



# **MINOR PROJECT REPORT**

## **AUDIO TO SIGN LANGUAGE TRANSLATOR**

Under the Guidance of

Prof. Padmavati

PEC University of Technology

Chandigarh

Submitted By:

Priyanshu Gautam (15103019)

Arushi Dhamija(15103029)

Sahil Khosla(15103040)

Rahul Garg(15103045)

Department of Computer Science and Engineering

PEC University of Technology, Chandigarh



## DECLARATION

We hereby declare that the project work entitled "Audio to Sign Language Translator" is an authentic record of our own work carried out at PEC University of Technology, Chandigarh as requirements of "Minor project" for the award of degree of B.E. Computer Science and Engineering, under the guidance and supervision of Prof. Padmavati.

We further declare that the information has been collected from genuine & authentic sources and we have not submitted this project report to this or any other university for award of diploma or degree of certificate examination.

Priyanshu Gautam (15103019)  
Arushi Dhamija(15103029)  
Sahil Khosla (15103040)  
Rahul Garg (15103045)

Date: \_\_\_\_\_

Place: \_\_\_\_\_

Certified that the above statement made by the students is correct to the best of my knowledge and belief.



## CERTIFICATE

Certified that the Project work entitled **Audio to Sign Language Translator** submitted by **Priyanshu Gautam, Arushi Dhamija, Rahul Garg and Sahil Khosla** for the fulfilment of Minor Project offered by PEC University of Technology during the academic year **2017-18** is a original work carried out by the student under my supervision and this work has not framed any basis for the award of and Degree, Diploma or such other titles.

Signature of the mentor

Date:

Name : Padmavati

Desgination: Assistant Professor, Department of Computer Science

Institution: PEC University of Technology



## ACKNOWLEDGEMENT

The completion of any inter-disciplinary project depends upon cooperation, co-ordination and combined efforts of several sources of knowledge. It gives us immense pleasure to take this opportunity to thank Dr. Manoj K. Arora for giving us such a great opportunity to do our Minor project in their esteemed organization PEC University of Technology, Chandigarh.

We owe our profound gratitude to our project Mentor, Prof. Padmavati, who took keen interest in our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

Finally, no word will be enough to express my deepest reverence to family and friends who have willingly helped us out with their abilities and without whose enthusiasm and support, We wouldn't have been able to pursue our goals.

Priyanshu Gautam (15103019)  
Arushi Dhamija(15103029)  
Rahul Garg (15103045)  
Sahil Khosla (15103045)

## ABSTRACT

Sign language is a visual language used by deaf and dumb people as their mother tongue. Unlike acoustically conveyed sound patterns, sign language uses body language and manual communication to fluidly convey the thoughts of a person. It can be used by a person who has difficulties in speaking or by a person who can hear but can not speak and also, by normal people to communicate with hearing disabled people. As far as a deaf person is concerned, having access to a sign language is very important for their social, emotional and linguistic growth.

Our project aims to bridge the gap between these Deaf people and normal people with the advent of new technologies of web applications, Machine Learning and Natural Language Processing. The main purpose of this project is to build an interface which accepts Audio/Voice as input and converts them to corresponding Sign Language for Deaf people. It is achieved by simultaneously combining hand shapes, orientation and movement of the hands, arms or body. The interface works in two phases, first converting Audio to Text using speech to text API (python modules or Google API) and secondly, represent the text using Parse Trees and applying the semantics of Natural Language Processing (NLTK specifically) for the lexical analysis of Sign Language Grammar.

The work builds upon the rules of ISL(Indian Sign Language) and follows the ISL rules of Grammar.

# TABLE OF CONTENT

## Chapter 1 - Insight

1.1 - Introduction.....	8
1.2 - Indian Sign Language Grammar .....	9
1.3 - HamNoSys: A notation System for Sign Language.....	9
1.4 Importance of HamNoSys in Generation of DignLangugae.....	10
1.5 SIGML Language.....	13
1.6 Avatars.....	13

## Chapter 2 - Background

2.1 -Direct Translation Systems.....	14
2.2 Sceernshots.....	32
2.3 Existing Research.....	15

## Chapter 3 - Proposed Work

3.1 System Architexture.....	24
------------------------------	----

## Chapter 4 -Implementation

4.1 - Website.....	31
4.2 - Screenshots of Website.....	23
4.3 UML Diagrams.....	35

## Chapter 5 - Results and Discussions

5.1 Accuracy.....	38
-------------------	----

## Chapter 6 - Conclusions and Future Scope.....45

References.....	46
-----------------	----

## LIST OF FIGURES

1.1 Structure Of HamNoSys.....	10
1.2 HamNoSys symbols and there descriptions .....	11
1.3 HamNoSys Of Woman.....	12
1.4 Sign Of Woman.....	12
1.5 SiGML for word “DEAF” .....	13
1.6 Different Avatars with their names.....	13
2.1 Structure of TESSA System.....	15
2.2 Architecture Of SignSynth System.....	16
2.3 Architecture of ASL Workbench.....	19
2.4 Architecture Of ViSiCAST System.....	20
2.5 Architecture of Text to Indian Sign Language Machine Translation.....	21
2.6 Architecture Of INGIT .....	23
3.1 System Architecture.....	23
3.2 Workflow of System.....	29
4.1 Main Page.....	31
4.2 Website’s LOGO.....	31
4.3 Login Portal.....	32
4.4 Screenshot of configuration.....	32
4.5 Screenshot of SiGML.....	33
4.6 USE Case Diagram.....	34
4.7 Class Diagram.....	35
4.8 Activity Diagram.....	37
5.1 Screenshot Of Website.....	38
5.2 Output Figure 1.....	39
5.3 Output Figure 2.....	39
5.4 Output Figure 3.....	40
5.5 Output Figure 4.....	40
5.6 Output Figure 5.....	41
5.7 Output Figure 6.....	41
5.8 Output Figure 7.....	42
5.9 Output Figure 8.....	42
5.10 Output Figure 9.....	43
5.11 Output Figure 10.....	43

## CHAPTER-1 INSIGHT

### 1.1 Introduction

A sign language (SL) is a natural visual-spatial language that uses the three-dimensional space to articulate linguistic utterances instead of sound to convey meaning, simultaneously combining handshapes, orientation and movement of the hands, arms, upper body and facial expressions to express the linguistic message. The language came into existence because of the deaf, dumb and hard of hearing people in India. All around the world there are different communities of deaf and dumb people and thus the language of these communities will be different. Just like there are many spoken languages in the world like English, French, and Urdu etc. Similarly there are different sign languages and different expressions used by hearing disabled people worldwide. The Sign Language used in USA is American Sign Language (ASL); British Sign Language (BSL) is used in Britain; and Indian Sign Language (ISL) is used in India for expressing thoughts and communicating with each other. The interactive systems are already developed for many sign language e.g. for ASL and BSL etc.

In India, approximately 5.07 million people who suffer from hearing disability. Among them, more than 30% people are below 20 years of age and about 50% are between 20 years and 60 years of age. These people are generally unable to speak properly because of which they use sign language to communicate with others. As sign languages do not have well defined structure or grammar, therefore there is no or very less acceptability of these signs outside the small world of these differently abled people. In research on American Sign Language proved that sign language is a full-fledged language with its own grammar, its own syntax, and other linguistic attributes. To prove the same for other sign languages, there are some efforts including Indian Sign Language. Particularly, research on ISL started in 1978 and it has been found that ISL is a complete natural language with its own grammar and syntax. Communication for the hearing impaired people in common places like railway stations, bus stands, banks, hospitals etc., is very difficult because a hearing person may not understand the sign language used by the deaf person to communicate. Also, a hearing person cannot convey any message to deaf person as he/she may not know the sign language. To make the communication between deaf and non-deaf community, the language translation is must.

According to 2011 census, in India

- 6.3% of the total population (i.e. 63 million) are suffering from significant hearing loss
- Out of these people, 76-89% of the Indian Deaf have no knowledge of language, either signed or spoken/written

Reason behind the low literacy rate can be either of the following

- Lack of Sign Language interpreters.
- Unavailability of ISL tool.
- Lack of researches on ISL.





Due to their inability, communication for the deaf community in common places like railway, bank, and hospitals is difficult. To help them communicate better with the rest of the world, a system is needed which will enable the conversion of text to Indian Sign Language and vice versa. These systems will increase the quality of living of this community. Sign languages have not been studied as extensively as spoken languages, and there is still much left to be learned about them. is the concept of people, individually or as a group, appearing at a location for a previously scheduled event

Audio to sign language translator is a web-based application developed for deaf or hard to hearing people. It translates English audio into Indian Sign Language. The system takes simple English sentences as input and generates ISL-gloss which may then be converted into the Hamburg Notation System (HamNoSys). The HamNoSys representation will provide signing instructions to the sign synthesis module, to generate an animated representation of ISL to the user. Dependency trees is used to represent ISL syntax.

## 1.2 Indian Sign Language Grammar

Like other languages, Indian Sign Language has its own grammar. It is not dependent on the spoken language – English or Hindi. The sign language is not same as the manual representation of spoken English or spoken Hindi. It has certain unique and distinct features like:

1. All the sign representation for numbers are done with appropriate hand gesture for every number. Eg. the sign for 45 will be the representation of four followed by sign representation of 5.
2. The signs for family relationships are preceded by signs for 'male/man' and 'female/woman'. The interrogative sentences having words like WHAT, WHERE etc. are represented by placing these questions at the end of sentences.
3. The ISL consists of various non-manual gestures including mouth pattern, mouth gestures, facial expressions, body postures, head position and eye gaze. ISL has essentially a Subject-Object-Verb word order (unlike English which is Subject-VerbObject).

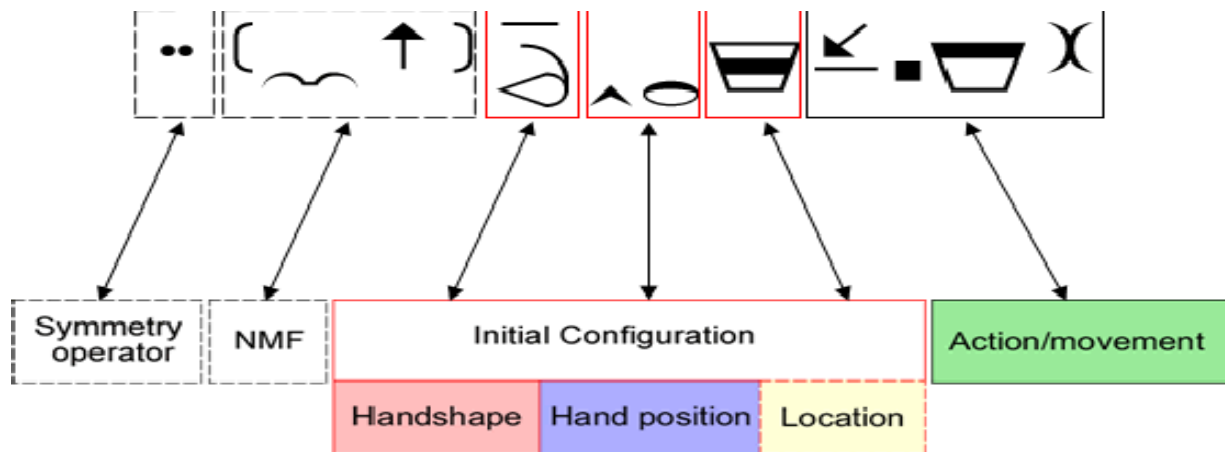
## 1.3 HamNoSys: A Notation System for Sign Language

Sign language does not have a particular written form. For defining, a sign there must be a notation system that can help to write signs. The Hamburg sign language notation system known as HamNoSys is one such system [8]. It uses phonetic transcription system to transcribe signing gestures. HamNoSys is a syntactic representation of a sign, which provides computer processing for signs [9]. HamNoSys has its roots in the Stokoe notation system which introduced an alphabetic system to describe the sub lexical parameters including hand location, hand configuration and hand movement to give a phonological description of signs [10]. First version of HamNoSys was defined in 1984. Objectives of HamNoSys are given as below.

1. HamNoSys transcriptions should be useful for all sign languages in the world, and the notations should not rely on conventions which differ from country to country.
2. Meaning of alphabets should be clear and easy to remember for the users.
3. It should be possible to transcribe any signed utterance with HamNoSys; notation of majority of signs should make use of such principles which results in shorter notation for the average sign.
4. The notation system should be usable in standard text processing and database applications with their computer supported transcription.
5. HamNoSys should allow both for a general evolution and specializations. New versions of the system should be compatible with the present ones.

## 1.4 Importance of HamNoSys in Generation of SignLanguage

HamNoSys is a Stokoe based notation system which is having 200 largely iconic characters [11]. These HamNoSys symbols are used to represent hand orientation, hand shape, and hand location corresponding to other parts of the body. For writing signs, these parameters should be assigned with some value. HamNoSys provides symmetry operator for representing two handed signs non-manual components. Non-manual phonological can be represented by replacing hand graphemes with the symbols for body part such as head [12]. But facial movements like raised eyebrows or puffed out cheeks are complex to represent. The parameters of a sign are written in the order of symmetry operator, non-manual components, hand shape, hand position, hand location and hand movement as shown in figure 1.1



**Figure 1.1 : Structure of HamNoSys**

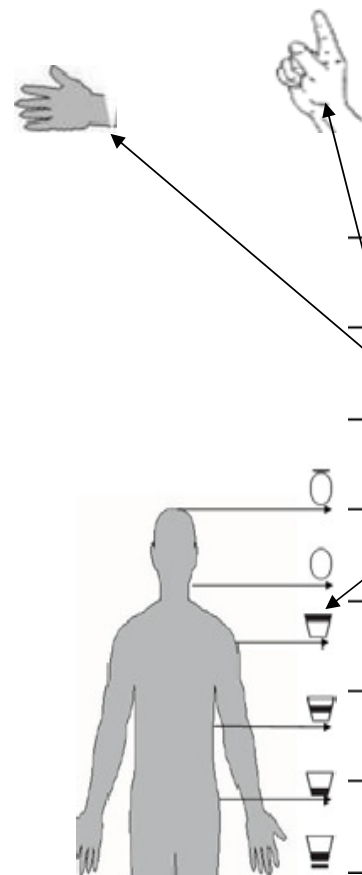
HamNoSys for a single sign describes the initial posture of the non-manual features, hand shape, hand orientation and hand location plus the actions which change this posture in a sequence or parallel. In case of two handed signs, initial posture notation is preceded by the symmetry operator that defines how the description of the non dominant hand gets copied into the dominant hand.








### Example

Traditional Stokoe based notation system includes only basic hand parameters. HamNoSys has extended it by expanding sign representation parameters. These parameters include:

- Dominant hand's shape,
- Location of the dominant and the non-dominant hand with respect to the body,
- Extended finger orientation of both dominant and non-dominant hand,
- Palm orientation of both hands,
- Movements (straight, curved, or circular) and,
- Non-manual signs.

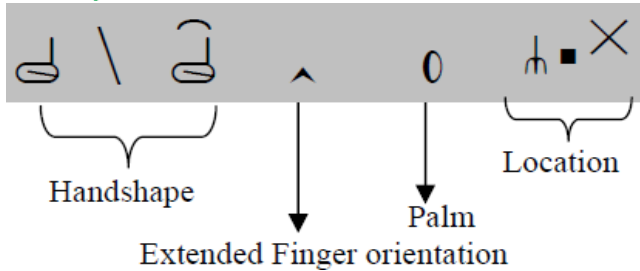
Figure 1.2 shows examples of different HamNoSys symbols and their descriptions.



Symbol	Description
	index finger stretched
	extended finger ahead
	palm orientated left
	location shoulder height
	fully stretched out
	hand move ahead
	hand move right

**Figure 1.2 : HamNoSys symbols and there descriptions**

Figure 1.3, shows the HamNoSys of the word "WOMAN" whereas its sign is shown in figure 1.4



**Figure 1.3: HamNoSys of "WOMAN"**



**Figure 1.4. Sign of WOMAN**

HamNoSys creates language independence because it is not specific to a particular Sign Language, it can be used to describe any signs of any language. HamNoSys creates visual phonetics of signs which are language independent [13]. HamNoSys is used as the basis for Signing Gesture Markup Language (SiGML), which is further used to generate animation using animated signing avatar.

## 1.5 SIGML Language

SiGML is Signing Gesture Mark-up Language. It was developed for specifying signing sequences in ViSiCAST project. It defines HamNoSys symbols into XML tags form. It was developed at the University of East Anglia. It provides communication tools in form of animated figures. SiGML representation made from HamNoSys notation of sign language is readable by 3D rendering software [14]. SiGML for word "DEAF" is shown in figure 1.5.

```
<sigml>
  <hns_sign gloss="Deaf">
    <hamnosys_nonmanual>
    </hamnosys_nonmanual>
    <hamnosys_manual>
    <hamfinger23/>
    <hamextfingerui/>
    <hambetween/>
    <hamextfingerul/>
    <hampalmd/>
    <hamear/>
    <hamtouch/>
    </hamnosys_manual>
  </hns_sign>
</sigml>
```

**Figure 1.5: SiGML for word "DEAF"**

## 1.6 Avatars

Avatars are described as “digitally created humanoids” or “virtual bodies”. They take input as SiGML language or XML text and generate animation corresponding to inputted text. Animation frame definitions are given as input to avatar in a sequence. They describe the static pose of the avatar. These sequences also describe the avatars time stamp, *i.e.*, at what time avatar will be placed in that pose [15]. When these avatars are placed in sequence of poses then rendering software produces signing animation corresponding to specified frame definitions. Many avatars have been developed for generating sign animations. They are given names as “Anna”, “Marc” and “Francoise” as shown in figure 1.6.



**Figure 1.6: Different Avatars with their names**

## Chapter-2 BACKGROUND

### 2.1 Direct Translation Systems

These systems perform their processing on the individual words of the source language string; translation is achieved without performing any form of syntactic analysis on the original input text. Generally the word order of the sign language remains the same as that of the English text. But in the case of English to Indian Sign Language, target sign language may not allow the same word order. With this, system also assumes a strong knowledge of both the English as well as the target sign language. Systems based on this approach are TESSA and SignSynth project.

### 2.2 Transfer Systems

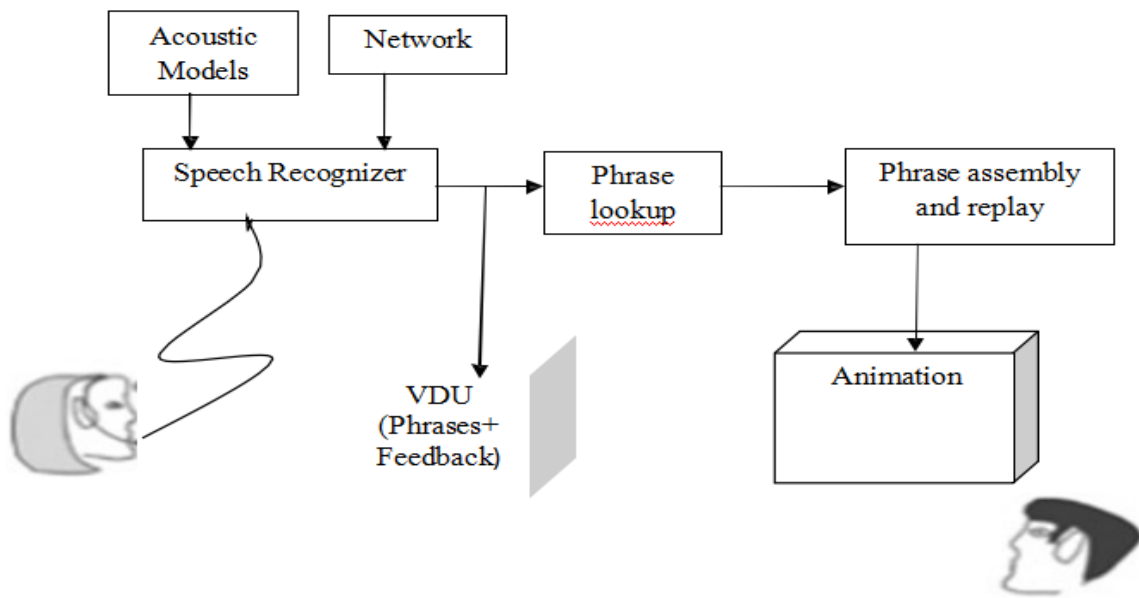
These systems analyze the input text to some syntactic or semantic level, according to that a special set of transfer rules are employed which read the information of the source language structure and produce a syntactic or semantic structure in the target language. Afterwards, a generation component is used to convert this linguistic structure into a target language surface form. The transfer grammar approach is used in text to SL MT systems and also in text to text MT systems. Systems based on this approach are TEAM, ASL workbench and ViSiCAST translator.

### 2.3 Existing Research

Researchers all over the world have been working on automatic generation of sign language. These automated systems take text or speech as input and produce animation for it.

#### 2.3.1 TESSA

Cox *et al.* (2002) had developed TESSA system based on direct translation approach. It is a speech to British Sign Language translation system which provides communication between a deaf person and a post office clerk. TESSA takes English as input text, look up each word of the English string in the English-to-Sign dictionary, concatenates those signs together, and blends them into an animation. In this system, formulaic grammar approach is used in which a set of predefined phrases are stored for translation and translated by using a phrase look up table. Architecture of system is shown in figure 2.1

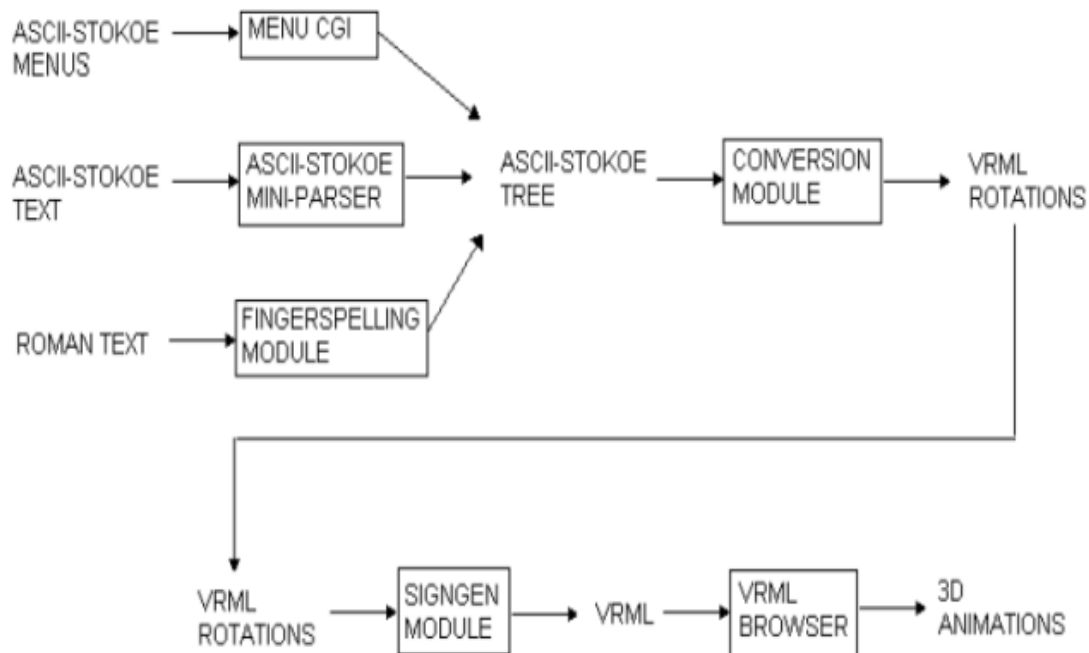


**Fig. 2.1 Structure of TESSA System**

The post office clerk uses headset microphone and speech recognizer. In speech recognizer, legal phrases from the grammar are stored. When clerk speaks a phrase, speech recognizer matches it with legal stored phrases. Clerk's screen displays topics available corresponding to uttered phrase, for example, "Postage", "Bill Payments" and "Passports". From these phrases clerk select one phrase according to requirement and sign of that phrase is displayed on the screen. Because of use of a small set of sentences as templates TESSA is a very domain specific system. Currently there are around 370 phrases stored in this system .

### 2.3.2 SignSynth Project

It is a prototype sign synthesis application which is under development at the University of New Mexico. It converts input text into American Sign Language. Sign synthesis and speech synthesis performs almost same task. The only difference is in the form of outputs. Thus the architectures of both of these are also almost same. Sign synthesis uses Perl scripts through the common gateway interface (CGI) for performing signing animation. Architecture of SignSynth is shown in figure. 2.2



**Fig. 2.2 Architecture Of SignSynth System**

It has three main interfaces. First interface, MENU CGI offers menus for signs by which user can specify the phonological parameters. Additional menus help such users who know nothing about ASCII-Stokoe. They can directly select the hand shape, hand location and hand orientation for each hold. Second interface, ASCII-Stokoe mini-parser is for more advanced users to type with additions for timing and non-manuals. The finger spelling module helps the user to type in the Roman alphabet. This module outputs an ASCII-Stokoe tree which becomes as input to the conversion module. Conversion module further produces Web3D rotations for joints which are used. After the creation of rotations, they become the input for SignGen module. SignGen module integrates them with Web3D humanoid for creating complete web file with animation data. Then with the help of plug-in, animation is played. SignSynth is free and open-source. It has simple humanoid and Perl CGI which runs on any web server.



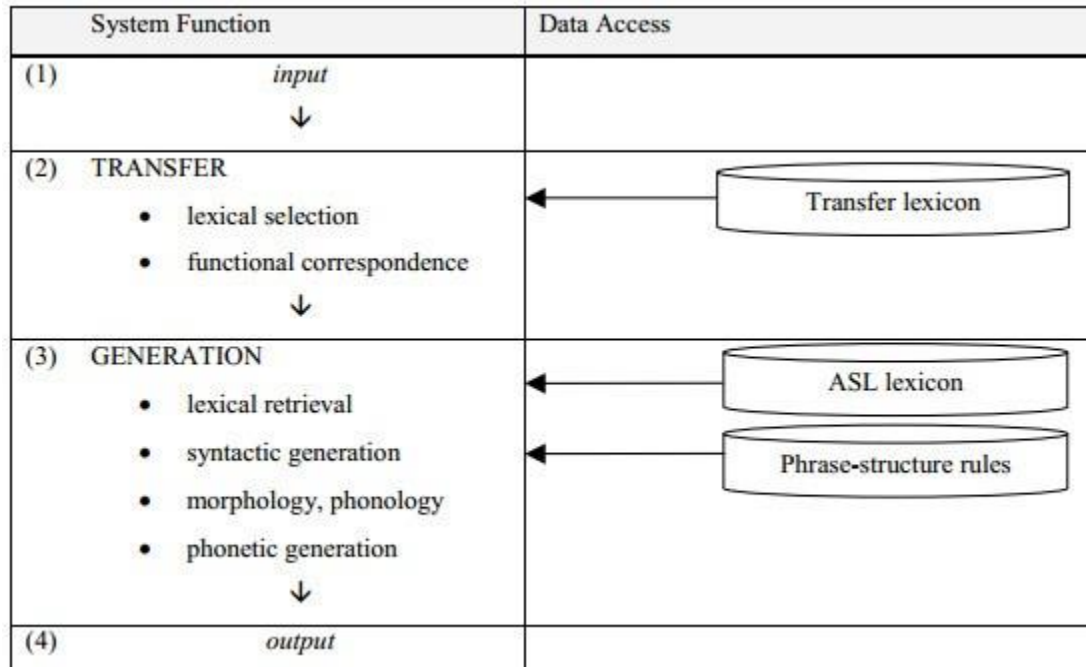


### 2.3.3 TEAM

Zhao *et al.* (2002) had developed a system TEAM based on machine translation. It is a translation system which converts English text into American Sign Language. In this system TAG parser analyzes the input text. It involves two major steps. First step includes the translation of input English sentence into intermediate representation. It considers syntactic, grammatical and morphological information. In second step, its interpretation is performed. Its representation is done as motion representation which actually controls the human model and produce ASL signs. This system uses gloss notations for generating intermediate representation. Firstly it analyzes the word order and then generates glosses which represent facial expressions, sentence types, and morphological information. It uses a synchronous tree adjoining grammar (STAG) to map information from English text to ASL. It is first translation system which considers visual and spatial information along with linguistic information associated with American Sign Language. It is not limited to ASL; it is expandable to other signed languages because of its flexibility.

### 2.3.4 The ASL Workbench

Speers (2001) had proposed and implemented a system ASL workbench. It is a Text to ASL Machine Translation system. It analyses the input text up to an LFG-style f- structure. Its representation abstracts some of syntactic specifics from input text and replaces them with linguistic features of the text. Architecture of ASL workbench is shown in figure 2.4. Workbench system included a set of specially written rules for translating an f-structure representation of English into one for ASL. It implements transfer module and generation module. Input for translation module is an English LFG f- structure. It is converted into an ASL f-structure using structural correspondence and performing lexical selection. The ASL f-structure becomes input to the generation module. Generation module creates American Sign Language c-structure and p-structure corresponding to the sentence. ASL workbench performs fingerspell of the word if the lexical element is noun.



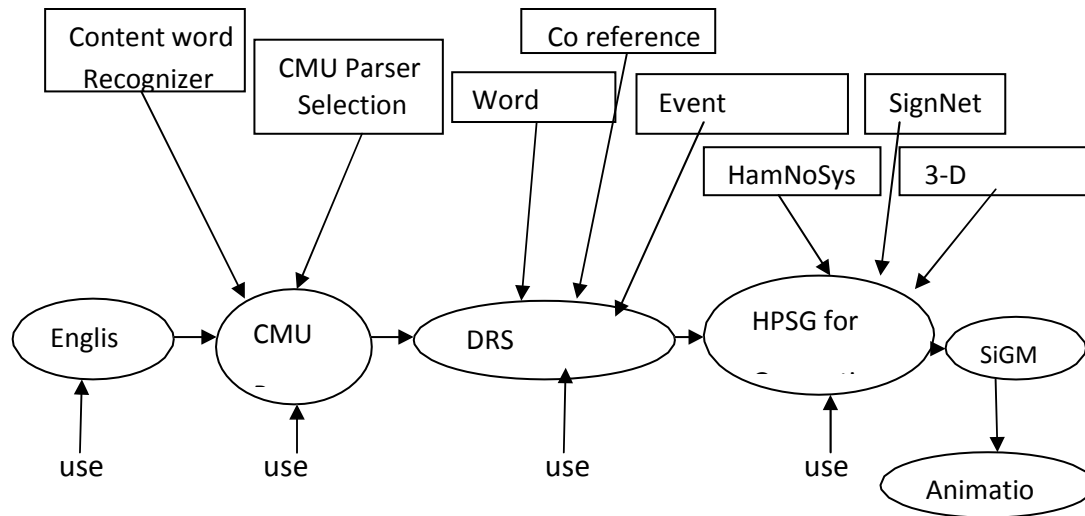
**Fig. 2.3 Architecture of ASL Workbench**

If element is other than noun then translation fails, although translator can create entry corresponding to the word in ASL lexicon. It can also create entry in transfer lexicon if necessary and re attempt the translation.

### 2.3.5 ViSiCAST Translator

Safar and Marshall (2001) had developed English to British Sign Language translation system. It uses semantic level of representation for performing English analysis to the BSL generation. It includes investigation of sign language delivery using different technologies. The architecture of ViSiCAST is shown in figure 2.5. This system is user friendly. User enters English text into the system, at this stage user can change or alter the text according to the requirement. After that, at the syntactic stage inputted text is parsed with CMU (Carnegie Mellon University) link grammar parser. From this parser, an intermediate semantic representation is made in the form of Discourse Representation Structure (DRS). From this representation, morphology and syntax of sign

generation is defined in Head Driven Phrase Structure Grammar (HPSG). Here, signs are shown in form of HamNoSys and can be edited.



**Fig. 2.4 Architecture of ViSiCAST System**

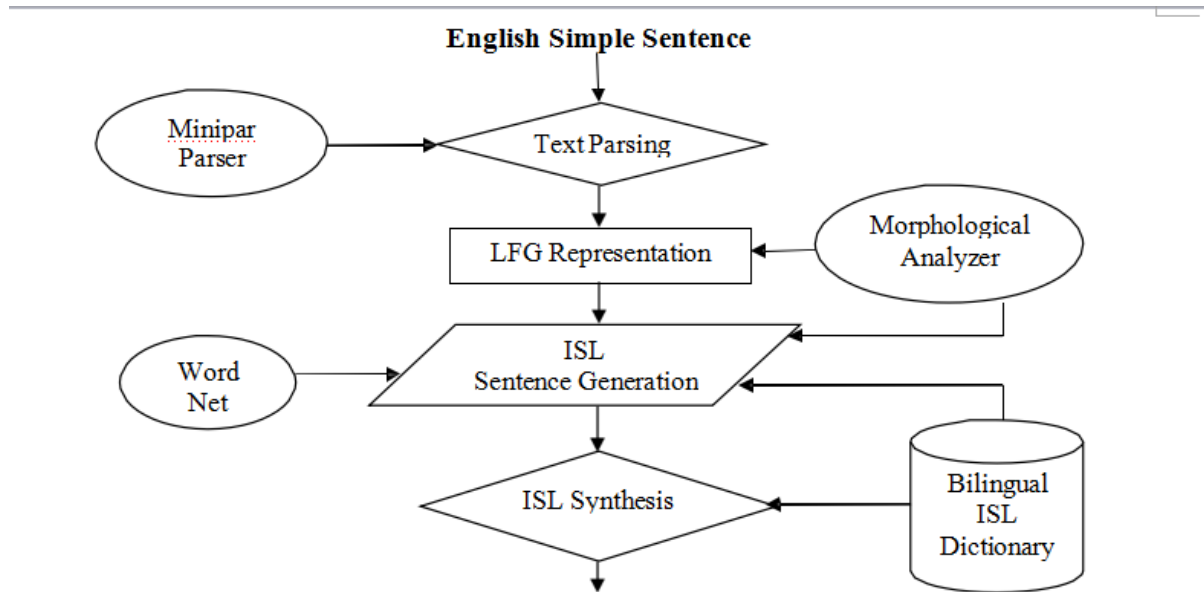
After this, signing symbols are linked with SiGML, which describes signing notations into XML form. SiGML is easily understandable by 3D rendering software, which plays animation corresponding to inputted text.

### 2.3.6 Machine Translation System from Text-To-Indian Sign Language

Dasgupta et al. (2008) had developed a system based on machine translation approach. It takes English text as input and generates signs corresponding to the inputted text. Architecture of system is illustrated in figure.

Figure shows four essential modules of the system which are, input text preprocessor and parser, LFG f-structure representation, Transfer Grammar Rules, ISL Sentence Generation and ISL synthesis. Simple English sentence is inputted to the parser. Simple sentence means which sentence has only one main verb in it. Minipar parser parses the sentence and make dependency tree. A phrase lookup table

of around 350 phrases and temporal expressions is made before parsing. English morphological analyzer identifies plurality of nouns.



**Fig. 2.5 Architecture of Text to Indian Sign Language Machine Translation**

LFG functional structure (f-structure) encodes grammatical relation of the input sentence. It also includes the higher syntactic and functional information representation of a sentence. This information is represented as a set of attribute-value pairs. Where attribute is for name of a grammatical symbol and value is for feature possessed by the constituent. It becomes as input to the generation module which apply transfer grammar rules on it so that it could transfer source sentence into target structure. Lexical selection and word order correspondence are two main operations that are performed during generation phase. Lexical selection is done using English to ISL bilingual lexicon. For example, word like “BREAKFAST” in English is replaced by “MORNING FOOD” in ISL. ISL uses Subject-Object-Verb (SOV) word order. Example sentence given in 2.1 shows the change in word order in ISL.

**English** “I have a LAPTOP”

**ISL** “I LAPTOP HAVE”.



The final Indian Sign Language structure is achieved by the addition or deletion of words and restructuring of source representation.

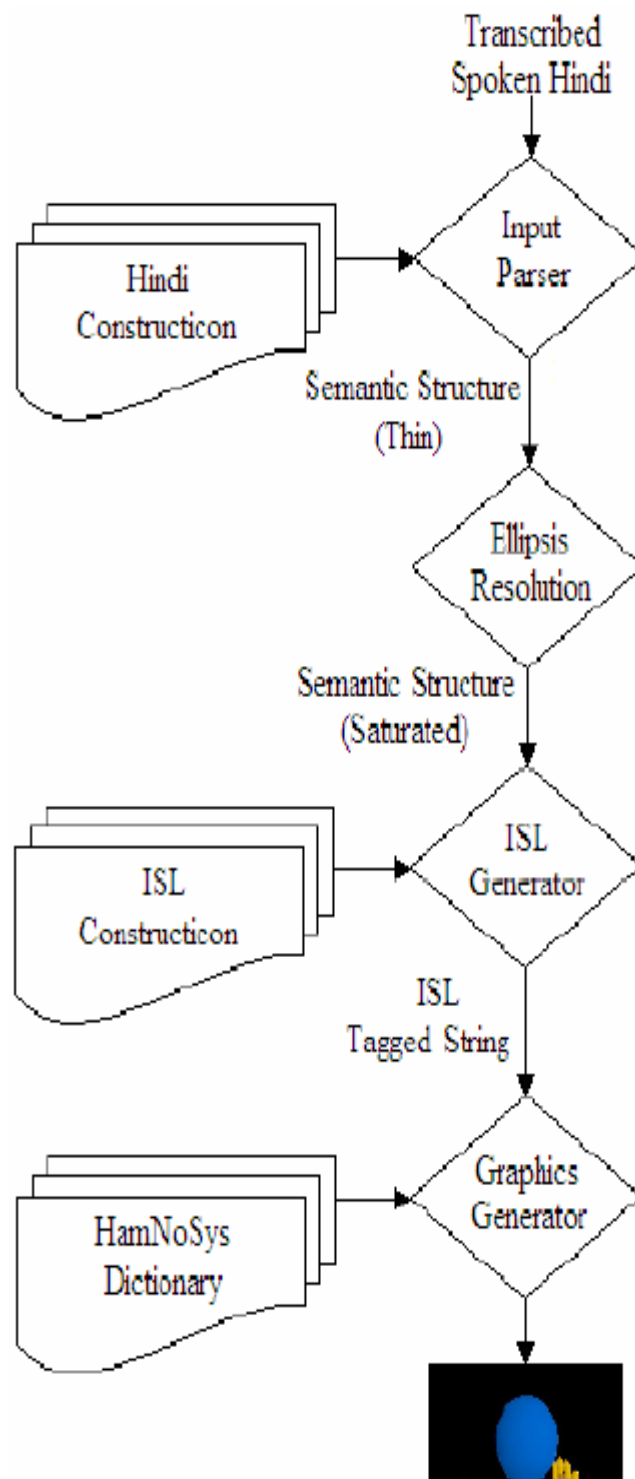
## 2.3.7 INGIT

A project being worked upon in the Indian Institute Of Technology, Kanpur, It is a prototype system designed as a proof-of-concept for a Hindi to ISL translator in the railway reservation domain, a common public need for citizens. The system, named INGIT 1 translates input from the reservation clerk into Indian Sign Language, which can then be displayed to the ISL user. INGIT currently accepts transcribed spoken language strings as input and generates ISL-gloss strings which are converted to a graphical display via HamNoSys (Prillwitz et al 1989) simulation. Only the utterances of the reservation clerk are translated since the deaf client can respond via the paper form.

INGIT works on strings of transcribed Hindi spoken text. A domain-specific construction grammar for Hindi, implemented in FCG, converts the input into a thin semantic structure which is input to ellipsis resolution, after which we obtain a saturated semantic structure. Depending on the type of utterance (statement, query, negation, etc) a suitable ISL-tag structure is generated by the ISL generator. This is then passed to a HamNoSys converter to generate the graphical simulation.

It is a crossmodel translation system from Hindi strings to Indian Sign Language for possible use in the Indian Railways reservation counters. The system translates input from the reservation clerk into Indian Sign Language, which can be then displayed to the ISL user. They have used Fluid Construction Grammar (FCG) [3], for constructing the grammar for Sign language. In this the domain-specific construction grammar for Hindi converts the input into a thin semantic structure which is an input to ellipsis resolution, after which a saturated semantic structure is obtained. Depending on the type of utterance (statement, query, negation, etc.) a suitable ISL-tag structure is generated by the ISL generator. This is then passed to a HamNoSys [4] converter to generate the graphical simulation. For validating the system, they collected small corpus on six different days. This corpus was based on interaction with speaking clients at a computer reservation counter. They after evaluation found the interaction constituted 230 words, of which many were repeated. The vocabulary of 90 words included 10 verbs in various morphological forms (e.g. work, worked, working etc.), 9 words related to time, 12 words specific to the domain (e.g. ticket, tatkal, etc.), Other words were numerals (15), names of months (12), cities (4) and trains (4) as well as digits particles etc. The INGIT system has three main modules:

1. Input parser   2. Ellipsis Resolution Module   3. ISL Generator (including ISL lexicon with Domain Bounded English to Indian Sign Language Translation Model



**Fig. 2.6 Architecture of INGNIT**

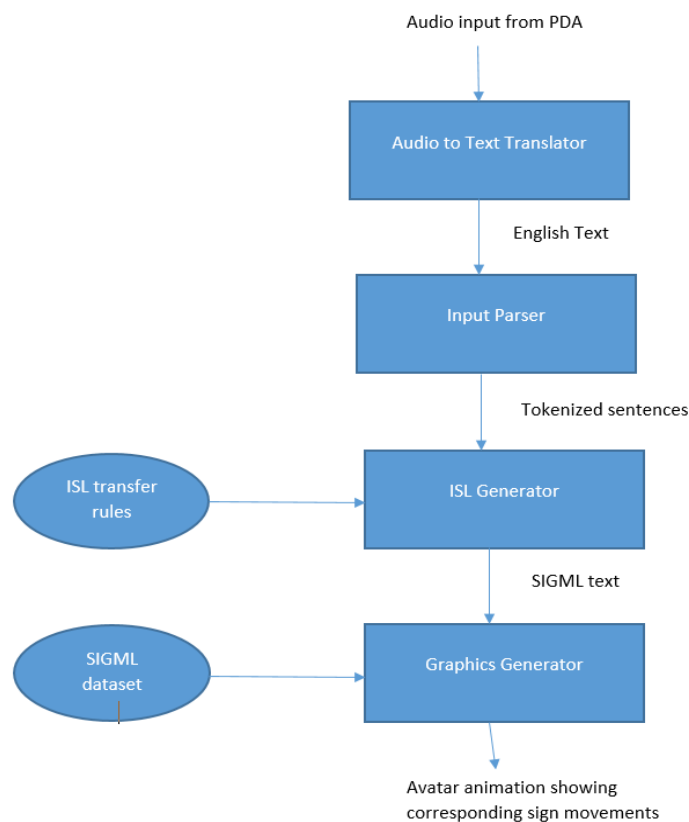
## Chapter 3 Proposed Work

The proposed translation system takes the audio text as input and system converts the audio into English text, which is further translated into ISL grammar by using parse tree structure. Machine based translation has been used to translate English into ISL based grammar. Rules have been defined for translating English into semi-structured parse tree and further bi-linguistic rules have been applied to translate this parse tree according to rules of ISL grammar. This modified tree is further converted to text, which is then processed to convert it into the form of ISL language.

### System Architecture

The system consists of following modules:

1. Audio to text converter
2. Input Parser
3. ISL Generator
4. Graphics Generator



**Fig. 3.1 System architecture**

## 1. Audio to text Translator

External or in-built microphone on any Personal Digital Assistant (PDA) is used to give input for the module. The module uses Google Cloud Speech for audio to text. The input can be given in any language. The output is given as English text string. The module has been developed in python script. The speech is recognized by the module and punctuation is added accordingly to the input text.

## 2. Input Parser

The input paragraph is tokenized into sentences. Using machine learning and Natural Language Processing tools, each sentence is tokenized into words. The output of this module is a list of token for each line of text.

Tokenization:

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called *tokens* , perhaps at the same time throwing away certain characters, such as punctuation. Here is an example of tokenization:

Input: Friends, Romans, Countrymen, lend me your ears;

Output: 

Friends	Romans	Countrymen	lend	me	your	ears
---------	--------	------------	------	----	------	------

Tokenizer:

A tokenizer divides text into a sequence of tokens, which roughly correspond to "words".

## 3. ISL Generator

The purpose of this module is to convert the input text with grammar of English into text with grammar of ISL. This has been achieved by using Transfer-based translation.

### **Transfer-based Translation:**

In Transfer Based Translation systems, plain text is given as an input to the system. The system then analyses it syntactically and also semantically. It is then transferred into a sign language.

In this system, source language is transformed into some intermediate abstract text, some linguistic rules are then applied to that text to transfer it into target language.

Transfer based



Translation is thus also known as “Rule based Translation” since some special set of rules are used which read the information of the source language and produce a semantic or syntactic structure in target language.

In our system, source language is converted as phrase structure trees by a parser. A natural language parser is a program that works out the grammatical **structure of sentences**, for instance, which groups of words go together (as "phrases") and which words are the **subject** or **object** of a verb. Probabilistic parsers use knowledge of language gained from hand-parsed sentences to try to produce the *most likely* analysis of new sentences. In our system, we have used Stanford parser which is an implementation of probabilistic natural language parsers, both highly optimized PCFG and lexicalized dependency parsers, and a lexicalized PCFG parser.

After converting the source language to phrase structure trees, we use ISL grammar rules to modify structure of phrase structure tree such that modified tree now represents the structure and grammar of ISL.

### Grammar Rules for Conversion of English Sentence to ISL Sentence

Translation of one spoken language to another spoken language is complex task if both the languages have different grammar rules. The complexity is increased many folds when source language is spoken language and the target language is sign language. For translating English text to Indian sign language, a comparison of grammar of both the languages is must:

English language grammar	Indian sign language grammar
English grammar is well structured and a lot of research work has been carried out to define the rules for it. English grammar follows the subject-verb-object order.	ISL is invented by deaf and a little work has been done to study the grammar of this language. The structure of sentences of ISL follows the subject-object-verb order[13].
English language uses various forms of verbs and adjectives depending upon the type of the sentence. Also, a lot of inflections of the words are used in English sentences.	ISL does not use any inflections ( gerund, suffixes, or other forms ), it uses the root form of the word.
English language has much larger dictionary	Indian sign language has a very limited dictionary, approximately 1800 words[10].
Question word in interrogative sentences is at the start in English	In Indian sign language, the question word is always sentence final[1].
A lot of helping verbs, articles, and conjunctions are used in the sentences of English	In Indian sign language, no conjunctions, articles or linking verbs are used

**Table 3.1 : Comparison of Grammar of English and Indian Sign Language**

For conversion of English sentence to a sentence as per ISL grammar rules, all the verb patterns (20 patterns) are studied and rules are formed to convert English sentence into ISL sentence. The parsed sentence is the input to this module where the noun phrase and the prepositional phrase are freezed but if there is any verb phrase present in the sentence, it is checked recursively because the verb phrase may further be composed of noun phrase, prepositional phrase, verb phrase or even the sentence. Some of the rules of conversion are given in the table:

Verb Pattern	Rule	Input Sentence	Parsed Sentence	Output Sentence
verb + object	VP NP	go school	(VP (VB Go ) (NP (NN school ) ) )	school go
subject + verb	NP V	birds fly	(NP (NNS birds ) ) (VP (VBP fly ) )	birds fly
subject + verb + subject complement	NP V NP	his brother became a soldier	(NP (PRP\$ his ) (NN brother ) ) (VP (VBD became ) (NP (DT a ) (NN soldier ) ) )	his brother a soldier became
subject + verb + indirect object + direct object	NP V NP NP	i lent her my pen	(NP (FW i ) ) (VP (VBD lent ) (NP (PRP her ) ) (NP (PRP\$ my ) (NN pen ) ) )	i her my pen lent
subject + verb	subject + verb	show me your hands	(VP (VBP show ) (NP (PRP me ) ) ) (NP (PRP\$ your ) (NNS hands ) )	me your hands show
subject + verb + direct object + preposition + prepositional object	NP V NP PP	she made coffee for all of us	(NP (PRP She ) ) (VP (VBD made ) (NP (NN coffee ) ) (PP (IN for ) (NP (DT all ) ) (PP (IN of ) (NP (PRP us ) ) ) ) ) )	she coffee for all of us made
subject + verb + indirect object + direct object	V NP PP	show your hands to me	(VP (VB show ) (NP (PRP\$ your ) (NNS hands ) ) (PP (TO to ) (NP (PRP me ) ) ) )	your hands to me show
subject + verb + preposition prepositional object	NP V PP	we are waiting for suresh	(NP (PRP we ) ) (VP (VBP are ) (VP (VBG waiting ) (PP (IN for ) (NP (NN suresh ) ) ) ) )	we for suresh are waiting

**Table 3.2: Examples of Grammatical Reordering of Words of English Sentence**

In this module, step 1 is to modify SVO structure of English to SOV structure of ISL.

In next step, we look for interrogative sentences, the ones with W- questions.

## Handling WH-Interrogatives

WH-Interrogative markers (like who, what, when, and why) always occur at the end of the sentence.

**English:** "When is your birthday?"

**ISL:** "YOUR BIRTHDAY TIME + QUESTION".

This is done by parsing phrase structure tree of text and deleting node containing WH-interrogative words and appending this node as last child of the node.

## Eliminator

According to the rules of ISL, we never use am/is/are/was/were/ (linking verbs) or Do not use articles. (a, an, some, the). Such words are called **Stop Words**. These words are very common in source language but donot give any meaning and are not a part of dictionary of target language and thus should be eliminated. In this phase of module, we eliminate stop words out of reordered tokens.

## Stemming and Lemmatzation

In ISL, for any word we never use word-endings/suffixes or words ending with gerunds (-ing). Every word in ISL is present in its root form. In this phase of module, we convert every token into its root form by stemming/lemmatizing.

For grammatical reasons, documents in source language are going to use different forms of a word, such as *organize*, *organizes*, and *organizing*. Additionally, there are families of derivationally related words with similar meanings, such as *democracy*, *democratic*, and *democratization*. The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance:

am, are, is  $\Rightarrow$  be

car, cars, car's, cars'  $\Rightarrow$  car

The result of this mapping of text will be something like:

the boy's cars are different colors  $\Rightarrow$

the boy car be differ color

**Stemming** usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes.

**Lemmatization** usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the *lemma* . If confronted with the token *saw*, stemming might return just *s*, whereas lemmatization would attempt to return either *see* or *saw* depending on whether the use of the token was as a verb or a noun. The two may also differ in that stemming most commonly collapses derivationally related words, whereas lemmatization commonly only collapses the different inflectional forms of a lemma.

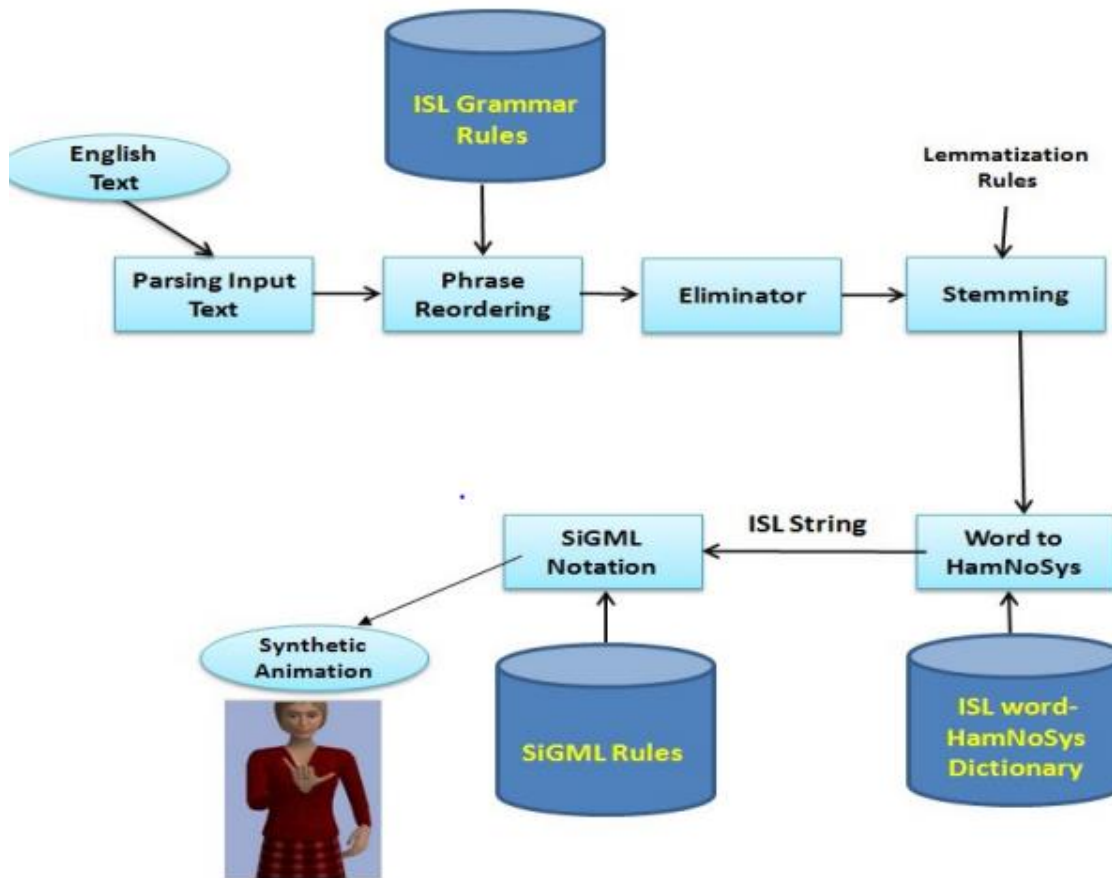
In our system, we have lemmatized every token and in order to improve accuracy we have also assigned parts-of-speech tagging to every word. This improves accuracy of word to root word conversion.

## Synonym Generation

As we have limited dictionary, all words present in source language may not be present in dictionary of ISL. Every token in the string can be one of the either:

1. Has SIGML file corresponding to word
2. Donot have SIGML file corresponding to it

In case 2, we use Wordnet to find synonyms of the word. For each possible synonym, we check for case 1, ie. Whether synonym of the word exists in current database or not and then that synonym replaces the word.



**Fig. 3.2 Workflow of system**

## 5. Graphics Generator

Input to this module is the ISL text string. In this module, Graphics generator map each token of text to a database. For every word present in database there is a corresponding file present.

e.g. for word bird , SIGML file is bird.sigml

So, every word is mapped to corresponding SIGML file. For the words which cannot be mapped we present them as concatenation of SIGML files of all alphabets.

e.g. Rahul ----->R.sigml + a.sigml + h.sigml +u.sigml + l.sigml

These SIGML files are given as input to avatar, which gives animated motion according to the SIGML file.

## Chapter 4 Implementation

### 4.1 Website

#### 4.1.1 Main Features

- Converts Speech to Sign Language.
- Uses Text as an intermediate.
- Tokenizes text using ISL rules
- Web based interface (no installs necessary)
- Client/Server Architecture.
- Machine Learning: Uses NLP tools.
- Fully responsive: Everything is responsive ready so need to worry about how this site will look on mobile, tablet, and desktop.
- Easy to use
- User-friendly

#### 4.1.2 Tools

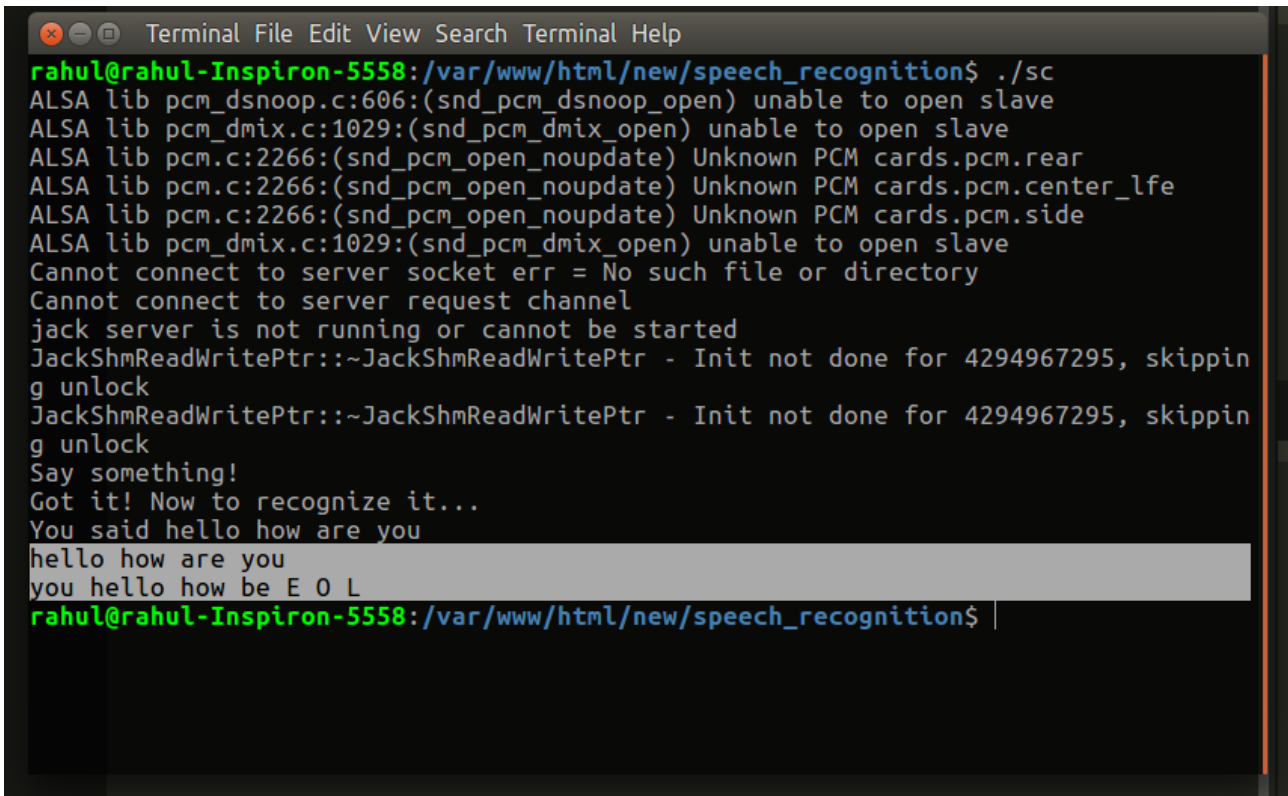
- **Front-end**  
(I) SigML URL App
- **Back-end**  
(I) OpenCV  
(II) phpMyAdmin

#### 4.1.3 Languages

- **Front-end**  
(I) HTML 5  
(II) CSS  
(III) SiGML
- **Back-end**  
(I) JavaScript  
(II) PHP  
(III) Python3

## 4.2 SCREENSHOTS

### 4.2.1 Speech Input



```
Terminal File Edit View Search Terminal Help
rahul@rahul-Inspiron-5558:/var/www/html/new/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said hello how are you
hello how are you
you hello how be E O L
rahul@rahul-Inspiron-5558:/var/www/html/new/speech_recognition$
```

**Fig 4.1 Mainpage**

- The web server is always listening for speech input which gets converted to text using Indian Sign Language Standards.
- The logo of the website is:



**Fig 4.2 Website's logo**

## 4.2.2 Output

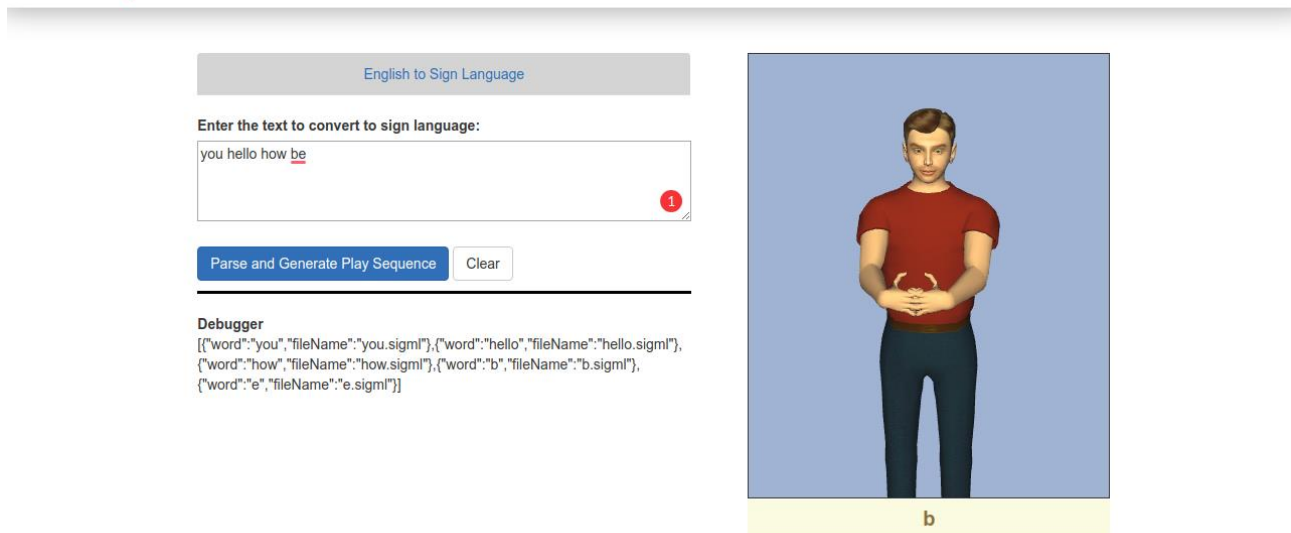


Fig 4.3 Login Portal

- ⑩ Gestures are displayed on web page using JASiGML URL App Player.

### Configuration File For Avatar:

```

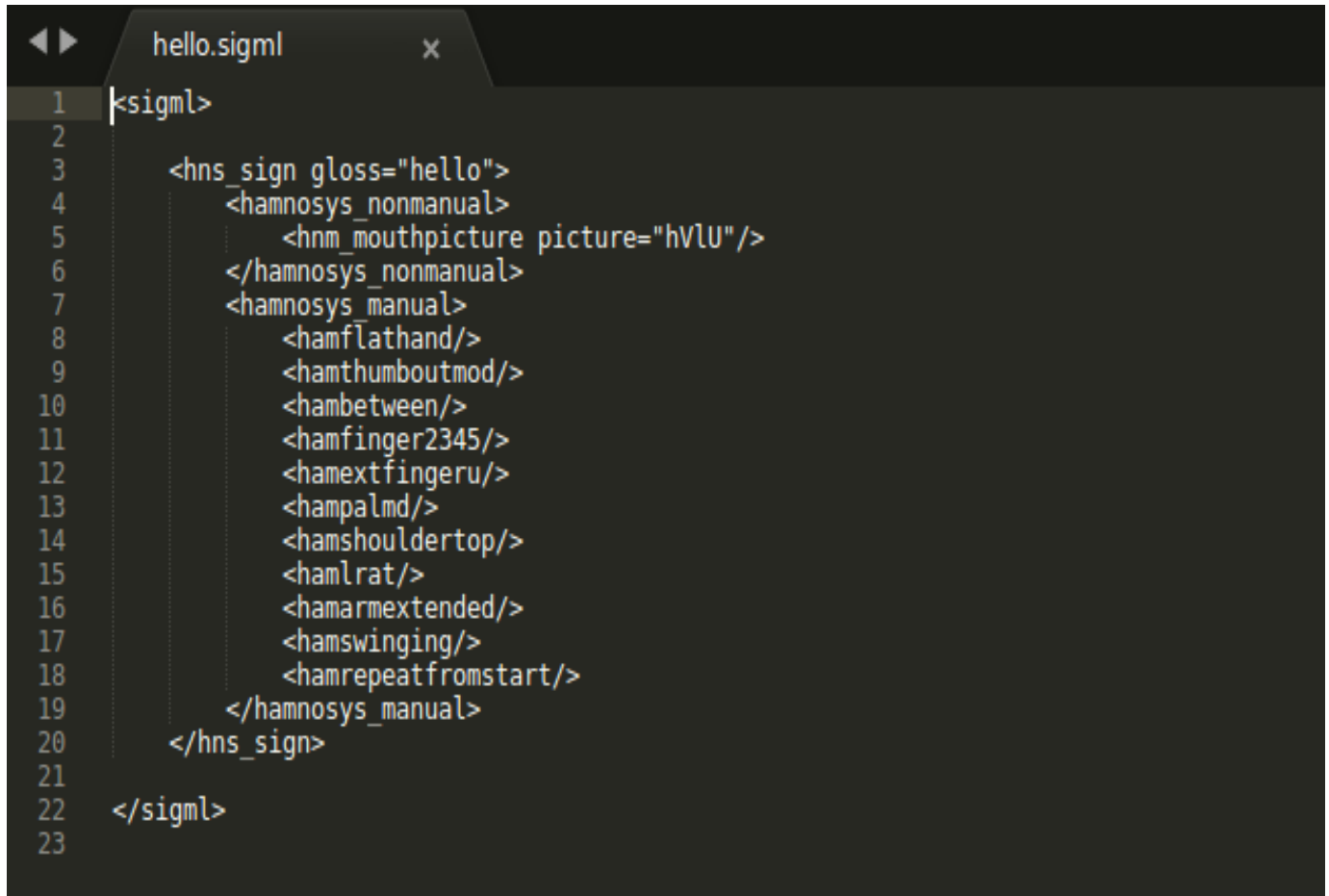
1  cwacfg.json
2  {
3    "description" : "==== CWASA installation configuration data =====",
4    "jasBase" : "http://vhg.cmp.uea.ac.uk/tech/jas/vhg2017/",
5    "jasVersionTag" : "vhg2017",
6
7    "sigmlBase" : "sigml",
8    "avJARBase" : "avatars",
9    "avJSONBase" : "avjson",
10   "useAvatarJARs" : true,
11
12   "animgenFPS" : 50,
13   "animgenServer" : "http://vhg.cmp.uea.ac.uk/cgi-bin/animgen/animgenserver.pl",
14
15   "avs" : [
16     "anna", "marc", "francoise"
17   ],
18
19   "avsfull" : [
20     "anna",
21     "beatrice", "candy", "darshan", "francoise", "genie", "luna",
22     "marc", "max", "monkey", "otis", "robotboy", "siggi"
23   ],
24
25   "avSettings" : [
26     {
27       "description": "Default if client configuration is missing or restricted",
28       "width": 384,
29       "height": 320,
30       "avList": "avs",
31       "initAv": "anna",
32       "initCamera": [ 0, 0.23, 3.24, 5, 18, 30, -1, -1 ],
33       "allowFrameSteps": true,
34       "initSiGMLURL": "iTakeMug.sigml",
35       "allowSiGMLText": true
36     }
37   ]
38 }
39

```

Fig.4.4 Screenshot of configuration



## Instructions For Avatar: SiGML File



```
1 <sigml>
2
3   <hns_sign gloss="hello">
4     <hamnosys_nonmanual>
5       <hnm_mouthpicture picture="hVlU"/>
6     </hamnosys_nonmanual>
7     <hamnosys_manual>
8       <hamflathand/>
9       <hamthumboutmod/>
10      <hambetween/>
11      <hamfinger2345/>
12      <hamextfingeru/>
13      <hampalmd/>
14      <hamshouldertop/>
15      <hamlrat/>
16      <hamarmextended/>
17      <hamswinging/>
18      <hamrepeatfromstart/>
19    </hamnosys_manual>
20  </hns_sign>
21
22 </sigml>
23
```

Fig.4.5 Screenshot of SIGML

## 4.3 UML DIAGRAMS

### USE CASE DIAGRAM

**Use case diagrams** are usually referred to as behaviour diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).

**ACTOR:** An Actor models a type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data), but which is external to the subject (i.e., in the sense that an instance of an actor is not a part of the instance of its corresponding subject). Actors may represent roles played by human users, external hardware, or other subjects

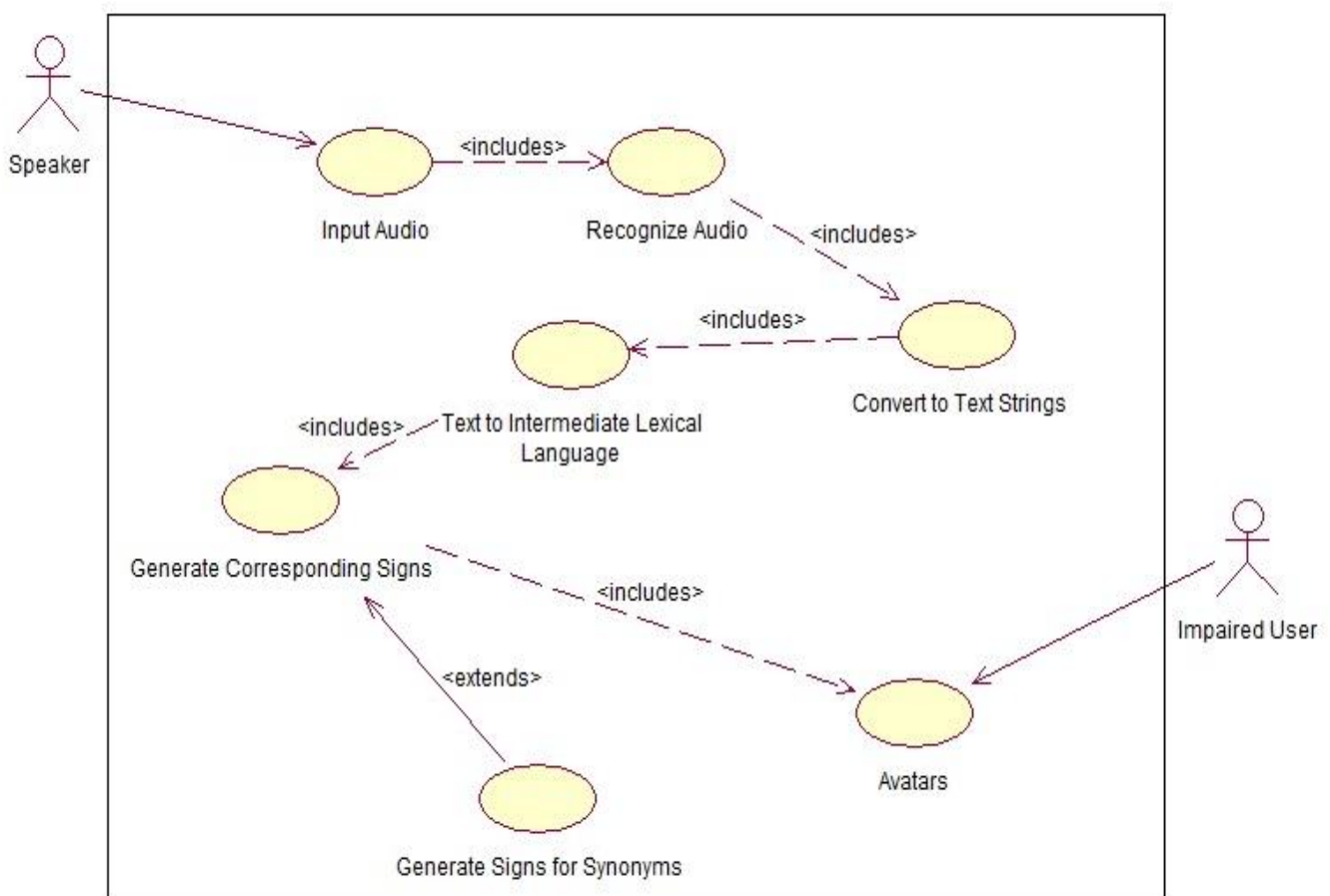
**ASSOCIATION:** An association specifies a semantic relationship that can occur between typed instances. It has at least two ends represented by properties, each of which is connected to the type of the end. More than one end of the association may have the same type.

**EXTENDS:** This relationship specifies that the behavior of a use case may be extended by the behavior of another (usually supplementary) use case. The extension takes place at one or more specific extension points defined in the extended use case.

**INCLUDE:** Include is a Directed Relationship between two use cases, implying that the behavior of the included use case is inserted into the behavior of the including use case. It is also a kind of NamedElement so that it can have a name in the context of its owning use case. The including use case may only depend on the result (value) of the included use case. This value is obtained as a result of the execution of the included use case.

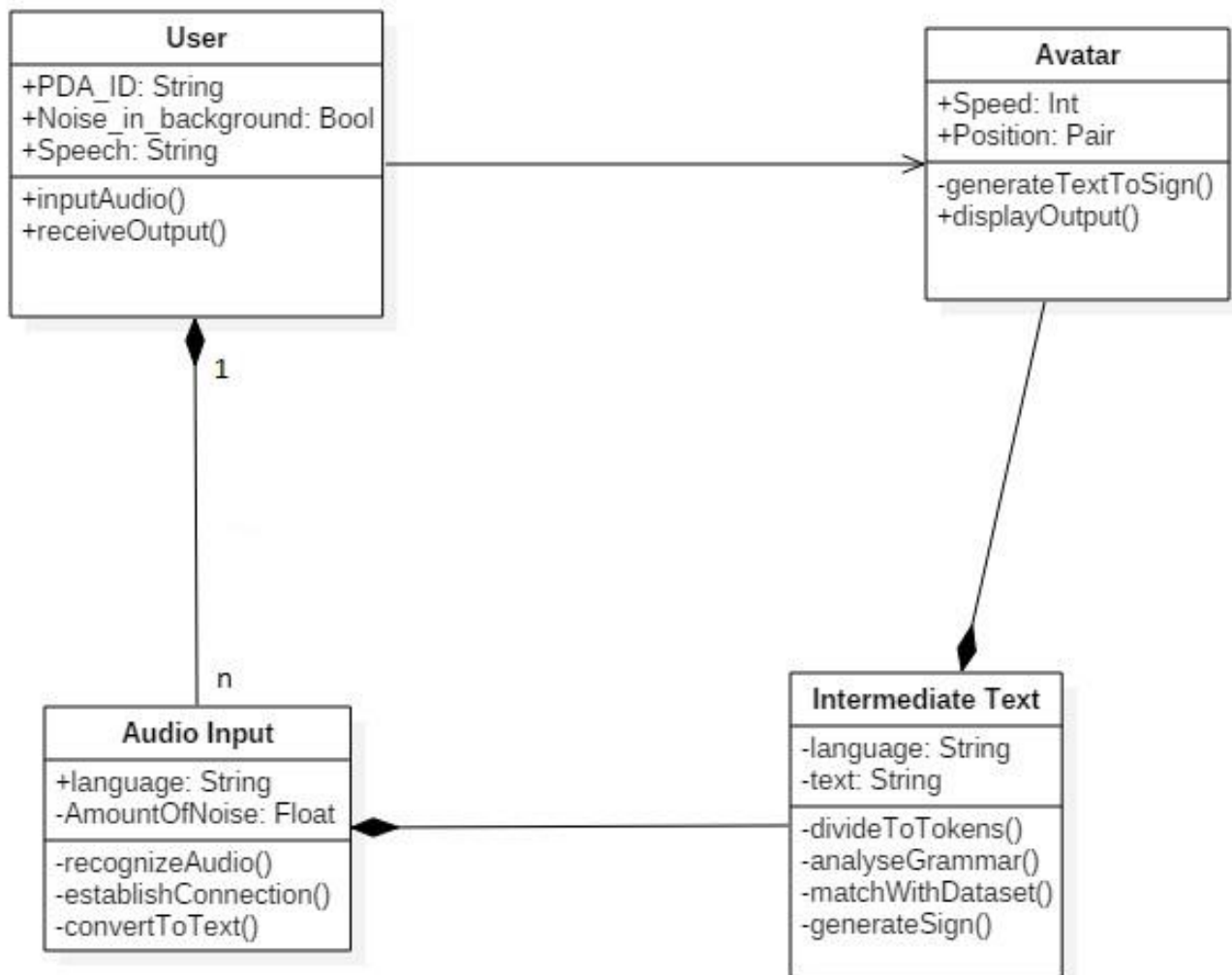
**SYSTEM BOUNDARY:** If a subject (or system boundary) is displayed, the use case ellipse is visually located inside the system boundary rectangle. Note that this does not necessarily mean that the subject classifier owns the contained use cases, but merely that the use case applies to that classifier.

**USE CASE:** A use case is the specification of a set of actions performed by a system, which yields an observable result that is, typically, of value for one or more actors or other stakeholders of the system.



**Fig.4.6 Use case diagram**

## CLASS DIAGRAM



**Fig.4.7 Class Diagram**

A **class diagram** in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

In most UML models these types include:

- a class
- an interface
- a data type

- a component.

### CLASS NAME:

The UML representation of a class is a rectangle containing three compartments stacked vertically, as shown in Figure 1. The top compartment shows the class's name. The middle compartment lists the class's attributes. The bottom compartment lists the class's operations. When drawing a class element on a class diagram, you must use the top compartment, and the bottom two compartments are optional.

### CLASS ATTRIBUTE LIST:

The attribute section of a class (the middle compartment) lists each of the class's attributes on a separate line. The attribute section is optional, but when used it contains each attribute of the class displayed in a list format.

### CLASS OPERATIONS LIST:

The class's operations are documented in the third (lowest) compartment of the class diagram's rectangle, which again is optional. Like the attributes, the operations of a class are displayed in a list format, with each operation on its own line. Operations are documented using the following notation:

name(parameter list) : type of value returned

## ACTIVITY DIAGRAM

Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

● **The start symbol** represents the beginning of a process or workflow in an activity diagram. It can be used by itself or with a note symbol that explains the starting point.



**The activity symbol** is the main component of an activity diagram. These shapes indicate the activities that make up a modelled process.



**The connector symbol** is represented by arrowed lines that show the directional flow, or control flow, of the activity. An incoming arrow starts a step of an activity; once the step is completed, the flow continues with the outgoing arrow.



**The join symbol, or synchronization bar,** is a thick vertical or horizontal line. It combines two concurrent activities and re-introduces them to a flow where only one activity occurs at a time.



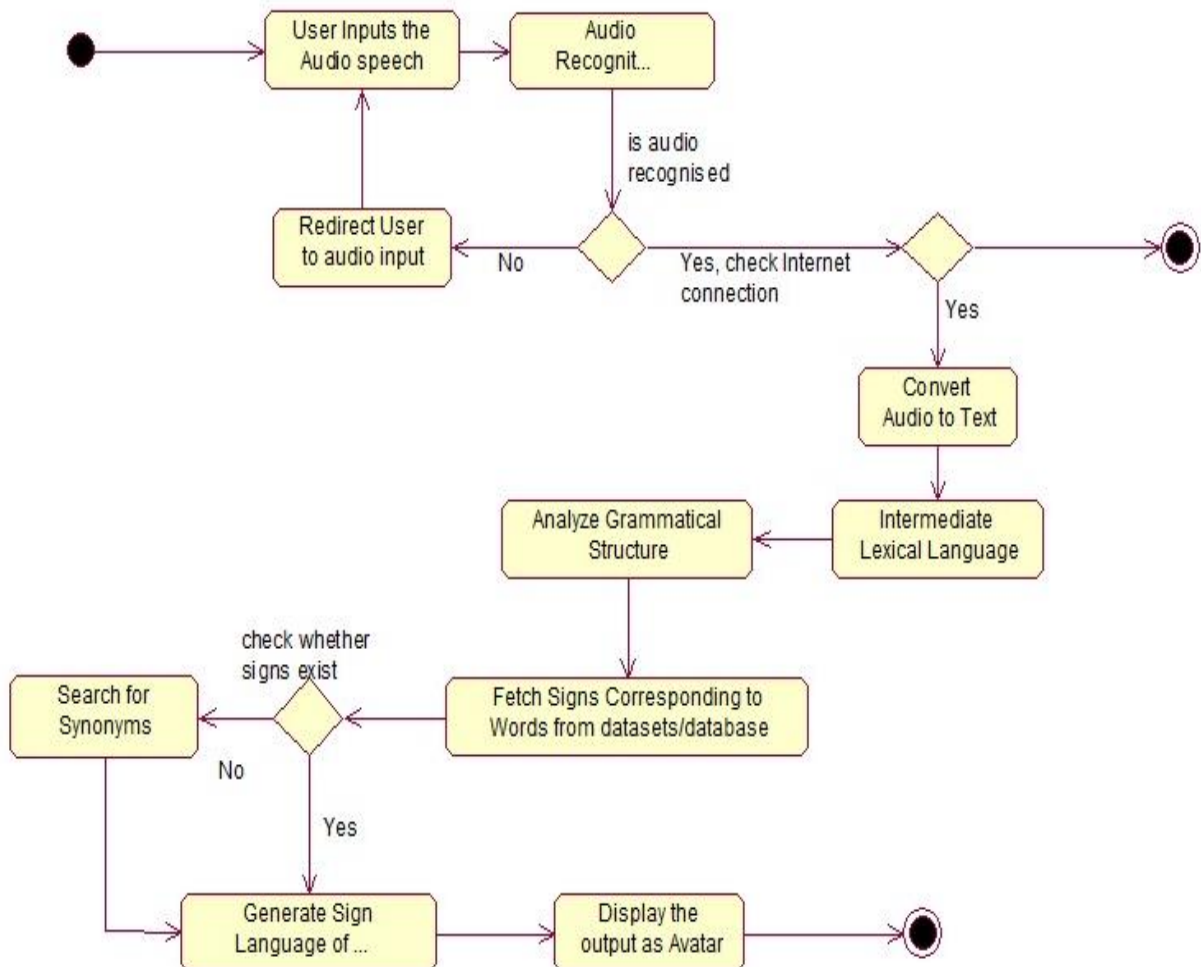
**A fork** is symbolized with multiple arrowed lines from a join. It splits a single activity flow into two concurrent activities.



**The decision symbol** is a diamond shape; it represents the branching or merging of various flows with the symbol acting as a frame or container.



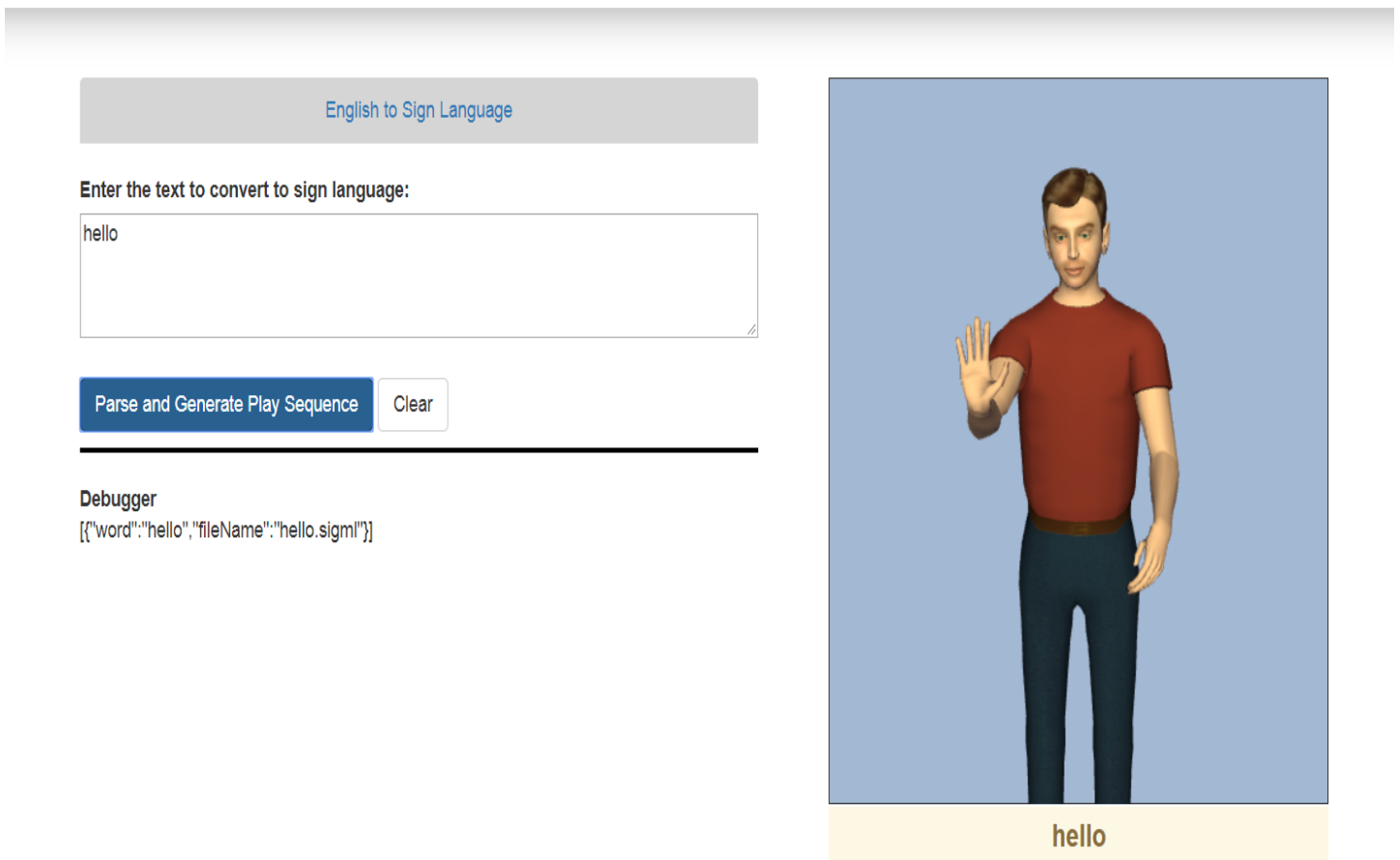
**The end symbol** represents the completion of a process or workflow.



**Fig.4.8 Activity Diagram**

## Chapter 5 RESULTS AND DISCUSSIONS

Web based portal to convert audio to sign language.



**Fig.5.1 Screenshot of website**

### Accuracy

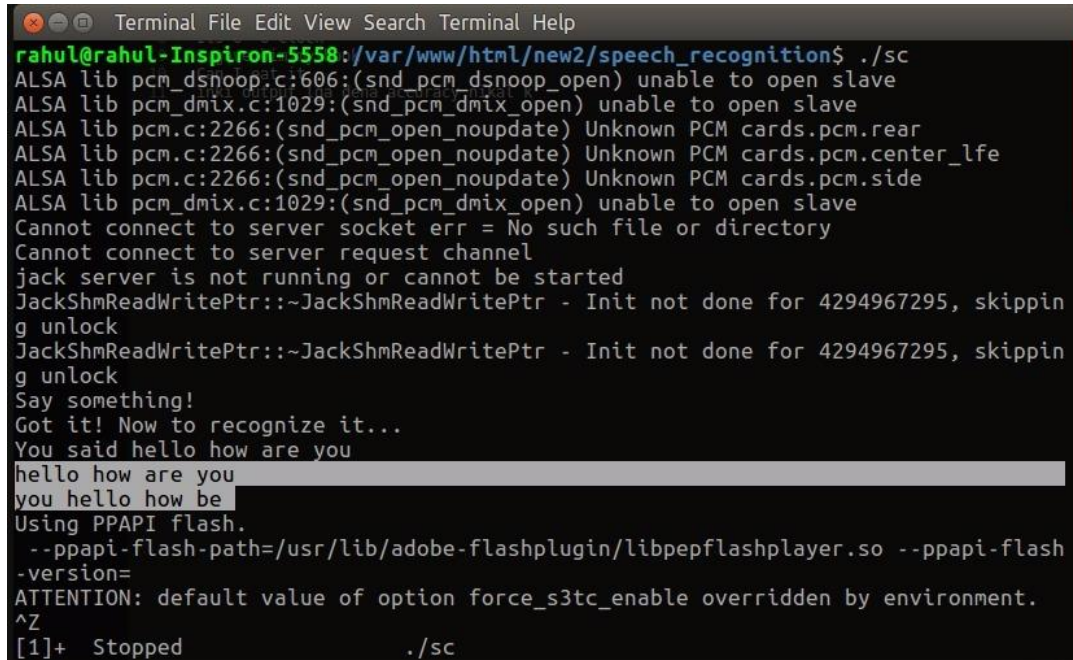
In order to check the accuracy of the system, we gave 10 sample test cases and system translated them into ISL grammar language. The results were checked according to standard.

Following are the sample test cases:



1. Hello how are you?

Output: you hello how be



```

Terminal File Edit View Search Terminal Help
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said hello how are you
hello how are you
you hello how be
Using PPAPI flash.
--ppapi-flash-path=/usr/lib/adobe-flashplugin/libpepflashplayer.so --ppapi-flash-version=
ATTENTION: default value of option force_s3tc_enable overridden by environment.
^Z
[1]+  Stopped                  ./sc

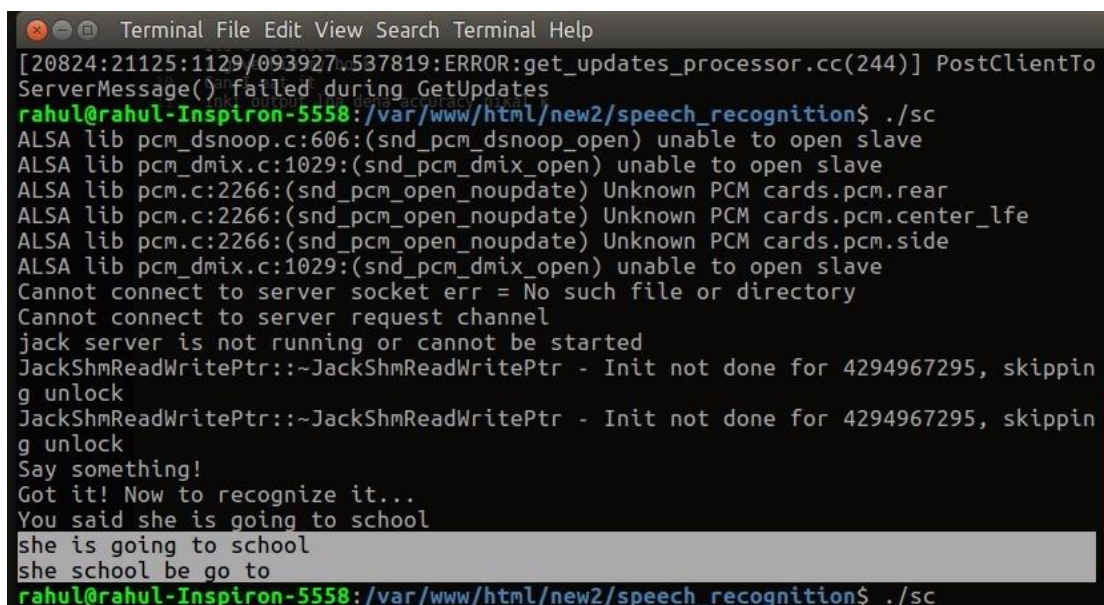
```

**Fig.5. Output 1**

Result: passed

2. She is going to school

Output: she school be go to



```

Terminal File Edit View Search Terminal Help
[20824:21125:1129/093927.537819:ERROR:get_updates_processor.cc(244)] PostClientTo
ServerMessage() failed during GetUpdates
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said she is going to school
she is going to school
she school be go to
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc

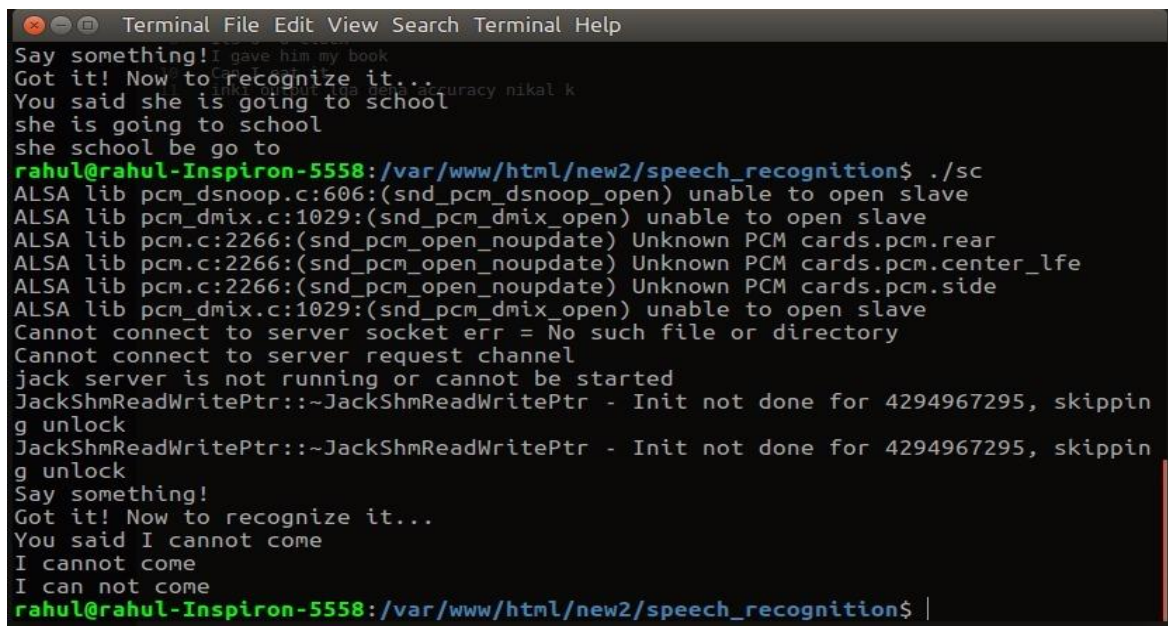
```

**Fig.5.3 Output 2**

Result: passed

3. I cannot come

Output: I can not come



```

Terminal File Edit View Search Terminal Help
Say something! I gave him my book
Got it! Now to recognize it...
You said she is going to school
she is going to school
she school be go to
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said I cannot come
I cannot come
I can not come
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$

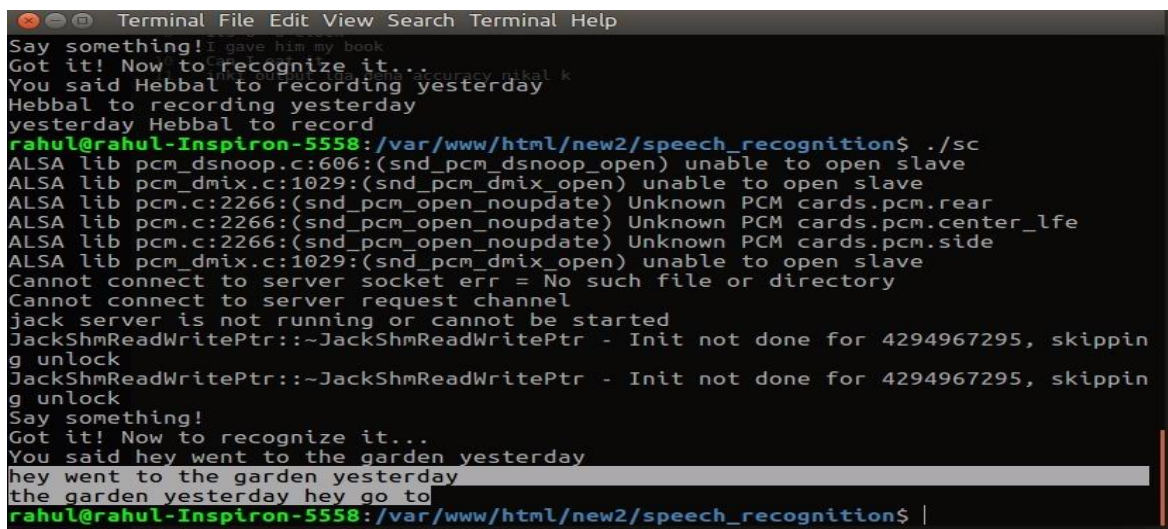
```

Fig.5.4. Output 3

Result: Passed

4. He went to garden yesterday

Output: the garden yesterday he go to



```

Terminal File Edit View Search Terminal Help
Say something! I gave him my book
Got it! Now to recognize it...
You said Hebbal to recording yesterday
Hebbal to recording yesterday
yesterday Hebbal to record
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said hey went to the garden yesterday
hey went to the garden yesterday
the garden yesterday hey go to
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$

```

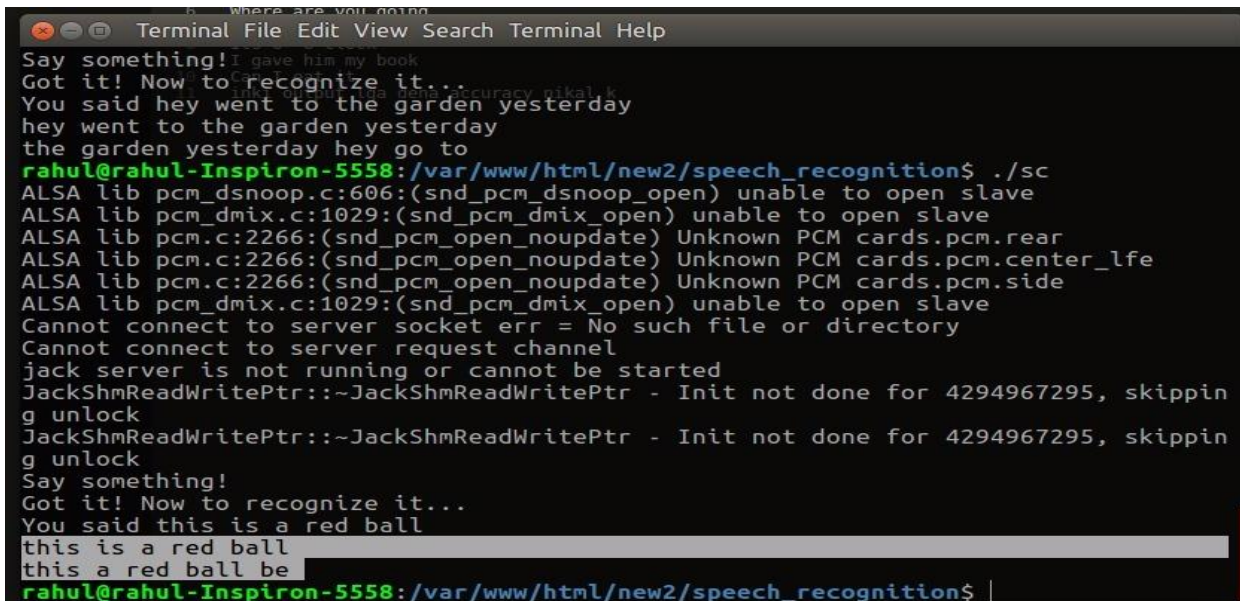
Fig.5.5 Output 4



Result: Passed

5. This is a red ball

Output: This red ball be



```

Terminal File Edit View Search Terminal Help
Say something! I gave him my book
Got it! Now to recognize it...
You said hey went to the garden yesterday
hey went to the garden yesterday
the garden yesterday hey go to
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said this is a red ball
this is a red ball
this a red ball be
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ |

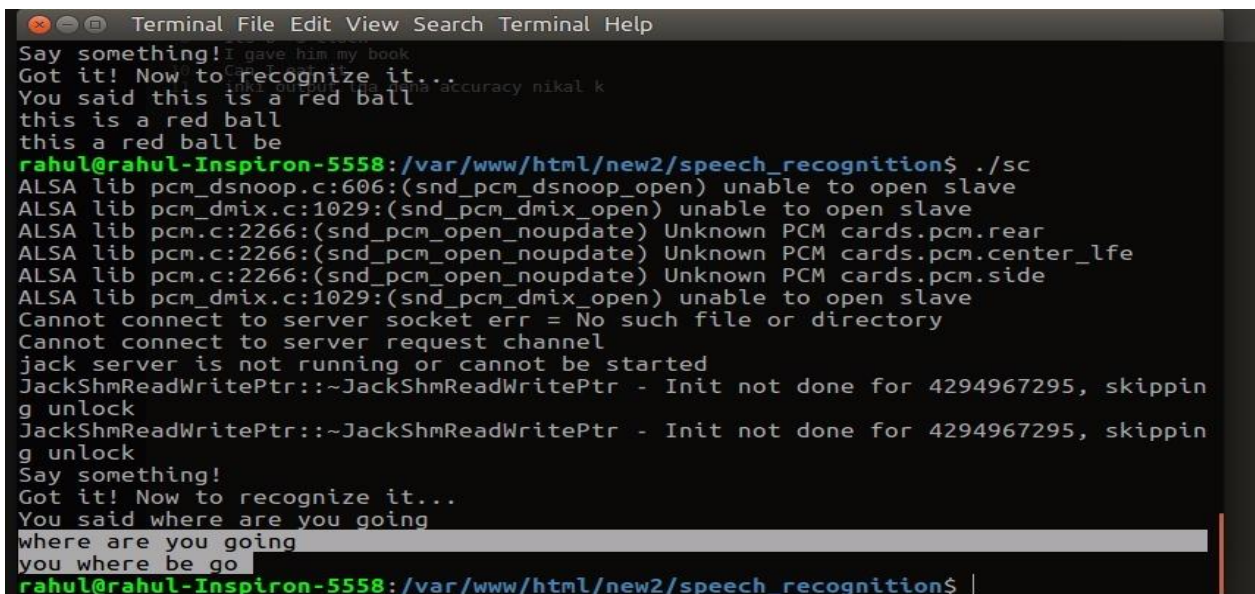
```

Fig.5.6 Output 5

Result: Passed

6. Where are you going?

Output: you where be go



```

Terminal File Edit View Search Terminal Help
Say something! I gave him my book
Got it! Now to recognize it...
You said this is a red ball
this is a red ball
this a red ball be
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said where are you going
where are you going
you where be go
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ |

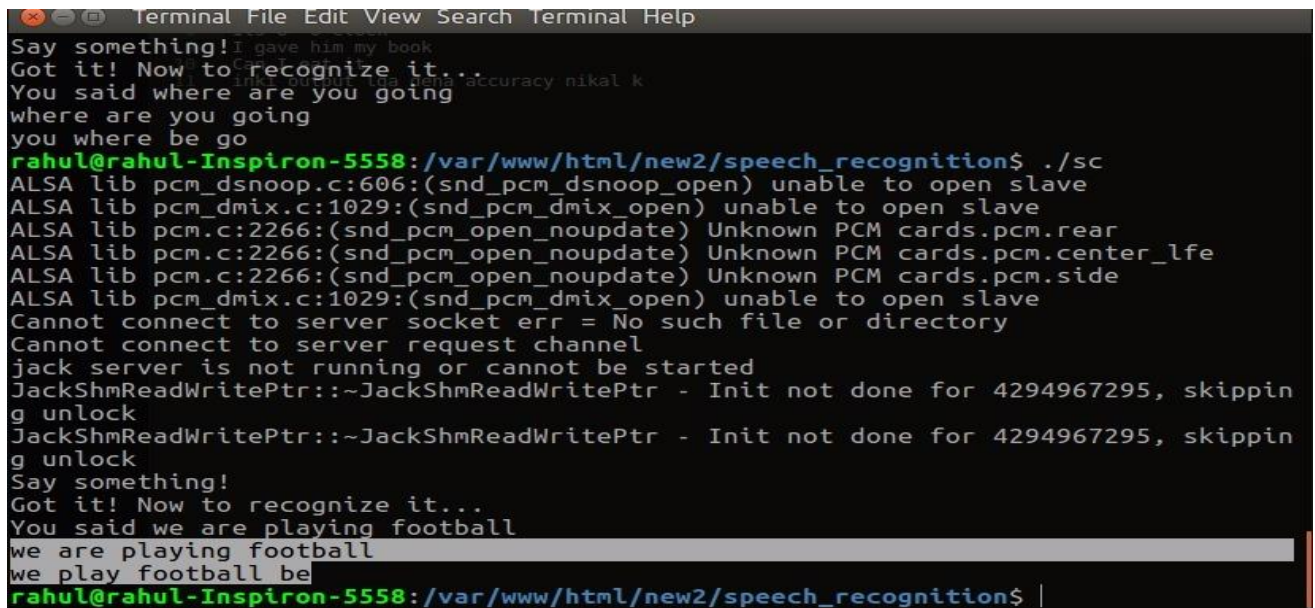
```

Fig.5.7 Output 6

Result: Passed

7. We are playing football

Output: we play football be



```

Terminal File Edit View Search Terminal Help
Say something! I gave him my book
Got it! Now to recognize it...
You said where are you going
where are you going
you where be go
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said we are playing football
we are playing football
we play football be
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ |

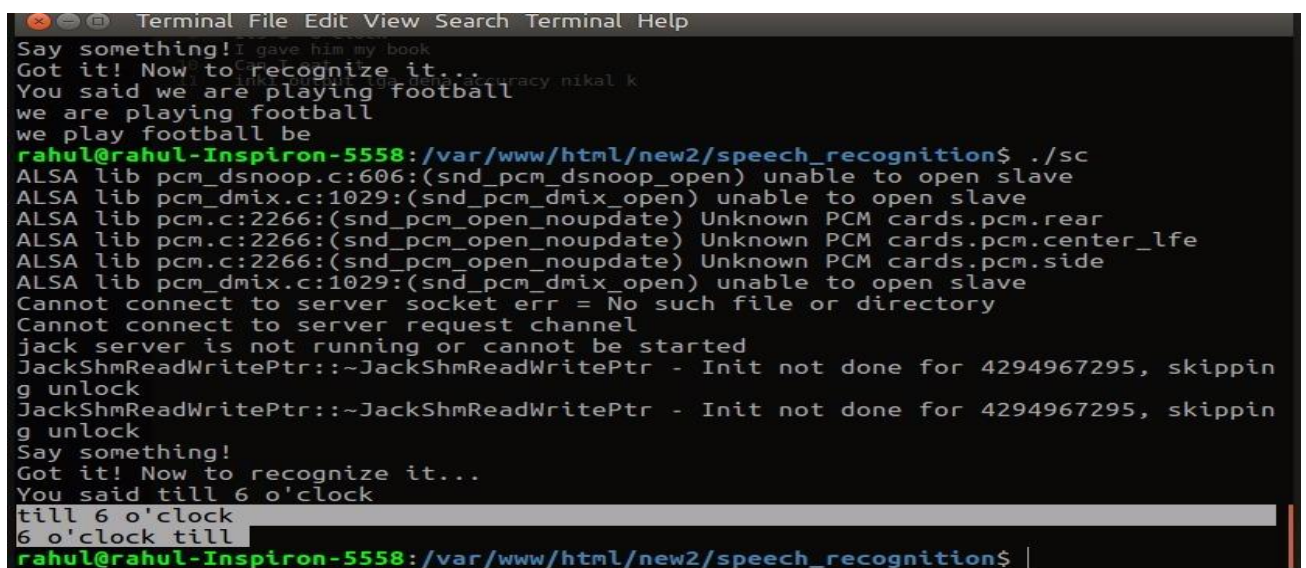
```

Fig.5.8 Output 7

Result: Passed

8. Till 6' o clock

Output: 6 o'clock till



```

Terminal File Edit View Search Terminal Help
Say something! I gave him my book
Got it! Now to recognize it...
You said we are playing football
we are playing football
we play football be
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said till 6 o'clock
till 6 o'clock
6 o'clock till
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ |

```

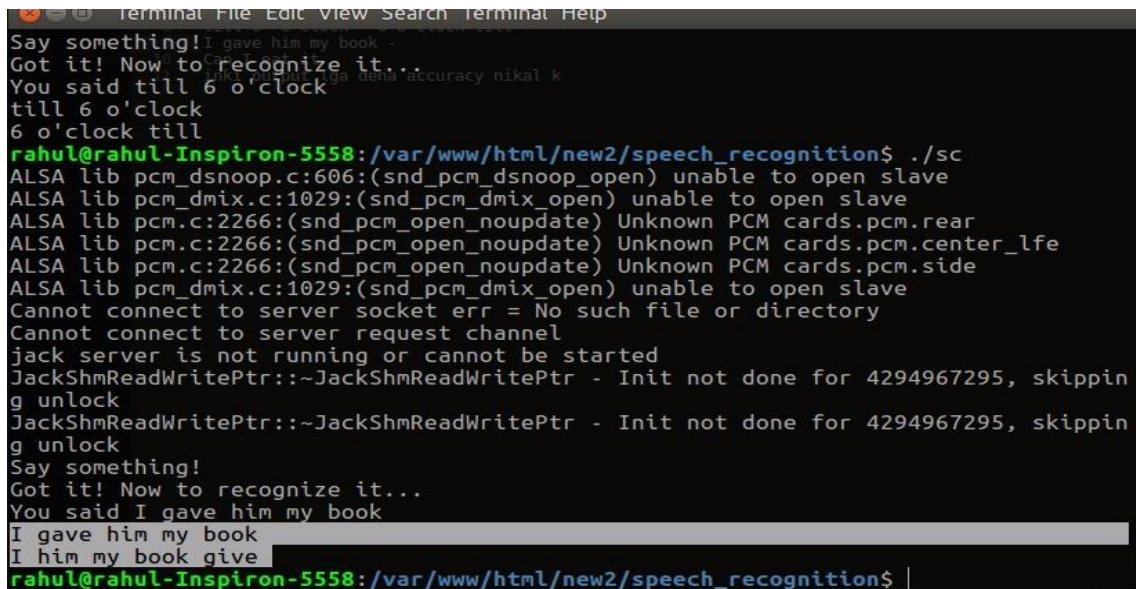
Fig.5.9 Output 8



Result: Not passed

9. I gave him a book

Output: I him my book give



```

Terminal File Edit View Search Terminal Help
Say something! I gave him my book -
Got it! Now to recognize it...
You said till 6 o'clock
till 6 o'clock
6 o'clock till
6 o'clock till
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said I gave him my book
I gave him my book
I him my book give
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ |

```

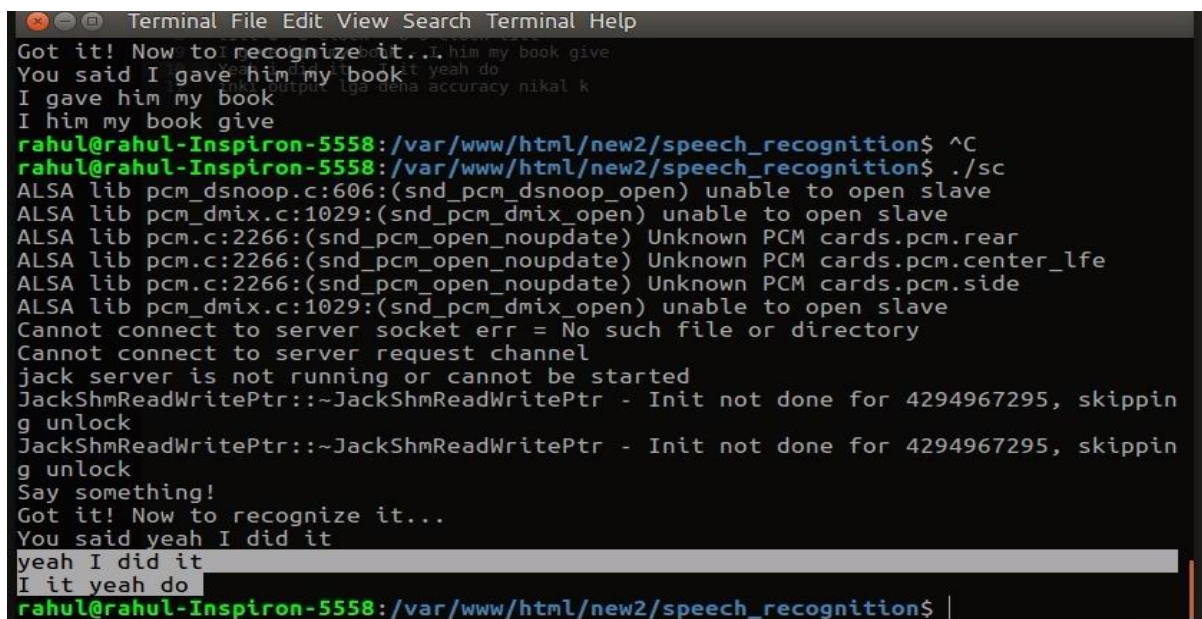
Fig.5.10 Output 9

Result: Passed

10. Yeah I did it.

Output: I it yeah do

Fig.5.11 Output 10



```

Terminal File Edit View Search Terminal Help
Got it! Now to recognize it... I him my book give
You said I gave him my book
I gave him my book
I him my book give
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ^C
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ ./sc
ALSA lib pcm_dsnoop.c:606:(snd_pcm_dsnoop_open) unable to open slave
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.rear
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.center_lfe
ALSA lib pcm.c:2266:(snd_pcm_open_noupdate) Unknown PCM cards.pcm.side
ALSA lib pcm_dmix.c:1029:(snd_pcm_dmix_open) unable to open slave
Cannot connect to server socket err = No such file or directory
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for 4294967295, skipping unlock
Say something!
Got it! Now to recognize it...
You said yeah I did it
yeah I did it
I it yeah do
rahul@rahul-Inspiron-5558:/var/www/html/new2/speech_recognition$ |

```



Result: Not Passed

Out of 10 test cases,

Test cases which passed the test = 8

Test cases which did not pass the test = 2

Therefore,

Accuracy =  $8/10 * 100$

= 80%

## Chapter 6 CONCLUSION AND FUTURE SCOPE

In this article, an efficient English audio to ISL sign language has been proposed. The sign languages like BSL, ASL have got the particular grammar which makes it feasible for the rule based systems and the syntax and semantic analysis can be performed to get the appropriate translation. In contrast for Indian Sign Language there is no particular grammatical rule which makes it difficult for the syntax and semantic analysis as there are no rules to compare the English text with. Thus appropriate translation of the English text is not feasible. In ISL, facial expressions denote negative and interrogative sentences. While ISL animation for the verb clause is played, the expressions change to denote the negation and interrogation in the sentence. This feature has not been yet accomplished by the system. The directionality and discourse are handled minimally. Further, ISL for phrases has not been included.

In the next stage of our work, we would like to handle non-manual components for the sentence as a whole, ISL for phrases will be included in dictionary. Also, directionality and discourse will be implemented. Since, a mobile based version of the application will increase the reach to more people so in next phase we shall be introducing an android based application for the same. Further, the system can be integrated with hand gesture recognition system using computer vision for establishing 2-way communication system.

## REFERENCES

1. INGIT: Limited Domain Formulaic Translation from Hindi strings to Indian Sign Language, Indian Institute of Technology, Kanpur
2. International Journal of emerging Technology in Computer Science and Electronics (IJETCSE) ISSN: 0976-1353 Volume 21 Issue 4, April 2016.
3. HamNoSys to SiGML Conversion System for Sign Language Automation, Multi conference on Information Processing.
4. Domain Bounded English to Indian Sign Language Translation Model, CSE, Sharda University, Noida.
5. Translation of Hindi Text to ISL and extension of ISL Dictionary with WordNet, Thapar University, Patiala.
6. <http://nlp.stanford.edu/software/stanford-ner-2015-12-09.zip> (nlp.stanford.edu)
7. <http://www.aclweb.org/anthology/W16-6319>
8. <http://www.icommunicator.com/productinfo/index.html>
9. <https://play.google.com/store/apps/details?id=com.Proactiva.ProDeafMoveI&hl=en>