A PROPOSED ENGLISH TO AMERICAN SIGN LANGUAGE TRANSLATION SYSTEM

Asmaa M. Hamandi, Alia Karim Abdul Hassan, Hala Bahjat

Computer Science Dept., University of Technology, Baghdad, Iraq.

ABSTRACT

A sign language (SL) is a visual language which uses manual communication and body language to convey meaning instead of acoustically conveyed sound patterns. SL is the communication mode among Deaf people. Different Sign languages according to different communities. One of the most widely used sign language is the American Sign Language (ASL). Translation between any pair of languages need many requirements, such as parallel dictionary, grammar rules, displaying tool. The proposed system takes input English text and produces ASL by passing into stages of processing. In order to build automatic translation system, a large dictionary has been built that contains ASL word with its corresponding movie. In order to evaluate the proposed grammar, English _ ASl corpus has been built. Good results have been obtained by evaluating the main stage of the system according to accuracy and time. Using Python programming language has the main contribution in producing good results. It provides the using of NLTK in addition to provide reducing time consuming.

Keywords: *Sign Language; Machine Translation; ASL; NLP toolkit; NLTK*

1. INTRODUCTION

Machine Translation (MT): is the use of computers to automate some or all of the process of translating from one language to another. Translation, in its full generality, is a difficult and intensely human endeavor, as rich as any other area of human creativity [Dan, 07]. The proposed MT system that translates English text into ASL. The system is fully automatic. The proposed system is considered to be bilingual unidirectional system since the translation is between the English language and the ASL, in one direction from English to ASL [Jon, 94].

ASL is a visual language expressed through hand gestures and facial expressions. It is a language used by many Deaf, hearing-impaired and hearing people in North American **[Hue, 05]**, [Cok, 94]. ASL has its own grammar, sentence construction, style, and regional variations and other characteristics that define any language**[Mar, 09]**, [Ric, 99]. ASL, as any Sign Language, has a form of fingerspell. Fingerspelling is the process of spelling out words by using signs that correspond to the letters of the word [Asm, 15]. In many ways finger spelling serves as a bridge between the sign language and the oral language that surrounds it **[Eli, 03] ,** [Wil, 11].

Any translation system is needing to apply Natural Language Processing (NLP). For NLP purposes, the proposed system used NLTK, it is a toolkit provides a built in and easy to use NLP

operations such as: tokenization, stemming, tagging, parsing, and semantic reasoning. That provides a trusted NLP with good results [NLTK, 01].

2. THE PROPOSED SYSTEM OVERVIEW

The proposed system consists of three stages. Each stage consists of sub stages and has its own importance. Figure (1) describes these three stages. Through these stages a translation from English input text into ASL has been performed:

- NLP stage: in this stage, POS tagging for the input English text has been produced in order to be passed to the next stage.
- Translation stage: in this stage an ASL sentence has been produced according the Proposed ASL Grammar that translate an English sentence with its POS tags that have been passed from the first stage. This stage produces an ASL sentence in order to be passed to the next stage.
- Matching stage: in this stage, a list of videos that corresponding to the ASL sentence ; has been passed from the second stage; has been produced. This task has been performed by searching the large dictionary that should be constructed previously. If the word has not been found in the dictionary, an image of fingerspell to the word has been produced.

The three stage are working in sequence such that the output of the previous stage will be used as input to the next stage, and then producing the output.

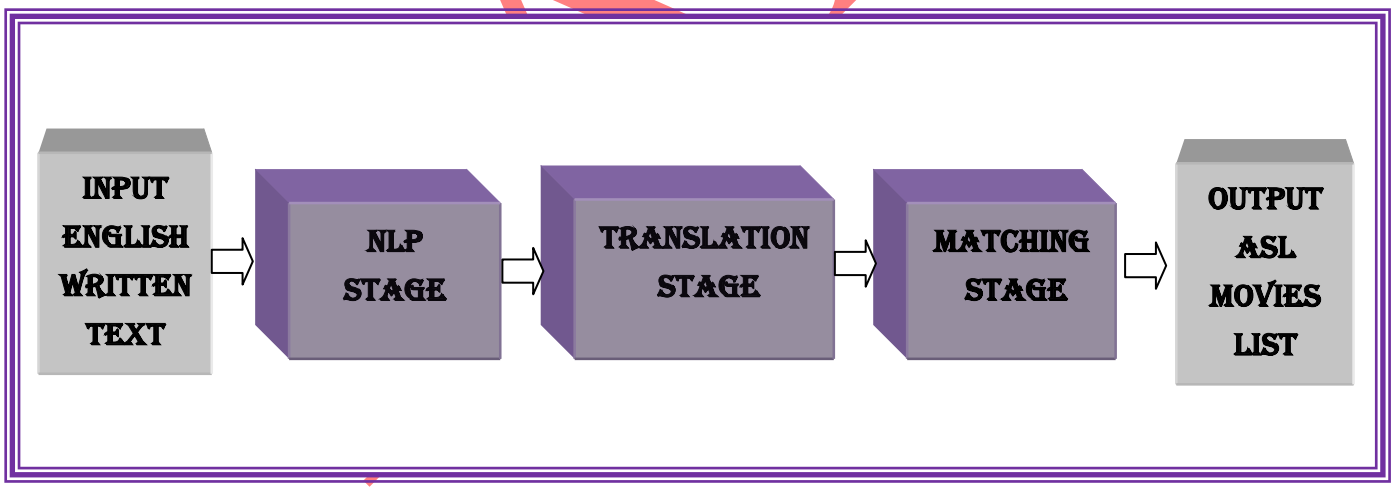

Figure (1): Block Diagram for the Online Mode of The System

3. PROPOSED SYSTEM ALGORITHMS

In this section, a description for each stage in details, in addition to the preprocessing of the system that should be prepared before applying the system.

3.1. System Preprocessing

The preprocessing composed of two important stages, that are the building of the dictionary and the building of the requirements that are necessary to implement the ASL

grammar. The first one: is building a large dictionary of 4677 word has been implemented. In order to represent each sign language word with its corresponding movie. **The second one** is the construction of ASL Grammar requirements. These requirements are specified according to the ASL grammar rules that must be applied in the second stage of the system, the translation stage. These requirements are as followed: ***Delete items, Time items, Continuous items and Negation items***.

3.2. NLP stage

After initialization (reading the input English text) the first stage of the system is the NLP stage, that takes the input English text as an input and produces the Part-Of-Speech (POS) tagging as an output for this stage according to several sub steps. Algorithm (1) represents the NLP stage main steps. In order to perform this task, an installation of the NLTK required, for getting a trust NLP tasks. The produced POS tags is according to the default tagger of NLTK that used Penn Treebank Tag Set that depends on max entropy approach.

Algorithm 1 (NLP stage: getting the POS tagged from English text)

input: English Text

output: POS tagged

{

calling NLTK toolkit;

Step 1: Get the input paragraph or sentence

str = readfile(sentence);

Step 2: Remove the punctuations from the input text

str1= remove-punctuations(str);

Step 3: Convert the input sentence into tokens

words= tokenization(str1);

Step 4: Produce the POS tags

tagged= POS (words);

Step 5: Rreturn tagged

}

3.3. Translation Stage

It is the second stage of the proposed system. After the first stage of the processing, the NLP stage, the second stage is translation stage, that takes the output of the first stage, the POS tagging, as an input and produces the ASL sentence as an output for this stage according to several sub steps as will be illustrated in the algorithm(2). In order to perform this task, several ASL grammar rules have been proposed to produce the ASL sentence as an output according to the input POS for the English text.

After reading the sentence with its POS tags saving them into two individual lists, the words have been normalized. The normalization process has been used is Capitalization (changing each letter of each word into its Uppercase form). After capitalization the grammar rules have been applied in order to produce the ASL sentence from the English sentence.

Algorithm .2 (Translation stage: translating POS tagged English sentence into ASL sentence)

input: English sentence , POS tagged

output: ASL sentence

{

Step 1: Get the input sentence with its POS tags

str = readfile(sentence, POS tags);

Step 2: Save the words of the sentence into words list

words= words-split(str);

Step 3: Save the tags that corresponds to the words list in parallel into tags list

tags= POS tags-split (str);

Step 4: Normalization for the words list, saving it into norm-words list

norm-words = capitalization(words);

Step 5: Applying the ASL Grammar Algorithm in order to produce the ASL sentence

ASL sentence= Grammar (norm-words, ASL grammar rules);

Step 6: Return ASL sentence

}

Each of these steps has its own importance. The importance of the splitting is that the using of each word with its corresponding POS tag through using two individual lists working in parallel all the processing. The importance of the Capitalization is to provide normalization that is very useful in searching and matching.

The last step in the algorithm(3) is applying the ASL grammar rules according to the rules for ASL, some items must be deleted and other items must be reordered in the sentence, another items must be stayed at the end of the sentence and others must be brought to the beginning of the sentence. Algorithm (3) describes applying the ASL grammar rules.

The proposed system has used the ASL grammar specifications for building (IF-Then-Grammar) Rules that converting the English sentence into ASL sentence. It is worth mentioning that, there no exists for such rules to be available for free use.

Thus, the proposed system allowed this proposed rule grammar for free use in scientific researches and available within reach to whom wants to use it, such as Deaf people and their family or friends or teachers or others in their community.

3.4. Proposed ASL Grammar Rules

One of the most important system contribution is the proposed grammar rules that have been applied to the English sentence with its corresponding POS tags in order to produce the ASL sentence.

1. Searching for Delete Items

Searching through English sentence or its corresponding POS tags to find the items to be deleted according to the grammar rules. When the system found one or more of the these items, the word and its corresponding tag must be deleted from the POS tagged as well as the word lists in order to build the new ASL sentence.

Algorithm 3 (Grammar sub-stage: applying ASL grammar to obtain the ASL sentence)

input: Norm-Words , POS tags

output: ASL sentence

{

Step 1: Searching the normalized words list and the POS tags list to find the items must be deleted in order to delete them

words= search-process(delete-items);

POS tags= search-process(delete-items);

Step 2: Searching the normalized words list to find the time items that must be brought to the beginning of the list

words= search-process(time-items);

Step 3: Searching the normalized words list and the POS tags list to find the items that refer to past tense in order to apply a past time flag at the beginning of the list

words= search-process(past-tense-items);

POS tags= search-process(past-tense-items);

Step 4: Searching the normalized words list and the POS tags list to find the items that refer to continuous tense in order to duplicate the verb in the list

words= search-process(continuous-items);

Step 5: Searching the normalized words list to find the items that refer to negation in order to apply a negation flag at the end of the list

words= search-process(negation-items);

Step 6: Searching the POS tags list to find the items that refer to adjectives and nouns in order to replace them in both lists

words= search-process(adjectives-and-nouns);

POS tags= search-process(adjectives-and-nouns);

Step 7: Searching the POS tags list to find the items that refer to adverbs and verbs in order to replace them in both lists

words= search-process(adverbs-and-verbs);

POS tags= search-process(adverbs-and-verbs);

Continue with Algorithm 3

Step 8: Searching the normalized words list to find dots in order to be deleted

words= search-process(dots-items);

Step 9: Now the produced list is representing the ASL sentence

ASL sentence= words ;

Step 10: Return ASL sentence

}

2. Searching for Time Items

Searching through English sentence to find the items that have been referred to the time, these items can be summarized as: "MORNING", "AFTERNOON", "EVENING", "NIGHT" , "YESTERDAY".

when the system found on or more of these items, they must be brought to the beginning of the new ASL sentence according to the ASL grammar rules.

3. Searching for Past-Tense Items

Searching through English sentence or its corresponding POS tags to find the items that have been referred to Past-Tense. When the system found one or more of the these items, the word "YESTERDAY" must be added in the beginning of the new ASL sentence if it wasn't already exist, while the each one of the above items has its own processing. For example the verb in the past tense will be returned to its base tense in the third stage of the system, while the auxiliary verbs will be deleted in other stage of the system. Each item of the sentence must be processed with its corresponding POS tag in order to build the new ASL sentence.

4. Searching for Continuous Items

Searching through English sentence or its corresponding POS tags to find the items that have been referred to Continuous, these items can be summarized as ("IS", "ARE", "AM", "WAS","WERE") as words in the English sentence and must be followed by a word that its corresponding POS tag is ("VBG": it refers to verb, present participle or gerund, for example: encrypting interrupting erasing).

When the system found one or more of the above items with the condition of must be followed by present participle verb, the following subtasks must be applied: a. Deleting the auxiliary verb ("IS", "ARE", "AM", "WAS","WERE") and its corresponding POS tag. b. Duplicating the verb (present participle or gerund).

These two subtasks have been applied according to ASL grammar in order to construct the new ASL sentence.

5. Searching for Negation Items

Searching through English sentence to find the items that have been referred to Negation, these items can be summarized as ("NOT" , "NONE" , "NEVER", "NOBODY" , "NOTHING" , "NOONE", "N'T").

When the system found one or more of these items, the following subtasks must be applied: a. Deleting these items for the sentence and its corresponding POS tag. b. Appending these items at the end of the sentence as well as its corresponding POS tag.

These two subtasks have been applied according to ASL grammar in order to construct the new ASL sentence.

6. Searching for Adjectives and Nouns Items

Searching through POS tags to find the items that have been referred to Adjectives. When the system found one or more of these items with the condition of must be followed by Noun, the system must exchange the position of the adjective and its noun, such that the noun must precedes the adjective in ASL sentence according to ASL grammar in order to construct the new ASL sentence. Switching the words in the word list must accompanied with switching to their corresponding POS tags in the tag list.

7. Searching for Adverbs and Verbs Items

Searching through POS tags to find the items that have been referred to Adverbs. When the system found one or more of these items with the condition of must be followed by Verb, the system must exchange the position of the adverb and its verb, such that the verb must precedes the adverb in ASL sentence according to ASL grammar in order to construct the new ASL sentence. Switching the words in the word list must accompanied with switching to their corresponding POS tags in the tag list.

8. Searching for DOTs

Searching through English sentence to find the Dots, since the dot is not considered within the punctuations that are removed from the sentence in the first stage of the system. Removing Dots from the word list must be accompanied with removing its corresponding POS tags in the tag list.

9. Return the ASL Sentence

After applying all the ASL Grammar Rules to the English sentence, the new constructed sentence is an ASL sentence that is ready to be displayed in any way that be suitable to the Deaf people and their community. Thus the system provides them with the ASL sentence and display it as movies, such that one movie for each ASL word after searching for the movie in a suitable

dictionary, that will be described in details in the next sections, as the third stage of the system "Matching Stage".

3.5. Matching stage

This stage is the third stage of the proposed system. After the second stage of the processing, the Translation stage, the third stage is matching stage, that takes the output of the second stage, the ASL sentence as an input and produces the list of movies to be displayed as an output for this stage as well as the whole system according to several sub steps. Algorithm (4) describes the Matching Stage main steps. In order to perform this task, an installation of the NLTK required as well as the large dictionary must be constructed.

Algorithm 4 (Matching stage: obtaining the list of ASL movies that corresponding to the ASL sentence)

```
input: ASL sentence
output: list-of-movies
{
Step 1: Calling NLTK toolkit;
Step 2: Get the ASL sentence
        str = readfile(ASL-sentence);
Step 3: Convert ASL sentence into list of words
        words= words-split(str);
Step 4: Process each word in the list
        for each word in words do
                {
                searching the dictionary to find the word corresponding movie
                        index-of-movie = search(dictionary);
                save the found index into list
                        list[word] = index-of-movie
                }
Step 5: Saving the list to be returned
        list-of-movies = list;
Step 6: Return list-of-movies
}
```

4. SYSTEM PRESENTATION

An experimental example with details of the proposed system. The proposed system consists of three processing stages have been applied in order to produce the series of ASL video movies that is corresponding to the input English sentence. Given an experimental example to describe the role of each stage in producing the output of the system.

4.1. The system GUI

The proposed system has been designed to be a useful tool for persons in Deaf community and around them that have a direct communication with Deaf people. Thus an understandable and easy to use interface is needed for providing simplicity and efficient use for the system. Figure (2) shows the first interface of the system when is being executed.

Figure (2): The System GUI

The second interface is produced when selecting (ASL Translator) choice. Figure (3) shows multiple options that the user may choose one of them. After writing the Input English text in the text box, the user may choose either Start choice to translate the sentence into series of movies, or choosing Fingerspell choice to translate the sentence into series of images that represents the ASL sentence as an ASL fingerspell. In both cases the user can repeat the show if needed by choosing Repeat button. Then the user can write a new input English sentence to translate it. Finally choosing close will close the second interface and return to the first one.

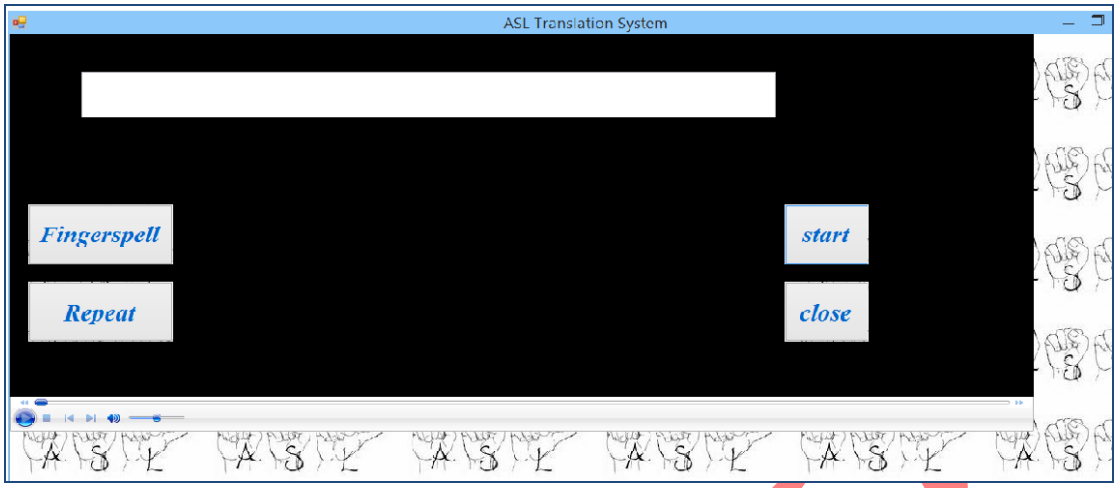

Figure (3): The Second System GUI

4.2. Detailed Example

Tracking an Input English sentence, applying the system processing steps, and then producing output. The system consists of three stages. The output of one stage used as an input to the next stage.

a. Getting Input English Sentence

English_ sentence :"I always depend on my wife, Nada, in keeping the house budget."

b. First stage is the NLP stage. Several sub stages have been applied in order to produce the POS tags for each word.

Stage-Input = :"I always depend on my wife, Nada, in keeping the house budget."

Stage-Output = [(I, PRP),(always, RB),(depend ,VBP),(on, IN),(my, PRP$),(wife, NN),(Nada, NNP),(in, IN),(keeping, VBG), (the, DT),(house, NN), (budget, NN)]

c. Second stage is the translation stage. Several sub stages have been applied in order to produce the ASL sentence that produced depending on input sentence and the POS tags that has been produced from the first stage.

Stage-Input = [(I, PRP),(always, RB),(depend ,VBP),(on, IN),(my, PRP$),(wife, NN),(Nada, NNP),(in, IN),(keeping, VBG), (the, DT),(house, NN), (budget, NN)]

Stage-Output= ['I', 'ALWAYS', 'DEPEND', 'MY', 'WIFE', 'NADA', 'KEEPING', 'HOUSE', 'BUDGET']

d. Third stage is the matching stage. Several sub stages have been applied in order to produce the series of movies that represent ASL sentence that has been produced as an output of the second stage, by searching the large dictionary constructed in offline mode. Figure (4) and figure (5) represent the show of one of the movies in the ASL sentence. If there is no movie corresponds to some ASL words, a fingerspell image produced to

represent the dictionary missing words. Figure (6) shows a fingerspell image for the word "Nada", since it is a proper noun that cannot be found in the dictionary.

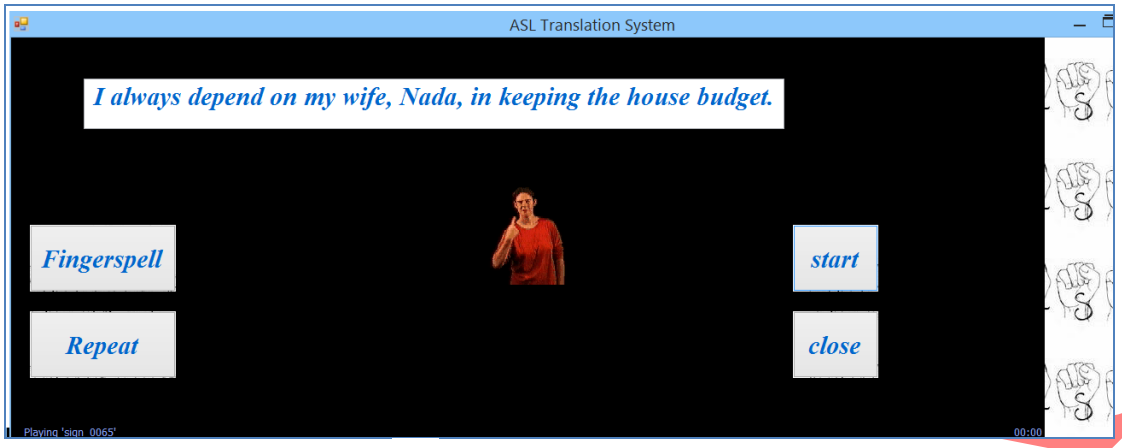

Figure (4): One of the ASL movies

Stage-Intput= ['I', 'ALWAYS', 'DEPEND', 'MY', 'WIFE', 'NADA', 'KEEPING', 'HOUSE', 'BUDGET']

Stage-Output= ['D:\program\Movies\sign_1276.mov',
'D:\program\Movies\sign_0496.mov',
'D:\program\Movies\sign_1392.mov',
'D:\program\Movies\sign_2667.mov',
'D:\program\data\fingerspell1.png',
'D:\program\Movies\sign_1066.mov',
'D:\program\Movies\sign_0920.mov',
'D:\program\Movies\sign_0294.mov']

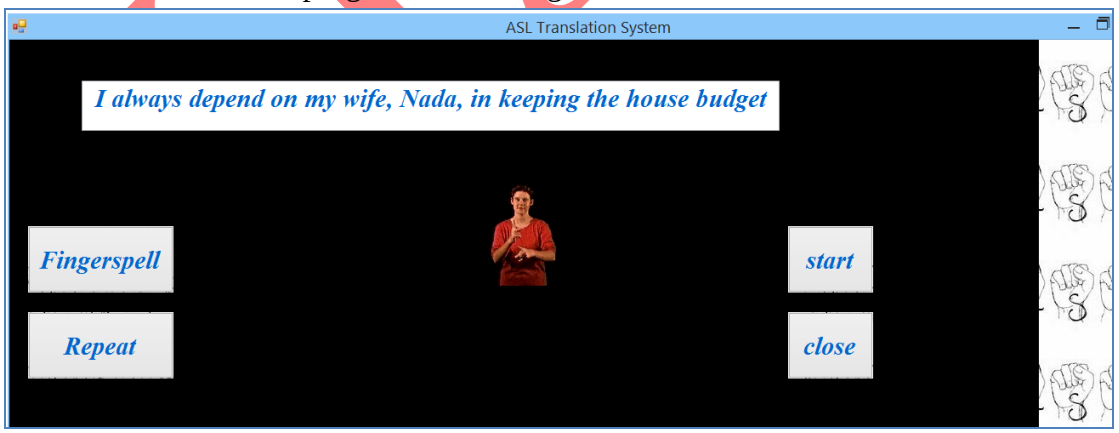

Figure (5): Another ASL movie

e. Displaying the output. Figures (4) and (5) show the display of a scene of one movie of the video series. While figure (6) shows the displaying of the fingerspell image for the word "Nada" that could not be found in the dictionary.

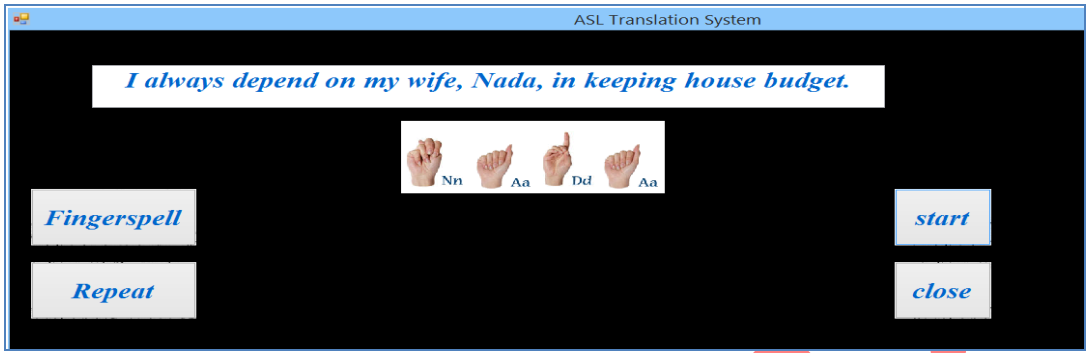

Figure (6): Fingerspell Image

5. SYSTEM EVALUATION AND DISCUSSION

The proposed system has been evaluated as any translation system according to the evaluation factor of accuracy. The proposed grammar is the main part of the system to be evaluated.

5.1. Building English- ASL corpus

The automated evaluation any translation system is based on building a corpus. Proposed system evaluation should be performed by building large English-ASL corpus that contains English sentence with its corresponding ASL sentence that is produced according to ASL grammar rule. The corpus also contains information about the POS tags for the sentences. The corpus consists of more than one hundred pair of sentences with its POS tag information. The corpus has been built for evaluation purpose only.

Since there is no existence of standard ASL grammar for free evaluation and comparing with the proposed system, the method for evaluating the proposed grammar will be available for future comparisons and evaluation metrics. To enlarge the constructed corpus, it would be time consuming, thus the corpus has been built for evaluation purposes only, and in future, it may be enlarged gradually over time to be a proposed standard English - ASL corpus available for free use. The English - ASL corpus that has been built for evaluation purpose is used to find the output ASL sentence corresponding to the English input sentence that has been produced by applying the proposed ASL grammar rules. Two evaluation factors would be used are accuracy and time.

5.2. Accuracy Evaluation Factor

For comparing the output ASL sentence for the proposed system with the ASL sentence of the corpus a similarity measure has been needed. One of the most common set similarity measures is the Jaccard Similarity Index, which is based on a simple set operations union and

intersection. The standard Jaccard Similarity equation is (Equ. 1), using this measure with additional processing steps to find the evaluation accuracy measure [Wil, 03].

Jaccard Similarity Ratio = Intersection(set A, set B) / Union (set A, set B)

Equ(1)

a. Evaluation Processing steps

Several processing steps have been applied to get the evaluation results. Algorithm (5) describes these steps.

Algorithm (5): Evaluation process

Processing the input sentence to find the index of the most similar sentence in the corpus.

Input: Input Sentence .

Output: The most similar sentence to the input one in the corpus.

step (1): Get the input paragraph or sentence

str = readfile(sentence);

Step (2): Remove the punctuations from the input text

str1= remove-punctuations(str);

Step (3): Remove English Stop Words from the sentence

str2= remove-stop-words(str1);

Step (4): Convert the input sentence into tokens

words= tokenization(str2);

Step (5): Normalization process by converting each token into Uppercase

words-norm= Uppercase(words);

Step(6): Searching the corpus to compute the Jaccard similarity ratio and finding the most similar one to the input sentence.

Semilar-sent=get-sent(Jaccardmax(words-norm,corpus-sentences));

Step(6): Return Similar-sent;

If the input text was a paragraph or a text file that contains several sentences, the algorithm has segmented it into individual sentences dealing with them one after another. The algorithm steps then would be applied to an individual sentence.

After reading the input sentence, a process of removing punctuations from the input sentence. After that, removing English stop words (words that are high frequency and low content like: but, is , or and if). Then, the normalization step by changing each letter in the sentence to its uppercase. Step (6) in the algorithm is to open the corpus to find out the most similar sentence in the corpus to the input sentence, the most similar sentence is identified by computing the Jaccard similarity.

For each input sentence, Algorithm (5) have been applied two times. First, applying the algorithm to find the most similar English sentence in the corpus to the input English sentence. Second, applying the algorithm to find the most similar ASL sentence in the corpus to the output ASL sentence that has been translated with the proposed ASL grammar. The success case is to return the same index in the two times. That means the translation process has been performed correctly. It is worth mentioning to know that most of the failure cases belongs to that the test sentence does neither exists in the corpus nor similar to any of the existence sentence in the corpus. This problem could be challenged by enlarge the corpus over time.

b. Experimental Results

Two types of experimental examples would be considered. First, in order to explain the algorithm(5) steps, an example of an English sentence would take place to find practically the effect of each step. Second, test text (group of test sentence) would be taken as input and results would be discussed.

b.1. Input English Sentence

Giving English sentence as an input to the algorithm and show the effect of each step. Input English Sentence: "In the eighteenth century, it was often convenient to regard man as a clockwork automaton."

Step (1): Get the input paragraph or sentence

str = "In the eighteenth century, it was often convenient to regard man as a clockwork automaton.";

Step (2): Remove the punctuations from the input text

str1= "In the eighteenth century it was often convenient to regard man as a clockwork automaton";

Step (3): Remove English Stop Words from the sentence

str2= "eighteenth century often convenient regard man clockwork automaton";

Step (4): Convert the input sentence into tokens

words= ['eighteenth', 'century', 'often', 'convenient', 'regard', 'man', 'clockwork', 'automaton'];

Step (5): Normalization process by converting each token into Uppercase

words-norm= ['EIGHTEENTH', 'CENTURY', 'OFTEN', 'CONVINIENT', 'REGARD', 'MAN', 'CLOCKWORK', 'AUTOMATON'];

Step (6): Multiple sub steps, these sub steps may be summarized as simple as possible to show only stemming and lemmatizing processes.

words-norm= ['EIGHTEEN', 'CENTURY', 'OFTEN', 'CONVINIENT', 'REGARD', 'MAN', 'CLOCKWORK', 'AUTOMATA'];

The list words-norm then would be matched with the all English sentences in the corpus to find the most similar one, and be returned as output.

b.2. Input Test Text

The results of the proposed ASL grammar are differ according to number of sentences, words, and the existence of the sentence itself or similar to it in the corpus or not. Selecting sentences for testing operation depends on whether the test sentences were the same sentences of English-ASL corpus that has been build for evaluation purpose or nearing them, or the test sentences may be different of them. Different samples of test sentences has been selected giving different results. Table (1) and table (2) shows these different results.

Table (1) contains different cases (English paragraphs) that are selected carefully, such that, most of the sentences in the paragraphs are similar to (not exactly the same) the sentences that have been saved in the corpus. Different sentences have been used, long sentences and short ones with different number of words.

Table (1): Different Results for Different Selected Test Cases

No. of Sentences	No. of words	Max Jaccard for English Sentence Ratio	Min Jaccard for English Sentence Ratio	Max Jaccard for ASL Sentence Ratio	Min Jaccard for ASL Sentence Ratio	Total Success Ratio
10	100	1.0	0.285	1.0	1.0	1.0
15	152	1.0	0.1428	1.0	0.25	0.9333
24	229	1.0	0.3333	1.0	1.0	1.0

25	230	0.8333	0	1.0	0	0.92

Maximum Jaccard Similarity for English sentences sometimes couldn't reach to (1.0), while for the ASL sentences could always reach to the maximum value(1.0), that belongs to the properties of ASL sentences (chapter two and three) that made the ASL sentences shorter than its equivalent English ones. The Minimum Jaccard Similarity may be the minimum value (0) for both English and ASL sentences, when there exists failure sentences in some cases. The Total Success Ratio, is the ratio that computed by dividing the number of success sentences by the number of all sentences for each case. The total success ratio may reach to (1.0) if all the sentences success and have a Jaccard similarity by finding its similar sentences (English and ASL) in the corpus.

Table (2): Different Results for Different Random Test Cases

No. of Sentences	No. of words	Max Jaccard for English Sentence Ratio	Min Jaccard for English Sentence Ratio	Max Jaccard for ASL Sentence Ratio	Min Jaccard for ASL Sentence Ratio	Total Success Ratio
9	93	0.1818	0	0.1818	0	0.2222
10	100	0.125	0	0.1666	0	0.3
13	170	0.3333	0	0.2857	0	0.2307
21	408	0.25	0	0.2222	0	0.3809

Table (2) shows the results for four cases in which produces random samples of paragraph from the web. Number of sentences is different from each case to another as well as number of words. It is obvious that the results in table (2) is less efficient than results in table (1). That belongs to the randomly selected English paragraph. Results in table (2) are still to be considered somewhat good because they referred to varietal the sentences that saved in the corpus. Such that there exists only more than one hundred sentences (because of the restricted time) and however that, the success results (even if it was few) is considered to be good mark that be given to both the grammar and the corpus.

According to results in both tables (1) and (2), each sentence is either computed as success of failure. The sentence computed as failure in one of two possibilities:

1. Couldn't find the similar sentence in the corpus.
2. Found an English sentence that is similar to the failure sentence, but couldn't find the ASL translation for it in parallel with the English one, or vice versa.

According to the first possibility, the problem could be solved by enlarging the English-ASL corpus to contain more different sentences in order to find similar sentences for all sentence

as more as possible. The second possibility is referring to an error in the grammar rules, or in the parallel English-ASL corpus. It is worth mentioning, according to all cases in the two tables (1 and 2) all failure sentences are belong to the first possibility, that all failure sentences couldn't be found in the corpus. That means, the grammar and the structure of the corpus have not recorded any error cases.

b.3. Time Evaluation Factor

As any translation system, the proposed system would be evaluated according to time factor. Since the system contains three sequential stages, take with consideration the time of each stage and the length of the sentence (number of words and total number of characters in the sentence).

Table (3): System Stages Time

No. of words	No. of characters	1st-stage Time in Seconds	2nd-stage Time in Seconds	3rd-stage Time in Seconds	Fingerspell (if any) Time in Second	Total Time in Seconds
12	63	2.36529	0.00044	2.28201	0.01642	4.66418
11	57	2.23514	0.00054	2.23096	4.27653e-07	4.46665
10	54	2.26948	0.00048	2.28285	1.28295e-06	4.55281
11	67	2.38614	0.00054	2.27355	8.55306e-07	4.66024
7	43	2.33522	0.00142	2.37944	8.55306e-07	4.71609
7	42	2.30316	0.00044	2.33622	0.04331	4.68315
4	17	2.26144	0.00048	2.30974	0.02105	4.59273
13	66	2.23671	0.00056	2.30287	4.27653e-07	4.54015
99	513	2.38541	2.24371	0.40569	0.20996	5.24479
161	958	2.43708	2.35889	0.66560	0.35244	5.81401

Table (3) shows the execution time of different samples selected randomly. There is no large effecting to the number of words or characters, that because each stage consists of different sub stages that are required for all sentences whatever its number of words or characters, these sub stages such as stemming, tokenization, lemmatization. Using NLTK provides time saving for all of these operations. Fingerspell technique may not be executed depending whether all the words of the sentence existing in the large dictionary or not. In addition to that, the number of words in the sentence refers to English stop words also have been computed (such as : a, the, to, in), while in most cases these words may be deleted them in the first few sub stages. Number of characters also encounters the spaces and the punctuations.

The execution time recorded in table (3) represents the execution time of the main processing steps without displaying time, since the display time depend on additional factors,

such as type of processor used and screen resolution and other technical factors. Using Python programming language has the main contribution in producing these results.

6. CONCLUSION AND FUTURE WORK

6.1. Conclusion

1. Using the NLTK for NLP in order to obtain the POS tags of the sentence resulting into trusted system in shorter time.
2. If an error has been found in the translated system, it may caused by the small error ratio of the NLTK toolkit during finding the POS tags.
3. Using this system as an important tool for the deaf and hearing impaired persons in their community in order to understand each other.
4. According to ASL grammar, many rules have flexibility during the translation. The system try to be stable according to the most famous grammar categories for the ASL language.
5. Using Python programming language has the main contribution in producing these results (Tables 1 and 2). It provides the using of NLTK in addition to provide reducing time consuming.
6. In order to evaluate any translation system, a corpus between the two languages must be used. Since ASLMT system are not wide distributed, there is no English _ ASL corpus that is available and free to use. In order to evaluate the system, the need of such corpus has been appeared. Building English _ ASL corpus for evaluation purpose may be an efficient method, this corpus contains pairs of English _ ASL sentence in addition to the POS tags for each sentence.
7. Using list of videos to display the ASL sentence has its advantage and disadvantage. The advantage, is by using video, there is no need to Know the nonmanual signs as well as orientation, movement and emotions. All of these factors have been recorded as a movie for each word. The disadvantage of using videos , its display time depend on additional factors, such as type of processor used and screen resolution and other technical factors. In addition to the needing of high quality storage for the large dictionary such that saving video for each word.

6.2. Future Work

1. It is very good if the system could be available on the web in order to be for free use by deaf communities.
2. The system may be improved to used for translation in specific subject, such as: weather forecasting, football matches. Such specification will make the vocabularies more limited, thus the error ratio will be small.
3. Developing the English _ ASL corpus by enlarging it with adding different sentences over time.
4. Selecting different way for displaying ASL sentence, such as animation. That needs saving multifactor for each ASL word, such as movement, orientation, emotions and

other non manual signs. Using these factors in causing moving of signing avatar graphically.

REFERENCES

[Asm, 15] Asmaa M. Hamandi, Aliaa K. Abdulhassan, Hala Bahjat. (2015). A Proposed Algorithm for Translating English Written Text to Fingerspell Language. Engineering and Technology Journal. **ISSN**:24120758. Iraq.

[**Cok, 94] Cokely Dennis , Charlotte Baker-Shenk . (1994).** American Sign Language: A Teacher's Resource . Washington: Clerc Books Gallaudet University Press .

[Dan, 07] Daniel Jurafsky & James H. Martin.C. (2007). Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition.

[Eli, 03] Elizabeth Keating , Gene Mirus . (2003). American Sign Language in virtual Space: Interactions between deaf users of computer-mediated video communication and the impact of technology on language practices . In Language in Society. United States of America: Cambridge University Press, USA.

[Hue, 05] M. Huenerfauth. (2005). "American Sign Language Generation: Multimodal NLG with Multiple Linguistic Channels", Proceedings of the ACL Student Research Workshop, pp. 37–42.USA.

[Jon, 94] W. Jone Hutchins & Harold L. Somer. (1994). An Introduction to Machine Translation. ACADEMIC PRESS INC. ISBN 0-12-362830-X. Cambridge.

[Mar, 09] Marek Hruz , and others. (2009). Input and Output Modalities used in a Sign-Language-Enabled Information Kiosk. Czech: University of West Bohemia, Pilsen, Czech Republic.

[NLTK, 01] Natural Language Tool Kit,(2001). Website: (http://www.nltk.org).

[Ric, 99] Richard A.Tennant, Marianne Gluszak Brown. (1999). The american sign language handshape dictionary. washington: clerkBooks Gallaudet University Press.

[Wil, 03] William W. Cohen, Pradeep R., Stephen E. Fienberg. (2003). A Comparison of String Distance Metrics for Name-Matching Tasks. American Association for Artificial Intelligence. USA.

[Wil, 11] William Vicars, E. (2011). American Sign Language Fingerspelling & Numbers: Introduction. (Bryan Eldredge). USA.