technologies for sign language recognition, translation, and speech synthesis to inform design decisions.Technical Implementation Details:Component Identification: Identify the key components required for the system, such as input sensors (e.g., cameras), gesture recognition algorithms, translation models, and speech synthesis engines.Architectural Design: Design the overall architecture of the system, including how different components interact and communicate with each other. This may involve a modular approach to facilitate scalability and maintenance.Algorithm Selection: Choose appropriate algorithms for sign language recognition and translation based on factors such as accuracy, speed, and computational resources.Data Flow Design: Define how data flows through the system, from input (sign language gestures) to output (text and speech), and design data structures and processing pipelines accordingly.Interface Design: Design user interfaces for interacting with the system, considering accessibility and ease of use for individuals communicating through sign language.Testing and Iteration: Continuously test and refine the system design through iterative development cycles, incorporating user feedback and improving performance and accuracy.Integration and Deployment:Integration of Components: Integrate individual components into a cohesive system, ensuring compatibility and interoperability between different modules.Deployment Strategy: Plan for deploying the ISL translation system in real-world settings, considering factors such as hardware requirements, software dependencies, and user training. Scalability and Maintenance: Design the system with scalability in mind to accommodate future growth and updates. Implement maintenance procedures to address issues and update software as needed.

Development:

Gesture Classification :

Convolutional Neural Network (CNN)

CNN is a class of neural networks that are highly useful in solving computer vision problems. They found inspiration from the actual perception of vision that takes place in the visual cortex of our brain. They make use of a filter/kernel to scan through the entire pixel values of the image and make computations by setting appropriate weights to enable detection of a specific feature. CNN is equipped with layers like convolution layer, max pooling layer, flatten layer, dense layer, dropout layer and a fully connected neural network layer. These layers together make a very powerful tool that can identify features in an image. The starting layers detect low level features that gradually begin to detect more complex higher-level featuresUnlike regular Neural Networks, in the layers of CNN, the neurons are arranged in 3 dimensions: width, height, depth.The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all of the neurons in a fully-connected manner.Moreover, the final output layer would have dimensions(number of classes), because by the end of the CNN architecture we will reduce the full image into a single vector of class scores.CNN**Convolutional Layer:**In convolution layer I have taken a small window size [typically of length 5*5] that extends to the depth of the input matrix.The layer consists of learnable filters of window size. During every iteration I slid the window by stride size [typically 1], and compute the dot product of filter entries and input values at a given position.As I continue this process well create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position.That is, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of color.

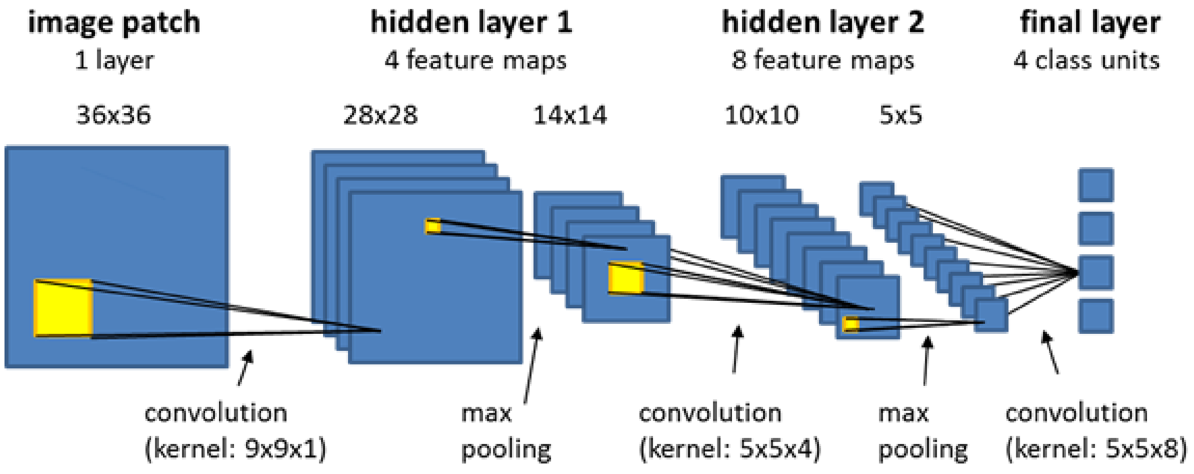


FIGURE 2

Pooling Layer:

We use pooling layer to decrease the size of activation matrix and ultimately reduce the learnable parameters.

There are two types of pooling:

a. Max Pooling:

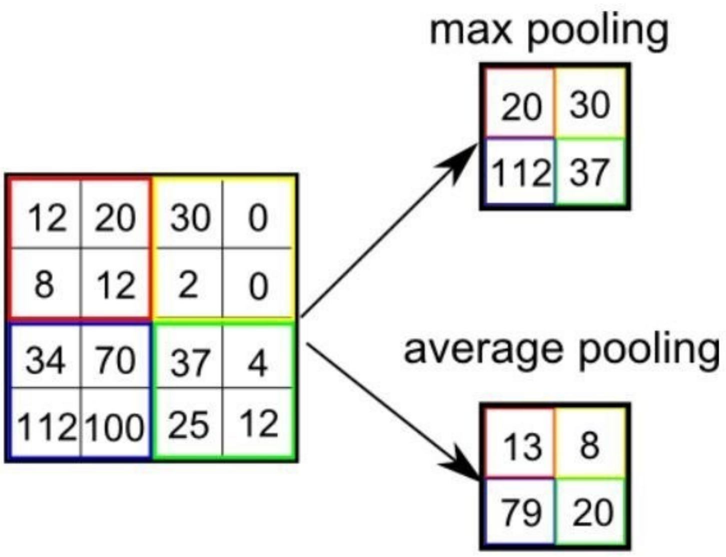
In max pooling we take a window size [for example window of size 2*2], and only taken the maximum of 4 values.

Well lid this window and continue this process, so well finally get an activation matrix half of its original Size.

b. Average Pooling:

In average pooling we take average of all Values in a window.

Pooling



Fully Connected Layer:

In convolution layer neurons are connected only to a local region, while in a fully connected region, well connect the all the inputs to neurons.

Fully Connected Layer

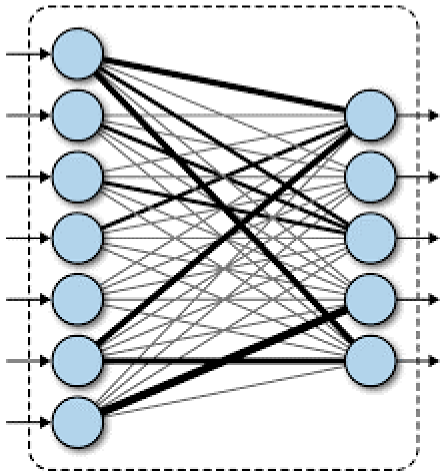


FIGURE-3

The pre-processed 180 images/alphabet will feed the keras CNN model.

4.1 HIGH-LEVEL DESIGN (ARCHITECTURAL)

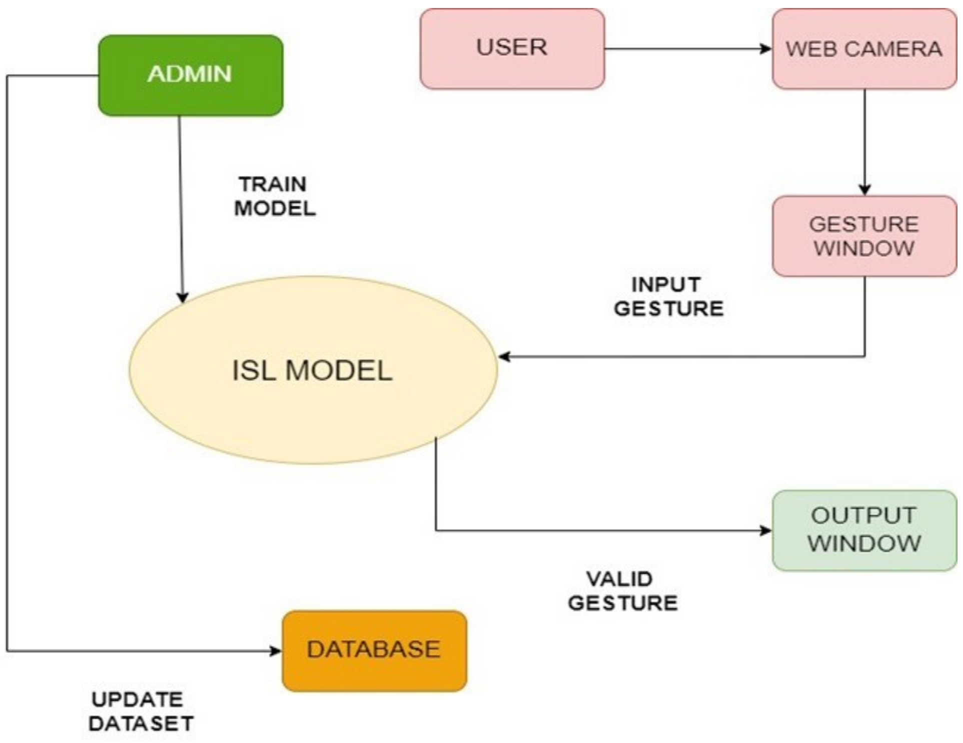


FIGURE- 4

The FIGURE-4 represents a system for recognizing and interpreting Indian Sign Language (ISL) gestures. The process begins with the user performing gestures, which are captured by a web camera and displayed in a gesture window for visual feedback. These gestures are then processed by the ISL model, a trained machine learning model designed to recognize and interpret ISL gestures. If the gesture is valid, its interpretation is shown in the output window. The system includes a database that stores the dataset of gestures used for training the ISL model. An admin is responsible for updating this dataset and retraining the model to enhance its accuracy. This continuous cycle ensures the system remains effective in recognizing and interpreting gestures accurately.

4.2 LOW-LEVEL DESIGN

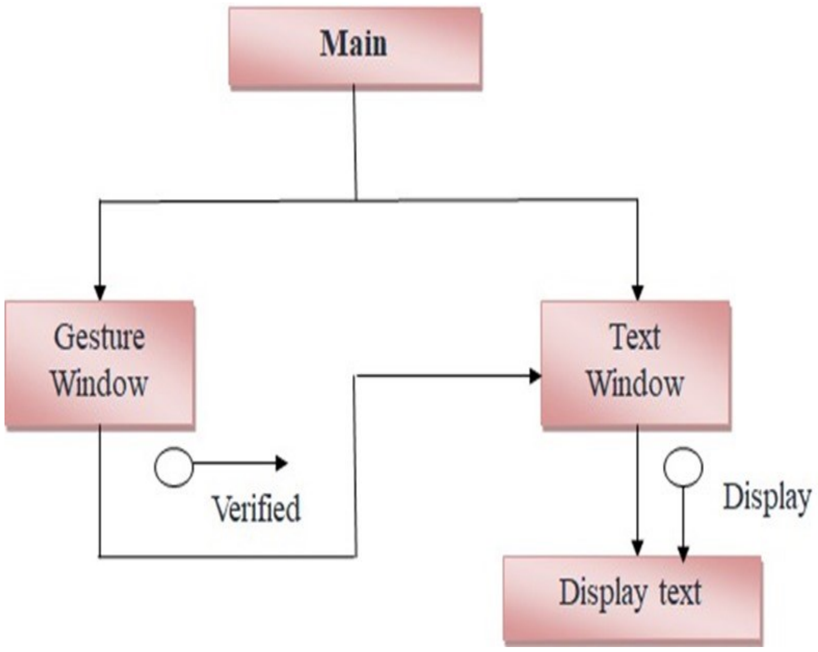


FIGURE-5

FIGURE-3 represents a system for converting gestures into text. The process begins with the "Main" control module, which orchestrates the overall workflow. Gestures are first captured and displayed in the "Gesture Window." Once the gestures are verified, they are passed to the "Text Window," where they are interpreted and converted into text. This text is then sent to the "Display Text" module, which shows the final interpreted text to the user. The system ensures

that captured gestures are accurately verified and translated into readable text, facilitating effective communication through gesture recognition.

4.3 DATAFLOW DIAGRAM

There are four components in DFD:

External Entity.

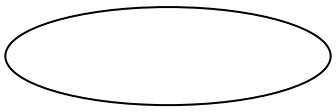
Process.

Data Flow.

Data Store

External-Entity :- It is an external system that sends or receives data, communicating with the system. Losing access to a system is a source of information and a destination. They can be external entities or individuals, computer systems or business systems. They are known as Terminator, Source and Sync or Actor. They are usually drawn on the edge of the figure. These are the sources and targets of the system's inputs and outputs.

Process: - It is just like a function that changes the data, producing an output. It might perform computations for sort data based on logic or direct the dataflow based on business rules.



Data Flow: - A dataflow represents a package of information flowing between two objects in the data-flow diagram, Data flows
are used to model the flow of information into the system, out of the system and between the elements within the system.

Data Store: - These are the files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label.

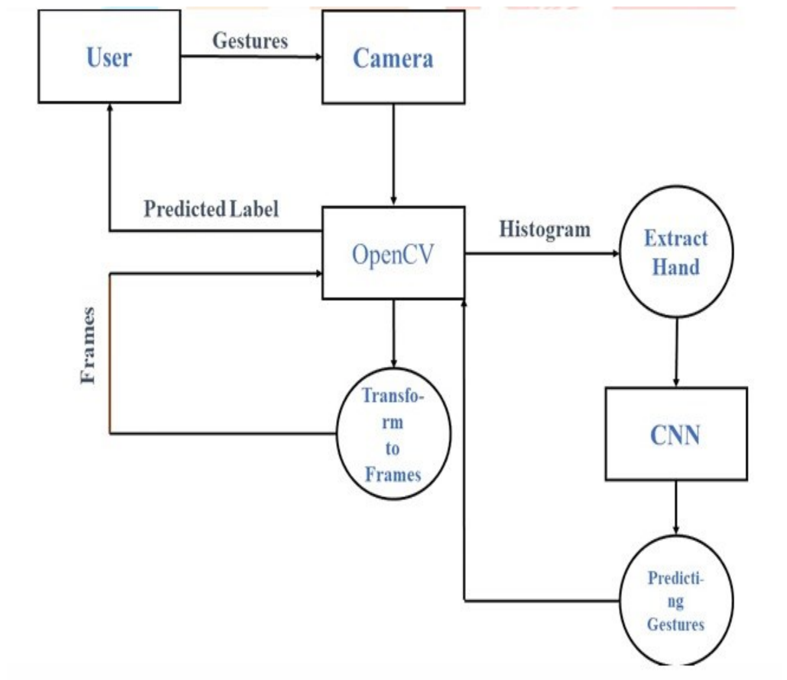


FIGURE-6

4.4 USE CASE DIAGRAM

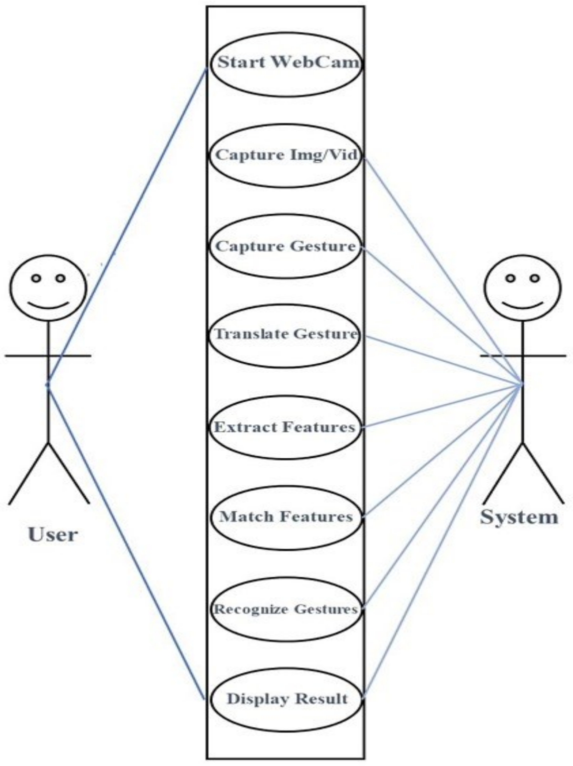
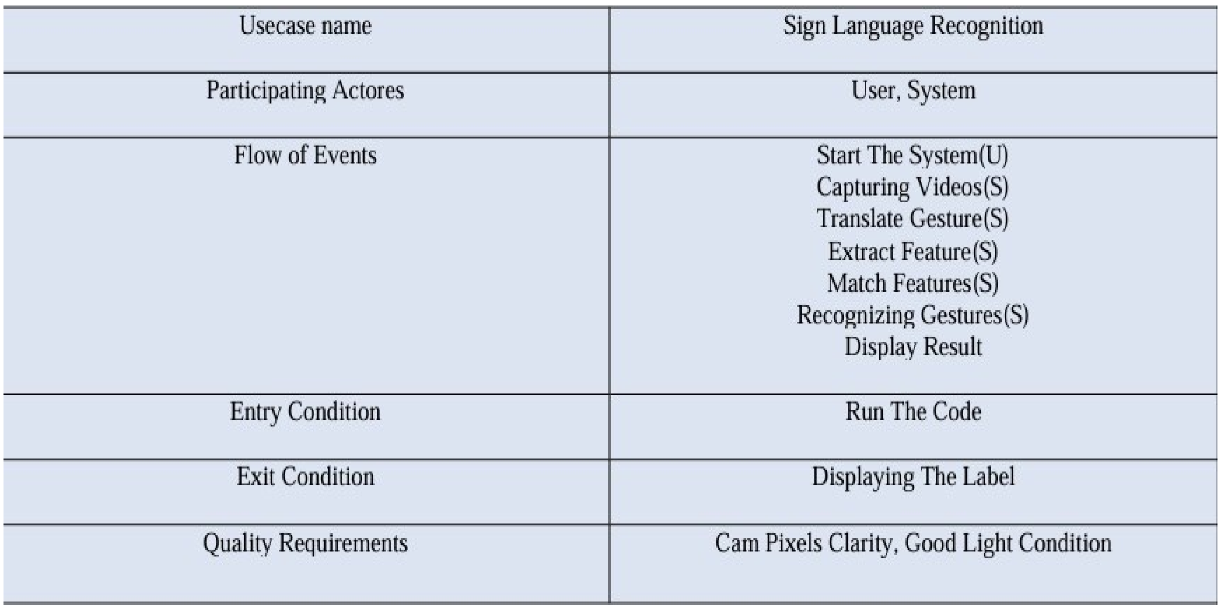


FIGURE-7

The FIGURE-7 illustrates a gesture recognition system involving interaction between a user and the system. The process starts with the user, who performs gestures captured by a web camera (Start WebCam). The camera captures images or videos (Capture Img/Vid) and identifies specific gestures within them (Capture Gesture). These gestures are then translated (Translate Gesture) and analyzed to extract relevant features (Extract Features). The system matches these features against a database of known gestures (Match Features) to recognize the performed gestures (Recognize Gestures). Finally, the recognized gesture is displayed to the user (Display Result), completing the cycle. This system allows seamless translation of physical gestures into digital outputs, enabling effective communication between the user and the system.

4.5 Use Case Scenario for Sign Language Recognition System



Usecase name	Sign Language Recognition
Participating Actores	User, System
Flow of Events	Start The System(U) Capturing Videos(S) Translate Gesture(S) Extract Feature(S) Match Features(S) Recognizing Gestures(S) Display Result
Entry Condition	Run The Code
Exit Condition	Displaying The Label
Quality Requirements	Cam Pixels Clarity, Good Light Condition

4.6 SEQUENCE DIAGRAM

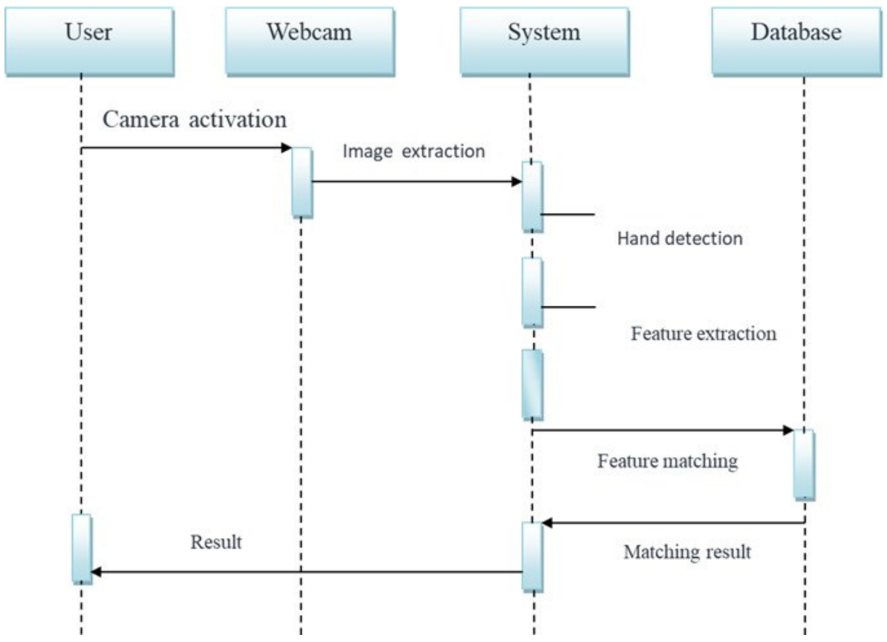


FIGURE-8

The FIGURE-8 represents the workflow of a gesture recognition system, detailing the interaction between the user, webcam, system, and database. The process begins with the user initiating the camera activation. The webcam captures images (image extraction) and sends them to the system. The system first performs hand detection to isolate the hand gestures from the captured images. Following this, it extracts features from the detected hand gestures (feature extraction). These extracted features are then matched against a database of known gesture features (feature matching). The database returns the matching result to the system, which interprets the recognized gesture. Finally, the system sends the result back to the user. This workflow ensures accurate detection, interpretation, and feedback of hand gestures.

4.7 ACTIVITY DIAGRAM

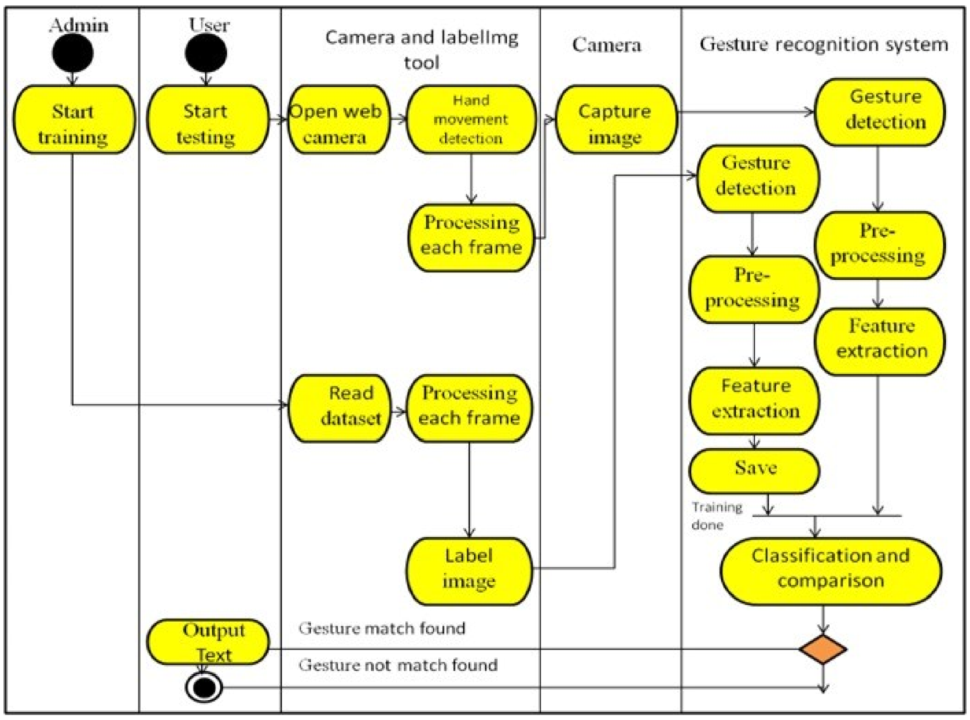


FIGURE-9

The FIGURE-9 outlines a comprehensive workflow for a gesture recognition system, illustrating the roles of both the admin and user, along with the processes involving camera tools and the gesture recognition system itself. The admin initiates the process by starting the training phase, while the user begins by starting the testing phase. The user then opens the web camera, which detects hand movements and processes each frame. Simultaneously, the system reads from the dataset and processes each frame to label images.

The camera captures images that are fed into the gesture recognition system. Within this system, gesture detection occurs, followed by pre-processing and feature extraction. These features are then saved for future reference. The classification and comparison phase ensues, where the system determines if a gesture match is found or not. If a match is found, the corresponding text output is displayed. This workflow ensures that gestures are accurately captured, processed, and recognized, providing real-time feedback to the user.

CHAPTER-5

CODING

5.1 PSEUDO CODE

```
import math
import cv2
from cvzone.HandTrackingModule import HandDetector
import numpy as np
from keras.models import load_model
import traceback

model = load_model('A:\desktop\progit\Sign-Language-To-Text-and-Speech-
Conversion\cnn8grps_rad1_model.h5')
white = np.ones((400, 400), np.uint8) * 255
cv2.imwrite("A:\desktop\progit\Sign-Language-To-Text-and-Speech-Conversion\white.jpg, white")

capture = cv2.VideoCapture(0)

hd = HandDetector(maxHands=1)
hd2 = HandDetector(maxHands=1)

offset = 29
step = 1
flag = False
suv = 0


def distance(x, y):
        return math.sqrt(((x[0] - y[0]) ** 2) + ((x[1] - y[1]) ** 2))


def distance_3d(x, y):
        return math.sqrt(((x[0] - y[0]) ** 2) + ((x[1] - y[1]) ** 2) + ((x[2] - y[2]) ** 2))


bfh = 0
dicttt=dict()
count=0
kok=[]

while True:
try:
_, frame = capture.read()
frame = cv2.flip(frame, 1)
hands = hd.findHands(frame, draw=False, flipType=True)
```

```
print(frame.shape)
if hands:
# #print(" --------- lmlist=",hands[1])
hand = hands[0]
x, y, w, h = hand['bbox']
image = frame[y - offset:y + h + offset, x - offset:x + w + offset]
white = cv2.imread("C:\\Users\\devansh raval\\PycharmProjects\\pythonProject\\white.jpg")
# img_final=img_final1=img_final2=0
handz = hd2.findHands(image, draw=False, flipType=True)
if handz:
hand = handz[0]
pts = hand['lmList']
# x1,y1,w1,h1=hand['bbox']

os = ((400 - w) // 2) - 15
os1 = ((400 - h) // 2) - 15
for t in range(0, 4, 1):
cv2.line(white, (pts[t][0] + os, pts[t][1] + os1), (pts[t + 1][0] + os, pts[t + 1][1] + os1),(0,
255, 0), 3)
for t in range(5, 8, 1):
cv2.line(white, (pts[t][0] + os, pts[t][1] + os1), (pts[t + 1][0] + os, pts[t + 1][1] + os1),
(0, 255, 0), 3)
for t in range(9, 12, 1):
cv2.line(white, (pts[t][0] + os, pts[t][1] + os1), (pts[t + 1][0] + os, pts[t + 1][1] + os1),
(0, 255, 0), 3)
for t in range(13, 16, 1):
cv2.line(white, (pts[t][0] + os, pts[t][1] + os1), (pts[t + 1][0] + os, pts[t + 1][1] + os1),
(0, 255, 0), 3)
for t in range(17, 20, 1):
cv2.line(white, (pts[t][0] + os, pts[t][1] + os1), (pts[t + 1][0] + os, pts[t + 1][1] + os1),
(0, 255, 0), 3)
cv2.line(white, (pts[5][0] + os, pts[5][1] + os1), (pts[9][0] + os, pts[9][1] + os1), (0, 255,
0),
3)
cv2.line(white, (pts[9][0] + os, pts[9][1] + os1), (pts[13][0] + os, pts[13][1] + os1), (0, 255,
0),
3)
cv2.line(white, (pts[13][0] + os, pts[13][1] + os1), (pts[17][0] + os, pts[17][1] + os1),
(0, 255, 0), 3)
cv2.line(white, (pts[0][0] + os, pts[0][1] + os1), (pts[5][0] + os, pts[5][1] + os1), (0, 255,
0),
3)
cv2.line(white, (pts[0][0] + os, pts[0][1] + os1), (pts[17][0] + os, pts[17][1] + os1), (0, 255,
0),
3)

for i in range(21):
cv2.circle(white, (pts[i][0] + os, pts[i][1] + os1), 2, (0, 0, 255), 1)
```

```
cv2.imshow("2", white)
# cv2.imshow("5", skeleton5)

# #print(model.predict(img))
white = white.reshape(1, 400, 400, 3)
prob = np.array(model.predict(white)[0], dtype='float32')
ch1 = np.argmax(prob, axis=0)
prob[ch1] = 0
ch2 = np.argmax(prob, axis=0)
prob[ch2] = 0
ch3 = np.argmax(prob, axis=0)
prob[ch3] = 0


pl = [ch1, ch2]

#condition for [Aemnst]
l=[[5,2],[5,3],[3,5],[3,6],[3,0],[3,2],[6,4],[6,1],[6,2],[6,6],[6,7],[6,0],[6,5],[4,1],[1,0],[1,
1],[6,3],[1,6],[5,6],[5,1],[4,5],[1,4],[1,5],[2,0],[2,6],[4,6],[1,0],[5,7],[1,6],[6,1],[7,6],[2,
5],[7,1],[5,4],[7,0],[7,5],[7,2]]
if pl in l:
if (pts[6][1] < pts[8][1] and pts[10][1] < pts[12][1] and pts[14][1] < pts[16][1] and pts[18][1]
<pts[20][1]):
ch1=0
#print("00000")

#condition for [o][s]
l=[[2,2],[2,1]]
if pl in l:
if (pts[5][0] < pts[4][0] ):
ch1=0
print("++++++++++++++++++")
#print("00000")



#condition for [c0][aemnst]
l=[[0,0],[0,6],[0,2],[0,5],[0,1],[0,7],[5,2],[7,6],[7,1]]
pl=[ch1,ch2]
if pl in l:
if (pts[0][0]>pts[8][0] and pts[0][0]>pts[4][0] and pts[0][0]>pts[12][0] and
pts[0][0]>pts[16][0] and pts[0][0]>pts[20][0]) and pts[5][0] > pts[4][0]:
ch1=2
#print("22222")

# condition for [c0][aemnst]
l = [[6,0],[6,6],[6,2]]
```

```
pl = [ch1, ch2]
if pl in l:
if distance(pts[8],pts[16])<52:
ch1 = 2
#print("22222")


##print(pts[2][1]+15>pts[16][1])
# condition for [gh][bdfikruvw]
l = [[1,4],[1,5],[1,6],[1,3],[1,0]]
pl = [ch1, ch2]

if pl in l:
if pts[6][1] > pts[8][1] and pts[14][1] < pts[16][1] and pts[18][1]<pts[20][1] and
pts[0][0]<pts[8][0] and pts[0][0]<pts[12][0] and pts[0][0]<pts[16][0] and pts[0][0]<pts[20][0]:
ch1 = 3
print("33333c")


#con for [gh][l]
l=[[4,6],[4,1],[4,5],[4,3],[4,7]]
pl=[ch1,ch2]
if pl in l:
if pts[4][0]>pts[0][0]:
ch1=3
print("33333b")

# con for [gh][pqz]
l = [[5, 3],[5,0],[5,7], [5, 4], [5, 2],[5,1],[5,5]]
pl = [ch1, ch2]
if pl in l:
if pts[2][1]+15<pts[16][1]:
ch1 = 3
print("33333a")

# con for [l][x]
l = [[6, 4], [6, 1], [6, 2]]
pl = [ch1, ch2]
if pl in l:
if distance(pts[4],pts[11])>55:
ch1 = 4
#print("44444")

# con for [l][d]
l = [[1, 4], [1, 6],[1,1]]
pl = [ch1, ch2]
if pl in l:
```

```
if (distance(pts[4], pts[11]) > 50) and (pts[6][1] > pts[8][1] and pts[10][1] < pts[12][1] and
pts[14][1] < pts[16][1] and pts[18][1] <pts[20][1]):
ch1 = 4
#print("44444")

# con for [1][gh]
l = [[3, 6], [3, 4]]
pl = [ch1, ch2]
if pl in l:
if (pts[4][0]<pts[0][0]):
ch1 = 4
#print("44444")

# con for [1][c0]
l = [[2, 2], [2, 5],[2,4]]
pl = [ch1, ch2]
if pl in l:
if (pts[1][0] < pts[12][0]):
ch1 = 4
#print("44444")

# con for [1][c0]
l = [[2, 2], [2, 5], [2, 4]]
pl = [ch1, ch2]
if pl in l:
if (pts[1][0] < pts[12][0]):
ch1 = 4
#print("44444")

# con for [gh][z]
l = [[3, 6],[3,5],[3,4]]
pl = [ch1, ch2]
if pl in l:
if (pts[6][1] > pts[8][1] and pts[10][1] < pts[12][1] and pts[14][1] < pts[16][1] and pts[18][1]
<pts[20][1]) and pts[4][1]>pts[10][1]:
ch1 = 5
print("55555b")



# con for [gh][pq]
l = [[3,2],[3,1],[3,6]]
pl = [ch1, ch2]
if pl in l:
if pts[4][1]+17>pts[8][1] and pts[4][1]+17>pts[12][1] and pts[4][1]+17>pts[16][1] and
pts[4][1]+17>pts[20][1]:
ch1 = 5
print("55555a")
```

```
# con for [l][pqz]
l = [[4,4],[4,5],[4,2],[7,5],[7,6],[7,0]]
pl = [ch1, ch2]
if pl in l:
if pts[4][0]>pts[0][0]:
ch1 = 5
#print("55555")

# con for [pqz][aemnst]
l = [[0, 2],[0,6],[0,1],[0,5],[0,0],[0,7],[0,4],[0,3],[2,7]]
pl = [ch1, ch2]
if pl in l:
if pts[0][0]<pts[8][0]  and  pts[0][0]<pts[12][0]  and pts[0][0]<pts[16][0]  and
pts[0][0]<pts[20][0]:
ch1 = 5
#print("55555")



# con for [pqz][yj]
l = [[5, 7],[5,2],[5,6]]
pl = [ch1, ch2]
if pl in l:
if pts[3][0]<pts[0][0]:
ch1 = 7
#print("77777")

# con for [l][yj]
l = [[4, 6],[4,2],[4,4],[4,1],[4,5],[4,7]]
pl = [ch1, ch2]
if pl in l:
if pts[6][1] < pts[8][1]:
ch1 = 7
#print("77777")

# con for [x][yj]
l = [[6, 7],[0,7],[0,1],[0,0],[6,4],[6,6] ,[6,5],[6,1]]
pl = [ch1, ch2]
if pl in l:
if pts[18][1] > pts[20][1]:
ch1 = 7
#print("77777")


# condition for [x][aemnst]
l = [[0,4],[0,2],[0,3],[0,1],[0,6]]
pl = [ch1, ch2]
```

```
if pl in l:
if pts[5][0]>pts[16][0]:
ch1 = 6
#print("66666")

# condition for [yj][x]
l = [[7, 2]]
pl = [ch1, ch2]
if pl in l:
if pts[18][1] < pts[20][1]:
ch1 = 6
#print("66666")


# condition for [c0][x]
l = [[2, 1],[2,2],[2,6],[2,7],[2,0]]
pl = [ch1, ch2]
if pl in l:
if distance(pts[8],pts[16])>50:
ch1 = 6
#print("66666")

# con for [l][x]

l = [[4, 6],[4,2],[4,1],[4,4]]
pl = [ch1, ch2]
if pl in l:
if distance(pts[4], pts[11]) < 60:
ch1 = 6
#print("66666")

#con for [x][d]
l = [[1,4],[1,6],[1,0],[1,2]]
pl = [ch1, ch2]
if pl in l:
if pts[5][0] - pts[4][0] - 15 > 0:
ch1 = 6


# con for [b][pqz]
l =
[[5,0],[5,1],[5,4],[5,5],[5,6],[6,1],[7,6],[0,2],[7,1],[7,4],[6,6],[7,2],[5,0],[6,3],[6,4],[7,5]
,[7,2]]
pl = [ch1, ch2]
if pl in l:
if (pts[6][1] > pts[8][1] and pts[10][1] > pts[12][1] and pts[14][1] > pts[16][1] and pts[18][1]
> pts[20][1]):
ch1 = 1
```

```
print("111111")


# con for [f][pqz]
l = [[6, 1],[6,0],[0,3],[6,4],[2,2], [0,6],[6,2],[7, 6],[4,6],[4,1],[4,2], [0, 2], [7, 1], [7,
4], [6, 6], [7, 2], [7, 5], [7, 2]]
pl = [ch1, ch2]
if pl in l:
if (pts[6][1] < pts[8][1] and pts[10][1] > pts[12][1] and pts[14][1] > pts[16][1] and
pts[18][1] > pts[20][1]):
ch1 = 1
print("111112")

l = [[6, 1], [6, 0],[4,2],[4,1],[4,6],[4,4]]
pl = [ch1, ch2]
if pl in l:
if (pts[10][1] > pts[12][1] and pts[14][1] > pts[16][1] and
pts[18][1] > pts[20][1]):
ch1 = 1
print("111112")

# con for [d][pqz]
fg=19
#print("_________________ch1=",ch1," ch2=",ch2)
l = [[5,0],[3,4],[3,0],[3,1],[3,5],[5,5],[5,4],[5,1],[7,6]]
pl = [ch1, ch2]
if pl in l:
if ((pts[6][1] > pts[8][1] and pts[10][1] < pts[12][1] and pts[14][1] < pts[16][1] and
pts[18][1] < pts[20][1]) and (pts[2][0]<pts[0][0]) and pts[4][1]>pts[14][1]):
ch1 = 1
print("111113")

l = [ [4, 1], [4, 2],[4, 4]]
pl = [ch1, ch2]
if pl in l:
if (distance(pts[4], pts[11]) < 50) and (pts[6][1] > pts[8][1] and pts[10][1] < pts[12][1] and
pts[14][1] < pts[16][1] and pts[18][1] < pts[20][1]):
ch1 = 1
print("1111993")



l = [[3, 4], [3, 0], [3, 1], [3, 5],[3,6]]
pl = [ch1, ch2]
if pl in l:
if ((pts[6][1] > pts[8][1] and pts[10][1] < pts[12][1] and pts[14][1] < pts[16][1] and
pts[18][1] < pts[20][1]) and (pts[2][0] < pts[0][0]) and pts[14][1]<pts[4][1]):
ch1 = 1
```

```
print("1111mmm3")

l = [[6, 6],[6, 4], [6, 1],[6,2]]
pl = [ch1, ch2]
if pl in l:
if pts[5][0]-pts[4][0]-15<0:
ch1 = 1
print("1111140")



# con for [i][pqz]
l = [[5,4],[5,5],[5,1],[0,3],[0,7],[5,0],[0,2],[6,2],[7, 5], [7, 1], [7, 6], [7, 7]]
pl = [ch1, ch2]
if pl in l:
if ((pts[6][1] < pts[8][1] and pts[10][1] < pts[12][1] and pts[14][1] < pts[16][1] and
pts[18][1] > pts[20][1])):
ch1 = 1
print("111114")

# con for [yj][bfdi]
l = [[1,5],[1,7],[1,1],[1,6],[1,3],[1,0]]
pl = [ch1, ch2]
if pl in l:
if (pts[4][0]<pts[5][0]+15) and ((pts[6][1] < pts[8][1] and pts[10][1] < pts[12][1] and
pts[14][1] < pts[16][1] and
pts[18][1] > pts[20][1])):
ch1 = 7
print("111114lll;;p")

#con for [uvr]
l = [[5,5],[5,0],[5,4],[5,1],[4,6],[4,1],[7,6],[3,0],[3,5]]
pl = [ch1, ch2]
if pl in l:
if ((pts[6][1] > pts[8][1] and pts[10][1] > pts[12][1] and pts[14][1] < pts[16][1] and
pts[18][1] < pts[20][1])) and pts[4][1]>pts[14][1]:
ch1 = 1
print("111115")



# con for [w]
fg=13
l = [[3,5],[3,0],[3,6],[5,1],[4,1],[2,0],[5,0],[5,5]]
pl = [ch1, ch2]
if pl in l:
```

```
if not(pts[0][0]+fg < pts[8][0] and pts[0][0]+fg < pts[12][0] and pts[0][0]+fg < pts[16][0]  and
pts[0][0]+fg < pts[20][0]) and not(pts[0][0] > pts[8][0] and pts[0][0] > pts[12][0] and
pts[0][0] > pts[16][0]  and pts[0][0] > pts[20][0]) and distance(pts[4], pts[11]) < 50:
ch1 = 1
print("111116")

# con for [w]

l = [ [5, 0], [5, 5],[0,1]]
pl = [ch1, ch2]
if pl in l:
if pts[6][1]>pts[8][1] and pts[10][1]>pts[12][1] and pts[14][1]>pts[16][1]:
ch1 = 1
print("1117")
#-------------------------condn for 8 groups  ends
frame = cv2.putText(frame, "Predicted " + str(ch1), (30, 80),
cv2.FONT_HERSHEY_SIMPLEX,
3, (0, 0, 255), 2, cv2.LINE_AA)

cv2.imshow("frame", frame)
interrupt = cv2.waitKey(1)
if interrupt & 0xFF == 27:
# esc key
break
except Exception:
print("==", traceback.format_exc())
dicttt = {key: val for key, val in sorted(dicttt.items(), key = lambda ele: ele[1], reverse =
True)}
print(dicttt)
print(set(kok))
capture.release()
cv2.destroyAllWindows()
```

CHAPTER-6

SOFTWARE TESTING (Test cases)

Testing a gesture recognition system involves creating a variety of test cases to ensure the system performs accurately and reliably under different conditions. Here's a detailed list of potential test cases for the described model:

FUNCTIONAL TEST CASES

- Web Camera Activation
  - Test Case: Verify the camera activation when the system starts.
  - Expected Result: The camera should turn on and start capturing frames.
- Image/Video Capture
  - Test Case: Verify the camera captures images/videos correctly.
  - Expected Result: The camera should capture clear images/videos without lag or distortion.
- Gesture Detection
  - Test Case: Verify the system detects hand gestures in various lighting conditions.
  - Expected Result: The system should accurately detect hand gestures in different lighting conditions (bright, dim, natural light).
- Pre-processing of Images
  - Test Case: Verify the pre-processing of captured images (e.g., noise reduction, normalization).
  - Expected Result: Pre-processing should correctly prepare images for feature extraction without losing important gesture details.
- Feature Extraction

- Test Case: Verify the feature extraction process from the pre-processed images.
- Expected Result: The system should extract relevant features accurately from the gestures.

➢ Feature Matching

- Test Case: Verify the system matches extracted features with the stored dataset.
- Expected Result: The system should correctly match features with high accuracy.

➢ Classification and Comparison

- Test Case: Verify the classification and comparison process for detected gestures.
- Expected Result: The system should correctly classify and compare gestures to known gestures in the database.

➢ Text Output Display

- Test Case: Verify the output text display when a gesture match is found.
- Expected Result: The correct text corresponding to the gesture should be displayed.

PERFORMANCE TEST CASES

➢ System Response Time

- Test Case: Measure the time taken for the system to process a gesture and display the result.
- Expected Result: The system should process and display the result within an acceptable time frame (e.g., less than 2 seconds).

➢ Concurrent User Handling

- Test Case: Verify the system's performance with multiple users simultaneously.
- Expected Result: The system should handle multiple users without significant performance degradation.

USABILITY TEST CASES

➢ User Interface Ease of Use

- Test Case: Verify the ease of use of the user interface for starting tests and viewing results.
- Expected Result: The interface should be intuitive and easy to navigate for users of varying technical proficiency.

➢ Error Handling and Messages

- Test Case: Verify the system's error handling and user notifications for various issues (e.g., camera not available, gesture not recognized).
- Expected Result: The system should display clear and helpful error messages.

BOUNDARY TEST CASES

➢ Edge Case Gestures

- Test Case: Verify the system's recognition of gestures that are at the edge of the training dataset.
- Expected Result: The system should accurately recognize gestures that are similar but not exactly the same as those in the training dataset.

➢ Gesture Speed Variations

- Test Case: Verify the system's recognition accuracy with gestures performed at different speeds (slow, normal, fast).
- Expected Result: The system should accurately recognize gestures performed at varying speeds.

Test Case ID	Scenario	Boundary Value	Expected Result	Actual Result	Status
1	Used in normal environment	>90%	In normal environment, hand gestures should be recognized easily.	Hand gestures were recognized easily and worked properly.	Passed
2	Used in bright environment	>60%	In bright environment, software should work fine as it easily detects hand movements, but in more brighter conditions it may not detect the hand gestures as expected.	In bright conditions, the software worked very well.	Passed
3	Used at a near distance (15 cm) from the webcam	>80%	At this distance, the software should perform perfectly.	It worked fine and all features worked properly.	Passed
4	Used at a far distance (35 cm) from the webcam	>95%	At this distance, the software should work fine.	At this distance, it worked properly.	Passed
5	Used at a farther distance (60 cm) from the webcam	>60%	At this distance, there will be some problem in detecting hand gestures but it should work fine.	At this distance, the functions of the software worked properly.	Passed

CHAPTER-7

CONCLUSION AND SCOPE FOR FUTURE ENHANCEMENT

The Indian Sign Language (ISL) translation model significantly bridges the communication gap between hearing and deaf communities by translating ISL gestures into text in real time. Rigorous testing, including unit, acceptance, and implementation testing, ensured the model's robustness, accuracy, and reliability across various environments. Unit testing verified the functionality of video processing, gesture recognition, translation, and output rendering. Acceptance and implementation testing confirmed real-world applicability, usability, and performance. The model demonstrated high accuracy in different lighting conditions, distances, and rapid movements, ensuring versatile and reliable use. User feedback from acceptance testing highlighted the system’s intuitiveness and effectiveness, underscoring its potential to improve communication for the deaf community. The comprehensive testing process addressed potential issues, ensuring high performance and stability under challenging conditions. This model promotes inclusivity and empowers deaf individuals to participate more fully in society, offering a practical solution to a longstanding challenge. Its success is a testament to meticulous design, thorough testing, and a user-centric approach, providing a strong foundation for future technological enhancements.

In conclusion, the ISL translation model is a powerful tool for promoting inclusivity and enhancing communication between hearing and deaf communities. Its real-time translation capability offers a practical solution to a longstanding challenge, empowering deaf individuals to participate more fully in society. The project’s success is a testament to meticulous design, thorough testing, and a user-centric approach. As technology evolves, this model provides a strong foundation for future enhancements, ensuring it remains a vital resource for inclusive communication.

Scope for Future Enhancement

Despite the significant achievements, there are several areas where the ISL translation model can be further enhanced:

- Expansion of Vocabulary:
  Integrate a broader range of signs, including regional variations and more complex gestures, to enhance the model's comprehensiveness and versatility.

- Multilingual Support:
  Extend the translation capabilities to support multiple languages, allowing for ISL to be translated into various spoken languages, enhancing accessibility for a global audience.

- Contextual Understanding:
  Incorporate natural language processing techniques to improve the model's ability to understand and translate signs in context, ensuring more accurate and meaningful translations.

- Enhanced Accuracy in Diverse Conditions:
  Improve the model's robustness to handle extreme lighting conditions, occlusions, and background complexities more effectively, ensuring consistent performance across all environments.

- Integration with Assistive Technologies:
  Collaborate with existing assistive technologies such as screen readers and alternative input devices to provide a seamless experience for users with additional disabilities.

- User-Customizable Features:
  Develop customizable features that allow users to train the model with their unique signing styles and preferences, enhancing personalized user experience and accuracy.

- Real-time Feedback and Correction:
  Implement real-time feedback mechanisms that provide users with immediate corrections and suggestions, helping them improve their signing and ensuring more accurate translations.

- Mobile and Wearable Device Compatibility:
  Expand the model's compatibility to include mobile devices and wearable technology, making the translation services more portable and accessible in everyday situations.

- Data Privacy and Security:
  Enhance data privacy and security measures to protect user data, especially when using cloud-based services for processing and storage.

CHAPTER-8

BIBLIOGRAPHY

[1] T. Pan, L. Lo, C. Yeh, J. Li, H. Liu and M. Hu, "Real-Time Sign Language Recognition in Complex Background Scene Based on a HierarchicalClustering Classification Method," 2016 IEEE Second InternationalConference on Multimedia Big Data (BigMM), Taipei, Taiwan, 2016, pp.64-67, DOI: 10.1109/BigMM.2016.44.

[2] M. Kim, L. Chau and W. Siu, "Keyframe selection for motion capture using motion activity analysis," 2012 IEEE International Symposium on Circuits and Systems (ISCAS), Seoul, Korea (South), 2012, pp. 612-615, DOI: 10.1109/ISCAS.2012.6272106.

[3] S. C.J. and L. A., "Signet: A Deep Learning based Indian Sign Language Recognition System," 2019 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 2019, pp. 0596-0600, DOI: 10.1109/ICCSP.2019.8698006.

[4] Thad Starner and Alex Pentland, “Real-Time American Sign Language Recognition from Video using Hidden Markov Models”, 1995.

[5] Washef Ahmed, Kunal Chanda, Soma Mitra, “Vision-based Hand Gesture Recognition using Dynamic Time Warping for Indian Sign Language”, 2016.

[6]https://tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/install.html#install-the-object-detection-api

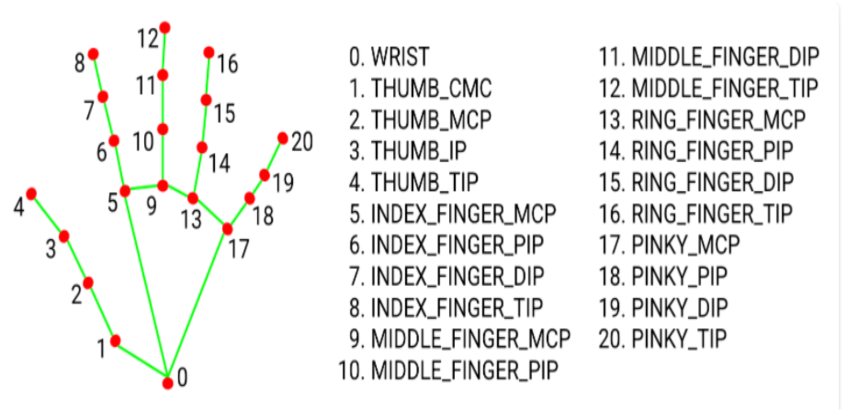
[7] https://opencv.org/

[8] https://en.wikipedia.org/wiki/TensorFlow

[9] https://en.wikipedia.org/wiki/Convolutional_neural_network

APPENDIX

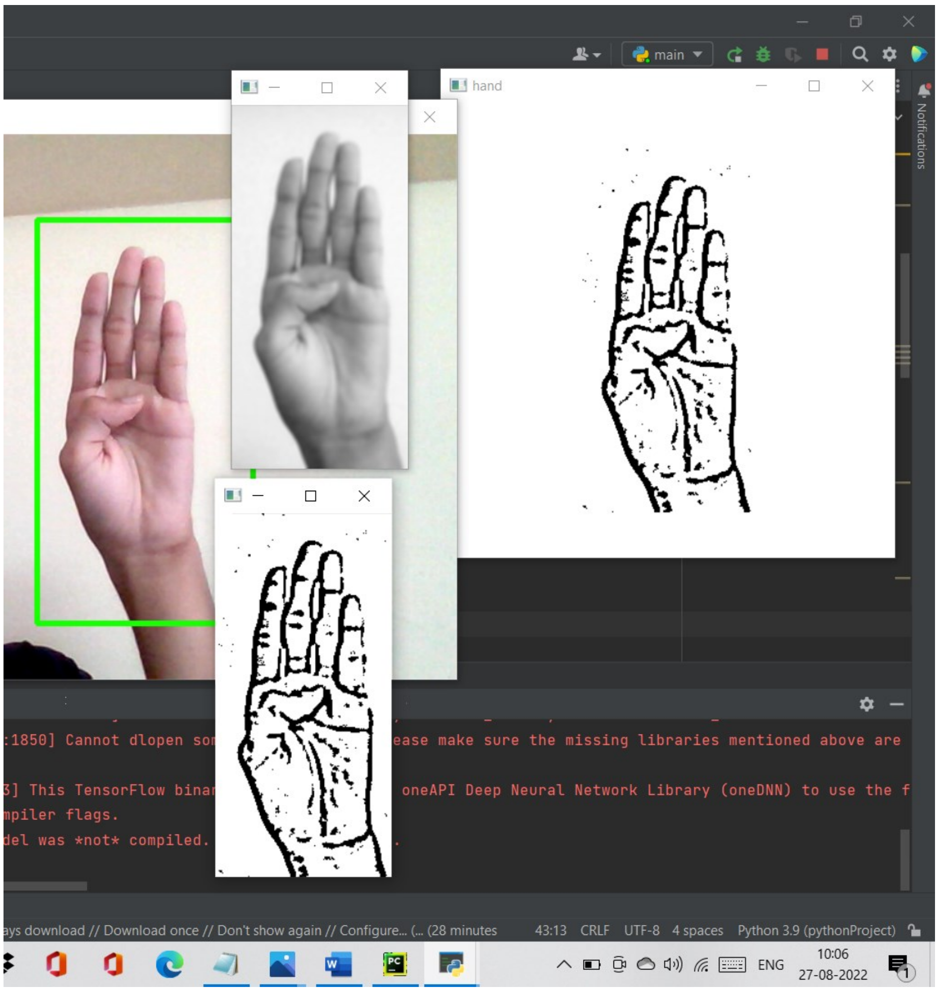
SNAPSHOTS



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 8- MEDIA PIPE LANDMARK SYSTEM

We have collected images of different signs of different angles for sign letter A to Z.

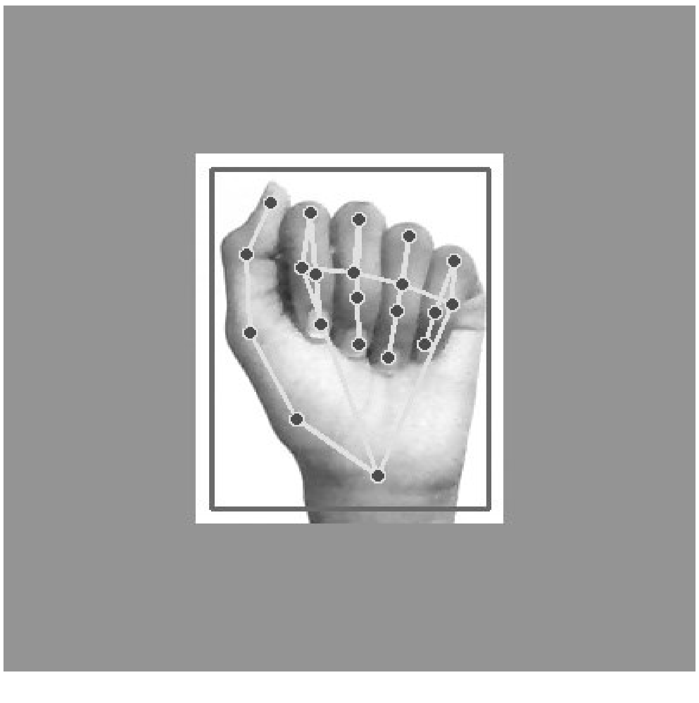


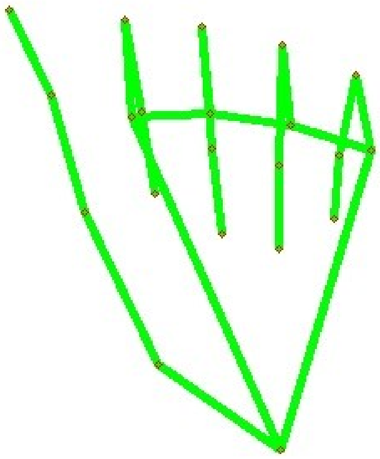
```
:1850] Cannot dlopen so          ease make sure the missing libraries mentioned above are

3] This TensorFlow binar          oneAPI Deep Neural Network Library (oneDNN) to use the f
mpiler flags.
del was *not* compiled.
```

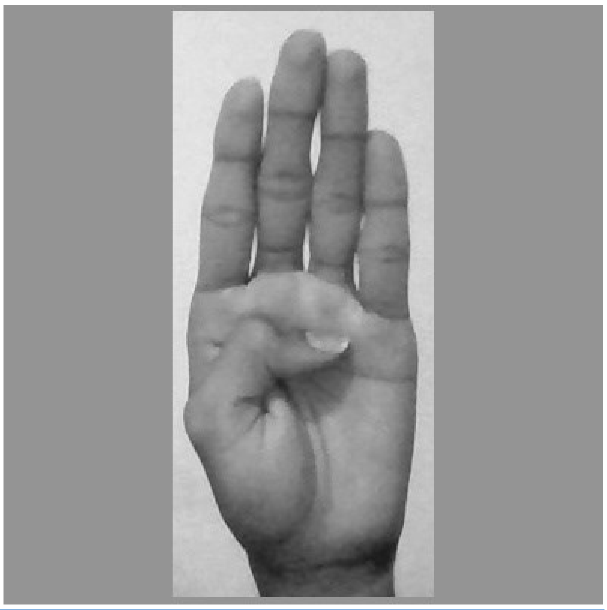
```
ays download // Download once // Don't show again // Configure... (... (28 minutes    43:13  CRLF  UTF-8  4 spaces  Python 3.9 (pythonProject)
ENG  10:06  27-08-2022
```

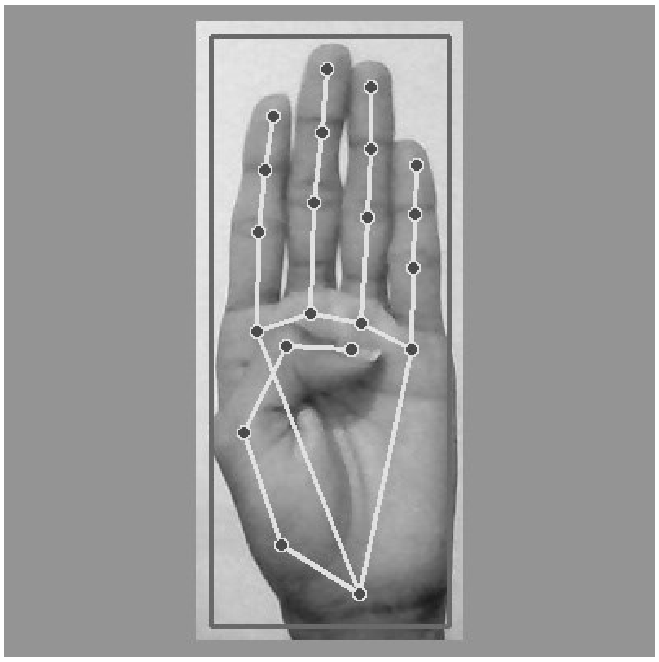
FIGURE 9- DATA COLLECTION

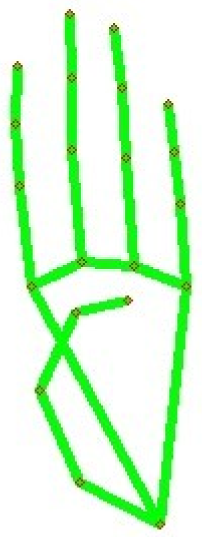
Mediapipe Landmark System:



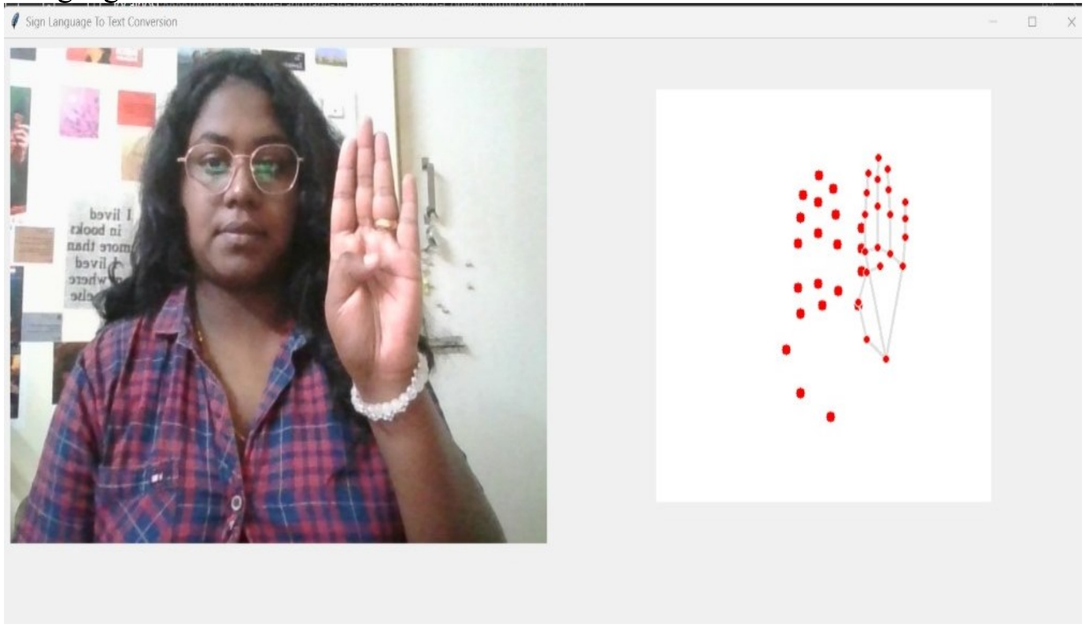


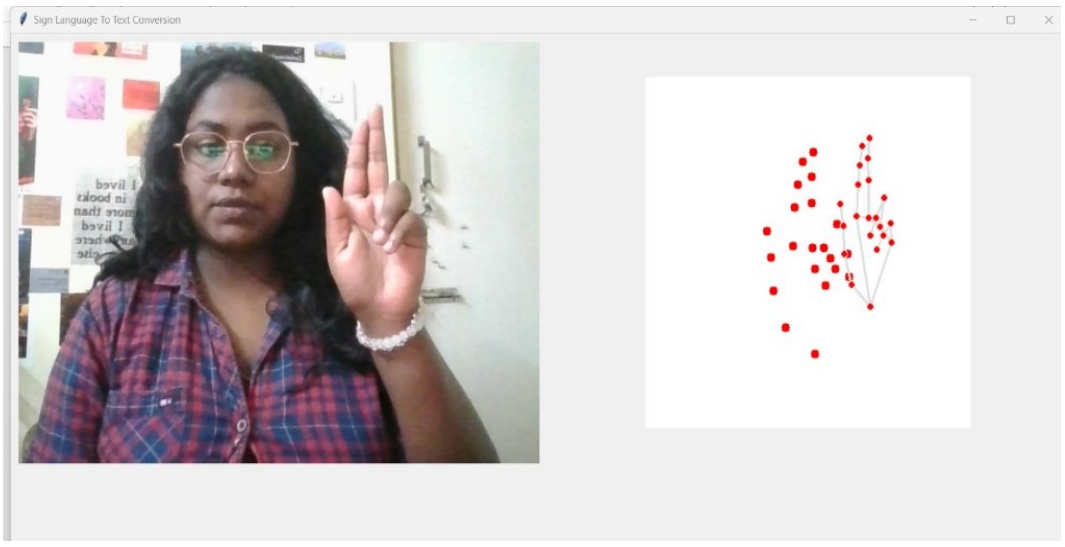


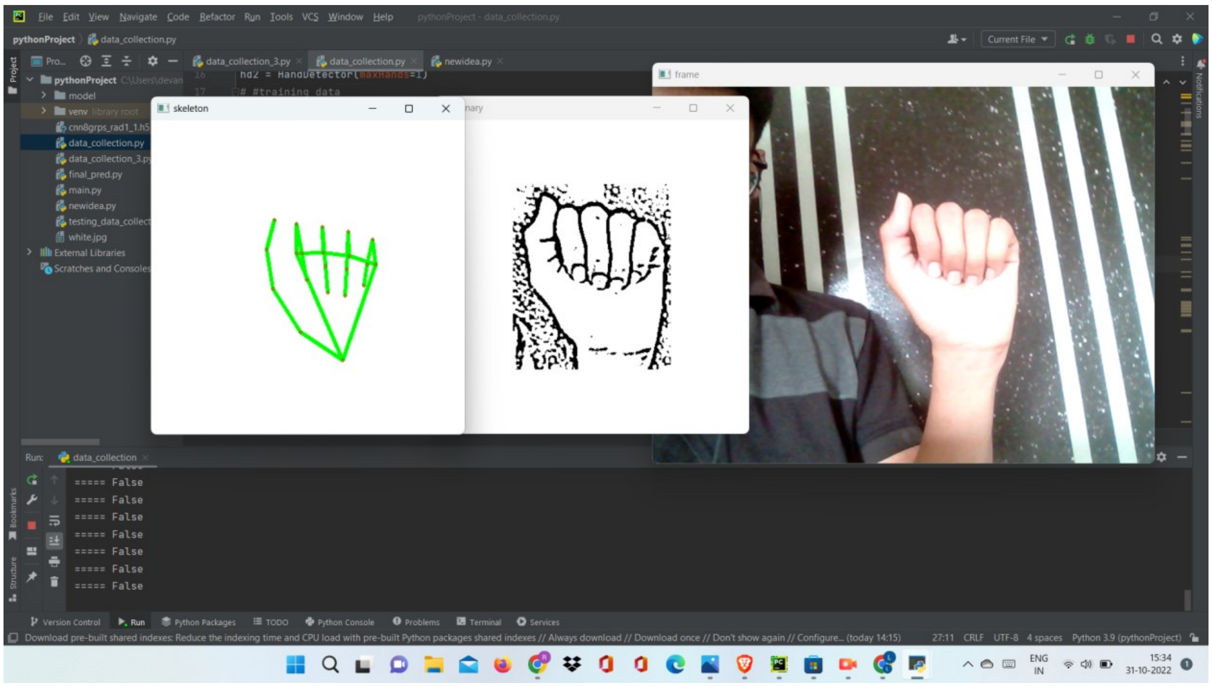




Now we get this landmark points and draw it in plain white background using opencv library.

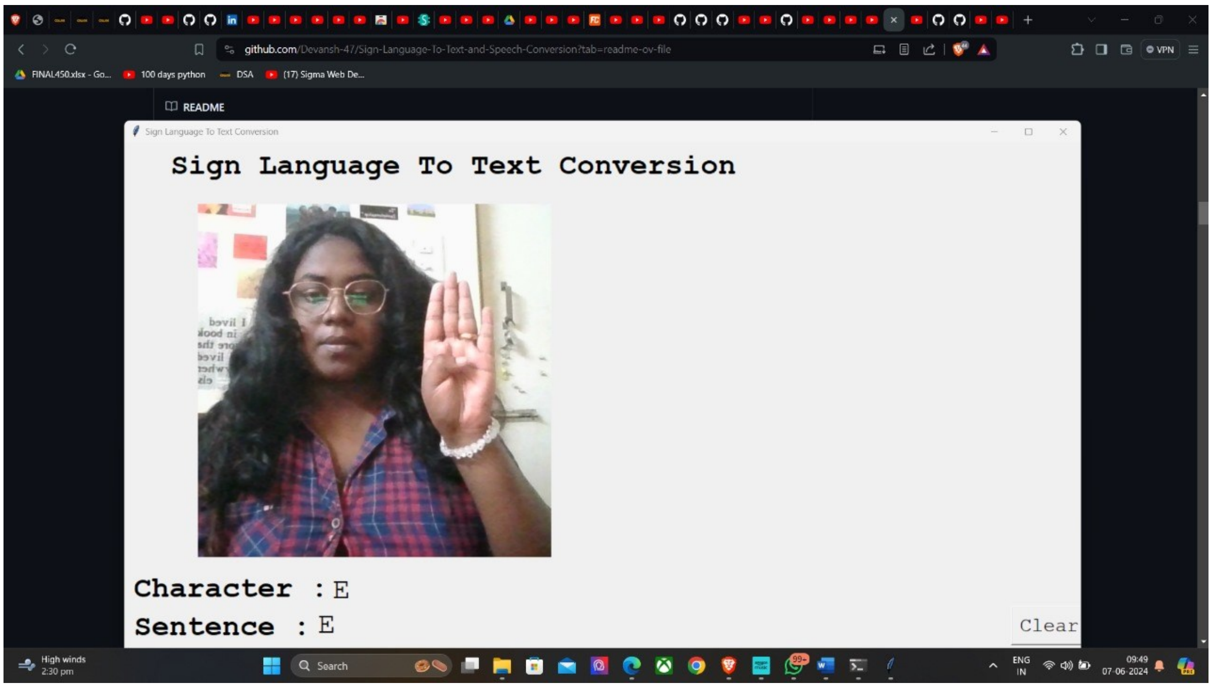

Sign Language To Text Conversion


Sign Language To Text Conversion



```
===== False
===== False
===== False
===== False
===== False
===== False
===== False
```

Now we get these landmark points and draw it in plain white background using opencv library -By doing this we tackle the situation of background and lightning conditions because the mediapipe labrary will give us landmark points in any background and mostly in any lightning conditions.



Sign Language To Text Conversion

Character : E

Sentence : E