Faculty of Computers & Artificial Intelligence

Benha University

Sign language conversation

A senior project submitted in partial fulfillment of the requirements for the degree of Bachelor of Computers and Artificial Intelligence.

Computer Science Departement,

Project Team

1- Mohamed Esam Mohamed Al-Azzazi

2- Mostafa Adel Gouda Abdel-Rahman

3- Mahmoud Hesham Salem Mahmoud

4- Mohamed Farid Fayez Fahim

5- Mohamed Taher Abdel-Hameed Mohamed

6- Mohamed Mostafa Mahmoud Mohamed

Under Supervision of

Dr. Hamada Nayel

A U G U S T – 2 0 2 2

ACKNOWLEDGMENT

It has been a great opportunity to gain lots of experience in real time projects, followed by the knowledge of how to design and analyze real projects. For that we want to thank all the people who made it possible for students like us. Special thanks to the graduation Project Unit for the efforts they did to provide us with all useful information and making the path clear for the students to implement all the education periods in real-time project design and analysis. Furthermore, we all the professors and visiting industry for the interesting lectures they presented which had great benefit for all of us. We would like to express our deepest gratitude to our graduation project supervisor **Dr. Hamada Nayel** for his patience and guidance along the semester. In addition, we would also like to thank the staff of our faculty for enabling us to visit their offices to observe their daily operations. Moreover, it is our duty to thank the staff of our faculty for enabling us to visit their offices to observe their daily operations.

At last, we would like to thank all the people who helped, supported, and encouraged us to successfully finish the graduation project Phase 1 whether they were in the university or in the industry.

A U G U S T – 2 0 2 2

DECLARATION

We hereby certify that this material, which we now submit for assessment on the program of study leading to the award of Bachelor of Computers and Artificial Intelligence in *(insert title of degree for which registered)* is entirely our own work, that we have exercised reasonable care to ensure that the work is original, and does not to the best of our knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of our work.

Signed: _____

Date: Day, xx Month Year.

ABSTRACT

hand gesture recognition is an area of computer vision and deep learning. Being a natural way of human communication, it is an area where many people in the machine learning field are working on, our goal is to make human computer interaction (HCI) easier, without the need for any extra devices. Communication is defined as the act of sharing or exchanging information, thoughts, or feelings. To establish communication between two persons, both of them are required to have knowledge and understanding of a common language. But in the case of deaf and mute people, the ways of communication are different, They communicate with themselves and other people with sign language, and if a person that is communicating with them lack the knowledge of sign language, then they won't be able to communicate with them, and that's the problem that our software will be fixing, it'll will act as a middle ground for communicating between people with disabilities, it also can translate voice to animated sign language on screen, so blind people can communicate with deaf people. It's crucial our software to work efficiently in real-time with high accuracy.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS/ABBREVIATIONS

ACRONYM Definition of Acronym

1 INTRODUCTION

Humans have a variety of methods for communicating with each other. This includes actions like bodily gestures, face expressions, spoken words, etc. However, people who are deaf are limited to communicate with hand motions. People with hearing disabilities and/or speech disabilities use a standard sign language which cannot be understood by people who do not know it.

We knew that Over 5% of the world's population – or 430 million people – require rehabilitation to address their 'disabling' hearing loss (432 million adults and 34 million children). It is estimated that by 2050 over 700 million people – or one in every ten people – will have disabling hearing loss. And in Egypt Experts say nearly 5 million of Egypt's 100 million residents are deaf, due in part to frequent intermarriage of close relatives. Notwithstanding their broad smiles and bubbling enthusiasm, deaf children in Egypt must cope with serious challenges at the same time the organizations that work to educate them struggle to run quality programs.

Also, learning sign language is hindered by their disability. A modern learning and translation tool for sign language implemented in Machine Learning can significantly affect the ease of Sign Language Communication. This tool will aim to do the following:

- Obtain a video feed from the camera
- Classify and display the equivalent Arabic alphabet for the Arabic sign language alphabet.
- Get a voice from the microphone
- Convert it into animated Arabic sign language alphabet.

However, the following factors pose some challenges to this system:

- The conditions of the illumination in the place where this system is used has to be considered, as it plays an important role in the accuracy of the recognition.
- It has to be made sure that the hand gesture is at an ideal distance from the camera.
- The camera must capture the hand up to the wrist at least, and it must not focus on background subjects.
- Identifying the features of the symbols that can be used in the system to get a better accuracy This system aims to consider these challenges and recognize Sign Language Symbols (except those which require movement of hands).

With this project it will be easier to communicate with them and they can live a natural life. And we will be glad to help them.

1.1 PROBLEM STATEMENT

In our progressive society, it is necessary to socialize with all people to whether for recreation or for a purpose. Communication is important for every human being. However, people who have a hearing disability and/or a speech disability need a different way to communicate other than vocal communication. They resort to sign language to communicate with each other. However, Sign Language requires a lot of training to be understood and learn and not every person may understand what the sign language gestures mean. Learning sign language is also time consuming as there are no effective, portable tool for recognizing sign language. Hearing or Speech disabled people who know Sign Language require a translator who also knows Sign Language to explain their thoughts to other people in an effective manner. To help overcome these problems, this system helps hearing or speech disabled people to learn as well as translate their sign language.

This is a very important matter relating to the deaf persons of the country. These people are obviously physically challenged. I understand that all sides of the House must have sympathy and definitely have responsibility towards them. Deaf people use hand signs to communicate, hence normal people face problem in recognizing their language by signs made. Hence there is a need of the systems which recognizes the different signs and conveys the information to the normal people.

To be very specific, there are some of the challenges face blind people:

- Lack of job opportunities: Job applications and interviews can be incredibly difficult to set up
- The hearing population feels sorry for them, and they don't need their sympathy.
- Education: Lack of sign language interpreters and teachers who now sign language
- Inaccessibility to information in public offices and private firms because most of the information is available in verbal mode.

1.2 DELIVERABLES

1. Hardware: None.
2. Software: A python-based application that make it easy to communicate with deaf people it’s designed for communicating with deaf people.

1.3 APPLICATION CONSTRAINTS

System constraints make rules and conventions commonly agreed to in each programming environment explicit and automatically checkable. System constraints are divided into 2 categories:

- Primary constraints: The application is subjected to the following four types of primary constraints, which must be satisfied by a solution to be considered as a valid one:
  1. Minimum system requirements must be satisfied.
  2. Libraries must be installed.
  3. Deaf person should make his hand out of his body.
  4. The room shouldn't be dark room.

- Secondary constraints: these are constraints that aren't great to concern but are still taken into consideration. They don't be satisfied but the solution will be better if they are satisfied:
  1. The room should be in a great illumination.
  2. The background of deaf person must be in one color.
  3. The room should be quite to get the voice correctly.

2 PLANNING

2.1 SCOPE INITIATION

This project seeks to make it easy to communicate with deaf people and make it easier to understand them quickly, the way which the project work is to convert sign language to text and sound that makes people understand quickly and vice versa.

The software consisting of two windows, The first window that is used from the first user containing the options to convert from sign language to text an audio , this window contains a frame to display the sign language animation that entered from the second user, another frame used to show the camera detection and hand tracking, a label display the characters and words that converted which equal to the sign language that entered from the user and finally three buttons , the first one is to create a space between signs, the second one is to start the camera on or make it off and converting from text to audio and the last one is to delete the last entered sign and the last converted letter.

The second window containing the options to convert from voice and text to sign language, this window contains a frame that display the first user camera, a label displays the text that converted from the sign language entered by the first user, a text box display the text converted from the recorded audio and finally three buttons, the first one is to clear the audio text, the second one is hold to record which the user use to record an audio and the last one is to send the output animation that converted from the text or audio to the first user.

2.1.1 Resource planning with Gantt chart

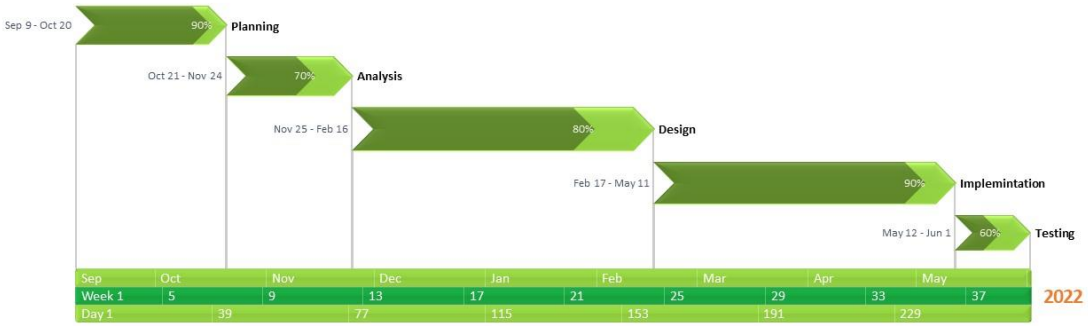


Figure 2.1-1:Gantt chart

Task	Start Date	End Date	Duration	Completion
Planning	9/10/2021	10/20/2021	40	90%
Analysis	10/21/2021	11/24/2021	34	70%
Design	11/25/2021	2/16/2022	83	80%
Implementation	2/17/2022	5/11/2022	83	90%
Testing	5/12/2022	6/1/2022	20	60%

Table 1

2.2 DEVELOPMENT REQUIREMENTS

- Cost estimating & Budgeting:
  -We used two tools from google which help to convert from audio to text and vice versa which allow the user a limited number of access to their data and this can be solved only by paying to get unlimited access.

  -The platform that allow us to use more GUI options gives us a limited number of features and needs to pay for it to use more features.

2.3 PROJECT RISKS

Ideally, sign recognition should be based on an image of a hand that's not moving showing only a single sign against a clear background in fine-lit conditions. But real-life conditions are never like that.

- **Google Servers**
  - Speech recognition and text-to-speech is handled by Google API, so if Google servers were to shut down, those two parts will stop working.
- **Internet Connection**
  - Poor internet quality will result in lag, both speech recognition and text-to-speech will be delayed, this affects the user experience.
- **Mic and camera quality**
  - Sign language recognition is heavily dependent on camera quality, noisy images will result in inaccurate sign detection. This also applies to mic quality in speech recognition.
- **Multiple hands in the image**
  - There can be more than one user communicating with sign language, which of them of the is the priority user?
- **Non-standard backgrounds**
  - Sign recognition is expected to bring good results no matter the background

2.4 SYSTEM REQUIREMENTS

1. Camera

   We used camera to detect hand gestures of Arabic alphabet sign language and translate it into a written sentence then convert it into Arabic voice.

2. Mic

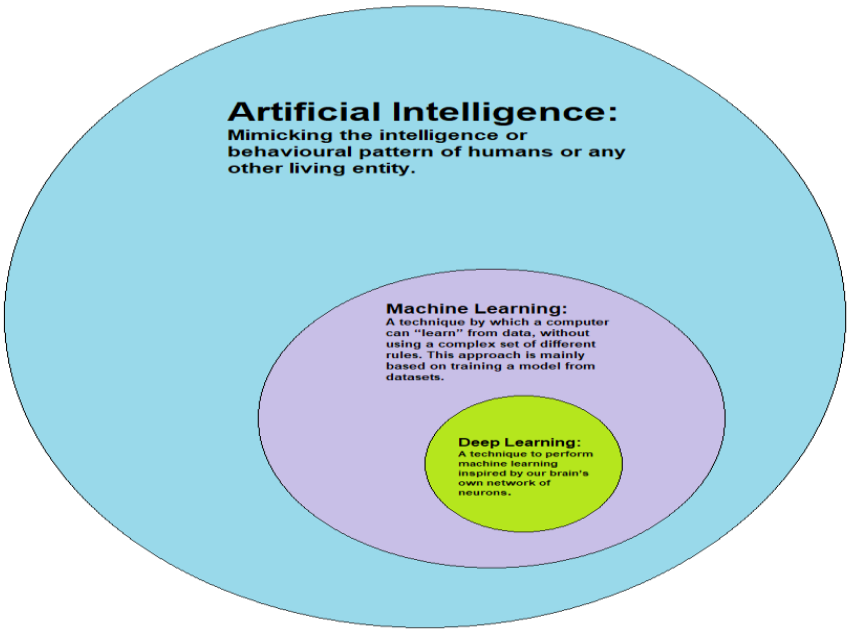
   Mic is used for voice recognition

2.5 TECHNOLOGIES.

Here we will discuss all what we used in the project and how does the project is implemented. We used many technologies to develop this project it is based on a python-based application the application is divided into two parts one is based on a machine learning model that can detect arabic alphabet sign language into arabic voice and the second part is animated arabic sign language in this part we used blender to make

2.5.1 artificial intelligence (AI)

the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings. The term is frequently applied to the project of developing systems endowed with the intellectual processes characteristic of humans, such as the ability to reason, discover meaning, generalize, or learn from past experience. Since the development of the digital computer in the 1940s, it has been demonstrated that computers can be programmed to carry out very complex tasks—as, for example, discovering proofs for mathematical theorems or playing chess—with great proficiency. Still, despite continuing advances in computer processing speed and memory capacity, there are for now no programs that can match human flexibility over wider domains or in tasks



requiring much everyday knowledge. On the other hand, some programs have attained the performance levels of human experts and professionals in performing certain specific tasks, so that artificial intelligence in this limited sense is found in applications as diverse as medical diagnosis, computer search engines, and voice or handwriting recognition.

Figure 2.5-1: AI

How does AI work?

As the hype around AI has accelerated, vendors have been scrambling to promote how their products and services use AI. Often what they refer to as AI is simply one component of AI, such as machine learning. AI requires a foundation of specialized hardware and software for writing and training machine learning algorithms. No one programming language is synonymous with AI, but a few, including Python, R and Java, are popular.

In general, AI systems work by ingesting large amounts of labeled training data, analyzing the data for correlations and patterns, and using these patterns to make predictions about future states. In this way, a chatbot that is fed examples of text chats can learn to produce lifelike exchanges with people, or an image recognition tool can learn to identify and describe objects in images by reviewing millions of examples.

2.5.2 Machine Learning

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.

Why is machine learning important?

Machine learning is important because it gives enterprises a view of trends in customer behaviour and business operational patterns, as well as supports the development of new products. Many of today's leading companies, such as Facebook, Google, and Uber, make machine learning a central part of their operations. Machine learning has become a significant competitive differentiator for many companies.

What are the different types of machine learning?

Classical machine learning is often categorized by how an algorithm learns to become more accurate in its predictions. There are four basic approaches: supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning. The type of algorithm data scientists choose to use depends on what type of data they want to predict.

Supervised learning: In this type of machine learning, data scientists supply algorithms with labelled training data and define the variables they want the algorithm to assess for correlations. Both the input and the output of the algorithm is specified.

Unsupervised learning: This type of machine learning involves algorithms that train on unlabelled data. The algorithm scans through data sets looking for any meaningful connection. The data that algorithms train on as well as the predictions or recommendations they output are predetermined.

Semi-supervised learning: This approach to machine learning involves a mix of the two preceding types. Data scientists may feed an algorithm mostly labelled training data, but the model is free to explore the data on its own and develop its own understanding of the data set.

Reinforcement learning: Data scientists typically use reinforcement learning to teach a machine to complete a multi-step process for which there are clearly defined rules. Data scientists program an algorithm to complete a task and give it positive or negative cues as it works out how to complete a task. But for the most part, the algorithm decides on its own what steps to take along the way.

How does supervised machine learning work?

Supervised machine learning requires the data scientist to train the algorithm with both labelled inputs and desired outputs. Supervised learning algorithms are good for the following tasks:

1. Binary classification: Dividing data into two categories.
2. Multi-class classification: Choosing between more than two types of answers.
3. Regression modelling: Predicting continuous values.
4. Ensembling: Combining the predictions of multiple machine learning models to produce an accurate prediction.

2.5.3 deep learning

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behaviour of the human brain—albeit far from matching its ability—allowing it to "learn" from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars).

How deep learning works?

Deep learning neural networks, or artificial neural networks, attempts to mimic the human brain through a combination of data inputs, weights, and bias. These elements work together to accurately recognize, classify, and describe objects within the data.

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This

progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called visible layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.

Another process called backpropagation uses algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model. Together, forward propagation and backpropagation allow a neural network to make predictions and correct for any errors accordingly. Over time, the algorithm becomes gradually more accurate.

The above describes the simplest type of deep neural network in the simplest terms. However, deep learning algorithms are incredibly complex, and there are different types of neural networks to address specific problems or datasets. For example,

Convolutional neural networks (CNNs), used primarily in computer vision and image classification applications, can detect features and patterns within an image, enabling tasks, like object detection or recognition. In 2015, a CNN bested a human in an object recognition challenge for the first time.

Recurrent neural network (RNNs) are typically used in natural language and speech recognition applications as it leverages sequential or times series data

2.5.4 Computer Vision

Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos, and other visual inputs — and take actions or make recommendations based on that information. If AI enables computers to think, computer vision enables them to see, observe and understand.

Computer vision works much the same as human vision, except humans have a head start. Human sight has the advantage of lifetimes of context to train how to tell objects apart, how far away they are, whether they are moving and whether there is something wrong in an image.

Computer vision trains machines to perform these functions, but it has to do it in much less time with cameras, data and algorithms rather than retinas, optic nerves and a visual cortex. Because a system trained to inspect products or watch a production asset can analyse thousands of products or processes a minute, noticing imperceptible defects or issues, it can quickly surpass human capabilities.

How does computer vision work?

Computer vision needs lots of data. It runs analyses of data over and over until it discerns distinctions and ultimately recognize images. For example, to train a computer to recognize automobile tires, it needs to be fed vast quantities of tire images and tire-related items to learn the differences and recognize a tire, especially one with no defects.

Two essential technologies are used to accomplish this: a type of machine learning called deep learning and a convolutional neural network (CNN).

Machine learning uses algorithmic models that enable a computer to teach itself about the context of visual data. If enough data is fed through the model, the computer will "look" at the data and teach itself to tell one image from another. Algorithms enable the machine to learn by itself, rather than someone programming it to recognize an image.

A CNN helps a machine learning or deep learning model "look" by breaking images down into pixels that are given tags or labels. It uses the labels to perform convolutions (a mathematical operation on two functions to produce a third function) and makes predictions about what it is "seeing." The neural network runs convolutions and checks the accuracy of its predictions in a series of iterations until the predictions start to come true. It is then recognizing or seeing images in a way similar to humans.

Much like a human making out an image at a distance, a CNN first discerns hard edges and simple shapes, then fills in information as it runs iterations of its predictions. A CNN is used to understand single images. A recurrent neural network (RNN) is used in a similar way for video applications to help computers understand how pictures in a series of frames are related to one another.

2.5.5 MediaPipe

What is MediaPipe?

MediaPipe is a Framework for building machine learning pipelines for processing time-series data like video, audio, etc. This cross-platform Framework works in Desktop/Server, Android, iOS, and embedded devices like Raspberry Pi and Jetson.

What is MediaPipe framework?

MediaPipe is Google's open-source framework, used for media processing. It is cross-platform, or we can say it is platform friendly. It is run on Android, iOS, web, and YouTube servers that's what Cross-platform means, to run everywhere.

How does MediaPipe hand detection work?

It employs machine learning (ML) to infer 21 3D landmarks of a hand from just a single frame. Whereas current state-of-the-art approaches rely primarily on powerful desktop environments for inference, our method achieves real-time performance on a mobile phone, and even scales to multiple hands.

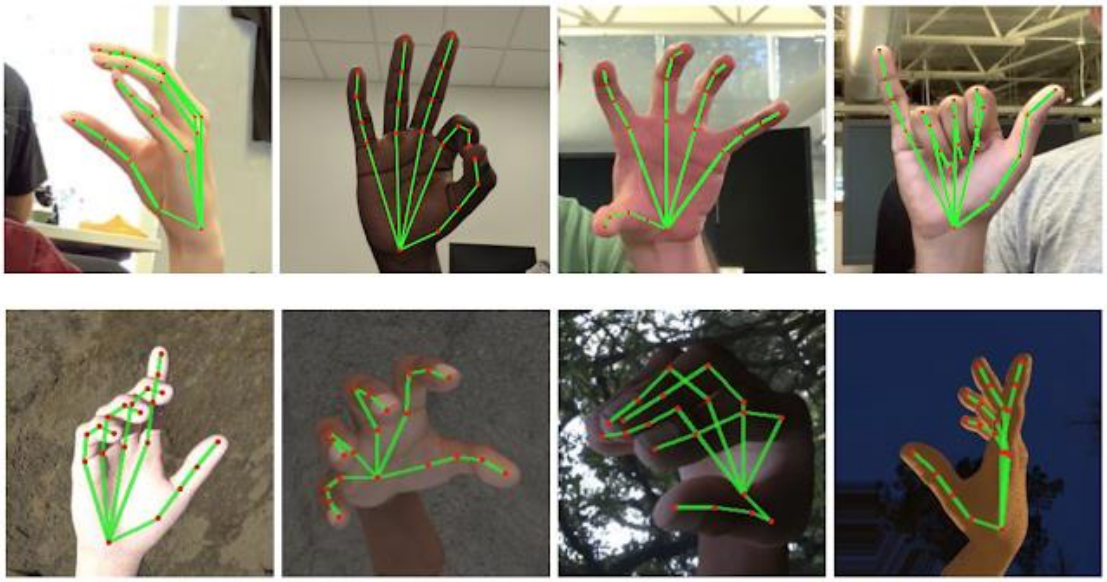
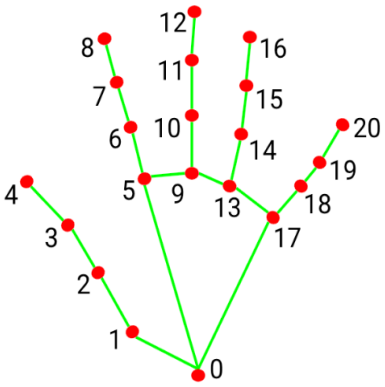


Figure 2.5-2:MediaPipe



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

Figure 2.5-3:MediaPipe

2.5.6 Graphical User Interfaces with Tk

Tkinter is an open source, portable graphical user interface (GUI) library designed for use in Python scripts. Tkinter relies on the Tk library, the GUI library used by Tcl/Tk and Perl, which is in turn implemented in C. Therefore, Tkinter can be said to be implemented using multiple layers. Several competing GUI toolkits are available to use with the Python language.

What are Tkinter Advantages?

1-Layered approach

The layered approach used in designing Tkinter gives Tkinter all of the advantages of the TK library. Therefore, at the time of creation, Tkinter inherited from the benefits of a GUI toolkit that had been given time to mature. This makes early versions of Tkinter a lot more stable and reliable than if it had been rewritten from scratch. Moreover, the conversion from Tcl/Tk to Tkinter is really trivial, so that Tk programmers can learn to use Tkinter very easily.

2-Accessibility

Learning Tkinter is very intuitive, and therefore quick and painless. The Tkinter implementation hides the detailed and complicated calls in simple, intuitive methods. This is a continuation of the Python way of thinking since the language excels at quickly building prototypes. It is therefore expected that its preferred GUI library be implemented using the same approach. For example, here is the code for a typical "Hello world"-like application:

```
from Tkinter import *
root = Tk( )
root.title("A simple application")
root.mainloop( )
```

The first 2 lines allow to create a complete window. Compared to MFC programming, it makes no doubt that Tkinter is simple to use. The third line sets the caption of the window, and the fourth one makes it enter its event loop.

3-Portability

Python scripts that use Tkinter do not require modifications to be ported from one platform to the other. Tkinter is available for any platform that Python is implemented for, namely Microsoft Windows, X Windows, and Macintosh. This gives it a great advantage over most competing libraries, which are often restricted to one or two platforms. Moreover, Tkinter will provide the native look-and-feel of the specific platform it runs on.

4-Availability

Tkinter is now included in any Python distribution. Therefore, no supplementary modules are required in order to run scripts using Tkinter.

2.5.7 Convolutional Neural Network

A convolutional neural network is a feed-forward neural network that is generally used to analyse visual images by processing data with grid-like topology. It's also known as a ConvNet. A convolutional neural network is used to detect and classify objects in an image.

How does it work?

The convolutional Neural Network CNN works by getting an image, designating it some weightage based on the different objects of the image, and then distinguishing them from each other. CNN requires very little pre-process data as compared to other deep learning algorithms. One of the main capabilities of CNN is that it applies primitive methods for training its classifiers, which makes it good enough to learn the characteristics of the target object.

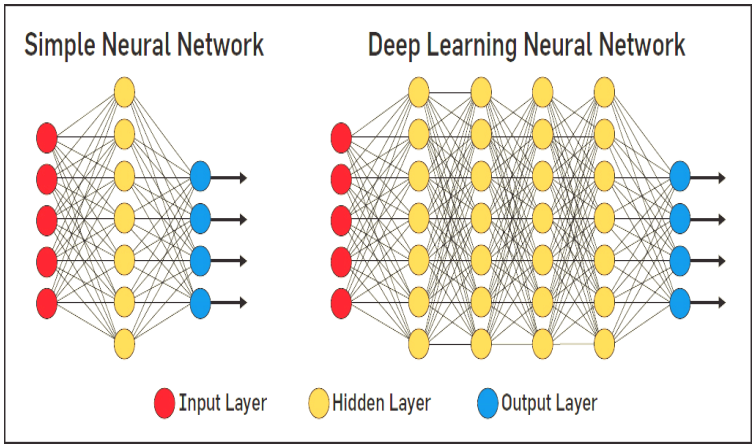


Figure 2.5: Deep Learning

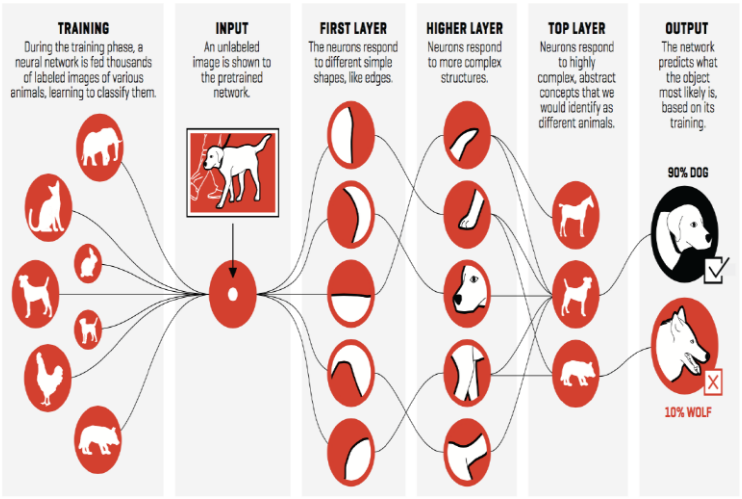


Figure 2.5-4:CNN

Here's how exactly CNN recognizes:
• The pixels from the image are fed to the convolutional layer that performs the convolution operation
• It results in a convolved map
• The convolved map is applied to a ReLU function to generate a rectified feature map
• The image is processed with multiple convolutions and ReLU layers for locating the features
• Different pooling layers with various filters are used to identify specific parts of the image
• The pooled feature map is flattened and fed to a fully connected layer to get the final output

Layers in a Convolutional Neural Network
A convolution neural network has multiple hidden layers that help in extracting information from an image. The four important layers in CNN are:

1. Convolution layer
2. ReLU layer
3. Pooling layer
4. Fully connected layer

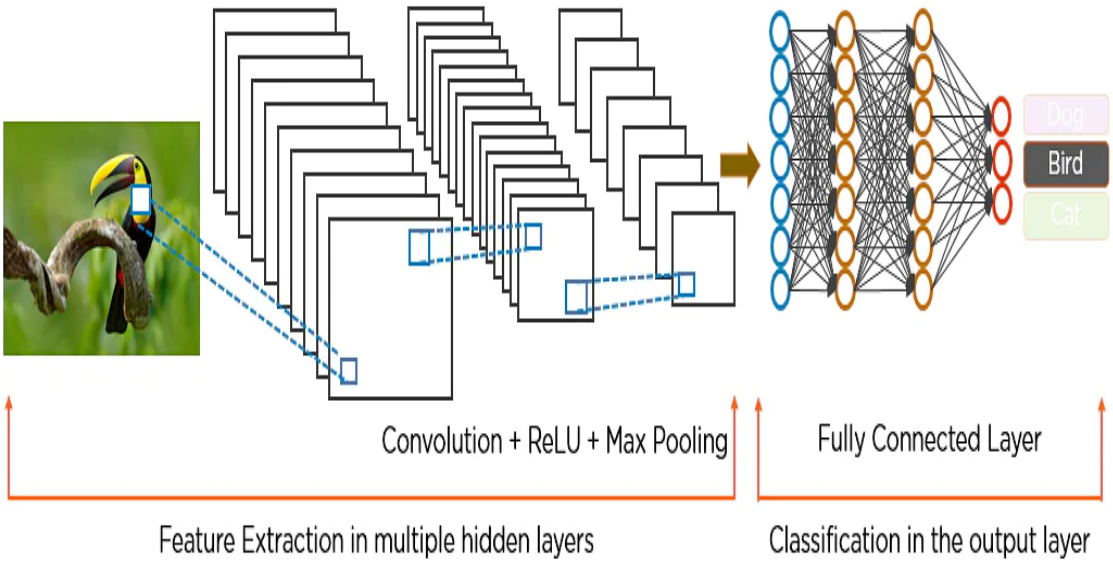


Figure 2.5-5:CNN Layers

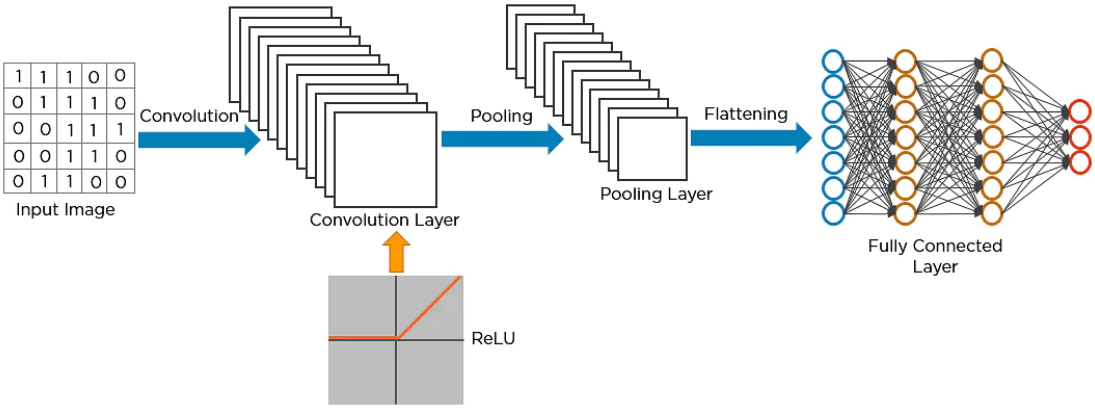


Figure 2.5-6:CNN Layers

2.5.8 Blender

Blender is a free and open-source 3D computer graphics software toolset used for creating animated films, visual effects, art, 3D-printed models, motion graphics, interactive 3D applications, virtual reality, and, formerly, video games. Blender's features include 3D modelling, UV mapping, texturing, digital drawing, raster graphics editing, rigging, and skinning, fluid and smoke

simulation, particle simulation, soft body simulation, sculpting, animation, match moving, rendering, motion graphics, video editing, and compositing.

Usage

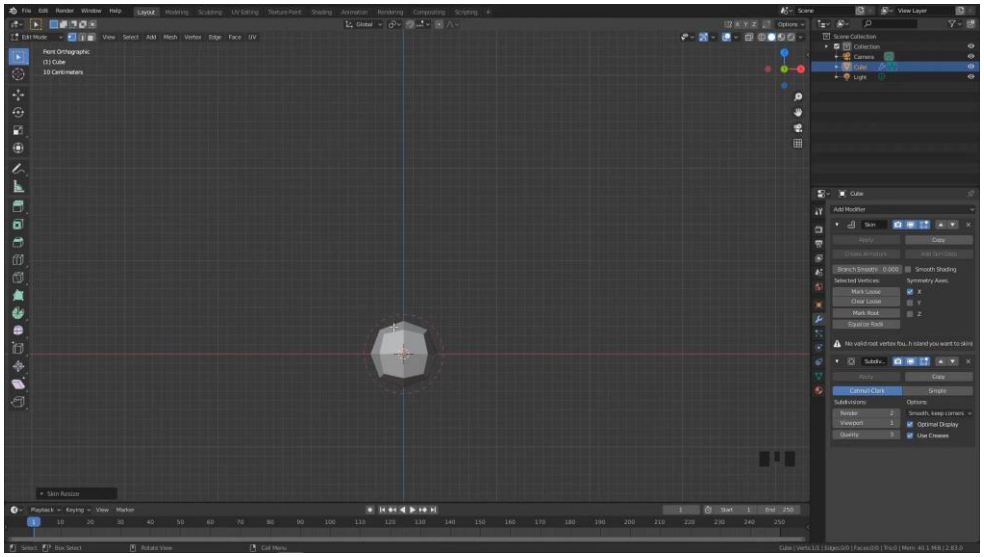
Figure 2.5-7:Creating the 3D model

Blender is a good tool overall for dealing and making 3D models and animating them, Blender was used in this project to make sign-language hand gestures animations, making animations consists of 5 parts:

1. Creating the model
2. Rigging
3. Shading
4. Animating
5. Rendering the results

Creating the model

First, we start from an arbitrary object like a cube or a circle.



Then the sculpting part: Sculpting is a feature that allows you to edit clay in Blender as if you were sculpting it with your own hands or using tools to make its surface uneven.

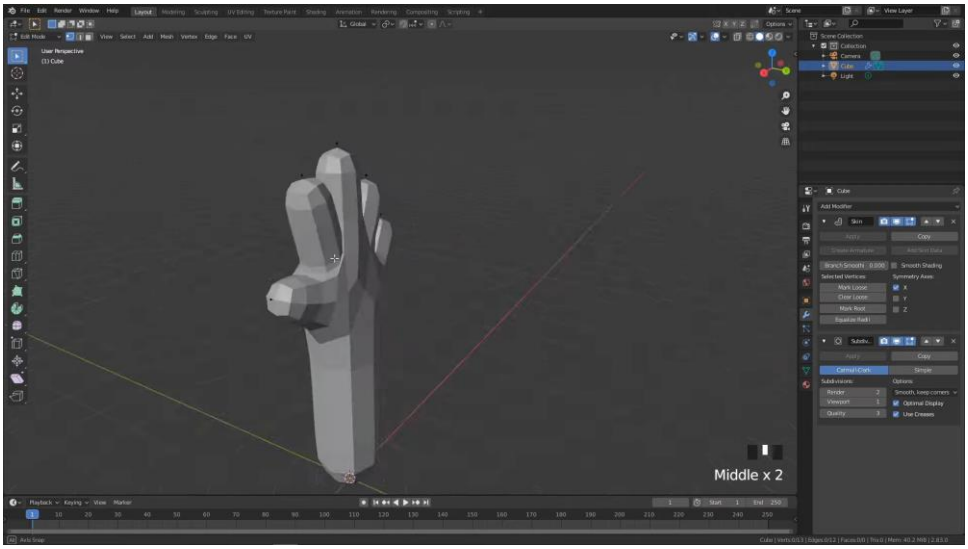


Figure 2.5-8:Creating the 3D model: Sculpting

Then the smoothing part to give it a more natural look.

Figure 2.5-9:Creating the 3D model: Smoothing

And that is the final model.

Rigging

Rigging is one part of the larger animation process.

After a 3D model has been created, a series of bones is constructed representing the skeletal structure. For instance, in a character there may be a group of back bones, a spine, and head bones.

These bones can be transformed using digital animation software meaning their position, rotation, and scale can be changed.

By recording these aspects of the bones along a timeline (using a process called keyframing) animations can be recorded.

Basically, to be able to animate our hand 3d model, there must be some points representing the joints, so we can move, rotate, and scale these points.

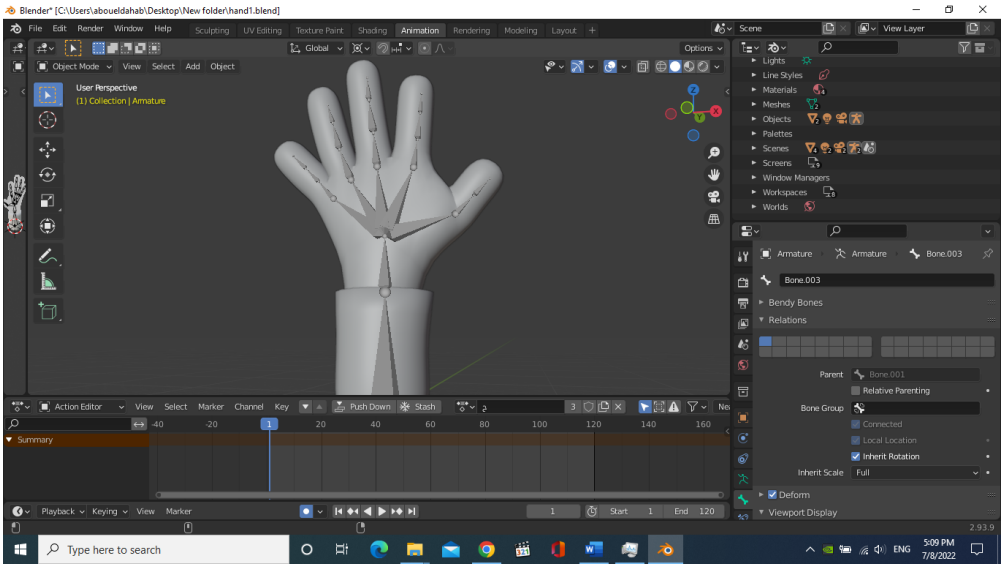


Figure 2.5-10:Rigging

The bones and joints, As demonstrated by the previous figure.

After that comes the animating process, it's done by moving and rotating the bones.

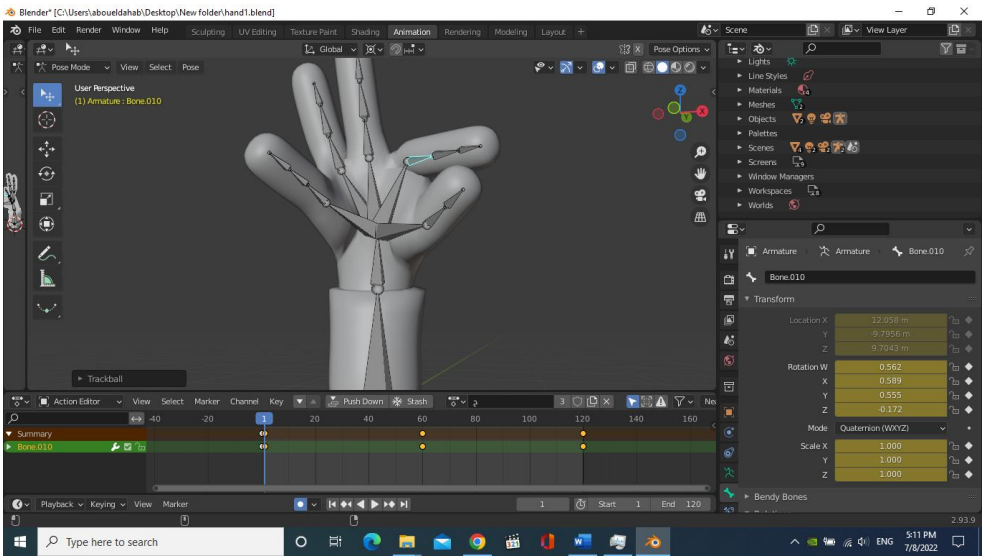


Figure 2.5-11:Rigging: Moving.

We can see here the result of rotating some bones.

Shading

shading refers to the process of altering the colour of an object/surface/polygon in the 3D scene, based on things like (but not limited to) the surface's angle to lights, its distance from lights, its angle to the camera and material properties. Shading basically gives 3d objects colour and tell 3d objects how to interact with light.

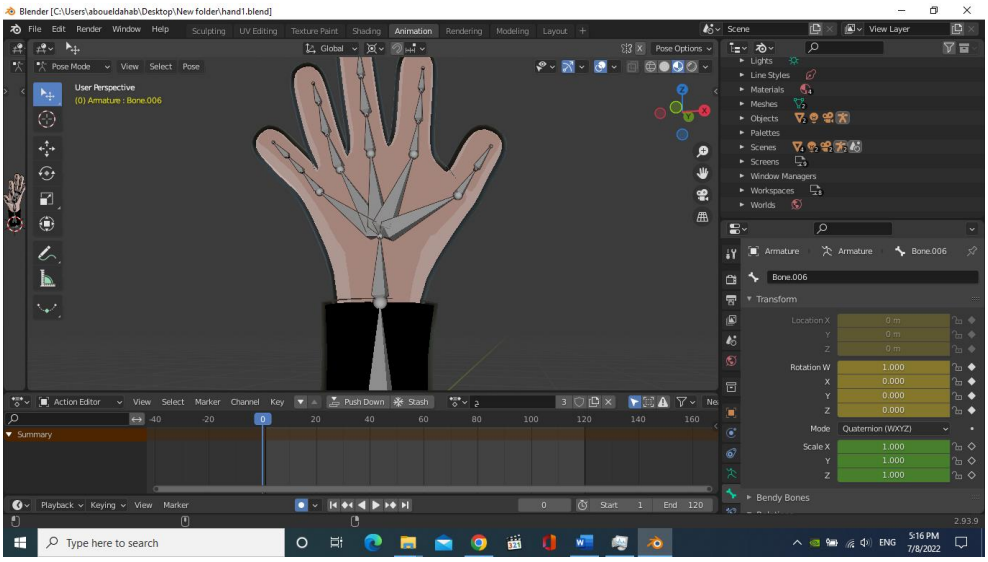


Figure 2.5-12:Shading

The result of shading, as demonstrated by this figure.

Now our model is ready for animating.

Animating

Animation is the process of taking a 3D object and getting it to move.

The animation of 3D model is done by manipulating bones' location, rotation, and scale, on a frame-by-frame basis.

Here in this figure, we see the animation process of the letter "☒"

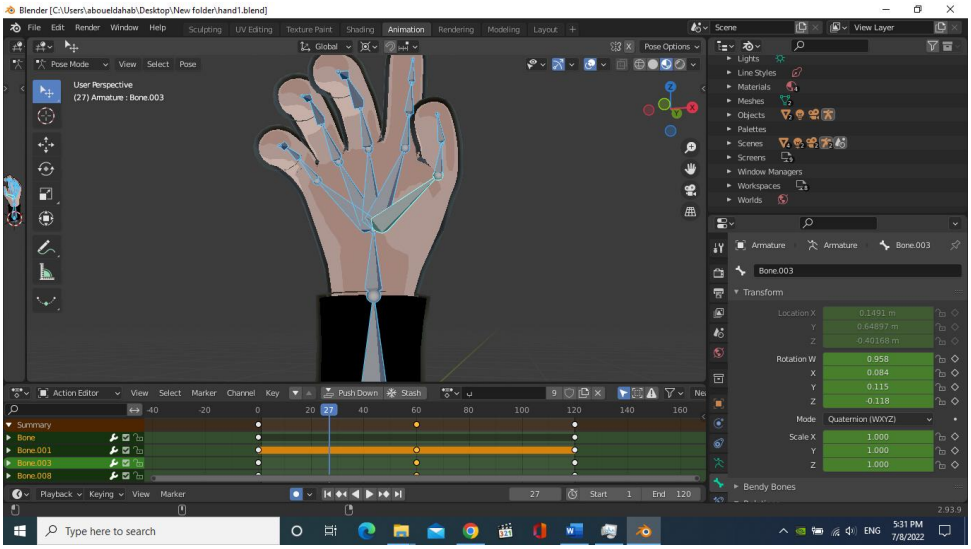


Figure 2.5-13:Animating-1

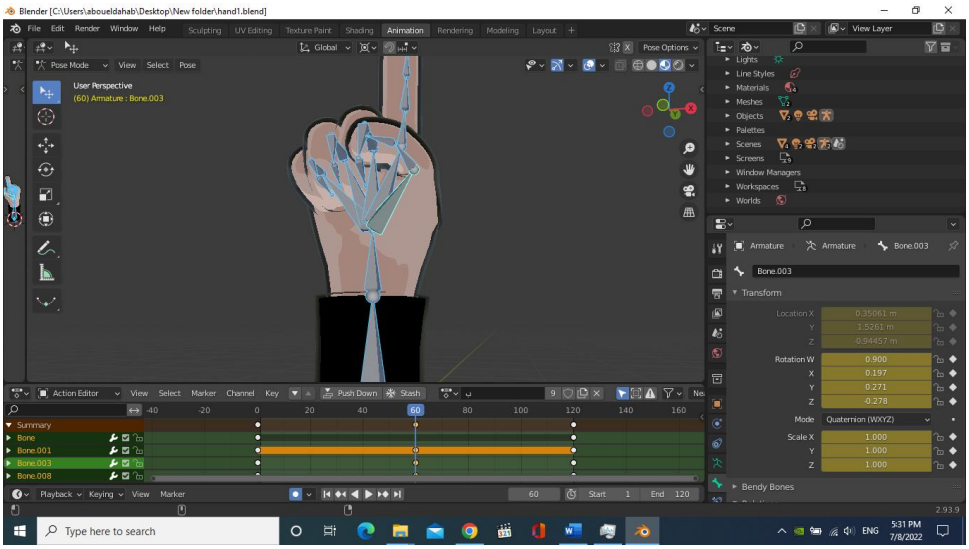


Figure 2.5-14:Animating-2

And that is the final result of animating.

Rendering

Rendering is converting the 3D model animation sequences into 2D photos, then package those photos together into files and use them in our software.

3 PROJECT ANALYSIS AND DESIGN

3.1 USE-CASE

The use case diagram can summarize the details of system's users (also known as actors) and their interactions with the system. The use case diagram can help to discuss and represent:
Scenarios in which the application interacts with people or external systems. Goals that your system or application helps those entities (known as actors) achieve. The scope of the system

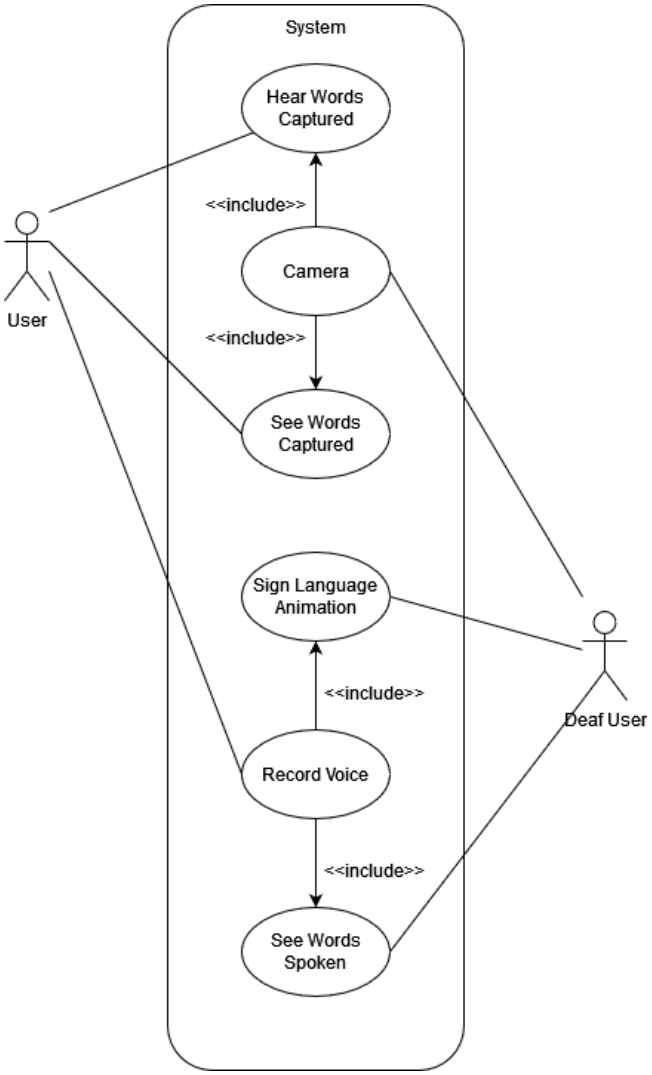


Figure 3.1-1: Use-Case diagram

3.1.1 DEAF user communicating with a user

ID:	**1**
Title:	Deaf User communicating with a user
Description:	Deaf user will use sign language in front of the camera, and it'll get translated in real-time to words on the screen and voice.
Primary Actor:	Deaf user.
Normal flow:	1. Deaf user will use the sign to text and voice mode. 2. Deaf user will communicate in sign language in front of the camera. 3. Sign language will be recognized and translated to voice in real-time.
Alternative flow:	Some sign language letters might get recognized incorrectly.

Table 2:Use-Case Deaf User

3.1.2 User communicating with a deaf user

ID:	**2**
Title:	User communicating with a deaf user
Description:	user will use their voice with the voice-to-sign-language mode, and words will be animated in real-time to sign language.
Primary Actor:	User.
Normal flow:	1. User will use voice-to-sign-language. 2. User will communicate with their voice. 3. words will be animated in real-time to sign language.
Alternative flow:	Some words might get recognized incorrectly.

Table 3:Use-Case User

3.2 CONTEXT DIAGRAM

Context diagram outlines how external entities interact with an internal software system. It's primarily used to help businesses wrap their heads around the scope of a system. As a result, they can figure out how best to design the system and its requirements or how to improve an existing system.
Context diagrams are high-level diagrams, meaning they don't go into the detailed ins and outs of the system. Instead, they map out an entire system in a way that's simple, clear, and easy to understand.

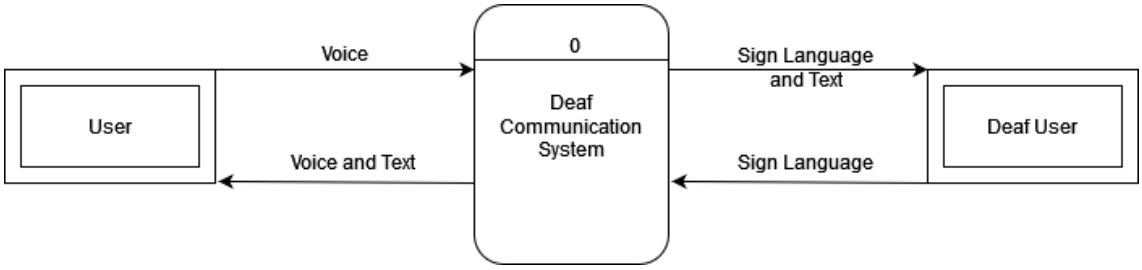


Figure 3.2-1:Context Diagram

3.3 DATAFLOW DIAGRAM

The data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. They can be used to analyse the system. Like all the best diagrams and charts, a DFD can often visually "say" things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That's why DFDs remain so popular after all these years.

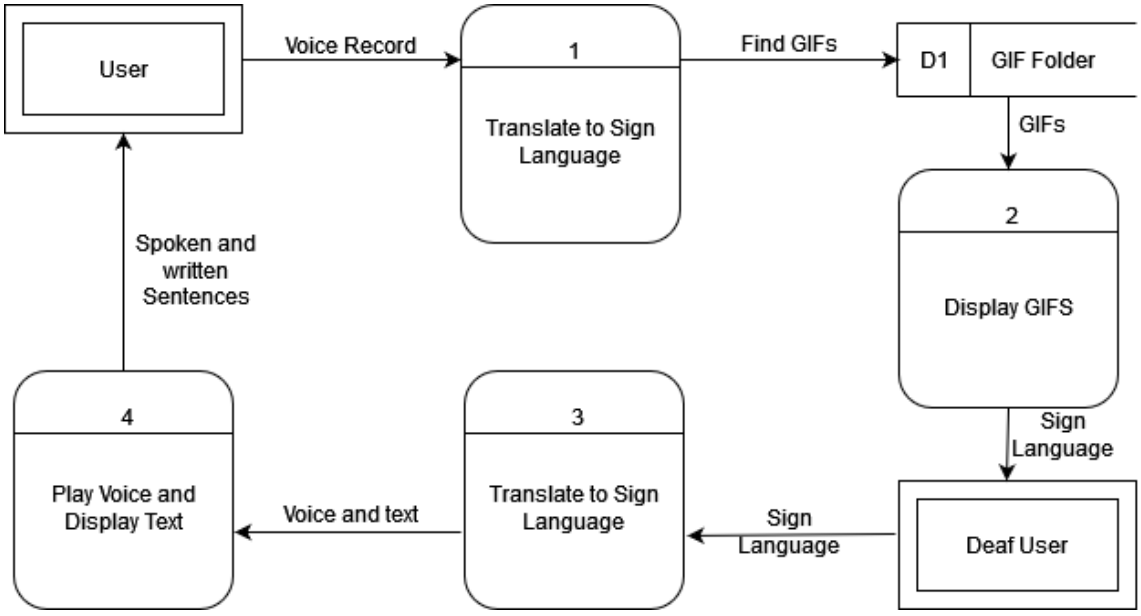


Figure 3.3-1:Dataflow Diagram

3.4 SEQUENCE DIAGRAM

Sequence Diagram is interaction diagram that detail how operations are carried out. It captures the interaction between objects in the context of a collaboration. Sequence Diagram is time focus and it show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

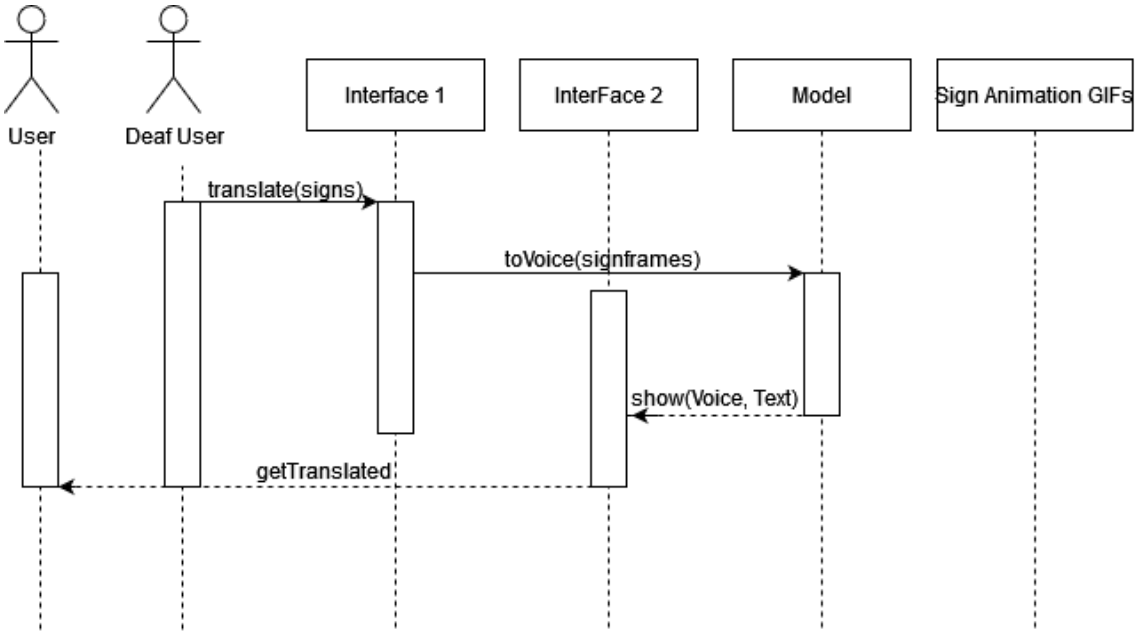


Figure 3.4-1:Sequence Diagram

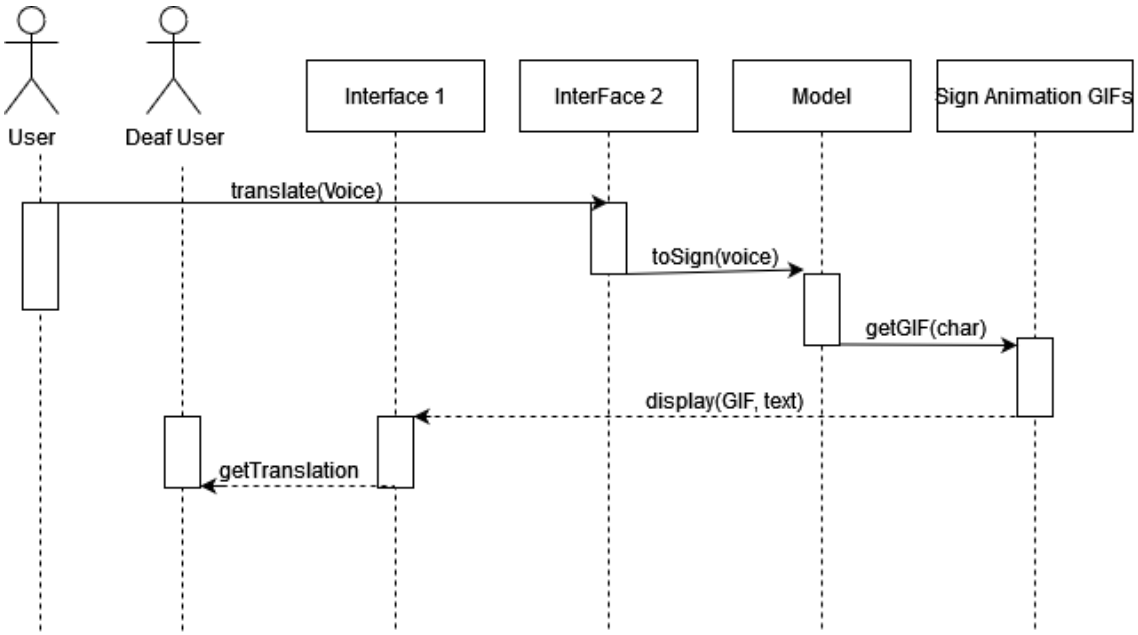


Figure 3.4-2:Sequence Diagram

3.5 ACTIVITY DIAGRAM

The activity diagram visually presents a series of actions or flow of control in a system. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent. In both cases an activity diagram will have a beginning (an initial state) and an end (a final state).

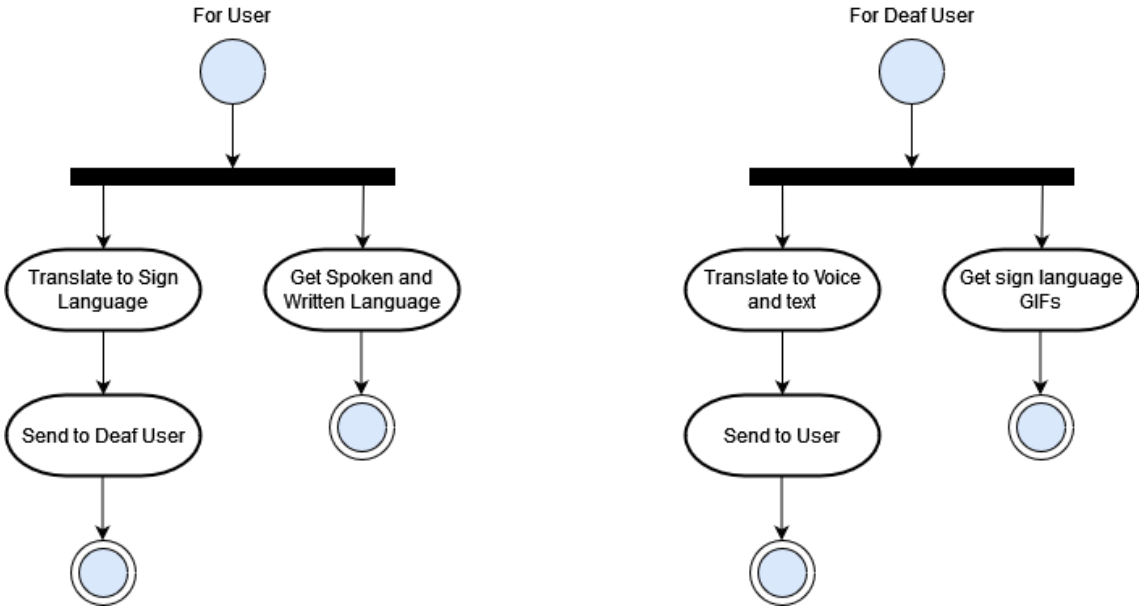


Figure 3.5-1:Activity Diagram

4 THE APPLICATION

4.1 TKINTER GUI

The application has two main windows one for the deaf and other for anyone who can't understand sign language.

First page "deaf page":

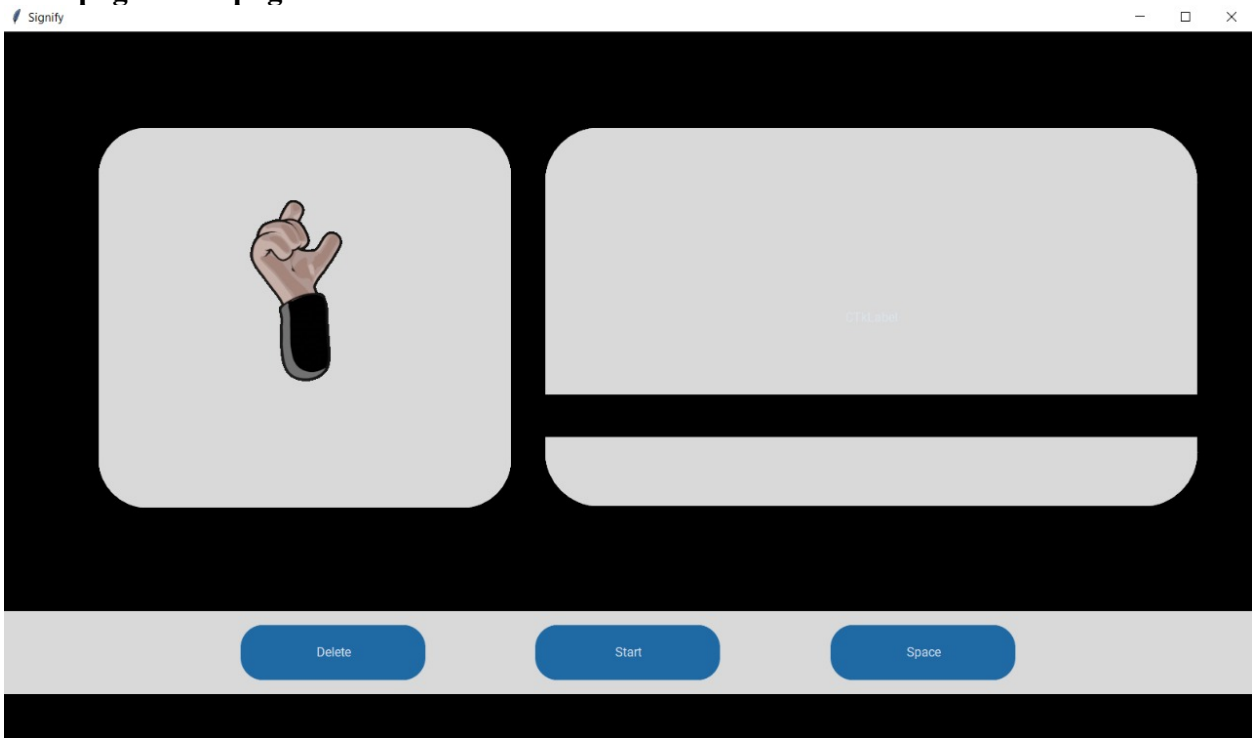


Figure 4.1-1 First page

Here the Deaf can start making alphabet signs and then send them as a voice record easily to understand. Or can receive a stream of alphabet sign that is send from the other person.

figure 4.1-2 first page buttons

This page has three buttons that allows the deaf user can delete, start, send or make a space.

1. First button allows deaf user to delete the last character of the sentence.
2. Second button allows deaf user to start the camera that obtains the signed characters and then make a complete sentence then can press for a second time to send a voice message So, it will be easy to recognize.
3. We couldn't find a sign that split every word from the other So, the third button allow the deaf user to make a space between every word

Figure 4.1-3 camera container 1

In this container deaf user can start camera in it and start making alphabet signs that make a sentence to make it easy to be send a voice record to the user who don't recognize sign language.
It has a subtitle label to write the sentence on it and send it to the other window.

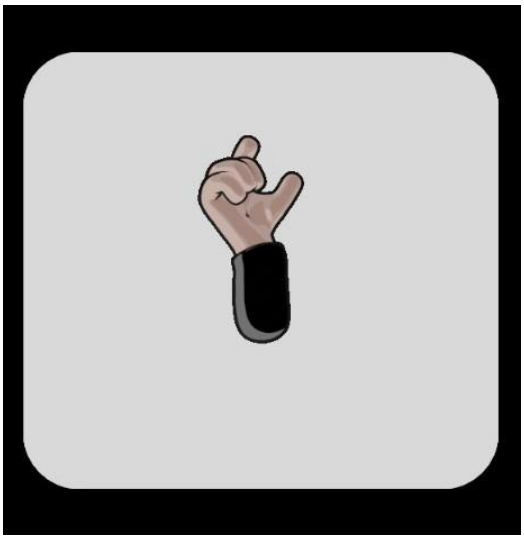


Figure 4.1-4 animation container

We made an alphabet sign animation for all Arabic alphabets So, we use them in this container to use it in describing the spoken sentences to make it easy for deaf understanding the spoken words.

Second page “other user page”:

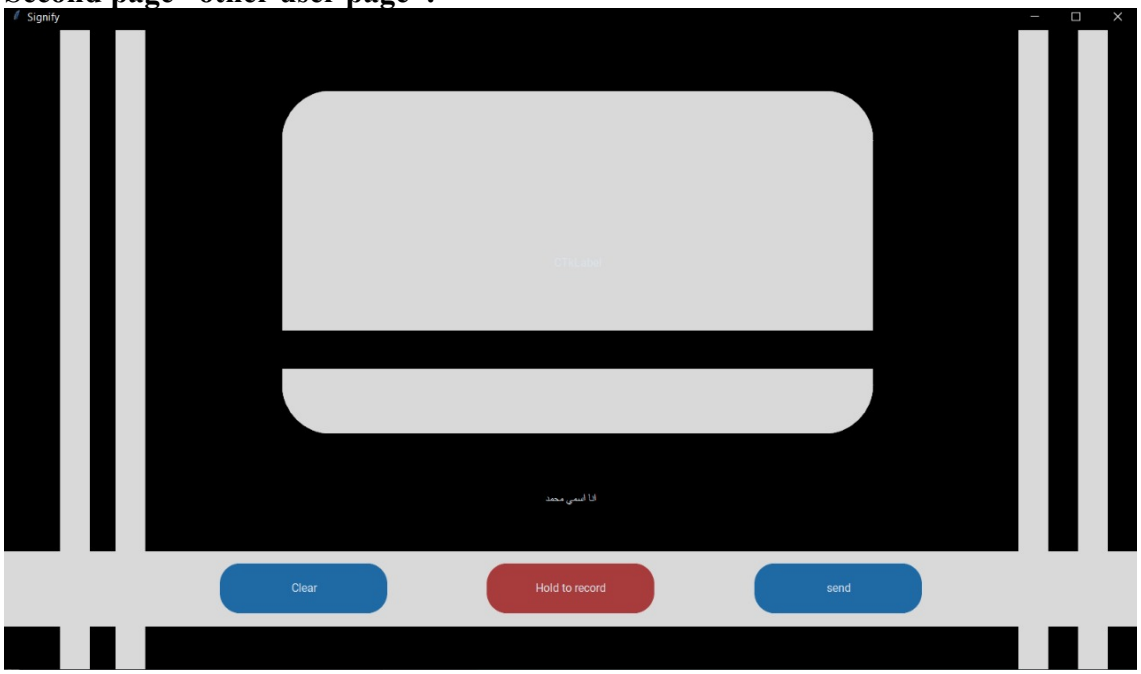


Figure 4.1-5 second page

Here in this page the other user can see the deaf one and his subtitle. He can start recording the voice and it will be written on a label in the center of the screen then he can send the animated arabic alphabet sign language.

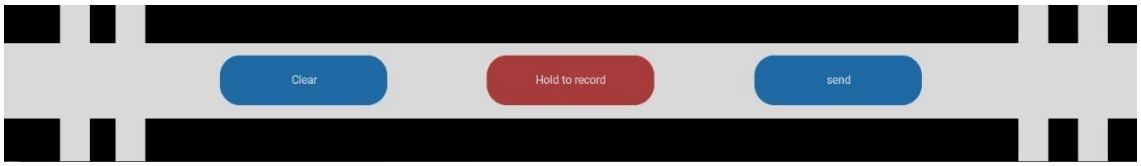


Figure 4.1-6 second page buttons

This page has three buttons that allows the deaf user can clear, start the record or send the record as animations.

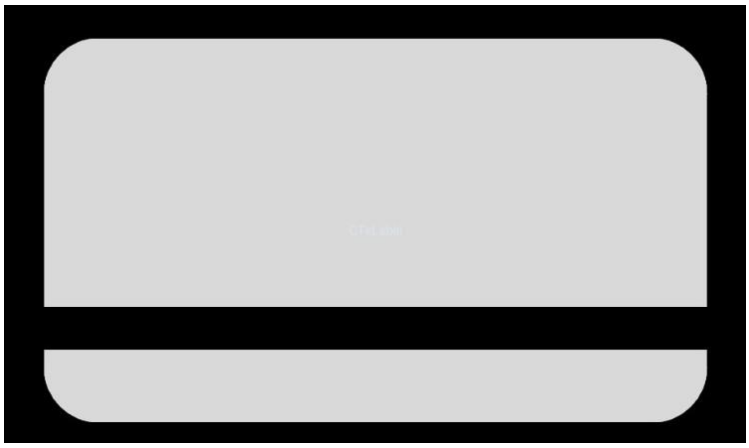


Figure 4.1-7 camera container 2

It has the same container that shows the deaf user to the other user to make better communication.

4.2 HOW WE DETECT THE HAND BY MEDIAPIPE

This is the whole pipeline of the image from the camera to the output.

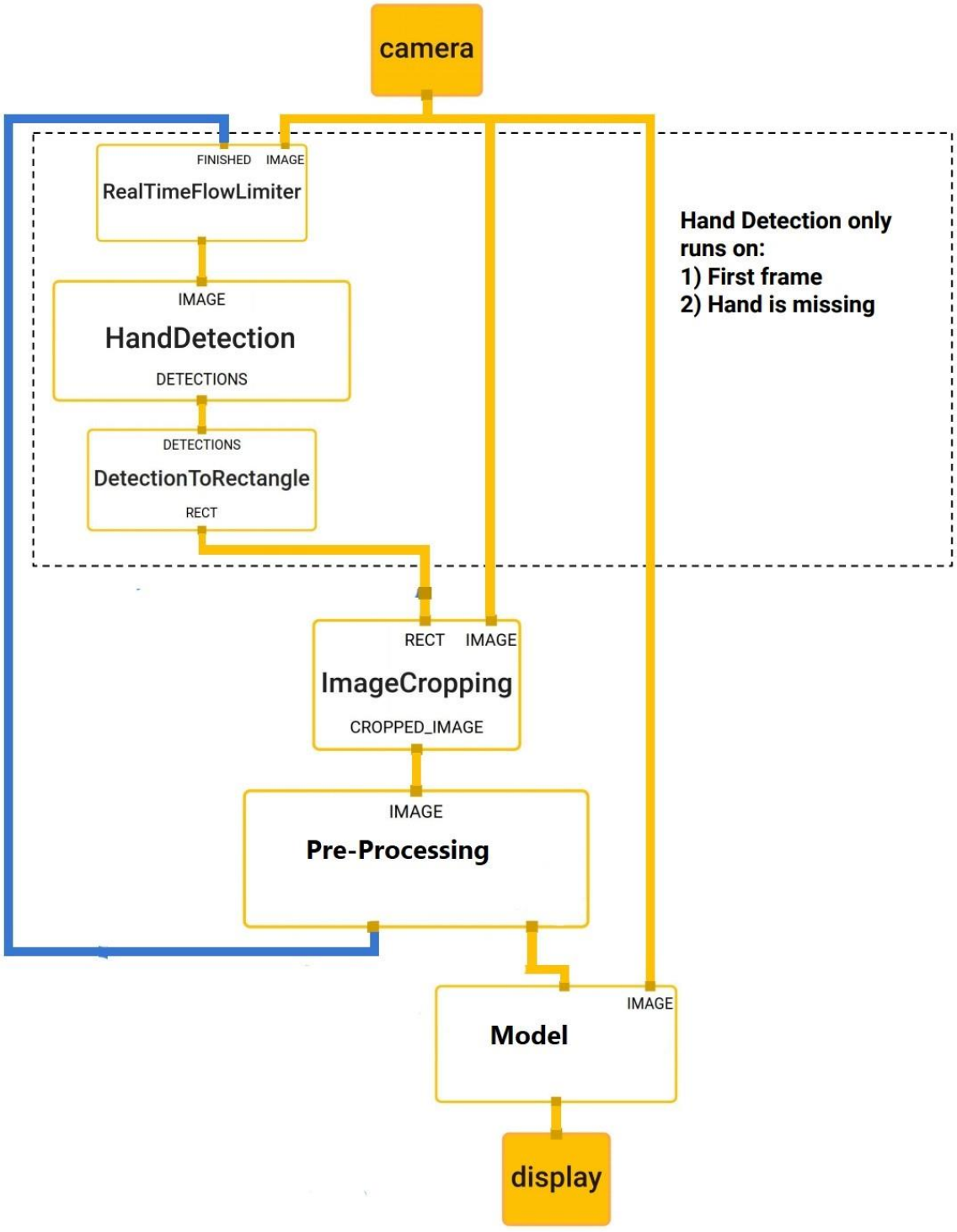


Figure 4.2-1 image pipline

Hand detection is the process in which a computer uses computer vision to detect a hand from an input image and keeps focus on the hand's movement and orientation. Hand detection allows us to develop numerous programs that use hand movement and orientation as their input.

4.2.1 Step 1 - Capturing an image input and processing it

The code below takes the image input from the webcam. It then converts the image from BGR to RGB. This is because MediaPipe only works with RGB images, not BGR.
It then processes the RGB image to identify the hands in the image:

```
while True:
    success, image = cap.read()
    imageRGB = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    results = hands.process(imageRGB)
```

Figure 4.2-2 mediapipe code

4.2.2 Step 2 - Working with each hand

```
# checking whether a hand is detected
if results.multi_hand_landmarks:
    for handLms in results.multi_hand_landmarks: # working w
        for id, lm in enumerate(handLms.landmark):
            h, w, c = image.shape
            cx, cy = int(lm.x * w), int(lm.y * h)
```

Figure 4.2-3 mediapipe code2

In the code above, we use the if statement to check whether a hand is detected. We then use the first for loop to enable us work with one hand at a time.

The second for loop helps us get the hand landmark information which will give us the x and y coordinates of each listed point in the hand landmark diagram. This loop will also give us the id of each point.

We will then find the height, width, and channel of our image using the image. shape function. We finally get the central positions of the identified hand points.

4.2.3 Step 3 - Drawing the hand landmarks and hand connections on the hand image

```
        if id == 20 :
            cv2.circle(image, (cx, cy), 25, (255, 0, 255), cv2.FILLED)

    mpDraw.draw_landmarks(image, handLms, mpHands.HAND_CONNECTIONS)
```

Figure 4.2-4 mediapipe code3

In the code above, we circle the hand point number 20. This is the tip of the pinkie finger.

Feel free to use the number of the hand point you want to circle as they are listed on the hand landmark diagram. We then draw the hand landmarks and the connections between them on the input image.

4.3 SIGN LANGUAGE DETECTION MODEL

4.3.1 Vgg16:

It is a Convolutional Neural Network (CNN) model proposed by Karen Simonyan and Andrew Zisserman at the University of Oxford. The idea of the model was proposed in 2013, but the actual model was submitted during the ILSVRC ImageNet Challenge in 2014. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was an annual competition that evaluated algorithms for image classification (and object detection) at a large scale.

4.3.2 VGG – The Idea

The model proposed the use of a very small 3 x 3 receptive field (filters) throughout the entire network with the stride of 1 pixel.

The idea behind using 3 x 3 filters uniformly is something that makes the VGG stand out. Two consecutive 3 x 3 filters provide for an effective receptive field of 5 x 5. Similarly, three 3 x 3 filters make up for a receptive field of 7 x 7. This way, a combination of multiple 3 x 3 filters can stand in for a receptive area of a larger size.

In addition to the three convolution layers, there are also three non-linear activation layers instead of a single one you would have in 7 x 7. This makes the decision functions more discriminative. It would impart the ability to the network to converge faster.

It also reduces the number of weight parameters in the model significantly. Assuming that the input and output of a three-layer 3 x 3 convolutional stack have C channels, the total number of weight parameters will be 3 * 32 C2 = 27 C2. If we compare this to a 7 x 7 convolutional layer, it would require 72 C2 = 49 C2, which is almost twice the 3 x 3 layers. Additionally, this can be seen as a regularization on the 7 x 7 convolutional filters forcing them to have a decomposition through the 3 x 3 filters, with, of course, the non-linearity added in-between by means of ReLU activations. This would reduce the tendency of the network to over-fit during the training exercise.

The consistent use of 3 x 3 convolutions across the network made the network very simple, elegant, and easy to work with.

4.3.3 VGG Configurations

A stack of multiple (usually 1, 2, or 3) convolution layers of filter size 3 x 3, stride one, and padding 1, followed by a max-pooling layer of size 2 x 2, is the basic building block for all of these configurations. Different configurations of this stack were repeated in the network configurations to achieve different depths. The number associated with each of the configurations is the number of layers with weight parameters in them.

The convolution stacks are followed by three fully connected layers, two with size 4,096 and the last one with size 1,000. The last one is the output layer with Softmax activation. The size of 1,000 refers to the total number of possible classes in ImageNet.

VGG16 refers to the configuration “D” in the table listed below. The configuration “C” also has 16 weight layers. However, it uses a 1 x 1 filter as the last convolution layer in stacks 3, 4, and 5. This layer was used to increase the non-linearity of the decision functions without affecting the receptive field of the layer.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 4.3-1 VGG16 layers

The left-most “A” configuration is called VGG11, as it has 11 layers with weights – primarily the convolution layers and fully connected layers. As we go right from left, more and more convolutional layers are added, making them deeper and deeper. Please note that the ReLU activation layer is not indicated in the table. It follows every convolutional layer.

4.3.4 VGG 16 Architecture

Of all the configurations, VGG16 was identified to be the best performing model on the ImageNet dataset.

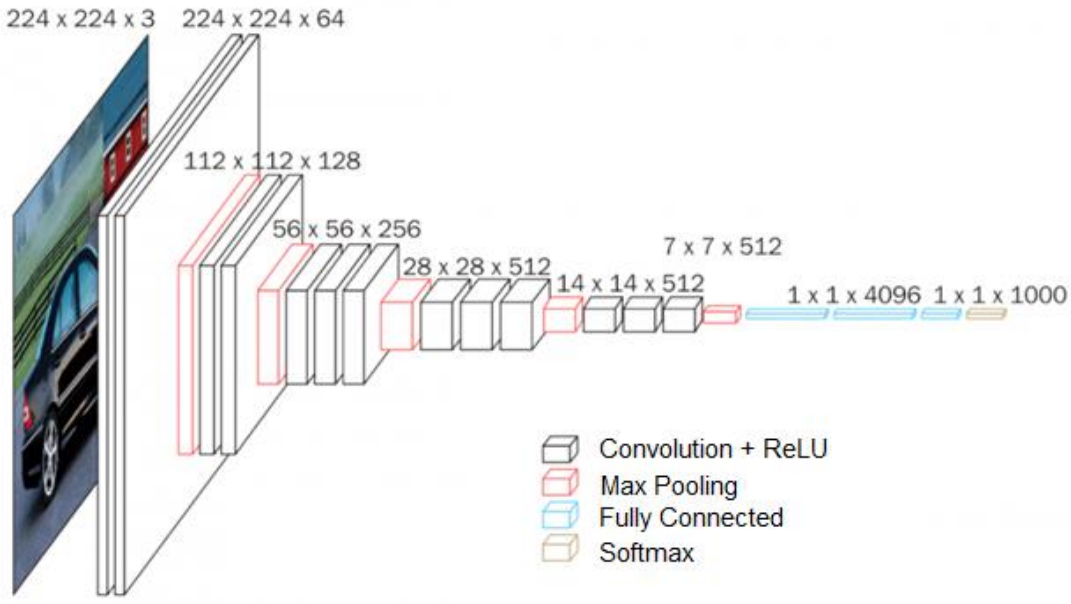


Figure 4.3-2 VGG16 1

The input to any of the network configurations is considered to be a fixed size 224 x 224 image with three channels – R, G, and B. The only pre-processing done is normalizing the RGB values for every pixel. This is achieved by subtracting the mean value from every pixel.

Image is passed through the first stack of 2 convolution layers of the very small receptive size of 3 x 3, followed by ReLU activations. Each of these two layers contains 64 filters. The convolution stride is fixed at 1 pixel, and the padding is 1 pixel. This configuration preserves the spatial resolution, and the size of the output activation map is the same as the input image dimensions. The activation maps are then passed through spatial max pooling over a 2 x 2-pixel window, with a stride of 2 pixels. This halves the size of the activations. Thus the size of the activations at the end of the first stack is 112 x 112 x 64.

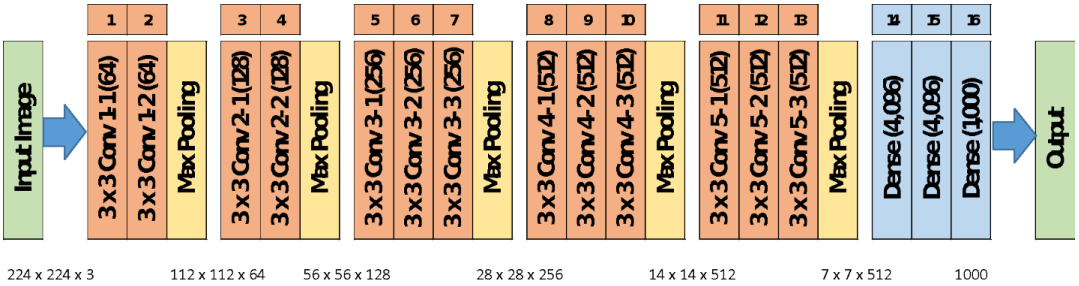


Figure 4.3-3 VGG16 2

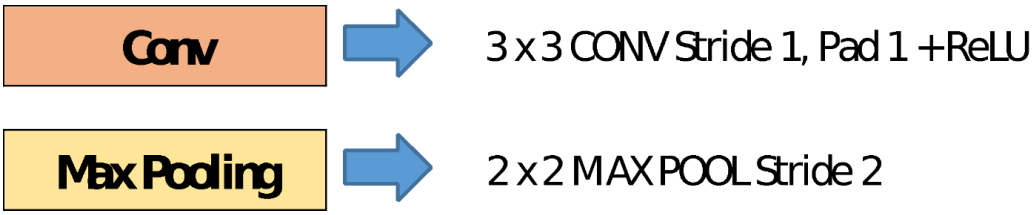


Figure 4.3-4 VGG16 3

1. Convolution using 64 filters
2. Convolution using 64 filters + Max pooling
3. Convolution using 128 filters
4. Convolution using 128 filters + Max pooling
5. Convolution using 256 filters
6. Convolution using 256 filters
7. Convolution using 256 filters + Max pooling
8. Convolution using 512 filters
9. Convolution using 512 filters
10. Convolution using 512 filters+Max pooling
11. Convolution using 512 filters
12. Convolution using 512 filters
13. Convolution using 512 filters+Max pooling
14. Fully connected with 4096 nodes
15. Fully connected with 4096 nodes
16. Output layer with Softmax activation with 1000 nodes.

The activations then flow through a similar second stack, but with 128 filters as against 64 in the first one. Consequently, the size after the second stack becomes 56 x 56 x 128. This is followed by the third stack with three convolutional layers and a max pool layer. The no. of filters applied here are 256, making the output size of the stack 28 x 28 x 256. This is followed by two stacks of three convolutional layers, with each containing 512 filters. The output at the end of both these stacks will be 7 x 7 x 512.

The stacks of convolutional layers are followed by three fully connected layers with a flattening layer in-between. The first two have 4,096 neurons each, and the last fully connected layer serves as the output layer and has 1,000 neurons corresponding to the 1,000 possible classes for the ImageNet dataset. The output

layer is followed by the Softmax activation layer used for categorical classification.

4.3.5 Training VGG16 model

We can create the VGG16 model using a pre-trained VGG16 model in the Keras Applications library.

```
tf.keras.applications.vgg16.VGG16(
    include_top=True,
    weights='imagenet',
    input_tensor=None,
    input_shape=None,
    pooling=None,
    classes=1000,
    classifier_activation='softmax'
)
```

Figure 4.3-5 VGG 16 code

4.3.6 Challenges

Though this is a very simple, elegant, and easy-to-use model, there are some challenges associated with it. The total number of parameters in this model is over 138M, and the size of the model is over 500MB. This puts some serious limitations on the usage of the model, specifically in edge computing, as the inference time required is higher.

Secondly, there is no specific measure available to control the problem of vanishing or exploding gradients.

4.3.7 VGG16 Use Cases

VGG16 keeps the data scientists and researchers worldwide interested despite the advent of many new and better scoring models since the time VGG was originally proposed. Here are a few use cases where you may find VGG16 practically in use.

1. Image Recognition or Classification – VGG16 can be used for disease diagnosis using medical imaging like x-ray or MRI. It can also be used in recognizing street signs from a moving vehicle.
2. Image Detection and Localization. it can perform really well in image detection use cases.
3. Image Embedding Vectors – After popping out the top output layer, the model can be used to train to create image embedding vectors which can be used for a problem like face verification using VGG16 inside a Siamese network.

We used this model for Image Recognition to classify the signs to Arabic letters.

4.3.8 Model analysis

4.4 ANIMATED SIGN LANGUAGE

We use blender application for designing the animations of arabic alphabet sign language we make a sign for every main character of arabic alphabet.

The characters we made is "أ,ب,ت,ث,ج,ح,خ,د,ذ,ر,ز,س,ش,ص,ض,ط,ظ,ع,غ,ف,ق,ك,ل,م,ن,ه,و,ي"

We thought that is the simplest characters that will describe any word to make a sentence.

That is an example for the animations:

Figure 4.4-1 Animation Ain

Figure 4.4-2 Animation Alf

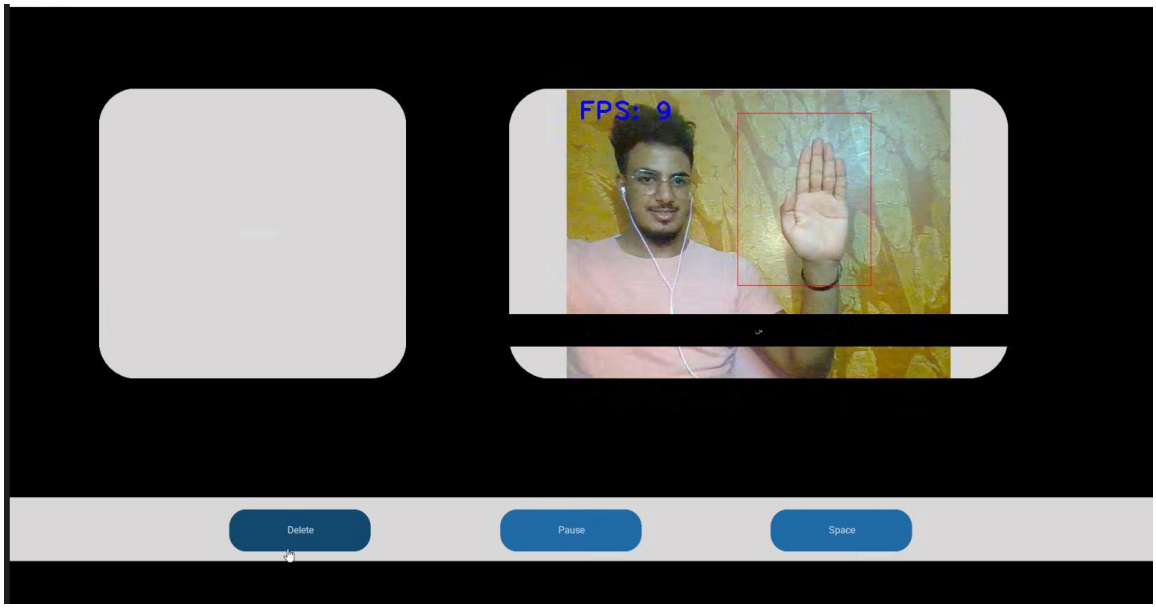


Figure 4.4-3 Running app 1

Screenshots of the project while running

Here in the deaf page the model starts to detect the characters this is the character "س" and push it on the string of the label on the end of the camera.
Then if the user push the button pause it will send the string in the label as a arabic voice to the other user.

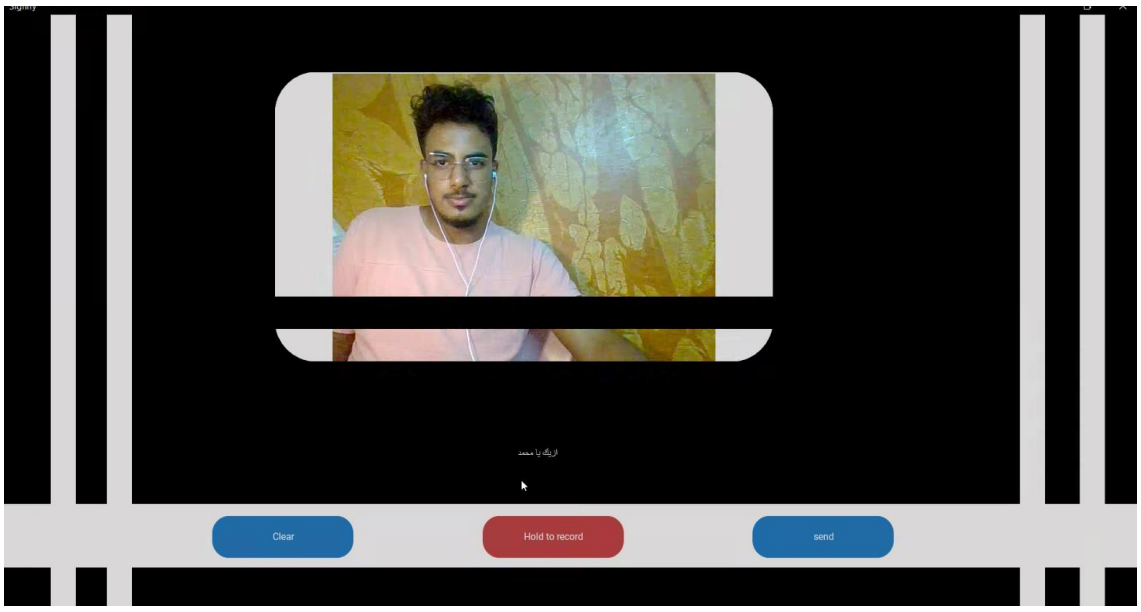


Figure 4.4-4 Running app 2

This screen the other user can record a voice that will be converted into stream of animations into the other screen.

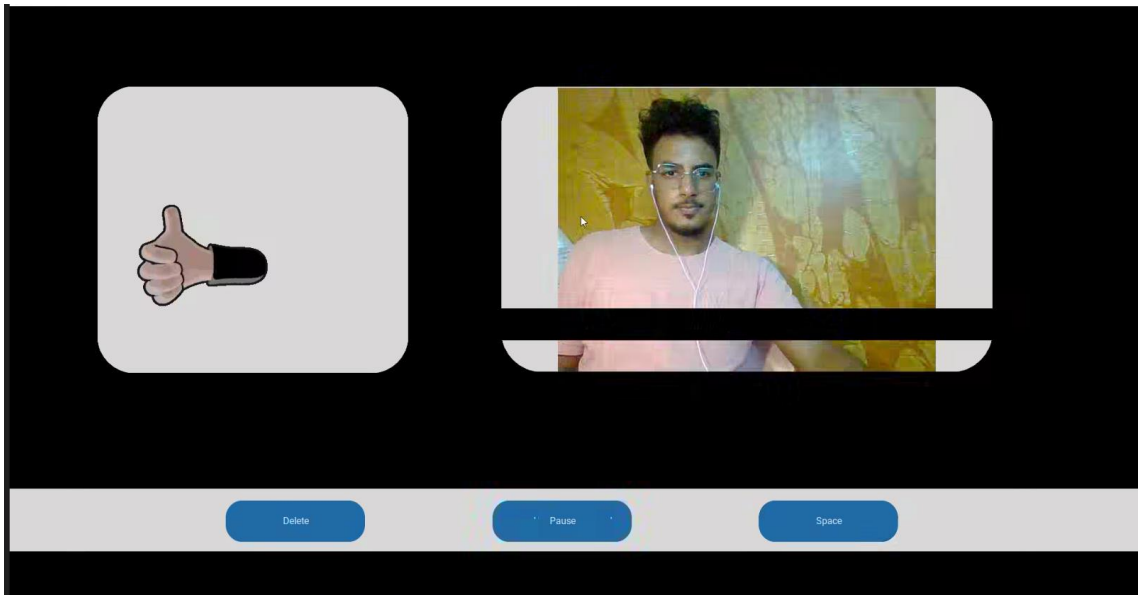


Figure 4.4-5 Running app 3

Here the animation will appear after it sent from the other screen. It will be appear as stream of arabic alphabet sign language.

5 CONCLUSION AND FUTER WORK

5.1 CONCLUSION

The main purpose of Project is providing a feasible way of communication between a Deaf and others by using Desktop Application which takes hand gesture and voice records. The proposed system can be accessed by using webcam or any in-built camera that detects the signs and processes them for recognition and a voice recorder to detect the voice.

From the result of the model, we can conclude that the proposed system can give accurate results under controlled light and intensity. Furthermore, custom gestures can easily be added and more the images taken at different angle and frame will provide more accuracy to the model. Thus, the model can easily be extended on a large scale by increasing the dataset. The model has some limitation such as environmental factors like low light intensity and uncontrolled background which cause decrease in the accuracy of the detection.

From the result of the Application, we can conclude that it's easy to display sign animation as GIF but there is a small delay between animations because it is processed before displaying. This problem can be solved by using a computer with high processing power. We also can conclude that detecting voice can works nicely but the computer should be connected to the internet.

5.2 IMPROVEMENT AND FUTURE WORK

5.2.1 Improvement

We need to improve our application by upgrade the contents of the application but this will be time consuming because of lack of time so this that we would like to improve in the future :

1- In the model, we can increase the dataset. This will increase the accuracy of the model. It's not easy to collect data with different hand shapes, different Brightness, different camera quality and different backgrounds. That needs a many people to get the dataset.

2- In the animation, we can reduce the animated time by make an animation depends on the previous and next sign, so we need about 26 * 26 * 26 GIFs to do this. It's a simulation for making signs in real world.

3- In front end, we can make more powerful application that has many features.

5.2.2 Future work

Many different adaptations, tests, and experiments have been left for the future due to lack of time (i.e. the experiments with real data are usually very time consuming, requiring even days to finish a single run). Future work concerns deeper analysis of mechanisms, new proposals to try different methods, or simply curiosity. There are some ideas that I would have liked to try during the description and the development of the fitness functions in Chapter 5. This thesis has been mainly focused on the use of EDAs for graph matching, and most of the fitness functions used to find the best result

where obtained from the literature of adapted from these, leaving the study of fitness functions outside the scope of the thesis. The following ideas could be tested:

1- Mobile application that will make it easy to communicate with all deaf people in the real world and the won't fell any differences between us and them they will use the application in they life.

2- Add learning sign language feature that will make the user to practice sign language every day with interesting exercises.

3- Make the application fully offline to make it easy to communicate any time anywhere.

4- Make deaf-blind communication that is really unacceptable in this time.

6 REFERENCES

Sample correct formats for various types of references are as follows.

Books:

[1] G. O. Young, "Synthetic structure of industrial plastics," in *Plastics*, 2nd ed., vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.

[2] W.-K. Chen, *Linear Networks and Systems*. Belmont, CA: Wadsworth, 1993, pp. 123–135.

Periodicals:

[3] J. U. Duncombe, "Infraredd navigation—Part I: An assessment of feasibility," *IEEE Trans. Electron Devices*, vol. ED-11, pp. 34–39, Jan. 1959.

[4] E. P. Wigner, "Theory of travelling-wave optical laser," *Phys. Rev.*, vol. 134, pp. A635–A646, Dec. 1965.

[5] E. H. Miller, "A note on reflector arrays," *IEEE Trans. Antennas Propagat.*, tobe published.

Articles from Conference Proceedings (published):

[6] D. B. Payne and J. R. Stern, "Wavelength-switched passively coupled single-mode optical network," in *Proc. IOOC-ECOC*, 1985, pp. 585–590.

Papers Presented at Conferences (unpublished):

[7] D. Ebehard and E. Voges, "Digital single sideband detection for interferometric sensors," presented at the 2nd Int. Conf. Optical Fibre Sensors, Stuttgart, Germany, 1984.

Standards/Patents:

[8] G. Brandli and M. Dick, "Alternating current fed power supply," U.S. Patent 4 084 217, Nov. 4, 1978.

Technical Reports:

[9] E. E. Reber, R. L. Mitchell, and C. J. Carter, "Oxygen absorption in the Earth's atmosphere," Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0200 (4230-46)-3, Nov. 1968.

6.1 REFERENCES TO ELECTRONIC SOURCES

The guidelines for citing electronic information as offered below are a modified illustration of the adaptation by the International Standards Organization (ISO) documentation system and the American Psychological Association (APA) style. Three pieces of information are required to complete each reference: 1) protocol or service; 2) location where the item is to be found; and 3) item to be retrieved. It is not necessary to repeat the protocol (i.e., http) in Web addresses after "Available" since that is stated in the URL.

Books:

[10] J. Jones. (1991, May 10). *Networks*. (2nd ed.) [Online]. Available: http://www.atm.com

Journals:

[11] R. J. Vidmar. (1992, Aug.). On the use of atmospheric plasmas as electromagnetic reflectors. *IEEE Trans. Plasma Sci.* [Online]. *21(3)*, pp. 876–880. Available: http://www.halcyon.com/pub/journals/21ps03-vidmar

Papers Presented at Conferences:

[12] PROCESS Corp., MA. Intranets: Internet technologies deployed behind the firewall for corporate productivity. Presented at INET96 Annu. Meeting. [Online]. Available: http://home.process.com/Intranets/wp2.htp

Reports and Handbooks:

[13] S. L. Talleen. (1996, Apr.). The Intranet Architecture: Managing information in the new paradigm. Amdahl Corp., CA. [Online]. Available: http://www.amdahl.com/doc/products/bsg/intra/infra/html

Computer Programs and Electronic Documents:

[14] A. Harriman. (1993, June). Compendium of genealogical software. *Humanist*. [Online]. Available e-mail: HUMANIST@NYVM Message: get GENEALOGY REPORT