****

**SIGN LANGUAGE DETECTION**

**A PROJECT REPORT**

*Submitted by*

**SIVAGURUMOORTHI M (910320104011)**

**DHANABALAN P (910320104003)**

**VASANTH S (910320104014)**

***In partial fulfilment for the award of degree***

***of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**

**CHENDHURAN COLLEGE OF ENGINEERING AND TECHNOLOGY, PUDUKKOTTAI.**

**ANNA UNIVERSITY::CHENNAI 600 025**

**MAY 2023**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“SIGN LANGUAGE DETECTION”**  is the bonafide work of **“SIVAGURUMOORTHI M (910320104011), P.DHANABALAN (910320104003), S..VASANTH (910320104014)”** who carried out the project work under my supervision, for the partial fulfilment of the requirement for the award of the Degree of Bachelor of Engineering in Computer Science and Engineering. Certified further that to the best of my knowledge and belief, the work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or an award was conferred earlier occasion.

SIGNATURE SIGNATURE

**Mrs. M.KIRITHIKA DEVI M.E, Ph.D\* Mr. I.RAMASAMY M.E.,**

**HEAD OF THE DEPARTMENT SUPERVISOR**

Assistant Professor, Assistant Professor,

Dept. of Computer Science & Engg., Dept. of Computer Science & Engg.,

Chendhuran College of Engg &Tech., Chendhuran College of Engg &Tech.,

Pudukkottai-622507. Pudukkottai-622507.

Submitted for the viva-voce examination held on\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**INTERNAL EXAMINER EXTERNALEXAMINER**

**ACKNOWLEDGEMENT**

We are much grateful to the management of Chendhuran College of Engineering and Technology, Pudukkottai for providing us an opportunity to undergo this project.

We have immense pleasure and satisfaction in expressing our hearty thanks to our beloved managing trustee **Shri. AVM. SELVARAJ Chairman**, who is the source of this great situation.

We express our sincere thanks to **Shri. R. VAIRAVAN Managing Director,** who provides support to the great success for this institution.

We are pleasure to expressing our hearty thanks to our beloved **C.E.O. Dr. AVM.S.KARTHICK B.E., M.B.A.,Ph.D.,** who is the source of spirit and strength of this great institution.

We express our sincere and special thanks to our Executive Director **Dr. M.PANDIKRISHNAN B.E., M.S., Ph.D.,** who provided us the hope to finish it.

We express our deep sense of gratitude and thank to our Principal **Dr. K.GANESH BABU B.E., M.Tech (IITM)., Ph.D(IITM).,** for permitting us to do the project work and allowing us to utilize all the facilities in the college.

We are grateful to **Mrs. M.KIRITHIKA DEVI M.E, Ph.D\*.,** Assistant Professor and Head of the Department, for her valuable advice, permission encouragement accorded to carry out this project successfully.

We express our heartfelt and special thanks to our project guide **Mr. I.RAMASAMY.M.E.,** Assistant Professor, Department of Computer Science and Engineering, our project Guide for not only showing path but also guiding us throughout the path.

Finally above all, we give all glory to the **PARENTS** and **GOD**, who has been the source of spirit and strength to us throughout the project.

**ABSTRACT**

The major communication tool for people with disabilities is sign language, which is a visual language that uses hand gestures, changes in hand shape, and track information to express meaning. Deafness in both hearing and speaking Recognize sign language can help to solve the issue of the increasing number of people who require assistance The number of people who use sign language is growing, as is its popularity.is inadequate, and provide a more comfortable approach to study, work, and socialize. A hand to create a locating network based on the Faster R-CNN, recognize the hand section in the video or the sign language video picture, as well as the recognition result, is passed along to. A 3D CNN feature extraction with additional processing based on a network and a sign-language recognition framework Coding and decoding in Long and Short-Term Memory The network is built for the sign language images of sequence The framework has the potential to enhance Learning the context of sign language improves recognition accuracy. Another approach to sign language recognition is to use deep learning techniques. Deep learning models can learn to recognize signs directly from the video or image data, without the need to extract features manually. This can lead to more accurate sign language recognition, but it also requires more data to train the models.

i

**திட்ட சுருக்கம்**

மாற்றுத்திறனாளிகளுக்கான முக்கிய தகவல்தொடர்பு கருவி சைகை மொழி ஆகும், இது கை சைகைகள், கை வடிவத்தில் மாற்றங்கள் மற்றும் அர்த்தத்தை வெளிப்படுத்த தகவலைக் கண்காணிக்கும் ஒரு காட்சி மொழியாகும். செவித்திறன் மற்றும் பேசுதல் ஆகிய இரண்டிலும் காது கேளாமை சைகை மொழியை அங்கீகரித்தல் உதவி தேவைப்படும் நபர்களின் எண்ணிக்கை அதிகரித்து வரும் சிக்கலைத் தீர்க்க உதவும். சைகை மொழியைப் பயன்படுத்துபவர்களின் எண்ணிக்கை அதிகரித்து வருகிறது, அதன் பிரபலம் போதுமானதாக இல்லை, மேலும் வசதியான அணுகுமுறையை வழங்குகிறது. படிக்கவும், வேலை செய்யவும், பழகவும். வேகமான R-CNN அடிப்படையில் ஒரு இருப்பிட நெட்வொர்க்கை உருவாக்க ஒரு கை, வீடியோவில் உள்ள கைப் பகுதியை அல்லது சைகை மொழி வீடியோ படம், அத்துடன் அங்கீகார முடிவு ஆகியவற்றை அடையாளம் காணவும். நெட்வொர்க் மற்றும் சைகை மொழி அங்கீகார கட்டமைப்பின் அடிப்படையிலான கூடுதல் செயலாக்கத்துடன் கூடிய 3D CNN அம்சம் பிரித்தெடுத்தல் மற்றும் நீண்ட மற்றும் குறுகிய கால நினைவகத்தில் குறியீட்டு மற்றும் டீகோடிங் நெட்வொர்க் வரிசையின் சைகை மொழி படங்களுக்காக கட்டமைக்கப்பட்டுள்ளது. சைகை மொழி அங்கீகாரம் துல்லியத்தை மேம்படுத்துகிறது. சைகை மொழி அங்கீகாரத்திற்கான மற்றொரு அணுகுமுறை ஆழமான கற்றல் நுட்பங்களைப் பயன்படுத்துவதாகும். ஆழமான கற்றல் மாதிரிகள், அம்சங்களை கைமுறையாக பிரித்தெடுக்க வேண்டிய அவசியம் இல்லாமல், வீடியோ அல்லது படத் தரவிலிருந்து நேரடியாக அடையாளங்களை அடையாளம் காண கற்றுக்கொள்ளலாம். இது மிகவும் துல்லியமான சைகை மொழி அங்கீகாரத்திற்கு வழிவகுக்கும், ஆனால் மாடல்களைப் பயிற்றுவிக்க அதிக தரவு தேவைப்படுகிறது.

ii

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT (ENGILISH)** | **i** |
|  | **ABSTRACT (TAMIL)** | **ii** |
|  | **LIST OF FIGURES** | **vi** |
|  | **LIST OF SYMBOLS** | **vii** |
| **1** | **INTRODUCTION** | **1** |
| **2** | **SYSTEM STUDY** | **2** |
|  | 2.1 LITERATURE SURVEY | 2 |
|  | 2.2 EXISTING SYSTEM | 4 |
|  | 2.2.1 LIMITATIONS | 4 |
|  | 2.3 PROPOSED SYSTEM | 4 |
|  | 2.3.1 ADVANTAGES | 4 |
| **3** | **REQUIREMENT SPECIFICATION** | **5** |
|  | 3.1 SOFTWARE SPECIFICATION | 5 |
|  | 3.2 HARDWARE SPECIFICATION | 6 |
| **4** | **SYSTEM DESIGN** | **7** |
|  | 4.1 PROPOSED ARCHITECTURE | 7 |
|  | 4.2 UML DIAGRAMS | 8 |
|  | 4.2.1 USE CASE DIAGRAM | 8 |
|  | 4.2.2 CLASS DIAGRAM | 9 |
|  | 4.2.3 ACTIVITY DIAGRAM | 10 |

iii

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| **5** | **SYSTEM IMPLEMENTATION** | **11** |
|  | 5.1 MODULES OVERVIEW | 11 |
|  | 5.2 MODULE DESCRIPTION | 12 |
|  | 5.2.1 HUMAN POSE ESTIMATION | 12 |
|  | 5.2.2 DATASETS, TRAINING AND TESTING SETUPS | 14 |
|  | 5.2.3 EVALUATION AND  RESULTS | 17 |
| **6** | **TESTING** | **20** |
|  | 6.1 OVERVIEW OF TESTING | 20 |
|  | 6.2 TYPES OF SOFTWARE TESTING | 20 |
|  | 6.2.1 WHITE BOX TESTING | 21 |
|  | 6.2.2 BLACK BOX TESTING | 21 |
|  | 6.2.3 UNIT TESTING | 22 |
|  | 6.2.4 FUNCTIONAL TESTING | 22 |
|  | 6.2.5 OUTPUT TESTING | 23 |
|  | 6.2.6 USER ACCEPTANCE TESTING | 23 |

iv

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| **7** | **APPENDIX** | **24** |
|  | APPENDIX A - SOURCE CODE | 24 |
| **8** | **RESULT** | **30** |
| **9** | **CONCLUSION** | **31** |
|  | **FUTURE ENHANCEMENT** | **32** |
|  | **REFERENCES** | **33** |

v

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **TITLE** | **PAGE NO.** |
| 4.1 | Architecture Diagram | 7 |
| 4.2 | Use case Diagram | 8 |
| 4.3 | Class Diagram | 9 |
| 4.4 | Activity Diagram | 10 |
| 5.1 | Module Diagram | 11 |
| 5.2 | Human Pose Estimation | 13 |
| 5.3 | Training Datasets for Sign Hello | 16 |
| 5.4 | Training Datasets for Sign Thanks | 16 |
| 5.5 | Output for sign Hello | 18 |
| 5.6 | Output for Sign Thanks | 19 |
| 8.1 | Output for sign Hello | 30 |
| 8.2 | Output for Sign Thanks | 30 |

vi

## LIST OF SYMBOLS

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **SYMBOL NAME** | **NOTATION** | **DESCRIPTION** |
| 1. | Initial Activity |  | This shows the Starting  point or first activity of flow. |
| 2. | Final Activity |  | The end of the Activity diagram is shown by a bull’s eye symbol. |
| 4. | Decision |  | A logic where a decision is to be made. |
| 5. | Actor |  | A role that a user plays with respect to system. |
| 6. | Object |  | A Real time Entity. |
| 7. | Message |  | To send message between the life of an object. |

vii

**CHAPTER 1**

**INTRODUCTION**

Hearing-impaired people use sign language as one of their communication tools. It is a non-verbal visual language comprised of both manual and non-manual signs. Facial expressions and lip movements are examples of non-manual signs and head motions, for example, whereas manual signs include hand and finger movements, gestures, and hand orientation etc.

The technology is capable of performing in a dynamic and minimally invasive manner. busy background with acceptable results since skin color segmentation is used to separate the gestures, i.e., concentrate on the hands and face. We delete the face using Viola-Jones face detection followed by deletion of the identified region because the vocabulary under consideration (letters a-z) does not include facial cues. The distance moved by the hand in the following frames is used to identify the gesture as static or dynamic.

The structure of sign language differs in terms of spatial and temporal information. Unlike spoken language, where one word can mean a lot one after another. However, as mentioned in Sign Language Recognition (SLR), the general configuration of a sign language sentence consists of: time, location, person, and predicate. SLR intends to construct an assistive system that automatically turns an input sign into text/speech. SLR devices can assist bridge the communication gap between hearing impaired people and the rest of society. As a result, such systems pave the way for novel Human-Computer-Interaction (HCI) applications. SLR systems have been created that function well for solitary words but fail to recognize and translate long sequences of gestures. Finding a modelling framework that can capture the sign motions and accompanying language is the fundamental issue in constructing a continuous SLR system.

## CHAPTER 2

## SYSTEM STUDY

**2.1 LITERATURE SURVEY:**

**Paper Title : A MODIFIED LSTM MODEL FOR CONTINUOUS**

**SIGN LANGUAGE RECOGNITION USING**

**LEAP MOTION**

**Author : Anshul Mittal; Pradeep Kumar; Partha Pratim Roy;**

**Raman Balasubramanian; Bidyut B. Chaudhuri**

**Abstract :** Sign language facilitates communication between hearing impaired peoples and the rest of the society. A number of sign language recognition (SLR) systems have been developed by researchers, but they are limited to isolated sign gestures only. In this paper, we propose a modified long short-term memory (LSTM) model for continuous sequences of gestures or continuous SLR that recognizes a sequence of connected gestures. It is based on splitting of continuous signs into sub-units and modeling them with neural networks. Thus, the consideration of a different combination of sub-units is not required during training. The proposed system has been tested with 942 signed sentences of Indian Sign Language (ISL). These sign sentences are recognized using 35 different sign words. The average accuracy of 72.3% and 89.5% has been recorded on signed sentences and isolated sign words, respectively.

## Drawbacks:

Most of existing recognition systems work with sequentially performed manual actions, due to their excessive use in gesticulation and ease of system development.

## Project Title : Sign language recognition with long short-term

## memory

**Author : Tao Liu; Wengang Zhou; Houqiang Li**

**Abstract :**

Sign Language Recognition (SLR) aims at translating the Sign Language (SL) into speech or text, so as to facilitate the communication between hearing-impaired people and the normal people. This problem has broad social impact; however, it is challenging due to the variation for different people and the complexity in sign words. Traditional methods for SLR generally use handcrafted feature and Hidden Markov Models (HMMs) modeling temporal information. But reliable handcrafted features are difficult to design and not able to adapt to the large variations of sign words. To approach this problem, considering that Long Short-Term memory (LSTM) can model the contextual information of temporal sequence well, we propose an end-to-end method for SLR based on LSTM. Our system takes the moving trajectories of 4 skeleton joints as inputs without any prior knowledge and is free of explicit feature design. To evaluate our proposed model, we built a large isolated Chinese sign language vocabulary with Kinect 2.0. Experimental results demonstrate the effectiveness of our approach compared with traditional HMM based methods.

## Drawbacks:

Statics deals with the detection of static gestures(2d-images) while dynamic is a real- time live capture of the gestures. This involves the use of the camera for capturing movements

**2.2 EXISTING SYSTEM**

* The present system has poor object recognition ability and not much accurate and can be easily manipulated by some experts.
* Another way to help deaf persons is to rely on another human who knows to use sign language which may be not possible at all times

## 

## 2.2.1 LIMITATIONS

* Should monitor in person.
* Low accuracy.
* Low frame capture.
* Depending on other humans

## 

## 2.3 PROPOSED SYSTEM

* The proposed system clearly comprises three phases, including a training phase, a testing phase, and a recognition phase.
* During the training phase, a multiclass support vector machine is used to train each class (MSVM). Hu invariant moment and structural shape descriptors are combined to form a combinational feature vector, which is retrieved from the input picture after pre-processing in the testing step.
* Different classes are utilized to test an input gesture during the recognition step. To recognize the motion, the outcome with the most likely group is identified. Finally, the meaning of the input image is presented on the screen when it has been recognized.

## ADVANTAGES

* Frame detection helps to identify clearly.
* We can get images with the help of image processing.
* Objects are detected with their names.

## CHAPTER 3

## REQUIREMENT SPECIFICATION

* 1. **SOFTWARE SPECIFICATION**

The software specification is the specification of the system. It should include both the specification and a definition of the requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide the basis for creating the software requirement specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team’s progress throughout the development activity.

## REQUIREMENTS

* Language : Python
* Platform : OpenCV, NumPy, Mediapipe, Keras, TensorFlow
* Tool : Deep Leaning toolbox

## HARDWARE SPECIFICATION

The Hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete specification of the whole system. They are used by the software engineers as the starting point for the system design. It shows what the system do not and how it should be implemented.

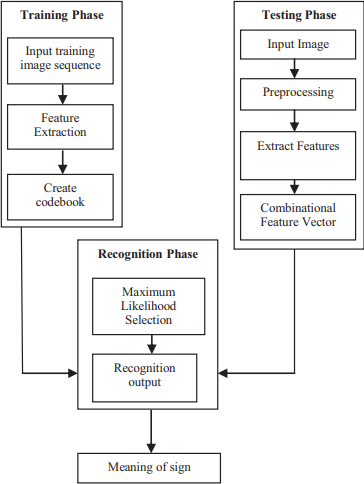
## REQUIREMENTS

* Processor : Intel® core ™ I [3-4030Ucpu@1.90GHZ](mailto:3-4030Ucpu@1.90GHZ)
* RAM : 4 GB or More
* System type : 64-bit operating system
* Keyboard : Normal or Multimedia
* Mouse : Compatible mouse
* Camera : Quality of 720p

## CHAPTER 4

## SYSTEM DESIGN

1. **PROPOSED ARCHITECTURE DIAGRAM:**



*Fig*: 4.1 Architecture

Models consisting of several CNN layers followed by numerous LSTM layers are often used to predict Real-Time Sign Language. The precision of these cutting-edge models, on the other hand, is quite low. This technique, Mediapipe Holistic with LSTM Model, on the other hand, provides significantly higher accuracy. This method yielded better outcomes with a smaller amount of data. This model trained much faster since it used less parameters, resulting in a shorter calculation time.

**4.2 UML DIAGRAMS:**

A UML diagram is a diagram based on the UML (Unified Modelling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system. It is based on diagrammatic representations of software components.

**4.2.1 USE CASE DIAGRAM:**

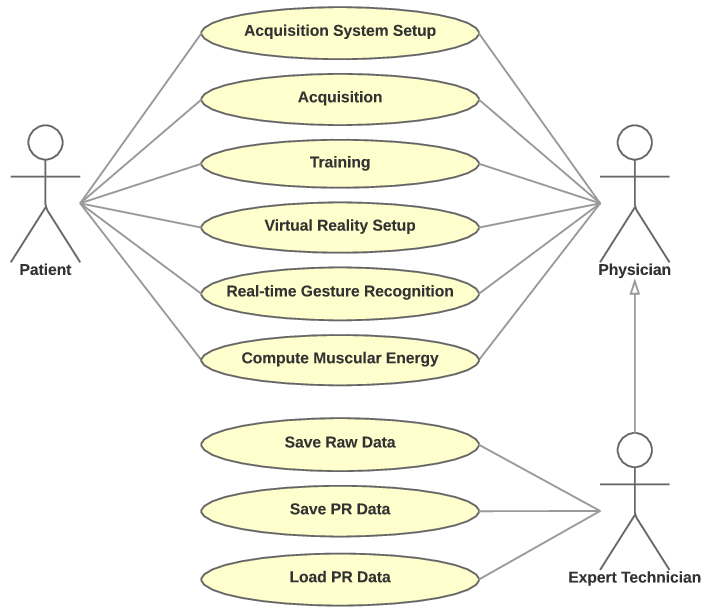
A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a “system” is something being developed or operated, such as a website. The “actors” are people or entities operating under defined roles within the system. Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities.

Fig. 4.2 Use Case Diagram

**4.2.2 CLASS DIAGRAM**

A Class diagram is a Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system’s classes, their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of object- oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

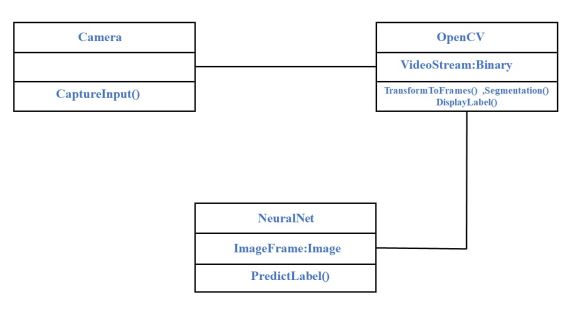
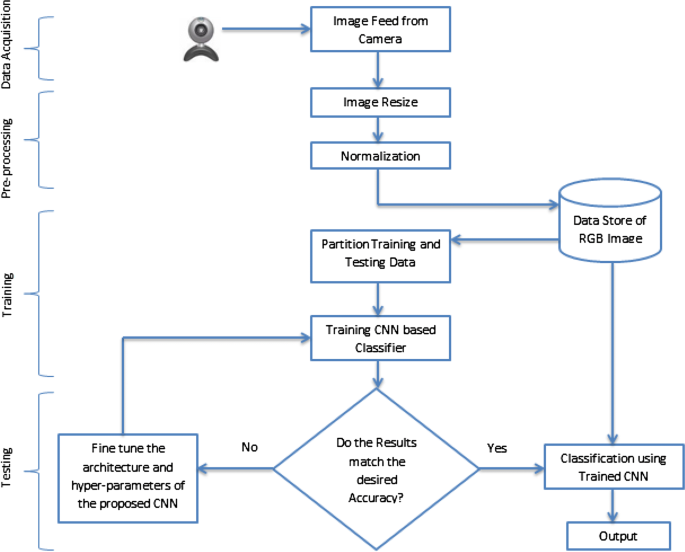


Fig.4.3 Class Diagram

## 4.2.3 ACTIVITY DIAGRAM

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity.The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join etc., The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system

.

*Fig*. 4.4 Activity Diagram

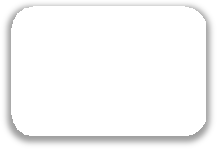
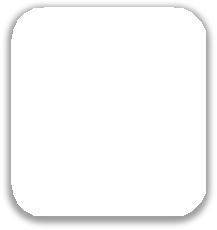
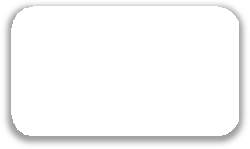
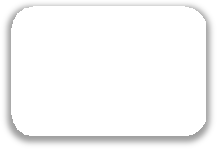
**CHAPTER 5**

**5 SYSTEM IMPLEMENTATION**

**5.1 MODULES OVERVIEW:**

Module is a logical separation of functionality within a project. They are basically used for reusability and better code maintenance. There are four modules used here.

* Human pose Estimation
* Pose Classification
* Scene Classification
* Datasets, Training and Testing setup
* Evaluation and Results



Pose

Estimation

Pose

Classification

Datasets,

Training and

Testing

Evaluation

and results

*Fig*.5.1 Module

**5.2 MODULE DESCRIPTION:**

**5.2.1 HUMAN POSE ESTIMATION:**

A model that mixes convolutional and recurrent neural network components, as well as a self-correcting feature that can enhance past predictions. This approach generates a 3D vector space using local inputs. It extracts partial body poses while taking into account the commotion The model was evaluated on an original dataset by the authors. and discovered that it is actually superior to any of the currently available alternatives. Depth imaging is also important in the proposes a solution known as Depth Ranking. Pose estimation for three- dimensional pictures. CNN network is a part of this concept. is used to choose between candidate couples in the first round. followed by a third step in which 3D pose predictions are made are created, integrating two-dimensional data with depth data pictures to Because body position estimation is crucial in many disciplines of research, including SLR, there have been numerous attempts to develop a viable model based on convolutional and recurrent deep learning networks. The use of three-dimensional imagery and the creation of depth maps has substantially improved the ability of such models to recognize each other. Scientific study aimed at improving the interpretation of body positions is still a hot topic. Researchers, in particular, Researchers are attempting to ensure that the exact locations of each joint can be detected even when the photos contain ambient noise or sections of the body are obscured. There has been significant progress in 3D mapping of body postures, but the fact that numerous 3D locations might correlate to a single 2D pose adds to the complexity. The difficulty of categorizing 3D joint images adds to the complexities, necessitating the adoption of technologically advanced input devices.

Such models lay the groundwork for future SLR research, on which new approaches can be built. Improved pose recognition and form prediction skills of new systems are further aided by technological advancements in capturing devices. Fusion of various types of data (e.g., thermal imaging or hybrid data) with vision-based indicators might improve system reliability under real-world situations, and so constitutes a potential research topic. The positions of critical points (i.e., limbs and joints) are directly conveyed using sensor technology, whereas image-based approaches infer those positions based on 2D images. Pose estimation performance can be influenced by a variety of things. Microsoft Kinect is one among them. Finally, excellent resources New methods for evaluating them are being developed. The availability of big SLR datasets motivates research. more thorough testing and puts us closer to the goal This technology is in the commercialization stage.

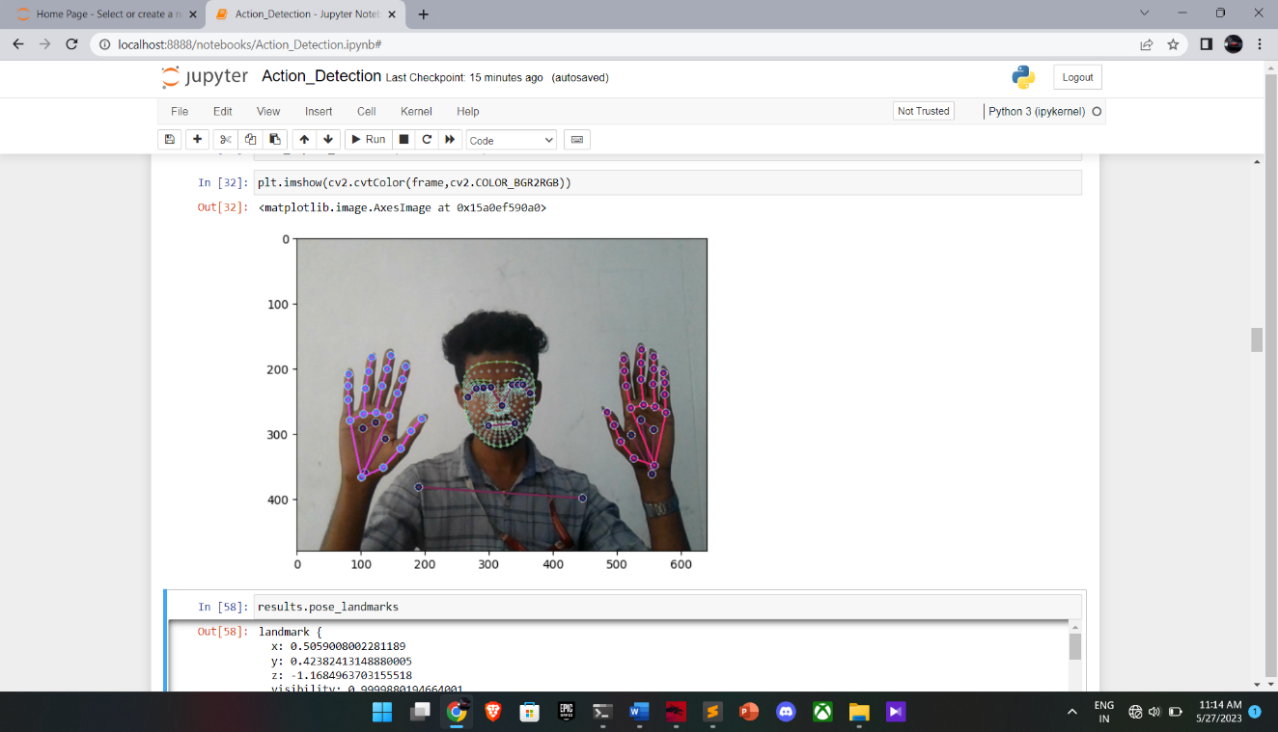


Fig.5.2 Human Pose Estimation

**5.2.2 DATASETS, TRAINING AND TESTING SETUP**

The dataset that was recorded using the proposed SLR frameworks is described in detail here. Following that, the recognition of continuous sign language was discussed. Finally, the results of recognition of isolated sign words have been displayed. We've registered six people to collect data on sign language collection. All participants are hearing aid students. India, has an impaired school. The data source There are 35 isolated sign words in this set. Each sign word has been carefully selected. Every signer must repeat it at least 15 times. As a result, a total of There were 3150 (35\*15\*6) sign words recorded. a list of everything Table I shows the sign words, with the '$' sign representing the transition gesture (i.e., when one user switches from one to another) between two continuous signals) between two continuous signs 157 sign sentences were recorded from each signer to test the efficiency of our proposed continuous-SLR method. This consisted of 2 to 6 sign words derived from these There are 35 different hand gestures. These have been repeated by each signer. sentences with signs as a result, there are 942 (6\*157) sign sentences in all. were taken down. In sentences, the distribution of sign words phrases containing three sign words have the maximum number of instances We created two Kinect sign language datasets and will make them available to the public in the future. Dataset I include 100 terms in sign language that are commonly used in everyday life. Each word is played five times by 50 signers. So, the dataset contains 25,000 samples, with 250 samples each word samples.

We created dataset II to demonstrate recognition performance on a large vocabulary dataset.500 sign words, 50 signers, 5 repetitions, 125,000 words total number of samples Each dataset is divided into two parts, one for each subset. for testing and one for training In the subsets for training, From the 50 signers in both groups, we chose 36 at random.

This section lists some of the most important and widely available datasets with hand gestures that can be used to test SLR techniques. The focus is on ensuring that dictionaries are large enough to allow for more robust testing and advanced applications. Depending on the chosen geographic variation of sign language, some high-quality sets can currently be utilized for this purpose. Researchers have access to numerous datasets for the UK form of sign language, including RWTH-Boston- 1, RWTH-Boston-50, and RWTHBoston-400, which contain anywhere from 10 to 400 individual signs. DGS Kinect-40, SIGNUM, and RWTHPHOENIX-Weather are only a few of the high-quality data corpora available for German sign language. There are between 35 and 1225 unique signs in the sets, including a considerable number of legitimate ones. The most important resource for learning American Sign Language is which comprises over 30 thousand signs performed by six different people. This is a named set as well, with specific frames at the start and finish of each statement. A typical dataset has several instances of the same thing. numerous signers, with the goal of making things easier following training, signer- independent recognition capacity Some the datasets given in the literature are far bigger.

Compared to others, and this factor should be considered while determining the consistency of outcomes The datasets in all of the reviewed studies were examined by us. The research was carried out using well-defined criteria how we relied on the talks in the written word Because all publications are essentially concerned with in elements of varied complexity in sign language The databases used have a lot of similarities and These characteristics allow them to be efficiently classified. For the most part, alphanumerical characters or words are employed. Continuous SLR experiments require longer or even longer portions of speech. The datasets also varied significantly in terms of size. It is critical to assess the size and complexity of the project.

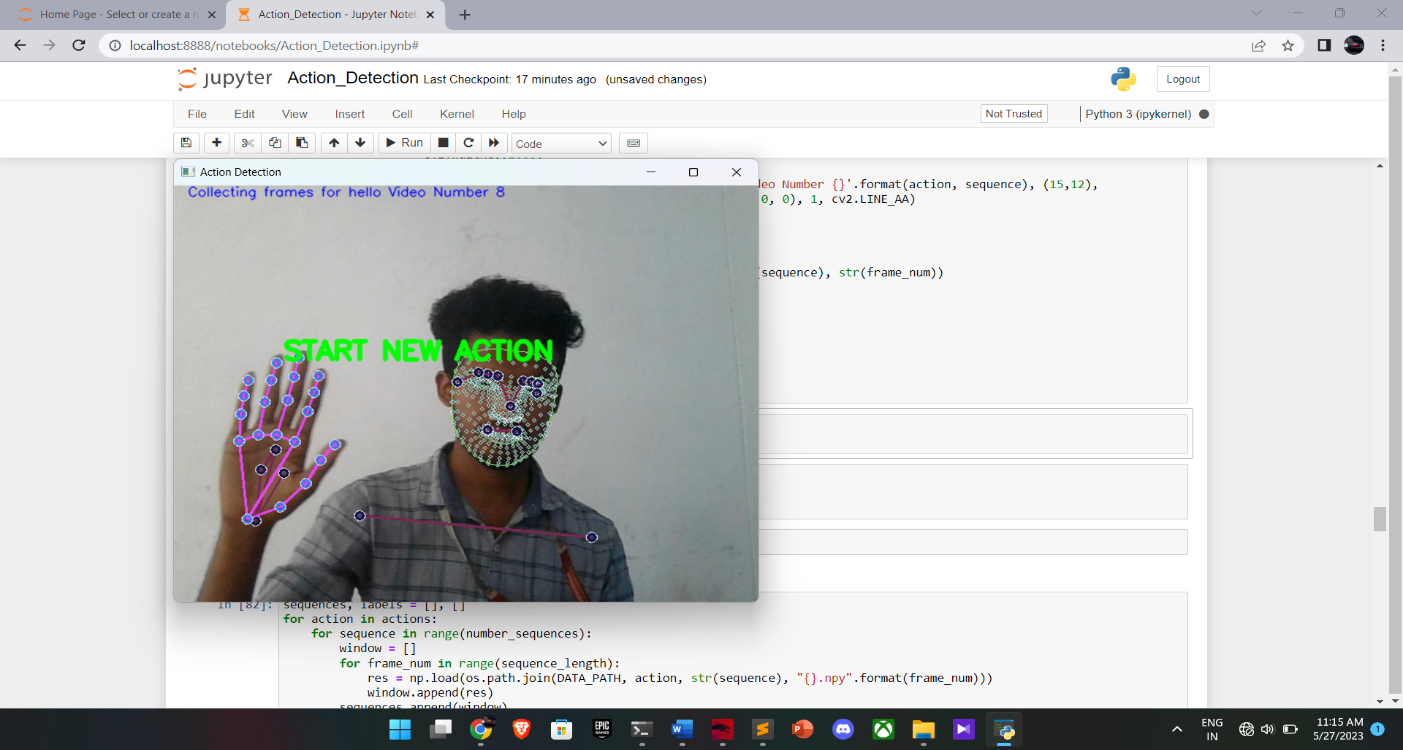


Fig.5.3 Training Datasets for Sign hello

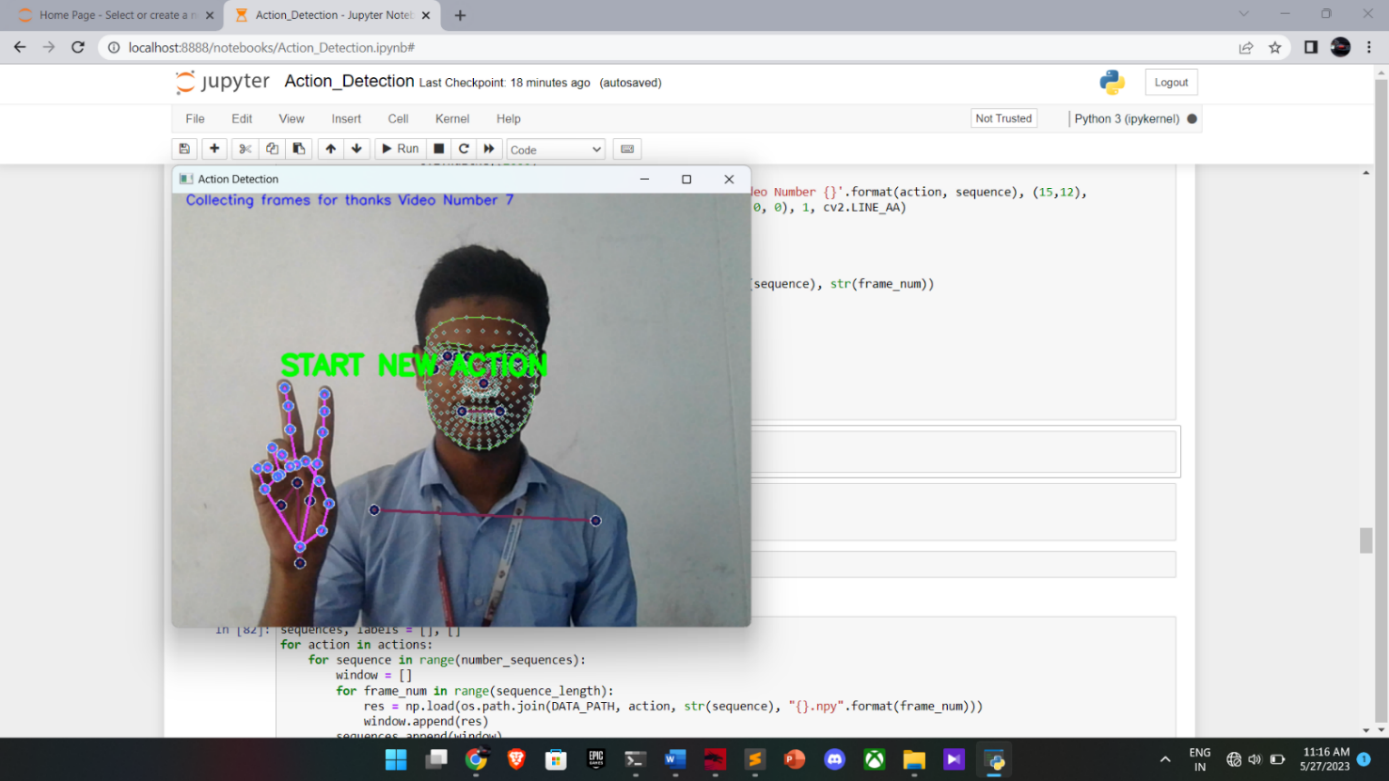


Fig.5.4 Training Datasets for Sign Thanks

**5.2.3 EVALUATION AND RESULTS**

We utilize the dataset we created to test the performance of our algorithm after we fix the network settings. which is a good match. curve from a manifold's perspective (CM VoM) and approach introduced in based on a trajectory model with HMM (HMM) (TM) In addition, we include another experiment. We utilize regular HMM and typical skeletal joints as input. temporal information modelling A variation is also tested.

Approach in classification reaches up to almost 96.23%. The experiments carried with the feature vector with one or more group of features. It is evident from that the proposed approach outperforms than other methods the proposed system's graphical user interface (GUI. The photos in (b)-(f) are similarly accurately identified, and their true meanings are displayed in the result window. We did a lot of experiments with 60 different signs and discovered that our categorization approach had a success rate of around 96.23 percent. the results of the feature vector study. with one or more sets of characteristics this shows the proposed strategy outperforms alternative approaches.

The proposed approach was implemented and tested using the following two sets of images: 1. Without angle measurements, the same collection of 320 photos was used for training. 2. A new set of 160 photos (5 images per sign) was loaded at runtime using angular measurement. With the exception of 4 sign 6, the initial batch of photos delivers perfect 100 percent recognition with practically all 32 signs. Which has a recognition rate of 70 to 80 percent. The second set of photos revealed a considerable improvement in sign recognition. In this situation, 60 to 80 percent recognition for two signs is accomplished, as illustrated in All other signs, on the other hand, produce 100% identification.

The proposed method's overall accuracy is presented below: No. of Patterns - No. of False Results Patterns Accuracy = X 100 percent for both test instances, the number of patterns obtained was 96.87 percent, and the accuracy attained with the second case was 98.125 percent. In the following artificial intelligence, computer vision, and other related topics, neural networks have contributed dynamic new methodologies to the study of sign language recognition based on vision. Starting with the task of common sign language word recognition and focusing on the topic of sign language locating and recognition using neural networks, this paper discusses the design of a deep learning-based hand locating algorithm, 3D CNN-based feature extraction, and LSTM-based recognition algorithm, and achieves better recognition results than other methods on the market.

Based on the findings, we discovered that we could construct and deploy the model during the development period of both desktop and mobile applications.

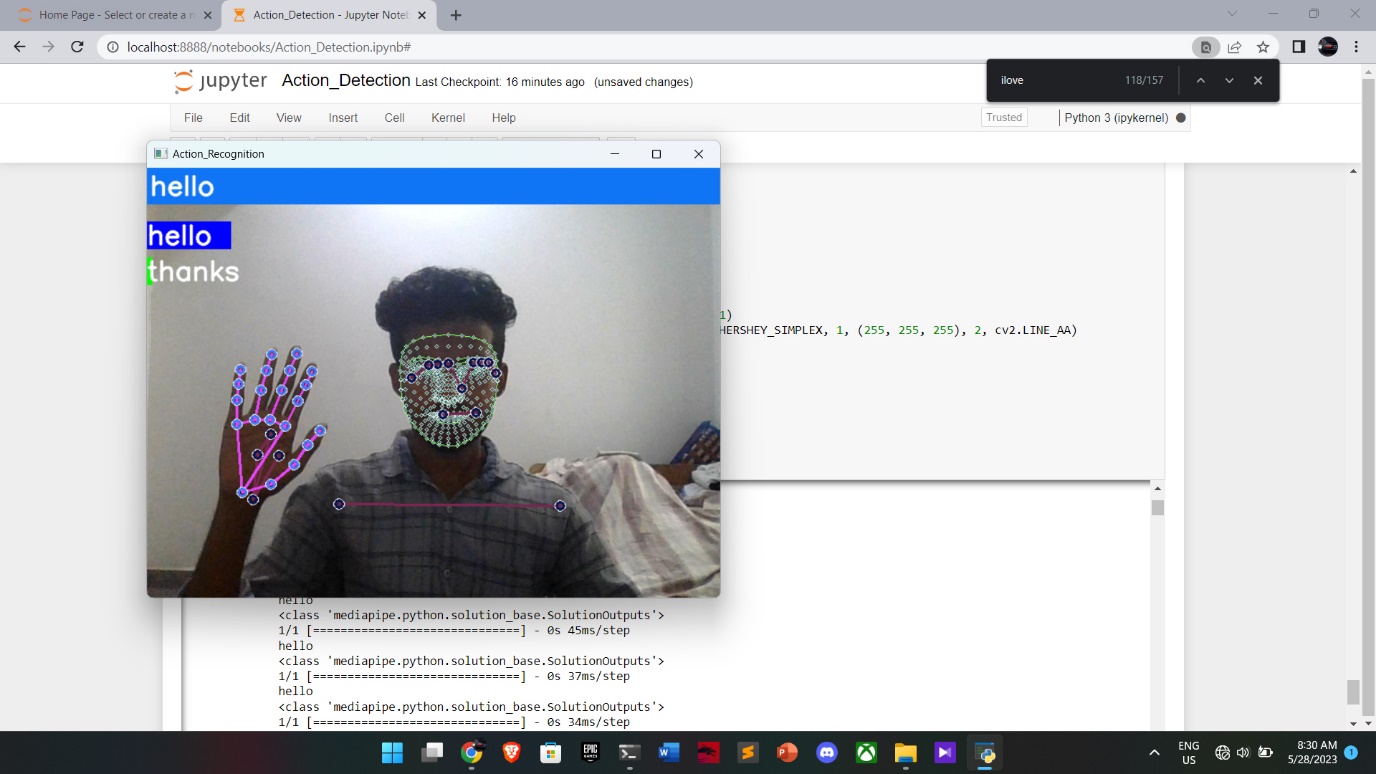
****

Fig.5.5 Output for sign Hello

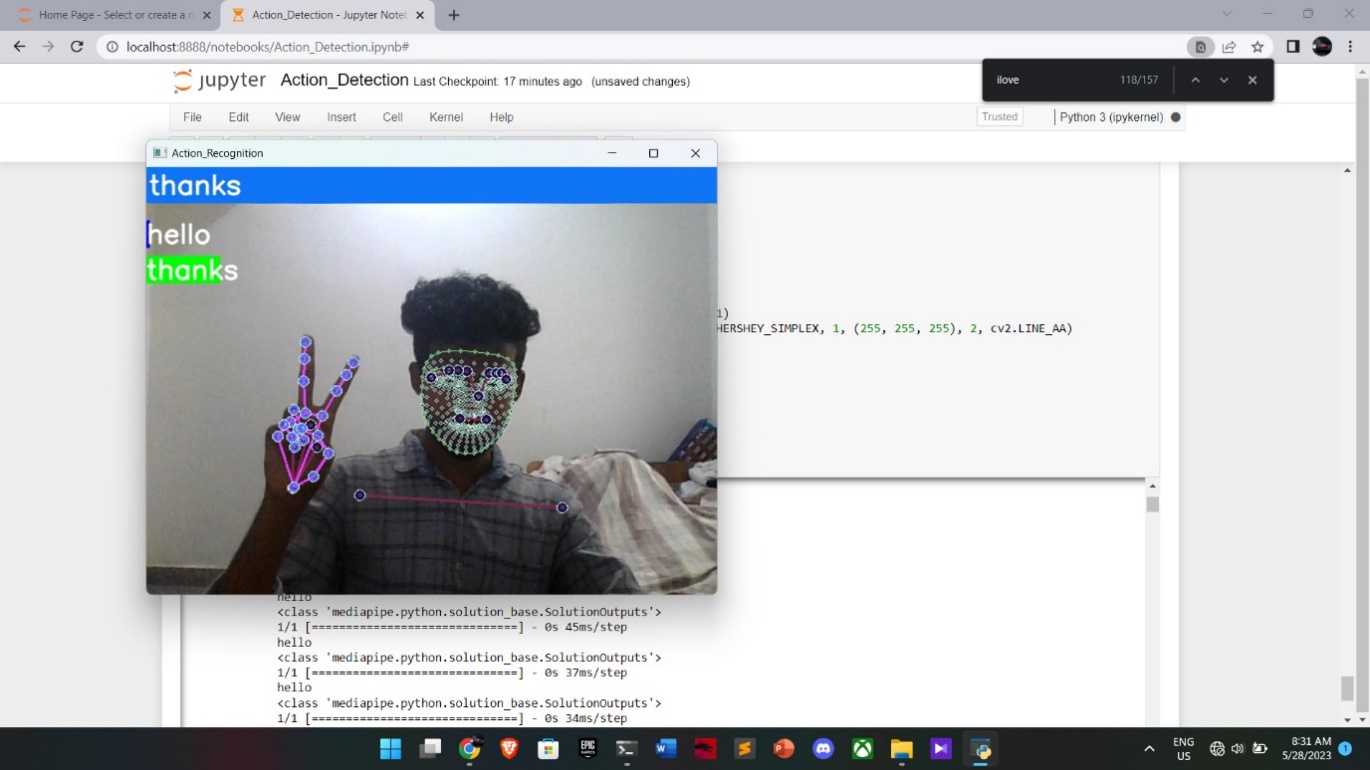
****

Fig.5.6 Output for Sign Thanks

**CHAPTER 6**

**6.1 TESTING**

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risk of software implementation. Test techniques include the process executing the program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

## 6.2 TYPES OF SOFTWARE TESTING:

* White box testing
* Black box testing
* Unit Testing
* Functional Testing
* Output Testing
* User Acceptance Testing

**6.2.1 WHITEE BOX TESTING**

White Box testing (also known as clear box testing, glass box testing, transparent box testing and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white box testing an internal perspective of the system as well as programming skills are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the expected outputs. White box testing can be applied at the unit, integration and system levels of the testing process. Although traditional testers tended to think of white box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration and between subsystems during a system level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

**6.2.2 BLACK BOX TESTING**

Black box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing like unit, integration, software, system and acceptance. It is sometimes referred to as specification-based testing. Black box testing, also known as behavioral testing is a software testing method in which the internal structure is not known to the tester. These tests can be functional or non- functional, though usually functional.

**6.2.3 UNIT TESTING**

* Unit testing, also known as Module testing, focuses verification efforts on the module. The module is tested separately and this is carried out at the programming stage itself
* Unit test comprises of the set of tests performed by an individual programmer before integration of the unit into the system.
* Unit test focuses on the smallest unit of software design the software component or module.
* Using component level design, important control paths are tested to uncover errors within the boundary of the module.
* Unit test is white box oriented and the step can be conducted in parallel for multiple components.

**6.2.4 FUNCTIONAL TESTING**

Functional testing is a type of software testing whereby the system is tested against the functional requirements specifications. Functions or features are tested by feeding them input and examining the output. Functional testing ensures that the requirements are properly satisfied by the application. This type of testing is not concerned with how processing occurs but rather with the results of processing. It simulates actual system usage but does not make any system structure assumptions. During functional testing, Black box testing technique is used in which the internal logic of the system being tested is not known to the tester. Functional testing is normally performed during the levels of system testing and acceptance testing.

**6.2.5 OUTPUT TESTING**

* Output of test cases compared with the expected results created during design of test cases.
* Asking the user about the format required by them tests the output generated or displayed by the system under consideration.
* Here, the output format is considered into two was, one is on screen and another one is printed format.
* The output on the screen is found to be correct as the format was designed in the system design phase according to user needs.
* The output comes out as the specified requirements as the user’s hard copy.

**6.2.6 USER ACCEPTANCE TESTING**

* Final Stage, before handling over to the customer which is usually carried out by the customer where the test cases are executed with actual data.
* The system under consideration is tested for user acceptance and constantly keeping touch with the prospective system user at the time of developing and making changes whenever required.
* It involves planning and execution of various types of tests in order to demonstrate that the implemented software system satisfies the requirements stated in the requirement document.

Two set of acceptance test to be run:

* Those developed by quality assurance group.
* Those developed by customer.

**CHAPTER 7**

**APPENDIX**

**APPENDIX A- Source code**

**#Importing Necessary Modules**

import cv2

import numpy as np

import os

from matplotlib import pyplot as plt

import time

import mediapipe as mp

**# Extract keypoints using MP Holistics**

mp\_holistic = mp.solutions.holistic

mp\_drawing = mp.solutions.drawing\_utils

def mediapipe\_detection(image, model):

image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image.flags.writeable = False

results = model.process(image)

image.flags.writeable = True

image = cv2.cvtColor(image, cv2.COLOR\_RGB2BGR)

return image, results

def draw\_landmarks(image, results):

mp\_drawing.draw\_landmarks(image, results.face\_landmarks, mp\_holistic.FACEMESH\_CONTOURS)

mp\_drawing.draw\_landmarks(image, results.pose\_landmarks, mp\_holistic.POSE\_CONNECTIONS)

mp\_drawing.draw\_landmarks(image, results.left\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS)

mp\_drawing.draw\_landmarks(image, results.right\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS)

def draw\_styled\_landmarks(image, results):

mp\_drawing.draw\_landmarks(image, results.face\_landmarks, mp\_holistic.FACEMESH\_CONTOURS,

mp\_drawing.DrawingSpec(color=(80,100,10), thickness=1, circle\_radius=1),

mp\_drawing.DrawingSpec(color=(80,250,120), thickness=1, circle\_radius=1))

mp\_drawing.draw\_landmarks(image, results.pose\_landmarks, mp\_holistic.POSE\_CONNECTIONS,

mp\_drawing.DrawingSpec(color=(80,20,10), thickness=2, circle\_radius=4),

mp\_drawing.DrawingSpec(color=(80,40,120), thickness=2, circle\_radius=2))

mp\_drawing.draw\_landmarks(image, results.left\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS,

mp\_drawing.DrawingSpec(color=(120,20,80), thickness=2, circle\_radius=4),

mp\_drawing.DrawingSpec(color=(120,40,250), thickness=2, circle\_radius=2))

mp\_drawing.draw\_landmarks(image, results.right\_hand\_landmarks, mp\_holistic.HAND\_CONNECTIONS,

mp\_drawing.DrawingSpec(color=(250,120,70), thickness=2, circle\_radius=4),

mp\_drawing.DrawingSpec(color=(250,70,230), thickness=2, circle\_radius=2))

**# Detecting Landmarks**

cap = cv2.VideoCapture(0)

with mp\_holistic.Holistic(min\_detection\_confidence=0.5,min\_tracking\_confidence=0.5) as holistic:

while cap.isOpened():

ret,frame = cap.read()

image,results = mediapipe\_detection(frame, holistic)

print(results)

draw\_styled\_landmarks(image,results)

cv2.imshow('check',image)

if cv2.waitKey(10) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

results

len(results.face\_landmarks.landmark)

mp\_holistic.POSE\_CONNECTIONS

frame

plt.imshow(frame)

plt.imshow(cv2.cvtColor(frame,cv2.COLOR\_BGR2RGB))

draw\_styled\_landmarks(frame,results)

plt.imshow(cv2.cvtColor(frame,cv2.COLOR\_BGR2RGB))

results.pose\_landmarks

# Extract Keypoint Values

for res in results.pose\_landmarks.landmark:

test = np.array([res.x, res.y, res.z, res.visibility])

test

results.pose\_landmarks.landmark[-1]

pose = []

for res in results.pose\_landmarks.landmark:

values = np.array([res.x, res.y, res.z, res.visibility])

pose.append(values)

len(results.pose\_landmarks.landmark)

len(pose)

len(results.right\_hand\_landmarks.landmark)

len(results.face\_landmarks.landmark)

pose

face

left\_hand

right\_hand

def extract\_keypoints(results):

pose = np.array([[res.x, res.y, res.z, res.visibility] for res in results.pose\_landmarks.landmark]).flatten() if results.pose\_landmarks else np.zeros(132)

face = np.array([[res.x, res.y, res.z] for res in results.face\_landmarks.landmark]).flatten() if results.face\_landmarks else np.zeros(1404)

left\_hand = np.array([[res.x, res.y, res.z] for res in results.left\_hand\_landmarks.landmark]).flatten() if results.left\_hand\_landmarks else np.zeros(63)

right\_hand = np.array([[res.x, res.y, res.z] for res in results.right\_hand\_landmarks.landmark]).flatten() if results.right\_hand\_landmarks else np.zeros(63)

return np.concatenate([pose, face, left\_hand, right\_hand])

extract\_keypoints(results)

extract\_keypoints(results).shape

**#Folder setup to collect keypoints for each frame**

DATA\_PATH = os.path.join('Feature\_Extraction')

actions = np.array(['hello', 'thanks', 'iloveyou'])

number\_sequences = 30

sequence\_length = 30

**# Feature Extraction**

cap = cv2.VideoCapture(0)

with mp\_holistic.Holistic(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as holistic:

for action in actions:

for sequence in range(number\_sequences):

for frame\_num in range(sequence\_length):

ret, frame = cap.read()

image, results = mediapipe\_detection(frame, holistic)

draw\_styled\_landmarks(image, results)

# Wait

if frame\_num == 0:

cv2.putText(image, 'START NEW ACTION', (120,200),

cv2.FONT\_HERSHEY\_SIMPLEX, 1, (0,255, 0), 4, cv2.LINE\_AA)

cv2.putText(image, 'Collecting frames for {} Video Number {}'.format(action, sequence), (15,12),

cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (255, 0, 0), 1, cv2.LINE\_AA)

cv2.imshow('Action Detection', image)

cv2.waitKey(2000)

else:

cv2.putText(image, 'Collecting frames for {} Video Number {}'.format(action, sequence), (15,12),

cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, (255, 0, 0), 1, cv2.LINE\_AA)

cv2.imshow('Action Detection', image)

keypoints = extract\_keypoints(results)

keypoint\_path = os.path.join(DATA\_PATH, action, str(sequence), str(frame\_num))

np.save(keypoint\_path, keypoints)

if cv2.waitKey(10) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

cap.release()

cv2.destroyAllWindows()

# Data Preprocessing

classes = {label:num for num, label in enumerate(actions)}

classes

sequences, labels = [], []

for action in actions:

for sequence in range(number\_sequences):

window = []

for frame\_num in range(sequence\_length):

res = np.load(os.path.join(DATA\_PATH, action, str(sequence), "{}.npy".format(frame\_num)))

window.append(res)

sequences.append(window)

labels.append(classes[action])

np.array(sequences).shape

np.array(labels).shape

X = np.array(sequences)

from tensorflow.keras.utils import to\_categorical

y= to\_categorical(labels).astype(int)

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.05)

X\_train.shape

X\_test.shape

**# LSTM Model**

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense

model = Sequential()

model.add(LSTM(64, return\_sequences=True, activation='relu', input\_shape=(30,1662)))

model.add(LSTM(128, return\_sequences=True, activation='relu'))

model.add(LSTM(64, return\_sequences=False, activation='relu'))

model.add(Dense(64, activation='relu'))

model.add(Dense(32, activation='relu'))

model.add(Dense(3, activation='softmax'))

model.compile(optimizer='Adam', loss='categorical\_crossentropy', metrics=['categorical\_accuracy'])

model.summary()

model.fit(X\_train, y\_train, epochs=100)

**# Predictions**

final\_result = model.predict(X\_test)

actions[np.argmax(final\_result[0])]

actions[np.argmax(y\_test[0])]

**# Save our model**

model.save('lstm\_model.h5')

y\_pred = model.predict(X\_test)

y\_true = np.argmax(y\_test, axis=1).tolist()

y\_pred = np.argmax(y\_pred, axis=1).tolist()

y\_true

y\_pred

**# Performance Evaluation**

from sklearn.metrics import multilabel\_confusion\_matrix

multilabel\_confusion\_matrix(y\_true,y\_pred)

**# Real Time Testing**

colors = [(255,0,0), (0,255,0), (0,0,255)]

def prob\_viz(res, actions, input\_frame, colors):

output\_frame = input\_frame.copy()

for num, prob in enumerate(res):

cv2.rectangle(output\_frame, (0,60+num\*40), (int(prob\*100), 90+num\*40), colors[num], -1)

cv2.putText(output\_frame, actions[num], (0, 85+num\*40), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255,255,255), 2, cv2.LINE\_AA)

return output\_frame

sequence = []

sentence = []

threshold = 0.8

cap = cv2.VideoCapture(0)

with mp\_holistic.Holistic(min\_detection\_confidence=0.5, min\_tracking\_confidence=0.5) as holistic:

while cap.isOpened():

ret, frame = cap.read()

image, results = mediapipe\_detection(frame, holistic)

print(results)

draw\_styled\_landmarks(image, results)

keypoints = extract\_keypoints(results)

sequence.append(keypoints)

sequence = sequence[-30:]

if len(sequence) == 30:

res = model.predict(np.expand\_dims(sequence, axis=0))[0]

print(actions[np.argmax(res)])

if res[np.argmax(res)] > threshold:

if len(sentence) > 0:

if actions[np.argmax(res)] != sentence[-1]:

sentence.append(actions[np.argmax(res)])

else:

sentence.append(actions[np.argmax(res)])

if len(sentence) > 5:

sentence = sentence[-5:]

image = prob\_viz(res, actions, image, colors)

cv2.rectangle(image, (0,0), (640, 40), (245, 117, 16), -1)

cv2.putText(image, ' '.join(sentence), (3,30), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE\_AA)

cv2.imshow('Action\_Recognition', image)

if cv2.waitKey(10) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

cap.release()

cv2.destroyAllWindows()

**CHAPTER 8**

**RESULTS**

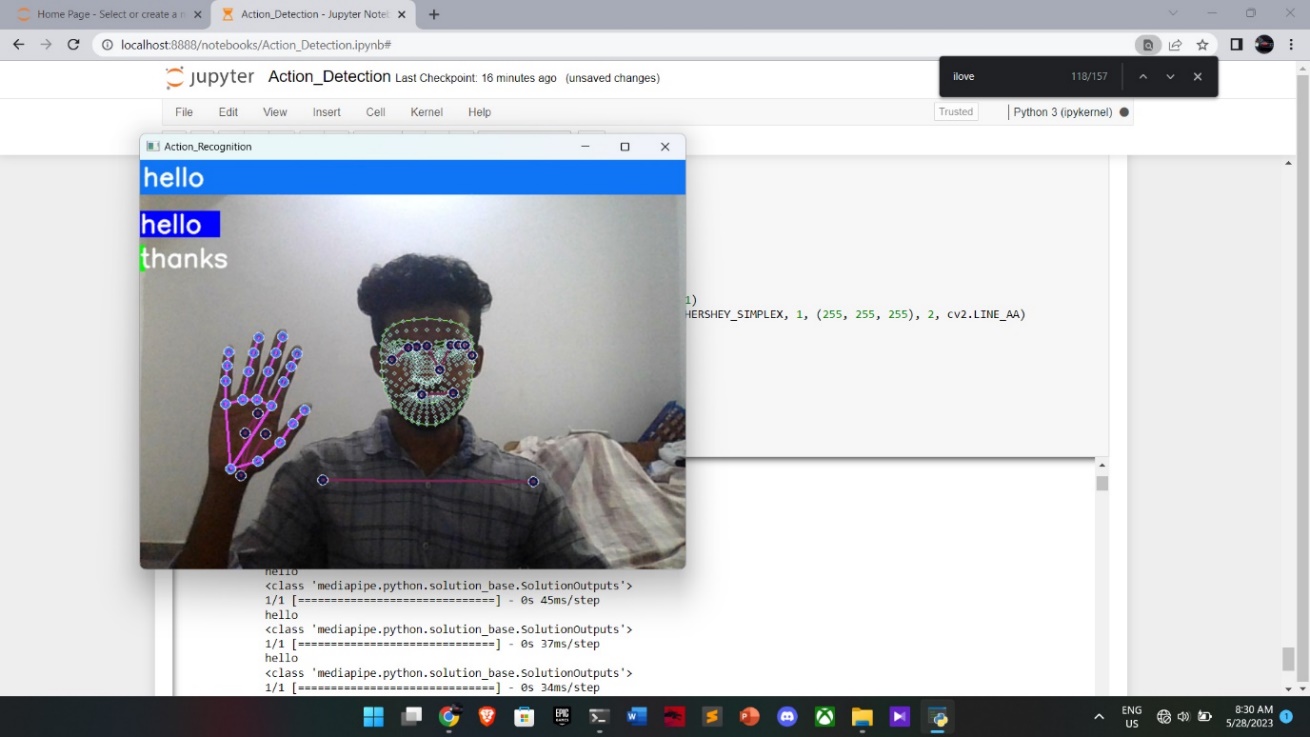
****

Fig .8.1 Output for Hello sign

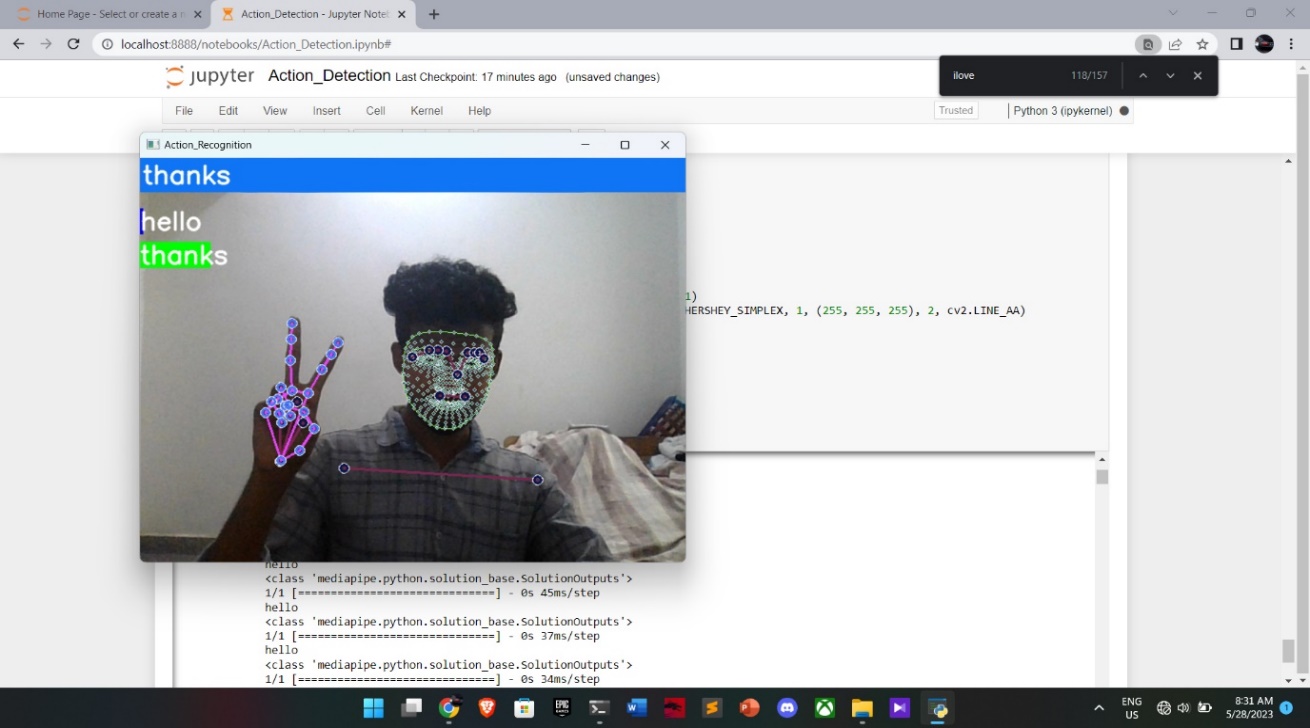
****

Fig .8.2 Output for Thanks Sign

**CHAPTER 9**

**CONCLUSION**

The ultimate goal of our project is to aid in communicating with those having vocal and hearing disabilities. Everyone can use this and communicate with everyone without any hesitation. Hand gestures are a powerful way for human communication, with lots of potential applications in the area of human computer interaction. Vision based hand gesture recognition techniques have many proven advantages compared with traditional devices. However, hand gesture recognition is a difficult problem and the current work is only a small contribution towards achieving the results needed in the field of sign language gesture recognition. This report presented a vision-based system able to interpret isolated hand gestures from the Argentinian Sign Language (LSA). Videos are difficult to classify because they contain both the temporal as well as the spatial features. We have used two different models to classify on the spatial and temporal features. CNN was used to classify on the spatial features whereas RNN was used to classify on the temporal features. We obtained an accuracy of 95.217%. This shows that CNN along with RNN can be successfully used to learn spatial and temporal features and classify Sign Language Gestures. This method for individual gestures can also be extended for sentence level sign language. Also, the current process uses two different models, training inception (CNN) followed by training RNN. For future work one can focus on combining the two models into a single model.

**FUTURE ENHANCEMENT**

* We can develop a model for Indian sign language word and sentence level recognition. This will require a system that can detect changes with respect to the temporal space.
* We can develop a complete product that will help the speech and hearing-impaired people, and thereby reduce the communication gap.
* In future work, proposed system can be developed and implemented using Raspberry Pi.
* Image Processing part should be improved so that System would be able to communicate in both directions i.e.it should be capable of converting normal language to sign language and vice versa.
* We will try to recognize signs which include motion.
* Moreover, we will focus on converting the sequence of gestures into text i.e., word and sentences and then converting it into the speech which can be heard.

**REFERENCES:**

* Anup Kumar, Karun Thankachan and Mevin M “Sign Language Recognition “IEEE 3rd InCI Conf. on Recent Advances in Information Technology I RAIT- 20161
* Siming He “Research of a Sign Language Translation System Based on Deep Learning” IEEE 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)
* Anusorn Chaikaew, Kritsana Somkuan, Thidalak Yuyen “Thai Sign Language Recognition: an Application of Deep Neural Network” IEEE The 6th International Conference on Digital Arts, Media and Technology (DAMT) and 4th ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (NCON)
* Anshul Mittal, Pradeep Kumar, Partha Pratim Roy, Raman Balasubramanian and Bidyut B. Chaudhuri “A Modified-LSTM Model for Continuous Sign Language Recognition using Leap motion” DOI 10.1109/JSEN.2019.2909837, IEEE Sensors Journal 1
* Tao Liu, Wengang Zhou, and Houqiang Li “SIGN LANGUAGE RECOGNITION WITH LONG SHORT-TERM MEMORY”
* Pratik Likhar, Neel Kamal Bhagat , Dr. Rathna G N “Deep Learning Methods for Indian Sign Language Recognition” 2020 IEEE 10th International Conference on Consumer Electronics (ICCE-Berlin
* Karishma Dixit , Anand Singh Jalal “Automatic Indian Sign Language Recognition System” DOI 10.1109/IAdCC.2013.6514343