Sign Recognition and Speech Translation Using OPENCV

Akshay Goel[1], Raksha Tandon[2], Mandeep Singh Narula[3]

[1]*Student, Dept. of ECE, Jaypee Institute of Information Technology, Noida, India*
[2]*Student, Dept. of ECE, Jaypee Institute of Information Technology, Noida, India*
[3]*Assistant Professor, Dept. of ECE, Jaypee Institute of Information Technology, Noida, India*

Abstract - *This paper proposes the recognition of numbers and letters from the Assembly Sign Language. Numeric and character identification is the backbone of this proposed and enhanced efficient system. Convolution with the combination of threshold image, contours and defects in the sequential order caters the need of the project. This work has a real time ASL converter which converts hand gestures(numbers and letters) into text and makes it available for the user to cater the information as per needs. With the help of PYGLET tools and Google Text to Speech, numbers are transferred into voice notation i:e Text To Speech. This algorithm processes numeric to voice.*

Key Words: ASL, OPENCV, Contours, Threshold, PYGLET, TTS, GTTS.

1.INTRODUCTION

Over 5% of the world population or 466 million people, have disabling hearing loss disorder and there are about thousands of people who can't speak. The language used by them to communicate with other people is Assembly Sign Language (ASL)[4]. To communicate with others, these people usually need a person along and to solve this problem we propose a real time ASL translator that is user friendly as well as efficient.

In this paper a way is suggested to convert ASL into human understandable language so that deaf and dumb[10] people can come at an equal platform with the regular people and they do not feel disadvantaged and can put their words forward. The work is made using python on JUPYTER and uses Open CV library. OpenCV uses visually based interfaces. An image is just a matrix of scalar and vector quantity and various OPENCV has features which are used to perform various operations and hence convert the hand images and show the required results. The algorithm used in this work involves finding contours, convex hull and locating convexity defects and thus identifying the number or the letter and further converting text into speech using PYGLET and GTTS[3]. Figure 1 depicts how this paper functions.

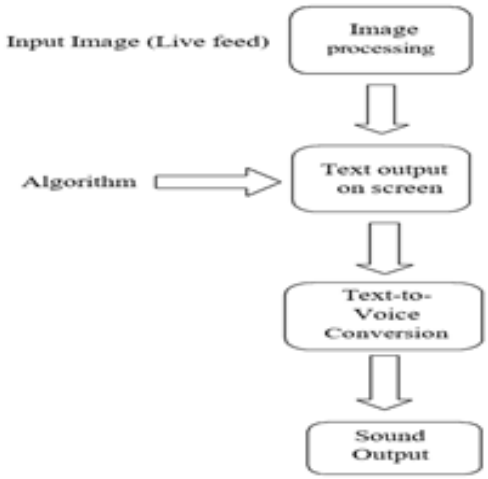


Fig -1: Structure Model

1.1 PROGRAM ALGORITHM

The Algorithm used has several steps involved which begins from taking input through a webcam at around 20-25 frames per second, Figure 2 depicts the algorithm. The live image is captured using a camera, Live feed in real time[1] (hand gesture).From the captured image a frame captures the hand image and the rest of the background is cropped. We apply Gaussian Blur[2], on the image that is inside the frame which is blurred. The image is blurred in order to filter the background noise which is done through morphological technique. The image is further converted from BGR scheme to HSV scheme i:e "Hue saturation value".

If in case, during the capturing of the frame, there is any error including that of capturing extra frames. Dilation and erosion have been used to filter out the image which help us to capture a true image. Below we have depicted the proposed work for the paper which gives a high efficiency.

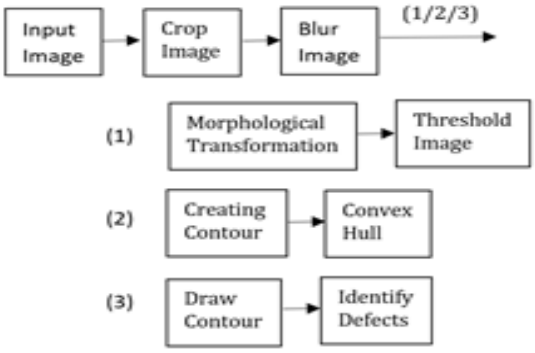


Fig -2: Algorithm

After these initial steps of image processing a threshold image is formed which is our main purpose. Threshold image gives us a clear idea about the contours projections. Figure 3 shows Threshold Image

Fig -3: Threshold Image

The hand is identified using contours which is an inbuilt function provided by OPENCV. It returns an array of coordinates of the contour formed. The contour is converted into a polygon which is called the hull that helps to detect the hand gesture. After processing the image data & analysing the frames using a convex hull. Figure 4 convex hull.

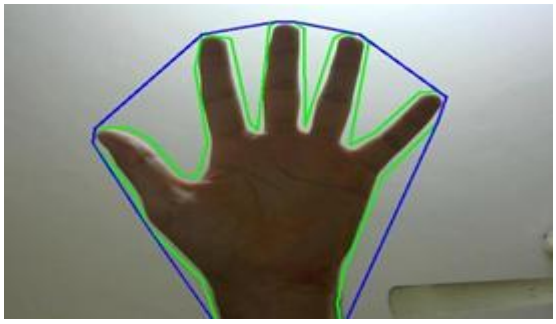
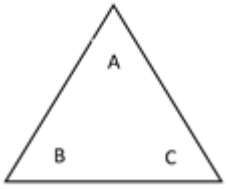


Fig -4: Convex hull

```
##
count_defects=0;
if(angle<=90)
   {
        count_defects=count_defects+1;
   }
```

We find defects from each frame of image which help us to detect the Hand Recognition. Based on the number of contours we find out the defects which we marked as points as red dots, they help us to recognize the gesture.

Cosine rule is used to locate these dots. A, B and C angles are calculated using these formulas. Suppose the angle A is greater than 90 degree. Then the count_defects is not incremented, i:e convexity defects are neglected, and if the angle is less than or equal to 90 degree, convexity defects increase by 1. This algorithm helps us find the number of convexity defects which can be further altered as per our needs.



$$CosA = (b^2 + c^2 - a^2)/2bc$$

$$CosB = (a^2 + c^2 - b^2)/2ac$$

$$CosC = (a^2 + b^2 - c^2)/2ab$$

```
##
if(count_defects==0)
    { cout<<"Digit is one"; }
if(count_defects==1)
    { cout<<"Digit is two"; }
if(count_defects==2)
    { cout<<"Digit is three"; }
if(count_defects==3)
    { cout<<"Digit is four"; }
if(count_defects==4)
    { cout<<"Digit is five"; }
```

Defects will count the numeric digit as we are trying to figure out the through our fingers, which is the aim of Assembly Sign Language, we can initialize the count_defects as per our need to ensure the required results,pre defined count-defects in the system will also enable us to use secretive symbol for alternate purpose.After obtaining the required results we display the stored image on the screen itself, which is stored in the system previously as the location of contours. Each symbol is stored in a database with XML extension[6]. Figure 3 depicts the research work.

2. TEXT TO SPEECH

We have displayed the desired output now another task is to convert it into audio to make this research work more compatible for the user or client whoever needs it.

PYGLET with the combination of GTTS[3] libraries in python takes care of the conversion process with the presence of audio drivers, pip install GTTS and pip install PYGLET are the two commands for installing GTTS and PYGLET respectively on 10th Generation Intel® Core™ i7 processor. Software requirement for speech output varies from operating system to operating system some are mentioned below.

Windows-Directx. Linux- OpenAI. MacOS-OpenAI.

Input at this moment is classified text and output is audio from which it can be transformed as per requirements of the user or client[5]. PYGLET and GTTS supports various formats as voice output MP3, AU, MP3, WAV and many more. After identifying the particular sentence. Shell script is done for the formation of the voice output. Text which was stored in XML[5] files previously were directed towards voice transformation with the help of GTTS and PYGLET.

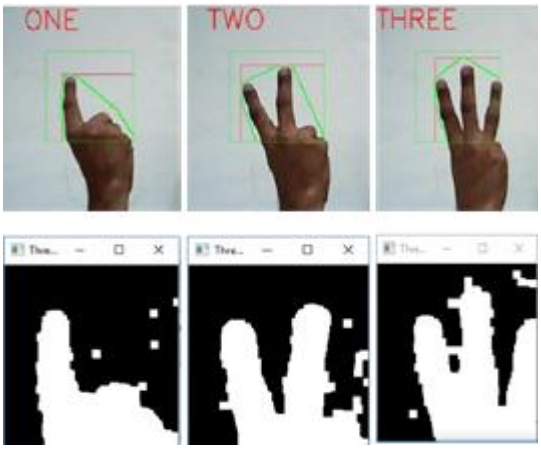


Fig -5: Desired Output

3. EXPERIMENTAL RESULTS

In the proposed approach we maintained the records for all prerecorded numbers 1-10 via each fingers, with the help of defects recorded and while testing it for 20 frames, after analyzing the confusion matrix[5] we have the results of this proposed approach for Assembly Sign Language, with the success rate of 99.1%, this model accurately identified the expression provided to the web camera as an input source and analyze it at approx 20-25 frames per second on 10th Generation Intel® Core™ i7 processor. Descriptive results are mentioned on Table 1 below.

Table -1: Results

Results	Percentage (%)
Accuracy	99.1%
Specificity	98.8%
Sensitivity	98.8%
Recall	99.1%

4. CONCLUSIONS

Our research work which is presented on this paper, which presents a suitable platform and approach for identification of numeric in Assembly Sign Language with the help of OPENCV library, many mathematical formulas and identified patterns using contours, convex hull, defects and threshold image, taking live feed from the hardware at around 20-25 frames per second. Later on with the support of GTTS and PYGLET input text classifier is converted in audio for a better presentation of results with a quite acceptable efficiency of about 99.1%.

REFERENCES

[1] Kar, Aradhana, and Pinaki Sankar Chatterjee. "An approach for minimizing the time taken by video processing for translating sign language to simple sentence in english." 2015 International Conference on Computational Intelligence and Networks. IEEE, 2015.

[2] Prateek, S. G., et al. "Dynamic Tool for American Sign Language Finger Spelling Interpreter." 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN). IEEE, 2018.

[3] Sangpal, Ravivanshikumar, et al. "JARVIS: An interpretation of AIML with integration of gTTS and Python." 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT). Vol. 1. IEEE, 2019.

[4] Kumar, Sujay S., et al. "Time Series Neural Networks for Real Time Sign Language Translation." 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, 2018.

[5] Truong, Vi NT, Chuan-Kai Yang, and Quoc-Viet Tran. "A translator for American sign language to text and speech." 2016 IEEE 5th Global Conference on Consumer Electronics. IEEE, 2016.

[6] Zhang, Chenyang, Yingli Tian, and Matt Huenerfauth. "Multi-modality American sign language recognition." 2016 IEEE International Conference on Image Processing (ICIP). IEEE, 2016.

[7] Mullah, Helal Uddin, Fidalizia Pyrtuh, and L. Joyprakash Singh. "Development of an HMM-based speech synthesis system for Indian English language." 2015 International Symposium on Advanced Computing and Communication (ISACC). IEEE, 2015.

[8] Madhuri, Yellapu, G. Anitha, and Ml Anburajan. "Vision-based sign language translation device." 2013 International Conference on Information Communication and Embedded Systems (ICICES). IEEE, 2013.

[9] Hakkun, Rizky Yuniar, and Achmad Baharuddin. "Sign language learning based on android for deaf and speech impaired people." 2015 International Electronics Symposium (IES). IEEE, 2015.