American International University-Bangladesh (AIUB)

Speech to New Zealand Sign Language Conversion in Real-Time

<u>Section: G84</u>

Submitted By-

Oindrila Chowdhury	**17-35431-3**
Md. Hasanul Nafiz Rafi	**17-35669-3**
Mostakimul Kabir	**17-35673-3**
Amit Saha	**18-36075-1**

A Thesis submitted for the degree of Bachelor of Science (BSc)in Computer Science and Engineering (CSE) at

American International University Bangladesh in December, 2021

Faculty of Science and Technology (FST)

Abstract

In this research paper, we have researched about how “Speech to Sign Language” method can be implicated in New Zealand Sign Language. To do so, we have gone through a process where we have taken user voice as input. Then for processing text-based gesture file search, user voice processing, database-based data processing and file extension has been used in the processing portion. Output of recognition of voice play and display match output through graphical user interface has been demonstrated. A meta human which has been created by Unreal Engine UE 4.62.2 (Quixel Bridge App). All work including this research paper has been totally done in purpose to help linguistic community and to reduce the communication gap between normal people and linguistic community. As a result, we found that our created system was able to interpret all inputs at 74% accuracy and give the output. The significance of the result clearly has demonstrated that, our system is capable of converting speeches into text with moderate accuracy. Further modifications and improvements will definitely increase the accuracy and efficiency of our system.

Declaration

This thesis is comprised entirely of our authorship and contains no content that has been republished or authored by someone else, unless otherwise noted in the text. Others' contributions to our thesis overall, such as statistical help, survey design, data analysis, important technological processes, professional editing guidance, economic assistance, and any other unique research effort used or documented in our thesis, have all been properly disclosed. The substance of our thesis is the outcome of our efforts from the start of the **Thesis Project**.

We recognize and agree that the copyright to all of the content in this thesis is owned by the copyright holder(s). To replicate material in this thesis, we received copyright clearance from the copyright holder where applicable, and we sought authorization from co-authors for any collaboratively created works contained in the thesis.

Oindrila Chowdhury

OINDRILA CHOWDHURY
Bachelor of Computer Science and Engineering
Department of Computer Science
American International University-Bangladesh

[illegible]

MD. HASANUL NAFIZ RAFI
Bachelor of Computer Science and Engineering
Department of Computer Science
American International University-Bangladesh

[illegible]

MOSTAKIMUL KABIR
Bachelor of Computer Science and Engineering
Department of Computer Science
American International University-Bangladesh

[illegible]

AMIT SAHA
Bachelor of Computer Science and Engineering
Department of Computer Science
American International University-Bangladesh

Approval

The thesis titled **"Speech to New Zealand Sign Language Conversion in Real-Time"** has been submitted to the following respected members - board of examiners of the department of computer science in partial fulfilment of the requirementsfor the degree of Bachelor of Science in Computer Science on **(06/01/2022)** and has been acceptedas satisfactory.

. . . .

PROF. DR. KHANDAKER TABIN HASAN
Professor, Head
Graduate Program [Graduate Program & DEPT. of MIS]
Department of Computer Science
American International University - Bangladesh

. . . .

DR. KAMRUDDIN MD. NUR
Associate Professor
Department of Computer Science
American International University-Bangladesh

. . . .

DR. KHANDAKER TABIN HASAN
Professor, Head
Graduate Program [Graduate Program & DEPT. of MIS]
Department of Computer Science
American International University - Bangladesh

. . . .

PROFESSOR DR. TAFAZZAL HOSSAIN
Dean
Faculty of Science & Information Technology
American International University-Bangladesh

. . . .

DR. CARMEN Z. LAMAGNA
Vice Chancellor
American International University-Bangladesh

Contributions Table

List of the significant contributions various authors made to the research, work, and writing represented and reported in the thesis. These could include significant contributions to the conception and design of the project, non-routine technical work; analysis and interpretation of research data; drafting significant parts of the work or critically revising it to contribute to the interpretation.

	Oindrila Chowdhury	**Md. Hasanul Nafiz Rafi**	**Mostakimul Kabir**	**Amit Saha**	**Contribution (%)**
	17-35431-3	*17-35669-3*	*17-35673-3*	*18-36075-1*	
Conceptualization	30.27%	27.23%	27.23%	15.27%	100 %
Data curation	30.27%	27.23%	27.23%	15.27%	100 %
Formal analysis	30.27%	27.23%	27.23%	15.27%	100 %
Investigation	30.27%	27.23%	27.23%	15.27%	100 %
Methodology	30.27%	27.23%	27.23%	15.27%	100 %
Implementation	30.27%	27.23%	27.23%	15.27%	100 %
Validation	30.27%	27.23%	27.23%	15.27%	100 %
Theoretical derivations	30.27%	27.23%	27.23%	15.27%	100 %
Preparation of figures	30.27%	27.23%	27.23%	15.27%	100 %
Writing – original draft	30.27%	27.23%	27.23%	15.27%	100 %
Writing – review & editing	30.27%	27.23%	27.23%	15.27%	100 %

Acknowledgments

First and foremost, we are grateful to our Almighty God for granting us the grace of finishing our thesis work on schedule and with a completed book. We need numerous backbones who will always inspire us to generate good innovation.

One of them is our respected research supervisor **Dr. KHANDAKER TABIN HASAN** Sir for his guidance, feedback, collaboration, supervision, directions, and the precious time. He spent guiding us down the right path while the thesis project was going on.

We would also like to give special thanks to **Dr. Md Akbar Hossain**, Senior Lecturer, "*Manukau Institute of Technology*". His enthusiasm and assistance made a significant contribution to the success of the thesis. Besides, we would also like to thank our external sir, **Dr Kamruddin MD. Nur**, Associate Professor, Department of Computer Science, American International University-Bangladesh (AIUB), for giving us his valuable time.

Last but not least, we are blessed by our mates and family members who have always been there by our side. For their care, support, and prayers, we are now going in a path to accomplish our grail.

Keywords

These are the important words frequently used throughout our research paper.

NZSL; UE; OpenCV; LSTM; KNN; HMM; CNN; DTW; PyQt5; gtts; QMovie

Table of Contents

List of Figures

32

List of Tables

List of Abbreviations

Mention all the abbreviations and the different symbols that is used in this document.

Abbreviations	
NZSL	New Zealand Sign Language
UE	Unreal Engine
OpenCV	Open-Source Computer Vision Library
LSTM	Long Short-Term Memory
KNN	K-Nearest Neighbors
HMM	Hidden Markov Model
CNN	Cable News Network
DTW	Dynamic Time Warping
PyQt5	Python binding of the cross-platform GUI toolkit
gtts	Google Text-to-Speech
QMovie	The QMovie class is a convenience class for playing movies with QImageReader.

Chapter 1

INTRODUCTION

1.1 Introduction

People who are deaf or hard of hearing use sign language as a medium of communication. People utilize its motions to convey their thoughts and wants through nonverbal communication. Speech insufficiency is a condition in which a person's ability to communicate via speech and hearing is impaired. Alternative coping strategies, such as sign language, are used by those who are afflicted. Hearing loss and disability speaking present daily challenges for more than 250–300 million people worldwide, some major, some minor.

Despite the fact that sign language has become more extensively utilized in recent years, non-sign language speakers still have trouble interacting with sign language speakers or signers. Non-signers, on the other hand, find it exceedingly difficult to comprehend, which is why experienced sign language interpreters are required for medical and legal consultations, as well as educational and training sessions.

Sign language is primarily a visual language in which meaning is expressed by hand movements, hand shape changes, and track details. People who are deaf or hard of hearing use it as their primary means of communication. The first challenge in studying sign language is collecting information.

This article examines the performance of several methods for converting sign language to speech format. A hand speak application that can translate real-time NZSL (New Zealand Sign Language) signals to voice is built using the best feasible approach after analysis time so that, the words of a normal person can be conveyed to a deaf/dumb person.

Our ultimate objective is to operate the combination of a model (avatar) to identify sign language motions by using Unreal Engine 4. We have developed the avatar using Quixel Bridge apps, which is another part of the Unreal Engine for creating Metahuman.

The purpose of this metahuman role will be to act like human expressions and emotions to communicate with hand gestures, which will interact with deaf-mute people.

In this circumstance, our goal is to utilize video data from the New Zealand Sign Language Dataset to train the model to recognize motions. Different motions are repeated many times in the dataset, allowing for diversity in context and video circumstances. Since sign language is gesticulated fluently and interactively like other spoken languages, a sign language recognizer must be able to recognize continuous sign vocabulary in real-time.

1.2 Proposed Contribution

We concentrated on our thesis milestone in this section:

The realization that signed languages are true languages is one of the great linguistic discoveries of the twentieth century. It is used by dumb and deaf people to por- tray what they want to say. It acts as a basic mode of communication for deaf and dumb people, without which communication between them and normal people might be difficult. This research is focused on developing an automated software system to convert speech to New Zealand sign language in real-time so that the words of a normal person can be conveyed to a deaf or dumb person. These systems have two main problems to solve:

(1) to translate a speech or written text from one language to another.

(2) to create movements for the virtual avatars that can be understood by deaf people, and, even better, with a certain level of fluency.

The problem we have broken down into the following modules:

1.2.1 Speech to Text Module

The Speech to Text module is responsible for collecting the audio (voice input) from the microphone connected to the computer or mobile, analyzing the information, and transforming it into the corresponding text. It contains two main elements: a speech recognizer to analyze the voice input and a message system to manipulate the text data.

1.2.2 Text to Sign Language Translator

This is the most important module. It receives a sentence from the previous modules and produces a translated sentence to the NZSL.

1.2.3 Avatar System Module

Controls and animates the avatar using a database of movements (vocabulary). From the NZSL sentence generates the correct animation movement and shows it to the user in real time.

1.3 Research Objective

As the deaf population is not able to speak or give feedback during a conversation, a specialized language is necessary to extract their thoughts. As a result of this circumstance, many countries have already come up with their own sign languages, like ASL (American Sign Language), BSL (British Sign Language), and many more. For our research purpose, we have built a sign language-based application on NZSL (New Zealand Sign Language) where the system will first take the verbal speech of a normal human as its "input" and then search for possible facial expressions, hand gestures, body posture, and lip movement in its database and give an output of those accumulated expressions through our "Avatar or Meta Human" as an output.

Our application will act as a universal translator for deaf people. Not everyone knows how to have a conversation with deaf or dumb people. Either normal people have to learn sign language to communicate with them or they have to find someone who knows and can translate sign language. To reduce this problem and take this drawback to the next level, we have created a system that can translate and also use any type of speech to sign language for deaf or dumb people for better communication.

Chapter 2

RELATED WORKS AND BACKGROUND

It is now universally accepted in the linguistic community that sign languages such as NZSL (New Zealand Sign Language), ASL (American Sign Language), CSL (Chinese Sign Language), and so on are natural languages with comparable expressive power to that of spoken languages.

This chapter provides a high-level summary of the key ideas and terminology explored in this thesis. The chapter begins with a discussion of the characteristics of sign language conversion. Then we showed earlier work on text to speech on sign language conversion and a similar technique to locating and answering relevant questions on speech to sign language conversion in real-time (NZSL).

We did check out some previous work regarding this thesis. There is a mobile app named "Hand Talk." That app translates English (USA) to sign language and Libra (Brazil) to sign language. There are two animated characters in this app who show the translated message. A user can type his commands and also record his or her voice. Then this app will simply translate that message. There is another project named "Kara Technology.". They made an animated character. This is a video-based project where the animated character shows sign language via video. This is a prerecorded video.

In some recent researches, researchers have come forward to help and assist disabled people who cannot do programming due to absence of hands in their body but those disabled persons have true potential of programming. For this reason, the researchers have built a platform where those disabled persons would speak and the system will write the code in C++. This type of research is getting much more prominent nowadays just because to help the impaired person and give them the pace to go forward in life just like a normal human being [1]. Another research has been done in 2018 in which researchers have pursued to extract text from motion and gesture recognition of a video. In this way, communication gap between "signers" and "non-signers" will be eradicated and anyone can make a conversation with a speech and hearing-impaired person [2]. The researchers have gone through several methods before they have seen the light of success in their research. In another research, researchers have used "Voice to Sign Language" approach to process speech into sign language [3].

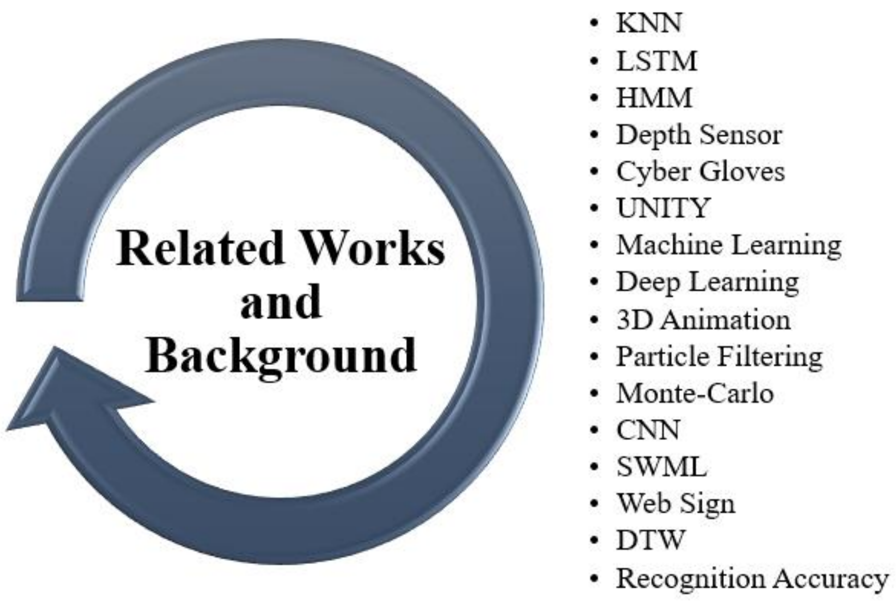


Figure 1: Uses of Different Technologies

The system or platform that has been used for this specific research paper had shown a massive recognition rate of 80.3%, which mean the software platform was able to recognize 80 from 100 speeches and convert them into sign language. This was a massive success back at that time. Since modern technology has been exponentially developed in few years, research and development about this "Sign Language" sector has also been developed a lot and it is developing each new day.

Chapter 3

LITERATURE REVIEW

2.1 Introduction

Sign language is the primary means of communication for the deaf population that carries data via numerous visual channels such as hand gestures, facial expressions, body posture, and lip movement. Sign language recognition technology is still in its early stages. Aside from the difficulty of extracting and modeling information from various channels, one of the primary causes is a lack of resources. Because, in contrast to spoken languages, sign language users are constrained. Furthermore, sign languages have a distinct vocabulary and syntax from the equivalent spoken language. British sign language, for example, is not a signed variant of British English. Additionally, even though the spoken language may be the same, the sign languages may change. American Sign Language, British Sign Language, and New Zealand Sign Language are all distinct sign languages. Swiss-German Sign Language (DSGS) and German Sign Language (DGS) are both distinct sign languages [4].

The rapid development of AI and other technologies has become a hot topic in HCI research. Sign language, as a particular gesture, is also the primary mode of communication for language-disabled people. It contains a great deal of information and has excellent communicative abilities. In order to solve this, we utilize a combination of two models to identify motions in sign language. For training the algorithm to detect gestures, we utilized video data from the unique New Zealand Sign Language Dataset. The dataset contains a variety of motions that were performed many times, resulting in a diversity of contexts and video circumstances. The goal of this study is to create an automated software system that converts speech to New Zealand sign language in real-time so that a normal person's words may be transmitted to a disabled person.

2.2 Research History

Communication is one of the key aspects of people's lives. Through spoken and written words, we have our own way of getting our vital resources such as education, healthcare, employment, etc. But for the deaf and muted community, these things become quite hard. We are less familiar with sign language as a hearing society. Over the years, many researchers have tried to find a way to reduce the distance between normal and deaf/muted people. Using sign language recognition and human- computer interaction, the researcher presented a cost-effective and portable way to detect Chinese sign language. Previously, two approaches have been used. One is vision-based and the other is device-based. In the vision-based system, sign language videos were collected via camera and tried to extract the motions. In the device-based technique, they used depth sensors, cyber gloves, and other components to collect motion. Between these two approaches, the device-based technique wins the war more precisely, but it is expensive. Later, in 1992, an image processing system was used to interpret ASL. Starner and Pentland got 99% accuracy using the Hidden Markov model [5].

In the CSL recognition system, the researcher used a Leap motion controller to acquire hand and finger motion data. This was a two-way communication system, and they put forward the implementation using k-NN [6]. Using the mathematical equation, finger, palm, wrist, and finger angles were calculated. In total, 5000 samples were used in an 80-20 split and they got 96.8% accuracy [7]. To implement the Thai sign language system, the scientists used character games and XNA to generate 3D animation of the sign language. Videos were collected via camera where hands and fingers were vividly shown and stored as an XML file to save storage. From the XML file, the finger motion will be connected to the Thai sign language word and show the output [8]. Using pose tracking and particle filtering, a gesture recognition system was proposed. Particle filtering is a sequential Monte Carlo method to find the target state based on observation. Using HMM and previously defined parameters, they tracked the pose and state of the model. The number of hidden states is the same as the number of gestures, and the number of observations is the same as the number of key poses in the pose library. In recognizing a single word, they got 99.8% accuracy. But the accuracy significantly decreased when the word count got higher. Maybe multiple HMM models could be used to solve this problem [9]. To recognize ASL's facial expressions, the researchers created a dataset based on a hundred different sign forms. They captured the videos at 60 fps and at 720p resolution. Then broke down the videos to 300 frames and resized them. Using CNN, they extracted the temporal features from the frames, which were used further to predict gestures based on the sequence of frames. Then used RNN to analyze the data. To get rid of RNN's drawback, LSTM was used as RNN has a vanishing gradient problem. They complied with the neural network on the GeForce GTX 920 GPU for 36 hours and got 99% accuracy on the training set [10]. An author proposed a manual animation system using SWML, which is materially based on XML. More or less, every sign encoded in SWML corresponds to a sign box tag that contains the collection of symbols that make up the notation as a whole. Each sign has a unique ID that identifies the feature of sign language to which it relates and the variation to which it was exposed. In addition, the parameters of each symbol's relative position on the two - dimensional canvas are shown [11]. There are two major flaws in the SWML encoding that make it nearly impossible to operate a signing avatar automatically. And there is no chronological sequence in which symbols are written for the sign to be appropriately interpreted. The designer of each sign can suggest a sign spelling sequence for internal symbol ordering. However, it should be noted that this sequence is an optional prefix and will not be found in all signboxes. As a result, this alternative option for organizing the notation elements cannot be used in this approach. The SWML format, on the other hand, encodes only the original glyphs, which means that certain information may be removed. Using sign language and gesture classification via WebSign, to animate an avatar for better visualization for a deaf or muted person. Their system allows users to generate their own sign descriptions and contribute to the growth of their community's lexicon. The animation engine utilized relies on the keyframe approach to pick the essential moments assumed for learning movement. However, this approach has challenges, particularly in terms of the time required to create postures and the lack of accuracy in the fabrication of gestures [12]. A multi-stage CNN was designed via an input layer, four convolutional layers, five ReLu, two stochastic pooling layers, one dense, and one SoftMax output layer. It was to acquire data from SL-video at 640x480 in aspect. Using supervised learning, the network was taught to learn the characteristics of each sign. Researchers described 200 ISL signals performed by five native Indian sign language users from five distinct viewing points. The entire dataset size is 5000 signs, with each sign recording normalized to 2 seconds, or 60 frames per second.

To properly detect the sign, the maximum activation neuron is extracted to know the feature representation learned by the CNN system. Finally, in higher layers, the feature maps were displayed by averaging the picture patches with the stochastic response [13]. Using the Tracking learning detection algorithm, a recognition system was proposed to extract the keyframes of sign language. By scaling and distorting the sample's time series, the DTW algorithm converts the global optimum issue into a local optimal problem. It seeks an optimum path between the two samples, using the total of the pathways' distances as the distance between the two-time series. The DTW algorithm was used to detect the sign language trajectory, and it is capable of distinguishing sign languages with substantial differences in motion trajectory.

The motion trajectory of many sign languages, however, is not very diverse, but there are some variances in the gestures in the process of showing sign language, and the gestures of the same sign language may be different for various individuals or different situations. As a result, sign language recognition is performed using the sign language trajectory as the primary recognition basis and keyframe matching as an auxiliary basis. Five rounds of tests were done based on 70 sign languages, and the average accuracy was 90.84% [12].

2.3 Research Question Discussion

- ***Is motion capture going to be space hungry?***

  Capturing the motion of a video is going to be less space-hungry than a manual animation system. Using a machine learning algorithm, a video's motion or subject's movement can be captured. For this, OpenCV or LSTM can be used. These frameworks are quite lighter than manual animation. OpenCV is less than 200 MB.

- ***After adding face expression will the model loose accuracy?***

  This is a probability. From the literature review, the researchers lost some accuracy after adding facial expression while recognizing ASL. They primarily used RNN. To get rid of this problem, they broke the video into 600 frames and used LSTM. Then the accuracy increased.

- ***How to be more precise at capturing finger and palm motion?***

  Some researcher had set the palm and finger position previously using SWML. But this approach cannot be used in real-time transition. And finger and palm has very tiny section of movement and the motion is very delicate. Keeping this in mind the Lidar sensor can be used. This sensor can capture any kind of subtle movement with fine tuning.

Chapter 4

METHODOLOGY

3.1 Introduction

At the very initial stage of this thesis, we analyzed more than 50 research papers, books, and journal articles. From that, we understood the basics of the song language and its working principles. We have got to know about some frameworks. We studied the process of animating a character. Speech to text conversion, database design, and real-time conversion of regular language to sign language are our basic modules. In the Speech to Text module, the user's speech will be converted to text. Then that text will be analyzed semantically with the help of sign language grammar. Then the text will be searched in the database. Once the text is matched with the database's words, it will show the animated character's video at the user's end. To animate a character, we used the Unreal Engine. After finishing all these reviews, we went out with the experimental methodology to conduct this research.

3.2 About Methods

Experimental methodology is broadly used in the Computer Science department to evaluate problems and their solutions. It is divided into two phases. In the exploratory phase, the researcher measures that will define what questions should be asked about the problem. Then the evaluation phase comes in to answer those questions. And this answer-finding process is actually a well-designed experimental methodology.

The first thing to get started with experimental methodology is record keeping. Each and every experiment's record should be organized. so that a researcher can trace back to his previous results whenever he wants. Then comes the experimental setup design. After the exploratory phase, the researcher stops all the documentation. Then he decides what kind of hardware and software should be used to conduct his research. As per the research questions and expected answers, this phase gets organized.

The researcher has to keep in mind what variables are in the entire process and which ones should be controlled. Reporting the experimental results When describing the outcomes of an experimental assessment, it is critical to describe what was learnt from the tests clearly and succinctly in simple English. Numbers in a paper or incorporated into a report should be there to answer a question or make a point. Graphical or tabular representations should be carefully chosen to emphasize the researcher's ideas. They must not deceive or twist the data. Data analysis based only on grouping is quite problematic since averages may be extremely misleading. Thus, even if the raw data is not included in a publication, the author should carefully analyze the raw data to acquire a better understanding of the experiment's outcomes.

3.3 Reason Behind Selecting

In our entire implementation, the subject's motion capture is a vital thing. For this, we had to try out various frameworks like OpenCV and LSTM. Different frameworks were providing us with different results. Face expressions and finger movements are very hard to capture with automation. Also, the semantic analysis of the speech was also a challenge. We had to break down the text into very small lexicons. Then they analyzed them as per Sign language grammar. We also used some algorithms to make this perform well. We also had to see how the results were presented at the user end. If the user finds it easy or not, So, the entire implementation process was kind of a trial-and-error process. That's why we went out for experimental methodology.

3.4 System Design Methodology

We had to keep in mind when designing the system that the majority of Deaf or muted people are underserved in terms of education and technology. If our entire system gets too complicated, then it may not be accepted positively and widely by the deaf or muted community.

There are three basic modules in our thesis. Speech to text module, Text to sign language translator, Avatar system module. First of all, we need to receive the audio from the speaker. Then we have to convert the audio into text format. After getting the text, we have to use a parser to extract semantic and syntax information using some algorithm. Database wise, we have to use the data collected from the parser. For this step, we may need to check parts of speech in the text data. By comparing our text data with the database, we have to get or collect the data as an animation script. It is like how the animated hand will move, what the facial expressions will be, etc. Using the animation script, we have to create the actual animation clip. To do this, we can use Blender, Maya, etc. We can get these animation clips in the "fdx" format for the Unity platform or in any other format, platform-wise. Next, we need to make human reggae for Unity to have some real-time movement. After that, we need to attach that reggae with animation. After completing all these steps, our code will be basic. It will be like this: our hearing module will listen to the voice of the user and convert it.

Firstly, to get the text data, we can use the "SpeechRecognition" API in Python. After getting the text data, we have to tokenize that data. Now we have to analyze those tokens. In this step, we may have to check syntax, parts of speech, sentence segmentation, parsing, etc. We can call this module "Sentence Analyzer." For this module, we need a grammatical system. Suppose we got a big lexicon. It may be a categorical grammar. Here we will have a small lexicon named "Combinatory Categorical Grammar" (CCG). Then we have to use the CCG and analyze it semantically to extract discourse information. Depending on the situation, we may need to add or remove some data in order to make things perfect. Then we may need to do "long-distance extraction" or linguistic phenomena or topicalization. This also depends on the situation. To do this, we have to use an algorithm named "Cocke–Younger–Kasami" (CYK). With this, we can parse and analyze the sentences dynamically. Then we will put the final analyzed sentence in a transformation function that has the database. This algorithm will follow some rules like object-in-phase positioning, word-directionality, word reordering, realizing phrases and other parts of speech, movement property, etc. Then we will collect the data as an object and compare it with the database. After that, we will have the most probable piece of data. The data will be in terms of animation. We will send this data to a new function that may be named "Animation_producer." This function will have some clips. This is motion data in "fdx" format. Depending on the animation script and given clips, the "Animation_producer" will play the animation sequentially. Like we have the words "eat", "sleep", and "code". The "Animation_producer" function plays these words' signs one by one.

Chapter 5

UE SYSTEM AVATAR

5.1 Introduction

Digital characters are used to make a visual impression. In general, character design is commonly done in 2D and 3D. On the basis of the project, it can vary which model is used. And it goes without saying that 3D characters can be crucial in achieving high-quality framework visualization. In general, the character design process comprises three phases: pre-production, production, and post-production [14]. Our team employed a 3D character design model for a better user experience, and getting an eye-catching result is our main ambition for this project. This is the only reason why we used this Unreal Engine app to create a 3D character with motions. In general, the Unreal Engine is more familiar with the game-developing sectors.

5.2 Digital 3D Character Design with Unreal Engine

Digital characters are used to make a visual impression on the user. As a result, the users can grasp the concept quickly. The Unreal Engine can create characters with custom motions. So, the Unreal Engine is a solution for creating digital characters. Depending on the needs of the user, the Unreal Engine can construct avatars with specific animations. And getting a 3D real-life avatar is the main purpose of using the Unreal Engine. 3D models are more effective at conveying information to users. The creation of personalized 3D avatars is promising due to the recent requirement of realistic AR/VR applications and the need for customized services, for example, VR games, virtual fitting, virtual fitness, etc. There are only a few recent research papers on personalized avatar creation [14] [15].

5.2.1 2D vs 3D Design

Two-dimensional avatars have been used for many years. It’s easier to create a 2D avatar than a 3D character. After many updates, people got acquainted with 2D to 3D. People are now focusing on 3D so that it can keep pace with the times. The 3D avatar is more eye-catching than the 2D animations. It helps to grasp the concept easily and get an improved quality by considering the external aspects.

5.3 Unreal Engine Capabilities

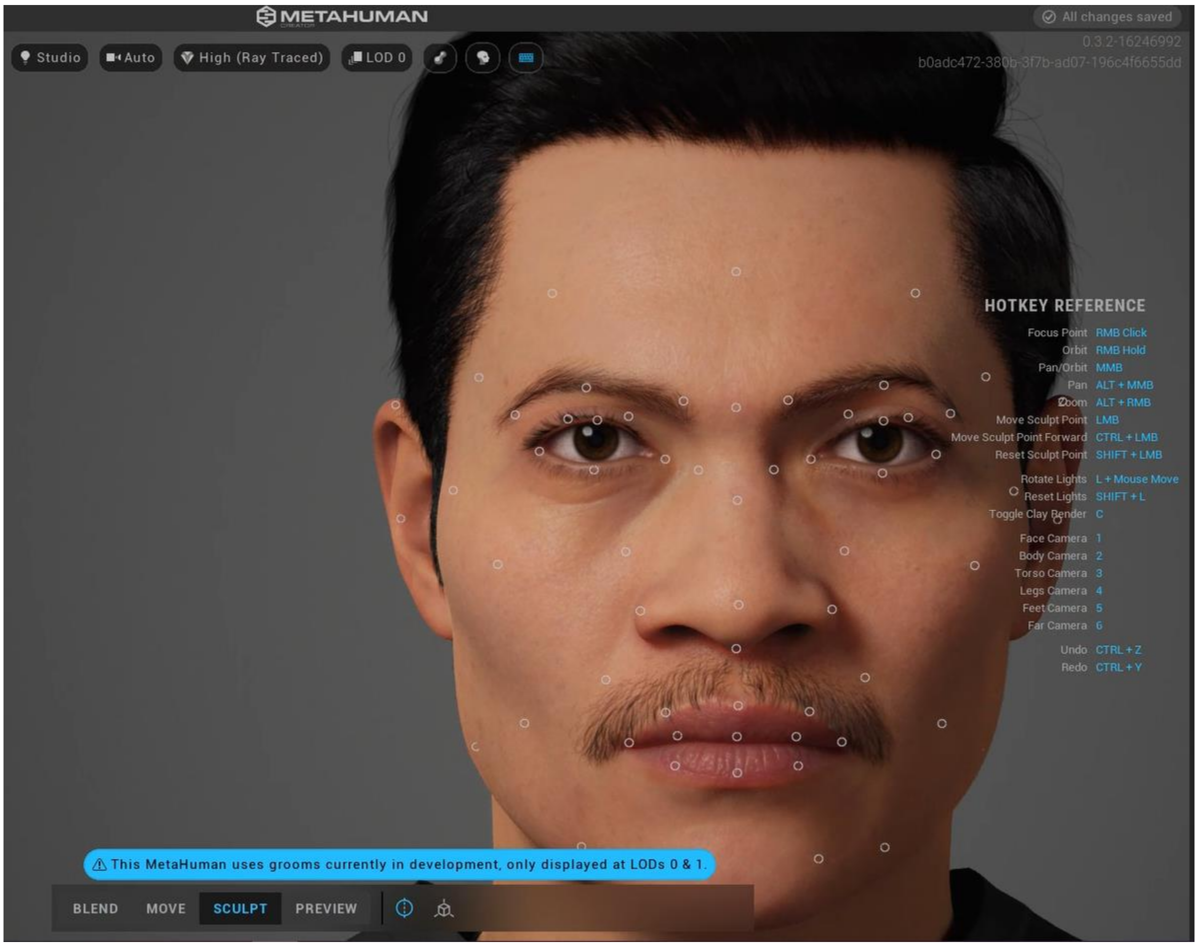
The digital world necessitates high-quality frameworks through which individuals might obtain eye-catching content. Apps for creating digital humans come in a variety of forms. People use a variety of apps depending on their perspective and skill level. However, if we're talking about keeping up with the times, an Unreal engine can generate a professional-level character in a very short amount of time. Once the character modeling, texturing, and rigging have been done, it has to do a retopology first, then import it into the Unreal Engine. The photo-real rendering in Unreal can be rendered while animating or previewing in high quality [16].

5.3.1 MetaHuman

Digital humans are the future, and it's a part of the future. MetaHuman Creator is a cloud-streamed app that allows users to create a 3D custom real-time digital human avatar in just a few hours while maintaining high quality, fidelity, and realism. MetaHuman terms are familiar with the game's developing sectors more. It's a part of the Unreal Engine. Epic game developments increase their level by developing this avatar/character creator app. This Gesture platform is much better than the other platforms where users can create their custom avatars. Custom avatars are created with the help of the Quixel Bridge app. This app works as a gesture editor. After creating the digital avatar character, a custom motion can be given to the gesture.

5.3.2 UE System Process

This section will go over the entire process of developing a digital human character using Quixel Bridge software. Quixel Bridge web and desktop software, Unreal Engine 4.62.2, and Epic Game Development desktop app are the main production tools. We must first register an Epic game development account, and then we need to open that web server and desktop app at the same time. It will automatically build a link with the desktop Quixel bridge software. After that, we have to start editing the digital human characters from the given samples. Then we have to start editing the digital human characters from the given samples. After creating a custom digital human character, we need to export it to the Unreal Engine to give it custom motion. For a better outcome, we selected the most recent version, 4.62.2. In this editor version, we can give a custom motion to the character which we have imported from Quixel Bridge. And then we have to save the file for use in the project.


METAHUMAN
All changes saved
Studio
Auto
High (Ray Traced)
LOD 0
0.3.2-16246992
b0adc472-380b-3f7b-ad07-196c4f6655dd
This MetaHuman uses grooms currently in development, only displayed at LODs 0 & 1.
BLEND
MOVE
SCULPT
PREVIEW

HOTKEY REFERENCE

Focus Point	RMB Click
Orbit	RMB Hold
Pan/Orbit	MMB
Pan	ALT + MMB
Zoom	ALT + RMB
Move Sculpt Point	LMB
Move Sculpt Point Forward	CTRL + LMB
Reset Sculpt Point	SHIFT + LMB
Rotate Lights	L + Mouse Move
Reset Lights	SHIFT + L
Toggle Clay Render	C
Face Camera	1
Body Camera	2
Torso Camera	3
Legs Camera	4
Feet Camera	5
Far Camera	6
Undo	CTRL + Z
Redo	CTRL + Y

Figure 2: UE Face Gesture

5.4 UE Limitation

In addition to the advantages, there are some limitations. To operate with Unreal Engine, need a device with a high configuration. We'll need three applications to make a custom avatar with motion: Quixel Bridge, Epic Game Development, and Unreal Engine. These are the primary tools for creating a custom-motion avatar. Apps for the Unreal Engine come in a variety of flavors. We used the most recent version, which is 4.62.2. Rendering is a big issue in this developing system. As we know, here all the apps have to be linked with each other. And have to export character to the Unreal Engine 4.62.2 for giving custom motion on it. For the purpose of developing, one's abilities and learning more about animation here is certain learning restrictions. After exporting the character to the unreal engine.

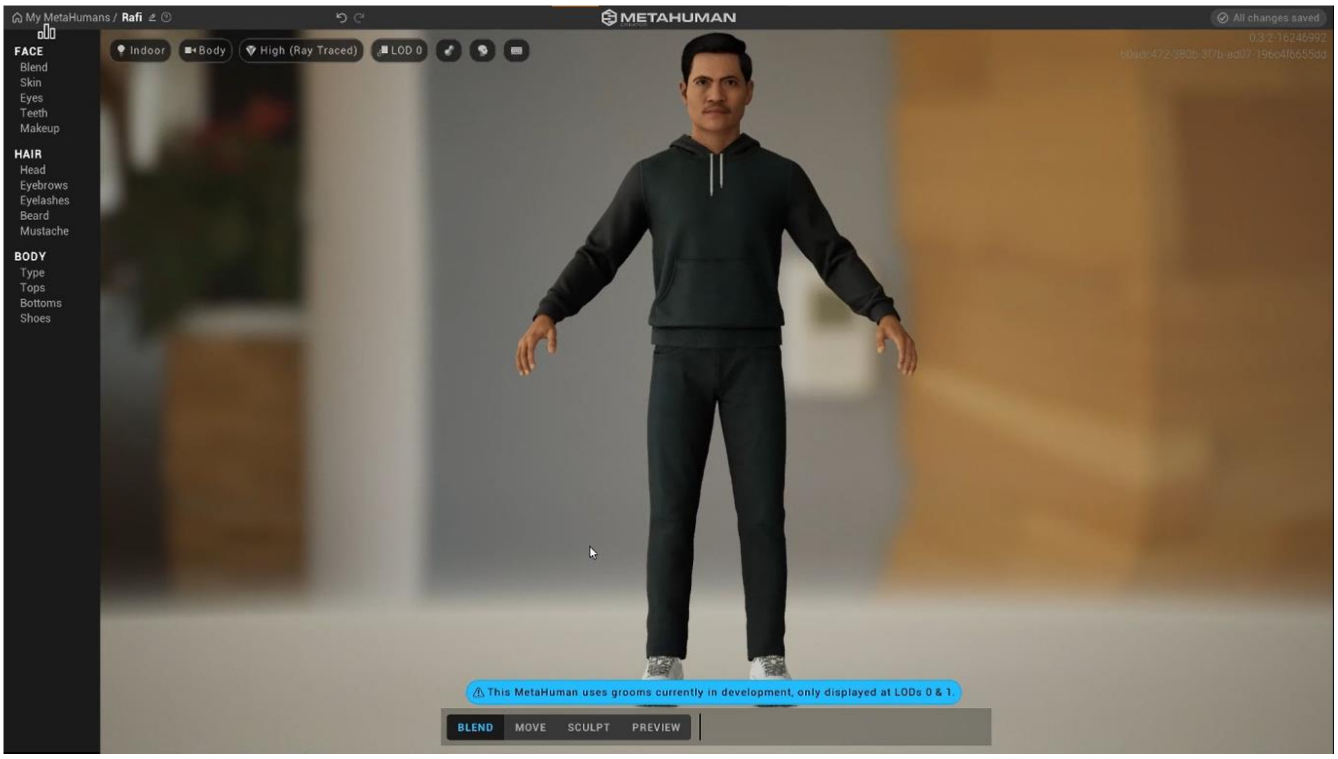


Figure 3: Meta Human System Gesture

5.4.1 Editor Flexibility

For creating a digital human character Quixel Bridge apps is the key of professionalism where custom digital human character can be developing like MetaHuman. Everything can be accomplished with this app. This app is a complete solution for creating a personalized digital human character in a compact package. Only a few prior skills are required. In this app, there are already some samples and in order to create a digital human character, we must have to choose a certain design. We can change the skin color, body shape, hair, eyes, ears, beard, and outfit of the character here. And we can get our desire custom digital human character in just few hours.

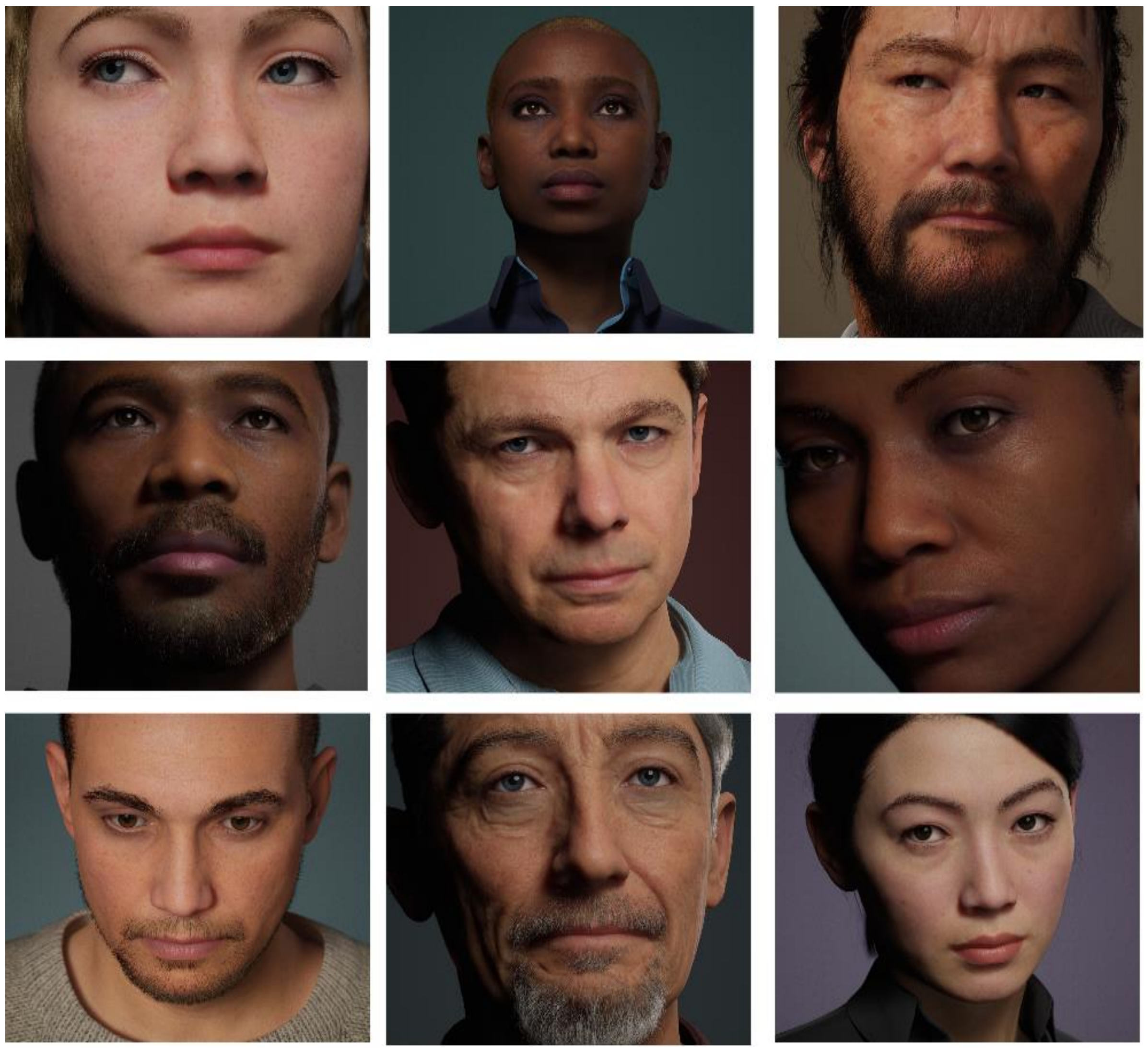


Figure 4: Meta Human Different Gesture

Chapter 6

SURVEY ANALYSIS

6.1 Introduction

By communicating with others, we people gain access to the vital resources in our daily life. We talk to people, ask something, give something. This is how society works. This is a general case for normal people. But for people who cannot talk or speak for biological reasons or accidents, life is not that much easy for them. In terms of getting education, healthcare, job and in so other activities in life they face difficulties. According to WHO (World Health Organization), there are more than 360 million people who are deaf or mute. As per the Rated digital library, more than 75% of deaf or muted people are illiterate. In the case of receiving education, on average an 18 years old deaf student gets knowledge of level 4's students in a deaf or muted specialized school. So, it is quite a clear picture that only for a physical disability a huge number of people are being deprived of basic human needs. Sign Language is a prominent means of communication among the deaf and the hearing-disabled people. It is a native language for them.

We did a survey based on making a sign language conversion application that can help the deaf community and can fill the place of a human sign language translator.

Our questions were -

1. Do you have any Deaf or Muted person in your family?
2. Have you ever communicated with a Deaf or Muted person?
3. Is it easy to communicate with a Deaf or Muted person?
4. Ever heard of Sign language?
5. Do you know how to use Sign language?
6. If I give you an application which will convert your language to Sign language, will you show any interest to use it?

7. What do you think will happen if "Speech to Sign Language" application provided to the Deaf and Muted community in terms of Education?
8. Do you have any past experience with this kind of application?
9. What kind of application will be more comfortable to use you think?

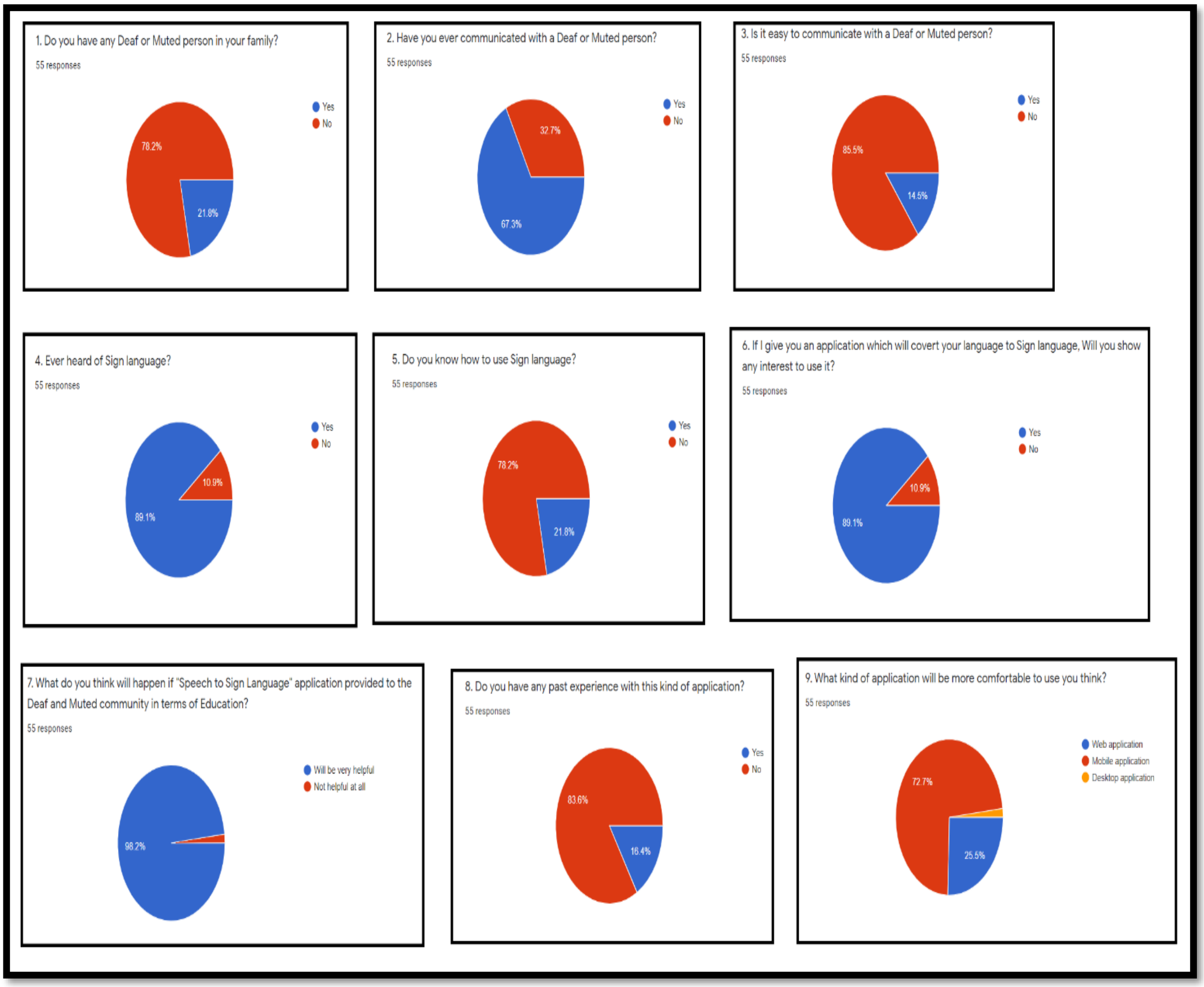


Figure 5: Survey Analysis Pie Chart

6.2 Summary

In the survey, our first question was to know if there were any deaf or muted people in their family. Based on the result, we came to know that 78% of the participants did not have any deaf or muted people in their family. This result appears to be fairly accurate, given that the Deaf community is a minority in our society. From the second question, we got to know that most of them have some sort of experience communicating with a hearing-impaired person. Although they faced some difficulty communicating with a deaf or muted person. 86% of the participants say that. For our 4th question, we wanted to know if they had ever heard of sign language. 89% of people responded positively. But from the result of the 5th question, it came to light that most of the people do not know how to use sign language. Our 6th question was to check the participant's level of interest if they get any kind of application that converts their language to sign language. And from the survey, we came to know that almost 89% of people were interested in that kind of application. On top of that, almost everyone believes that this kind of application will have a great impact in terms of education. From the survey, we also got to know that most people do not have any previous experience using this kind of app. In terms of the applications, most of them wanted a mobile application, some of them wanted a web application, but only a minor part wanted a desktop application.

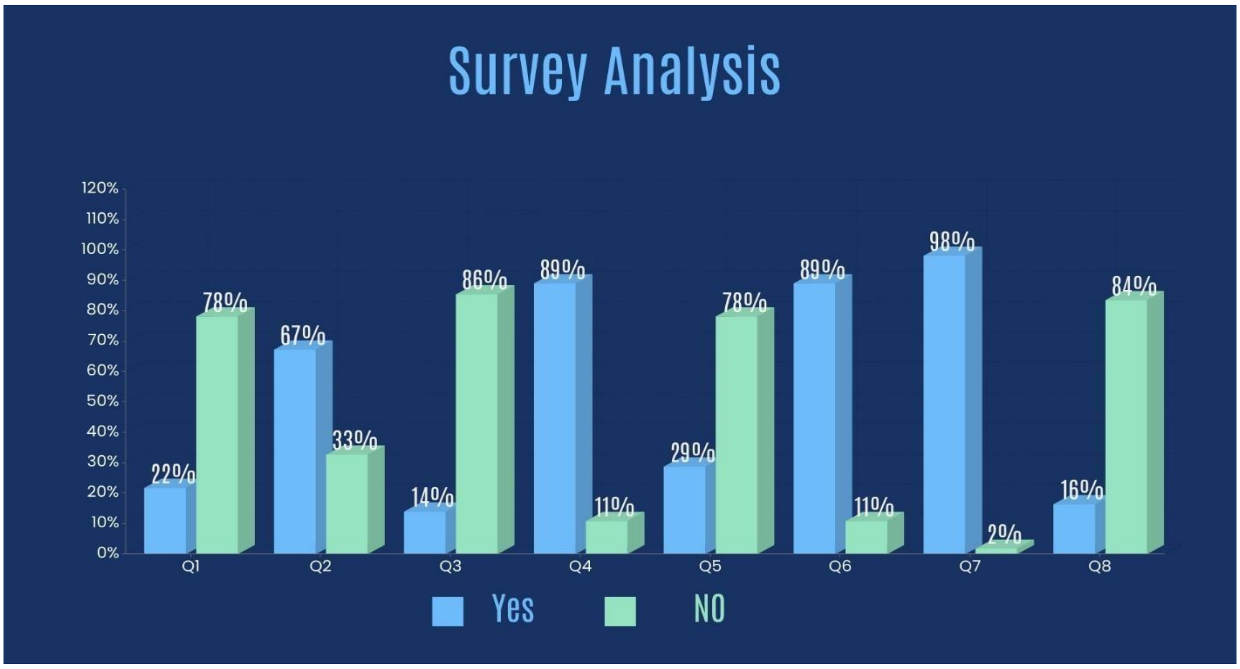


Figure 6: Survey Analysis Bar Chart

Chapter 7

Result and Discussion

7.1 Introduction

We used human voice as input to the NZSL system and displayed gesture data as output. We categorized the input voice into three parts which are:

(i) Word
(ii) Sentence and
(iii) Paragraph

We have used a three-set of data where each data set contains ten inputs. The human voice was inputted in the NZSL system and we measured the correctly recognized word and the response time. Here response time is measured based on the inputted time to gesture file display time. Table 1 shows the average response time of input voice based on the dataset. We also calculated ***Root Mean Square Error (RMSE)*** and ***Mean Square Error(MSE)*** for the data set. We calculated every input voice response time and correctly identified response data which are shown in **Table 1**.

Table 1: Data Set with RMSE and MSE

Data Type	Total Data	Correctly Identified	Wrongly Identified	Data Response AVG Time in Sec	MSE	RMSE	Accuracy (%AVG)	Deviation (%AVG)
Word	12	7	4	2	0.417	0.645	75%	25%
Sentence	10	9	1	4	0.100	0.316		
Paragraph	10	8	2	10	0.200	0.447		
Total	32	24	7	16	0.250	0.500		

For Calculating Mean Square Error and Root Mean Square Error we used Formula 1 And Formula 2.

$$Mean\ Square\ Error = \frac{\sum_{i=1}^{N}(Pi-Oi)^2}{N} \qquad (1)$$

$$Root\ Mean\ Square\ Error = \sqrt{\frac{\sum_{i=1}^{N}(Pi-Oi)^2}{N}} \qquad (2)$$

Here

P_i is Predicted value

O_i is Observed value

N is the total number of Data.

If any word, Sentence or Paragraph is fully identified then we set the data as 1(True) otherwise 0 (False). For Predicted (Pi) value we set 1 to all data and Observed Value (Oi) is set based on result if any inputted voice is correctly identified and displayed then gesture data then we set it to 1 otherwise 0. Then we calculated the MSE and RMSE value using formula 1 and 2.

From Table 1 we can see that Word correctly identification rate is less than Sentence and Paragraph because of our inputted voice accent. The total *MSE* and *RMSE* value for 32 data is *0.250* and *0.500* respectively.

We also calculated the average accuracy using Formula 3 which are shown in **Table 2**.

$$\text{Accuracy} = \left(\frac{Total\ Correctly\ Identified\ Data}{Total\ Input\ Data}\right) \times 100 \qquad (3)$$

Table 2: Identification Accuracy Table

Data Type	**Positive Classified**	**Negative Classified**
Word	58%	33%
Sentence	90%	10%
Paragraph	80%	20%
Total	75%	25%

Though Word correctly identified less than 60% but Sentence and Paragraph Correctly Identification rate is above 78%.

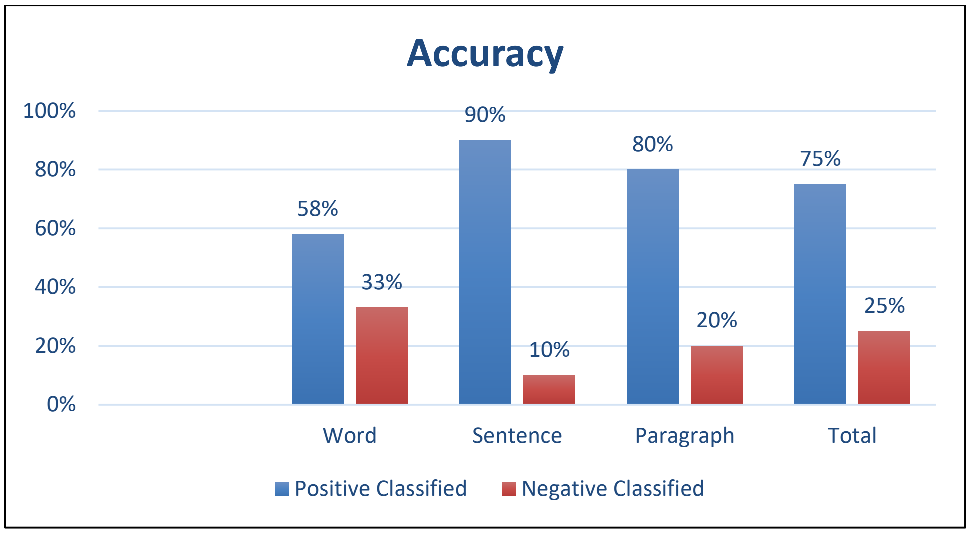


Figure 7: Accuracy Level Bar Chart

Though our system shows a nice performance on word-based gesture response On the other hand it shows less performance on paragraph-based gesture response. The performance graph shows in **Figure 7**. From **Figure 7** we can say that our system easily detects words which are 58% and sentence detection rate is 90% and paragraph detection rate is 80%. Sentence and Paragraph detection rate above 78% and the word detection rate below 60% in our NZSL system.

We have used google voice recognition which also has some limitations. Sometimes our delivered voice accent is not accurate. So the delivered voice is not easily recognized by voice recognizers. That time our system gets some error message or wrong data. The wrong data rate is increased when we input a paragraph as a voice.

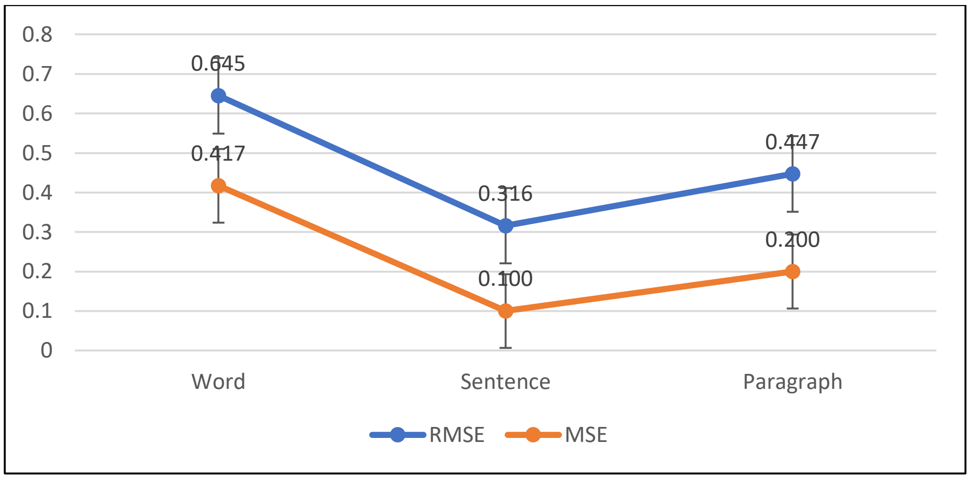


Figure 8: RMSE and MSE Line Chart

MSE and RMSE deviations for Word Sentence and Paragraph are shown in Figure 8. From Figure 8 we can say that Sentence RMSE and MSE rate is better than Word and Paragraph which are 0.316 and 0.100 respectively.

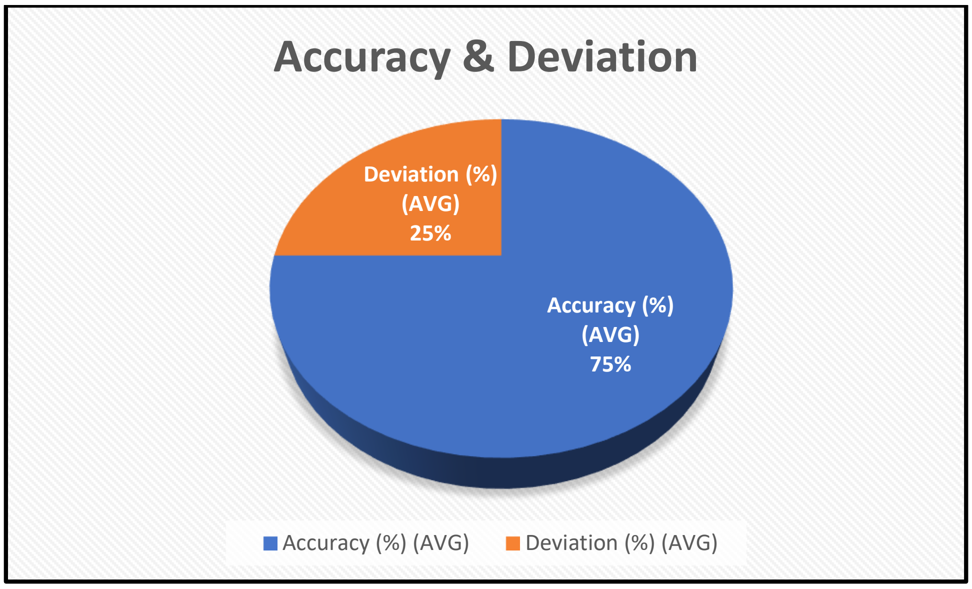


Figure 9: Accuracy Level Pie Chart

From Table 1 we calculated correctly identified and wrongly identified data and also calculated accuracy rate based on input data set. Accuracy rate calculated based on equation (3).

We have got overall accuracy for the three data sets is 75% and 25% as a deviation [Figure 2]. Though the rate is above 75% we are working on it to increase the rate by approximately 85%.

Chapter 8

CONCLUSION

8.1 Conclusion

Throughout this paper, we have worked on a system that can effectively assist people who are impaired to listen or speak. The system can effectively convert a normal human's voice into text, so that people who are unable to listen or speak can figure out what he or she has been told, asked or instructed to do and they can also reply their reaction or answer. Throughout the whole world, there are millions of people who are not able to listen or speak by born or by accidents. For this reason, most of them, not all, being left behind at any point of their lives. These impaired people have to face unnecessary situations in their personal, academic or professional life. To reduce the communication gap between these people with normal people, we have tried our best to build a system by using both a normal person and a hearing-impaired person can seamlessly communicate and help each other. Our system has shown significant accuracy while testing and further deep improvements and modifications will surely ensure our actual vision of this research.

8.1.1 Future Vision

In our process, we take voice as input and display gesture files as output. Now we are also working on some special features which are-

1. Direct text input
2. Text or Data Search.

Currently, we are disabled three widgets which are two text boxes and one search button. We worked with a set of limited data. In the future, we will add more data for the following purpose:

1. Sentence summary identify based on deep learning
2. Paragraph summary identify based on deep learning
3. Summary based gesture data display
4. Easily imputed text-based gesture data show based on machine learning
5. Inputted text can be easily searched through the search button and text box.

Bibliography

References

[1] M. Ayub and M. A. Saleem, "A speech recognition based approach for development in C++," *International Journal of Computer Science Issues (IJCSI),* vol. 10, no. 5, p. 52, September 2013.

[2] M. R. M. M.-D. Sandrine Tornay, "TOWARDS MULTILINGUAL SIGN LANGUAGE RECOGNITION," *IEEE,* vol. 978, 2020.

[3] K. Bantupali and Y. Xie, "American Sign Language Recognition using Deep Learning and Computer Vision," in *2018 IEEE International Conference on Big Data (Big Data),* Seattle, WA, USA, 2018.

[4] O. M. Foong, T. J. Low and W. W. La, "V2S: Voice to Sign Language Translation System for," in *International Visual Informatics Conference,* 2009.

[5] M. M. W. X. C. T. a. G. P.-M. T. Alldieck, "Video based reconstruction of 3d people models," *CVPR,* 2018.

[6] A. W. F. R. W. G. M. H. L. M. B. a. E. S. A. Shapiro, " Rapid avatar capture using commodity sensors. In Computer Animation and Virtual Worlds," 2014.

[7] J. M. a. T. O. K. Nop, "Development of Character Design Frameworks using Game Engine: Unreal Engine," 2019.

[8] N. T. Paramat Ittisam, "3D Animation Editor and Display Sign Language System Case Study: Thai Sign Language," *IEEE,* vol. 978, 2010.

[9] S. G. H. S. W. Q. Yaofeng Xue, "A Chinese Sign Language Recognition System Using Leap Motion," *IRVRV,* 2017.

[10] M. Z. I. J.-W. P. C.-W. L. Chi-Min Oh, "A Gesture Recognition Interface with Upper Body Model-based Pose Tracking," *IEEE,* Vols. V7-531, 2010.

[11] Y. X. Kshitij Bantupalli, "American Sign Language Recognition using Deep Learning and Computer Vision," *IEEE,* 2018.

[12] M. J. Yosra Bouzid, "An Animated Avatar to Interpret SignWriting Transcription".

[13] M. J. Nour Ben Yahia, "Animating signing avatar using descriptive sign language".

[14] K. P. A. G.Anantha Rao, "Deep Convolutional Neural Networks for Sign Language Recognition," *SPACES,* 2018.

[15] M. K. Deepali Naglot, "Real Time Sign Language Recognition using the Leap Motion Controller.".

[16] Z. L. Q. T. C. L. R. Z. Yufei Yan, "Research on Dynamic Sign Language Algorithm Based on Sign Language Trajectory and Key Frame Extraction.," *IEEE,* Vols. 978-1, 2019.

Appendix A

SOURCE CODE REVIEW

A. Introduction

In our research criteria, we can divide the source code into three module parts. We used python programming language to fulfill our goal. For different modules, we used different python classes which are listed below. The overall process is done through Graphical User Interface Parts. The overall process starts from the **_main_.py** class.

a) **Input**

    i. User voice Input [**Listener.py**]

b) **Processing**

    i. User voice processing (Speech to Text)

    ii. Text-based gesture file search

    iii. Database based Data Processing

    iv. File extraction and ready for display

c) **Output**

    i. Recognized voice play

    ii. Display the matches output through Graphical User Interface

d) **Graphical User Interface**

e) **Error Handling Part**

A.1 Explanation of Code

A. Input [Listener.py]

In this section, users input their voice through the device microphone.

1. User voice Input

   Inputs are taken through the python **speech_recognition** package. In the **Listener** class, we used the **listener** function which will take the user voice as input. Inputted voice is recognized through **Google voice recognizer**. Successfully recognized voice is played as output and return the recognized voice is to the processing part for converting it into text for further processing.

B. Processing [__main__.py, MySQLDBConnection.py]

The processing part is the heart of the whole section which is described below.

1. User voice processing (Speech to Text) [**__main__.py**]

   Input voice is converted into text using voice audio to text converter. The converted text is transferred to the gesture data search class.

2. Text-based gesture data search [**MySQLDBConnection.py**]

   Gesture files are listed in the MySQL database. This database contains a gesture file list. Table name with **speech_to_gesture** has some fields which are **speech_data, gesture_file_location, file_type created_at, updated_at**.

   The converted text is searched in the database using MySQL **LIKE** operation and finds out the exact data. This operation is done by a function named **get_speech_wise_gif_file** which is located in **MySQLDBConnection.py** class. This function performs the MySQL **LIKE** operation and finds out the matches data. If data is not found in the database it returns an empty row which is processed by further error handling parts.

   Matched data is returned as a row in the **__main__.py** class where the **changeGIF** function processes the raw data.

3. Database based Data Processing

In the **changeGIF** function under the **__main__.py** class process the raw data. If raw data contains an **empty value** or **null** it throws a warning using **PyQt5.QtWidgets** with **QMessageBox** warning.

If raw data has values, then it processes the data based on table fields location and finds out the gesture file data location. This data location is transferred to the data extraction and display section.

4. Data extraction and ready for display

After getting the data location it is prepared for display and sent to the display section for further processing.

C. Output [__main__.py, Speaker.py]

The output section is the main display part which plays video files or gif files against input voice.

1. Recognized voice play [**Speaker.py**]

Though input voice is recognized by Google **voice recognizer** but recognized text is played via **Speaker.py** class. **Speaker.py** has a speaker function that uses **gtts** and **playsound** packages for play sound of the recognized text.

2. Display the matches output through Graphical User Interface [**__main__.py**]

After getting the gesture data location it is displayed in **Graphical user Interface** through **QMovie** of **pyQT5** building function. It is displayed gesture file one after another based on user voice input. If data is not found, then it shows an empty box with a warning message box.

D. Error Handling Parts

Sometimes data sources cannot be found or codes may be broken down due to unwanted input. So we used try-catch blocks to handle the error and printed it in the console.

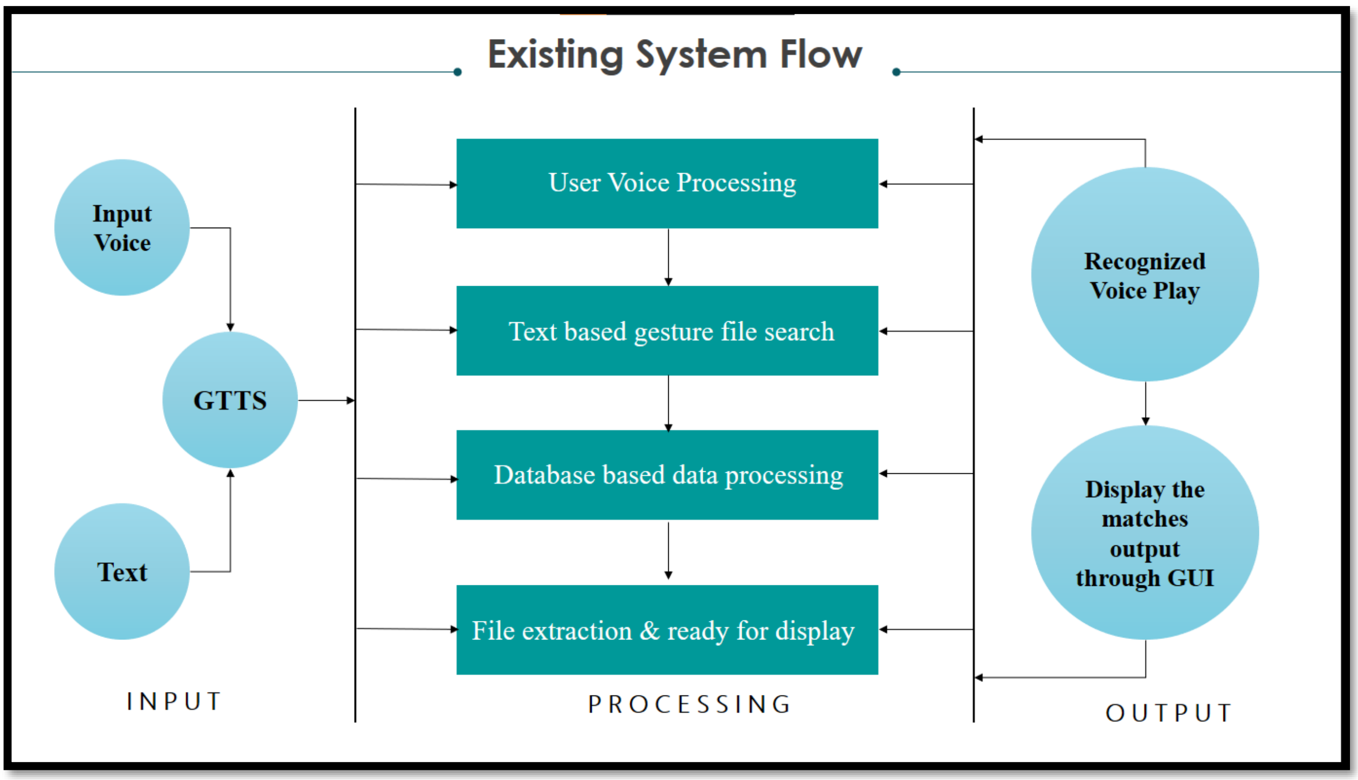

Existing System Flow
Input Voice
GTTS
Text
INPUT
User Voice Processing
Text based gesture file search
Database based data processing
File extraction & ready for display
PROCESSING
Recognized Voice Play
Display the matches output through GUI
OUTPUT

Figure 10: Existing System Flow

E. Graphical User Interface

The graphical user interface is the main user interface part that will interact with the end-user. The graphical part contains **main_window** which has a continue button and loader. After clicking the continue button the main the processing window will open which has some following features.

1. Gesture file display part

   It displays video or gesture files as an output. If data is not found it will be an empty box.

2. Home, Speak, and exit button.

   The home button is used for the main window and the exit button is used to quiet the application. Speak button is used for taking the voice input of the end-user.

3. Recognized voice display parts

   After correctly recognized voice will be displayed in the right window.

F. Limitations

Though the whole process is working fine it has some limitations. We have some limitations like google voice recognizer will not work without internet and data source limitations.

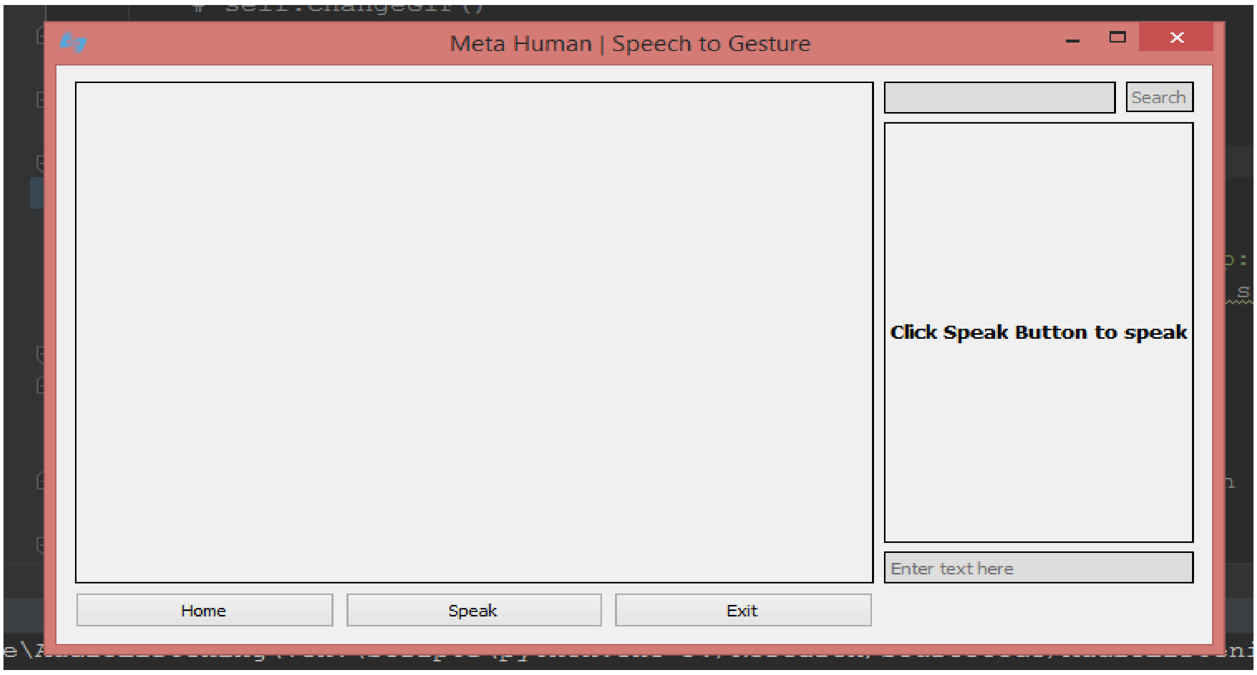


Figure 11: Main UI without Gesture Data

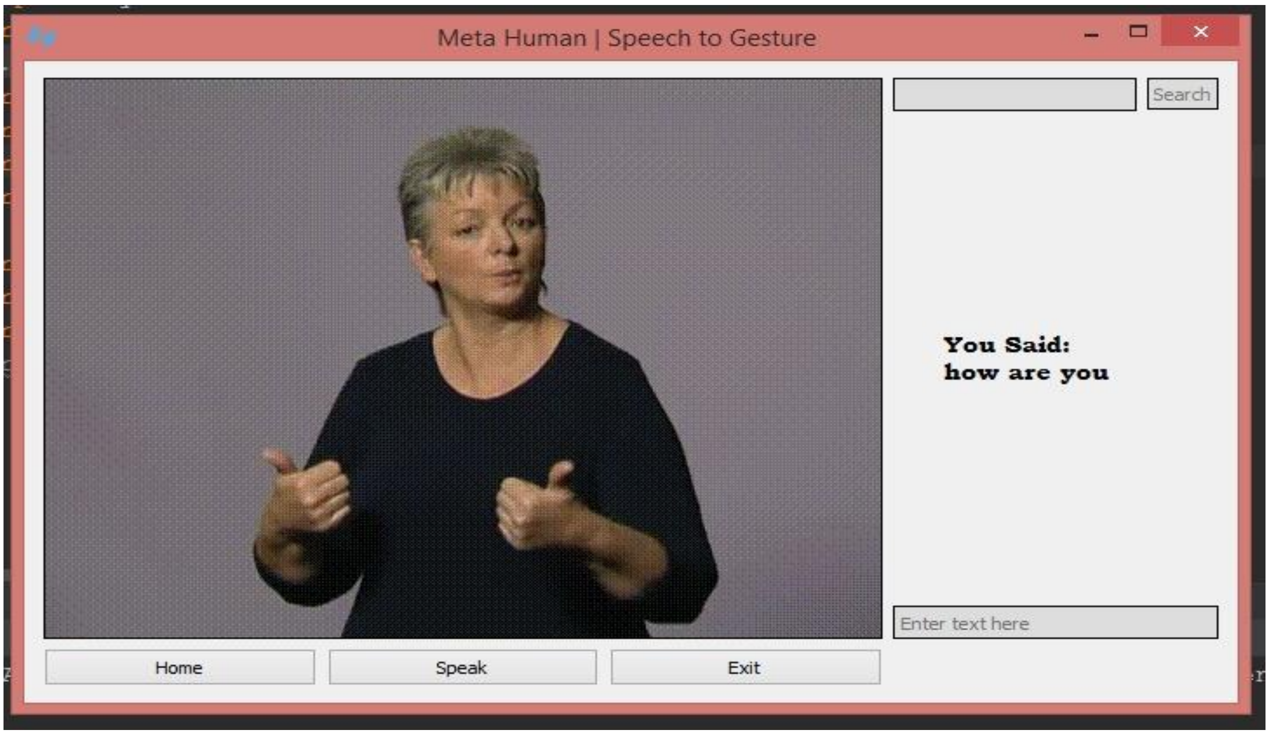
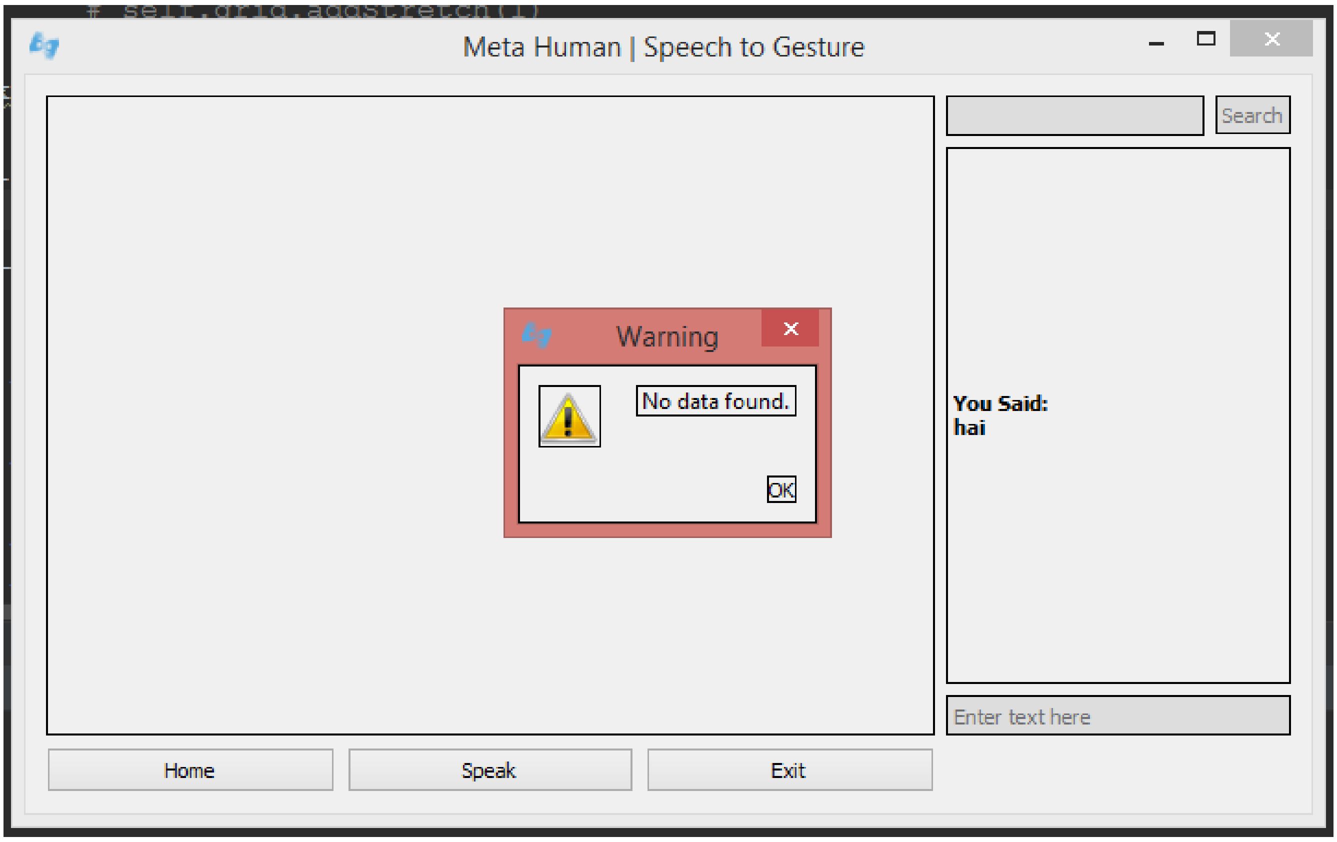


Figure 12: Main UI with Gesture Data



No data found.

You Said:
hai

Figure 13: Error Handling UI

Interface Feature Functionality

1. **Home Button** – To go back main home page
2. **Speak Button** – For voice input
3. **Search** – To search the desire Sentence/Word/Alphabet
4. **Text Message** – To give desire input
5. **Layout (Grid View)** –To show the desire video output
6. **Exit Button** – To close the window

Appendix B

SOURCE CODE

User Input take through Listen function

```python
def listen():
    try:
        with sr.Microphone() as source:
            # read the audio data from the default microphone
            print("START TALKING...")
            # os.remove("voice")
            speak("START TALKING...")
            audio_data = r.record(source, duration=5)
            # audio_data = r.record(source, duration=3)
            print("Recognizing...")
            # convert speech to text
            recognized = r.recognize_google(audio_data)
            print(recognized)
            print("YOU SAID: ", recognized)
            # speak(recognized)
            # sleep(2)
            return recognized

    except Exception as e:
        # print(exc)
        print(e)
        speak("Sorry. Voice can not recognized")
        return None
```

User inputed voice play with speak function

```
def speak(text):
    tts = gTTS(text=text,lang="en")
    lowercase_str = uuid.uuid4().hex
    filename = lowercase_str+".mp3"
    tts.save(filename)
    playsound(filename)
    os.remove(filename)
```

mySQL DB Connection

```
def db_connection():
    try:
        mydb = mysql.connector.connect(
            host="localhost",
            user="root",
            password="",
            database="metahuman"
        )
        return mydb
    except requests as exc:
        print("DB Exception {}", format(exc))
```

Gesture file Search in Database

```
def get_speech_wise_gif_file(speech_text):
    try:
        mydb = db_connection()
        mycursor = mydb.cursor()
        mycursor.execute("SELECT * FROM `speech_to_gesture` WHERE spe
ech_data LIKE %s", ("%"+speech_text+"%",))
        data_location = ""
        for x in mycursor:
            print(x)
            data_location = x[2]
        return data_location
    except requests as exc:
            print("DB Exception {}", format(exc))
```

Gesture file display in window

```
try:
    recognized_sentence = listen()

    if not recognized_sentence:
        data_location = ""
    else:
        data_location = get_speech_wise_gif_file(recognized_sentence)
    if not data_location:
        data_location = "style\data_not_found.gif"
        QMessageBox.warning(self, "Warning", "No data found.")
    movie = QMovie(data_location)
    self.setMovie(movie)
    movie.start()
except Exception as e:
    print(e)
```

Main Window Graphical UI

```
if __name__ == '__main__':
    app = QApplication(sys.argv)
    # app.setStyleSheet(stylesheet)
    MyApp = Window()
    # MyApp.setGeometry(358, 178, 650, 400)
    MyApp.setGeometry(358, 178, 680, 400)
    # MyApp.setFixedSize(640, 480)
    MyApp.setWindowTitle('Meta Human | Speech to Gesture ')
    MyApp.show()
    sys.exit(app.exec_())
```

For overall process we uses following packages

For Graphical User Interface

```
from PyQt5.QtWidgets import *
from PyQt5.QtGui import QMovie, QPixmap
from PyQt5 import QtCore
from PyQt5 import QtGui
from PyQt5.QtCore import QTimer
```

For mySQL DB Connection

```
import mysql.connector
import requests
```

For voice input

```
import speech_recognition as sr
import requests
```

For Play Sound

```
from gtts import gTTS
from playsound import playsound
import os
import uuid
```