Real-time Conversion of Sign Language to Text and Speech

Kohsheen Tiku
Department of Information Science
BMS College of Engineering
Bangalore, India
kohsheen.t@gmail.com

Jayshree Maloo
Department of Information Science
BMS College of Engineering
Bangalore, India
jayshreemaloo03@gmail.com

Aishwarya Ramesh
Department of Information Science
BMS College of Engineering
Bangalore, India
ash.cancer98@gmail.com

Indra R
Department of Information Science
BMS College of Engineering
Bangalore, India
indra.ise@bmsce.ac.in

Abstract—**This paper presents an analysis of the performance of different techniques that have been used for the conversion of sign language to text/speech format. Using the best possible method after analysis, an android application is developed that can convert real-time ASL (American Sign Language) signs to text/speech.**

Keywords—sign language, ASL, image processing, machine learning

I. INTRODUCTION

466 million people worldwide have impaired hearing loss (more than 5 percent of the world 's population), 34 million of whom are teenagers, according to the World Health Organization (WHO). Studies expect these figures would surpass 900 million by 2050. Moreover, most cases of debilitating hearing loss affecting millions of people are concentrated in low- and middle-income countries.

Sign Languages allow the dumb and deaf people to communication with each other and the rest of the world. There are over 135 different sign languages around the world which include American Sign Language (ASL), British Sign Language (BSL) and Australian Sign Language (Auslan) etc.

American Sign Language has been created to reach the wider public and acts as the primary sign language of the Deaf populations in the United States and much of Anglophone Canada, also including most of West Africa and areas of Southeast Asia.

People with hearing impairments are left behind in online conferences, office sessions, schools. They usually use basic text chat to converse — a method less than optimal. With the growing adoption of telehealth, deaf people need to be able to communicate naturally with their healthcare network, colleagues and peers regardless of whether the second person knows sign language.

Being able to achieve a uniform sign language translation machine is not a simple task, however, there are two common methods used to address this problem namely sensor based sign language recognition and Vision-based sign language recognition. Sensor based sign language recognition [12] uses designs such as the robotic arm with a sensor, smart glove, golden glove for the conversion of ASL Sign language to speech. But the issue is that many people do not use it. Also, one must spend money to purchase such a glove, which is not easily available. Vision based Sign Language Translation [13][14] uses Digital Image Processing. It is a framework which is utilized to perceive and interpret nonstop gesture-based communication to English content. In vision-based gesture recognition, a camera is used as input. Videos are broken down into frames before processing. Hence vision-based methods are preferred over gesture-based approaches as anyone with a smartphone can convert sign language to text/speech and it is relatively cost-effective.

In this paper, the method of developing an android application is demonstrated for the vision-based approach, of sign language to text/speech conversion without any sensors, by only capturing video of the hand gestures, completely free of any cost.

II. METHODOLOGY

A. Overview

In this paper, 26 ASL alphabets are used along with 1 customized symbol for 'Space' which is to be recognized in real-time using a smartphone. For this purpose, here the One Plus 6 smartphone with OxygenOS (based on Android Oreo) operating system has been used. The algorithm is developed on top of a Java-based OpenCV wrapper. The entire system was developed using images that are of 200 x 200 pixels in RGB format.

To design an appropriate model, the first thing is to understand what features will be the most appropriate to extract from static images. Examples of such features include Radial signature, Histogram of gradients (HOG) [1], centroid distance signature, Fourier descriptors.

The technique which is the most appropriate for this scenario is Histogram of gradients (HOG) descriptors. HOG is preferred because the appearance and shape of a local object can be easily detected by means of intensity gradients or edge directions. The image is divided into small connected regions called cells, and a histogram of gradient directions is compiled for the pixels within each cell. The descriptor is the concatenation of the histograms. For higher accuracy, local histograms are contrast-normalized by measuring the intensity variance over a wider area of the image, called a block, and then using this value to normalize all cells within a block. This normalization results in greater invariance with shifts in lighting and shadowing.

Support Vector Machine (SVM) [8][9][10], a machine learning algorithm uses HOG descriptors as the features of the image. Hence, SVM is used to train our model and this experimentation deals with using three different array parameters for SVM and comparing the results of each. The three array parameters are Detection Method, Kernel and Dimensionality reduction type. The following are the different types of array parameters that are used for training.

- **Detection Method** - Contour Mask, Canny Edges, Skeleton
- **Kernel** - Linear, Radial Basis Function (RBF)
- **Dimensionality Reduction Type** - None, Principal Component Analysis (PCA)

B. *DataSet Used*

The dataset used for this paper is the ASL Kaggle dataset [2], which contains 3000 images for every alphabet of the English vocabulary. Here another character has been introduced which is unique from all other hand gestures for the purpose of acting as an indication of completion for the previous word. This special sign called the ‘Space’ allows the user to form sentences in a very simple fashion. For example, this space gesture will be used to separate ‘hello’ and ‘world’ in the sentence ‘hello world’.

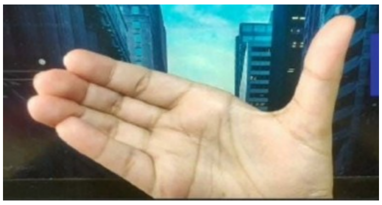


Figure 1: Hand gesture for ‘Space’

Since our dataset has 3000 images of all other characters, reduced it to 100 distinct images of each character for training since the SVM algorithm works more precisely with smaller datasets. The dataset created for space consists of 100 images of the gesture as well. In total, there are 27 classes (26 ‘Alphabet’ classes + 1 ‘Space’ class) where considered the ‘Space’ as a separate class.

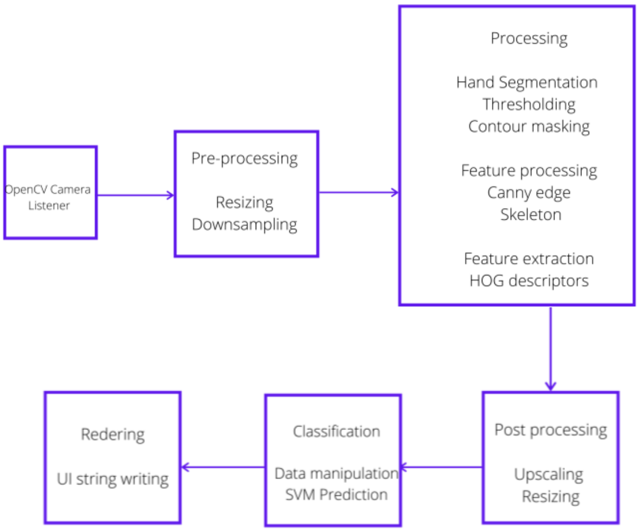


Figure 2: Architecture of the android application

III. IMPLEMENTATION

The application is designed and implemented using Android Studio and OpenCV [15] functions in Java.

A. *Calibration*

Here, color-based segmentation has been implemented provided using libraries present by OpenCV. This can be done by understanding all the different skin tones and their HSVA (Hue, Saturation, Value, Alpha) Configurations. The following lower and upper bounds define all the skin tones possible. Only if the image possesses pixel values in this range, the frame will be considered for classification else it will be discarded.

```
// H lowerBound.val[0] = 0; upperBound.val[0] = 25;
// S lowerBound.val[1] = 40; upperBound.val[1] = 255;
// V lowerBound.val[2] = 60; upperBound.val[2] = 255;
// A lowerBound.val[3] = 0; upperBound.val[3] = 255;
```

The image is then blurred using gaussian blur for easy processing. The next step is to find contours of the largest area of the frame wherein skin color is present. The main contour is applied to the largest area and child contour is also applied within the largest skin color area so that even if there are two patches of skin, say one full hand and the other some one’s finger, between those can be easily differentiated. A matrix is used to represent the contours of the skin area.

B. Processing of frame

The following diagram summarizes the steps involved in the processing of the frame.

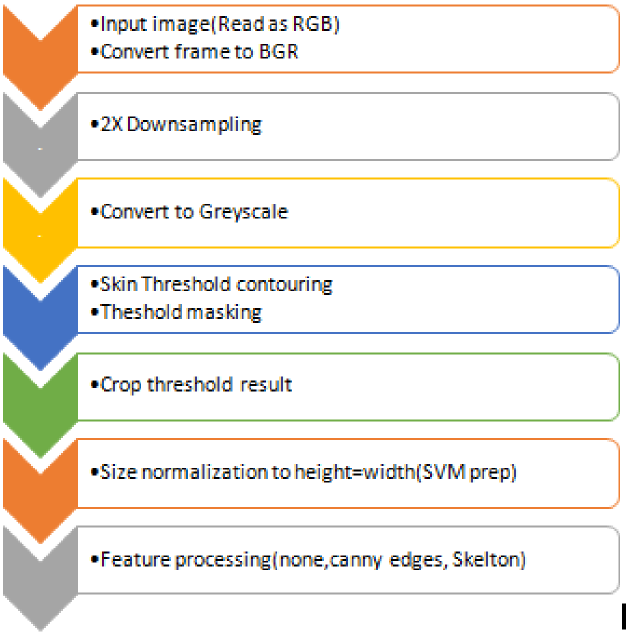

•Input image(Read as RGB)
•Convert frame to BGR
•2X Downsampling
•Convert to Greyscale
•Skin Threshold contouring
•Theshold masking
•Crop threshold result
•Size normalization to height=width(SVM prep)
•Feature processing(none,canny edges, Skelton)

Figure 3: Frame Processing Diagram

The steps involved in the processing of the frame are:

- An input image is read as BGR (Blue, Green, Red) Format since OpenCV uses BGR instead of RGB Format, then the image is converted to RGB for image processing.

- Downsampling [3] of the frame captured involves throwing away unnecessary image information discarding rows and columns of data at the edges of the image, this reduces storage requirement as shown in figure 4. The image is then converted to grayscale as shown in figure 5. Threshold Contouring is performed to segment the hand image from the background. Threshold masking is used to exclude unnecessary from image processing. After this, the image is cropped and normalization is performed to change the range of pixel intensity to increase contrast and make feature extraction easier.

- Different feature preprocessing algorithms can now be applied, which consist of contour mask, canny edges, skeleton explained further in the paper.

  All these techniques have been experimented with. All of them cannot be used together, results have been generated when each of these processes is used alone, as shown in table 1.

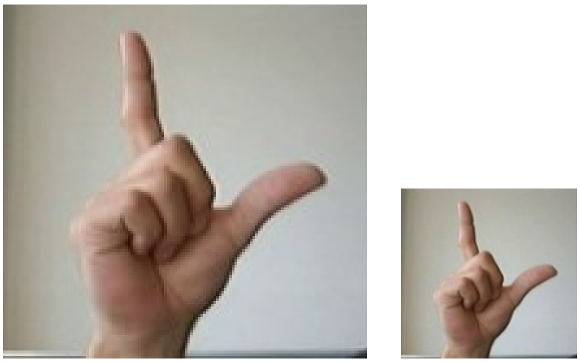


Figure 4: Down sampling of the frame

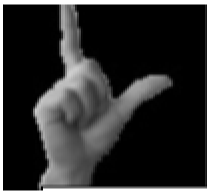


Figure 5: Converting image to Grayscale

C. Detection Method

1. Contour masking

Contours may be defined precisely as a curve that connects all the continuous points (along the boundary), with the same color or intensity. The contours are a valuable resource for the study of the structure and for identification and recognition of an object like a hand as shown in figure 6.

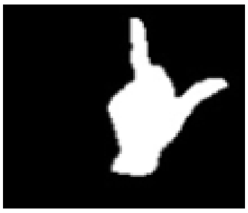


Figure 6: Contour Masking of hand gesture

2. Skeletonization

Skeletonization [4] is a method for reducing foreground regions to a skeletal remnant in a binary picture that essentially retains the magnitude and continuity of the original area while removing much of the original foreground pixels. The skeleton is valuable because it offers a clear and compact image of a form that retains much of the initial form's topological and scale characteristics. Refer figure 7 for the above.

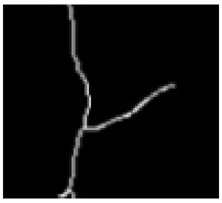


Figure 7: Skeleton form of the image

3.Canny Edges

Step by step process to implement Canny edges:[5]

- A Gaussian filter is applied to make the image smooth and remove the noise.
- Intensity gradients of the image are calculated.
- Non-maximum suppression is applied to remove the possibility of a false response.
- Double thresholding is done to detect or determine the possible edges.
- Edges are finalized by identifying and removing all other edges that are weak and not linked to strong edges.

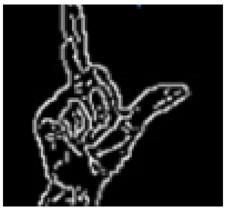


Figure 8: Canny Edges form of the image

D. Kernel

Kernels in SVM classification [6] refer to the function that is responsible for defining the decision boundaries between the classes. The SVM software has been used with the Linear Kernel and the RBF (radial basis function) kernel. Execution time for model selection is an important issue for practical applications of SVM.

1. Linear SVM [11]

Support vector learning is an algorithm which deals with finding a separate hyperplane with the has the greatest margin that separates the positive instances (labelled as +1) from the negative instances (labelled as -1). The hyperplane margin is defined as the shortest distance between the positive and negative occurrences closest to the hyperplane. The intuition behind searching for the large-margin hyperplane is that a hyperplane with the largest margin should be more noise resistant than a smaller-margin hyperplane.

Formally, assume all data meet the constraints.

f (x) = {+1, w * x_i + b >=1 and -1, w * x_i + b <= -1}

Where the w is the normal to the hyperplane, |b|/||w|| is the perpendicular distance from the hyperplane to the origin, and ||w|| is the Euclidean norm of w. [6]

2. Radial basis function kernel (RBF) Kernel [11]

This is a non-linear kernel which maps samples to a higher-dimensional space, unlike the linear kernel function. It can handle the case when the relation between class labels and attributes is nonlinear.

The RBF kernel is defined as:

$K_{RBF}(x, x') = exp[y\|x - x^1\|^2]$, where γ is a parameter that sets the "spread" of the kernel.

A kernel is any function of the form:

$k(x, x') = \left(\psi(x), \psi(x^1)\right)$, where ψ is a function that projects vector x into a new vector space. The kernel function computes the inner product between two projected vectors.

E. Dimensionality Reduction

Principal Component Analysis[7]

PCA uses a list of the principal axes to identify the underlying dataset before classifying it according to the amount of variance identified by each axis. PCA makes the maximum variability of the data set more visible by rotating the axes. The number of feature combinations is equal to the number of dimensions of the dataset and, in general, the maximum number of PCAs that can be constructed. The association between each main component should be zero as the residual variation is captured by the subsequent components. The similarity of any pair of own value/eigenvector is zero so that the axes are orthogonal, i.e. perpendicular to each other in the data space.

F. Classification

SVM (Support Vector Machine) is a supervised learning technique. The objective is to find a hyperplane that distinctly classifies data points into classes. 27 classes have been in our model each corresponding to letters in the English language. The SVM model will classify the images into the 27 classes to yield a result. SVM is used as it a supervised learning technique which is apt to solve the problem statement.

G. Post Processing

a. UI String Writing –User interface string writing is used to print a message for an error. This is to make the UI user friendly to the users.
b. Debugging – This function is mainly used for inspection of strings. This will help in removing errors and increasing the work efficiency of the app.

H. Use Case Diagram

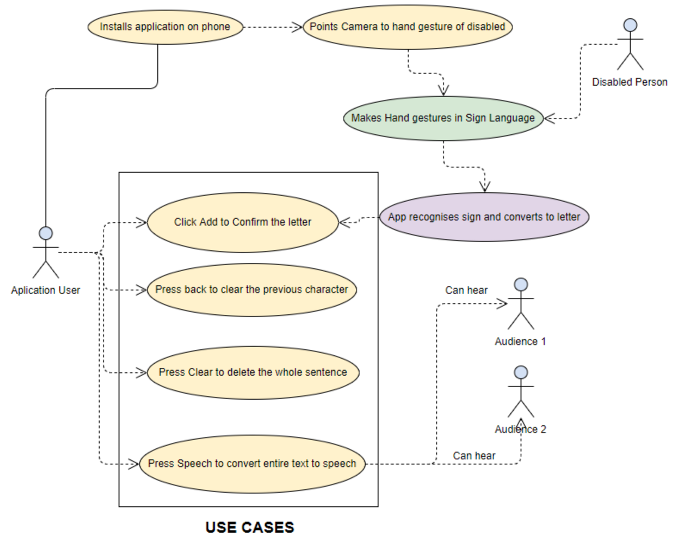

Installs application on phone
Points Camera to hand gesture of disabled
Disabled Person
Makes Hand gestures in Sign Language
App recognises sign and converts to letter
Click Add to Confirm the letter
Aplication User
Press back to clear the previous character
Can hear
Audience 1
Press Clear to delete the whole sentence
Audience 2
Can hear
Press Speech to convert entire text to speech
USE CASES

Figure 9: Use cases of application

In the above use case diagram, the communication is being explained between a deaf/dumb person with a person with all senses. The application user installs our application on his/her phone. He then points the camera to the deaf/dumb person who will make hand gestures to convey his/her message. The hand gestures are picked up by the app, and the due processes happen which has been explained above thoroughly. The app then recognizes this sign and prints it on the screen. The printed letters are then converted to speech by the application. The app has the following functionalities. The android application consists of 4 main features:

- Add alphabet - This functionality adds a new alphabet
- Back - This functionality erases previously detected alphabets.
- Clear- This functionality clears the entire sentence ;
- Speech- This function converts the entire text to speech format.

IV. RESULTS

A. Comparative Analysis of all array parameters in SVM

The table below presents a comparative analysis of the different detection methods, kernels and dimensionality reduction functions. Observing from the table, maximum accuracy and minimum average processing time for each image is achieved in the case of Canny Edges, RBF and PCA. Hence, these three array parameter values have been deployed in the SVM algorithm.

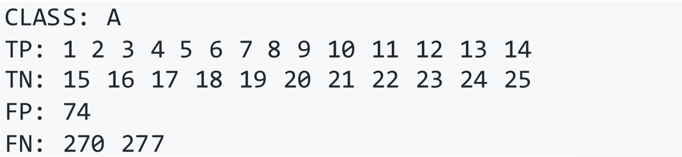
Table 1: Performance comparison of all array parameters in SVM

Sl. No	**Different combination of methods in SVM**				
	Detection Method	***Kernel***	***Dimensionality Reduction***	***Average per Image processing time (MilliSecond)***	***Accuracy***
1	Contour Masking	Linear	None	18.2	97.45
2	Contour Masking	Linear	PCA	18.0	97.98
3	Contour Masking	RBF	None	18.4	98.12
4	Contour Masking	RBF	PCA	17.8	98.34
5	Skeleton	Linear	None	17.9	98.22
6	Skeleton	Linear	PCA	17.7	98.25
7	Skeleton	RBF	None	18.4	98.56
8	Skeleton	RBF	PCA	18.0	98.89
9	Canny Edges	Linear	None	18.1	98.52
10	Canny Edges	Linear	PCA	17.5	98.67
11	Canny Edges	RBF	None	18.3	98.74
12	Canny Edges	RBF	PCA	15.0	98.82

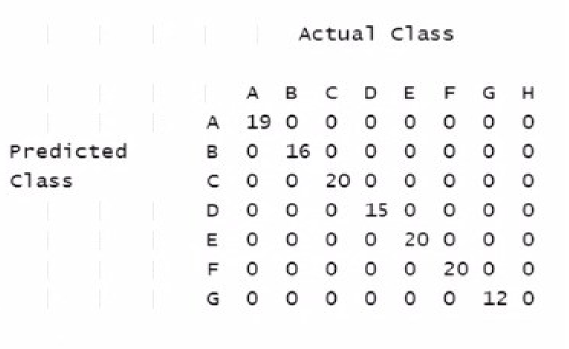
a.

B. Testing Results using the selected parameters(Canny Edges, RBF and PCA)

Testing is performed on 20% of the dataset that means 20 images of each alphabet and 'space' gesture. After testing a category matrix is obtained for each alphabet which gives sample number of images which were classified as True Positive(TP), True Negative(TN), False Positive(FP) and False Negative(FN). Following is the category matrix for the alphabet 'A'.



```
CLASS: A
TP: 1 2 3 4 5 6 7 8 9 10 11 12 13 14
TN: 15 16 17 18 19 20 21 22 23 24 25
FP: 74
FN: 270 277
```



		Actual Class							
		A	B	C	D	E	F	G	H
Predicted Class	A	19	0	0	0	0	0	0	0
	B	0	16	0	0	0	0	0	0
	C	0	0	20	0	0	0	0	0
	D	0	0	0	15	0	0	0	0
	E	0	0	0	0	20	0	0	0
	F	0	0	0	0	0	20	0	0
	G	0	0	0	0	0	0	12	0

Figure 10: Confusion Matrix

Then a confusion matrix is obtained of each class against each other class when testing. This matrix can be thought of as the following:
For example, in the confusion matrix the entry (1,1) indicates that for 19 images out of 20, the predicted class and actual classes were same thus giving a precision of 0.95. Similarly, for Class 'C', the entry (3,3) indicates 20 out of 20 images are classified correctly giving precision value as 1.00.

Then, for each alphabet the following is being calculated :

- Precision = TP/TP+FP
- Recall = TP / FN+TP
- F1 score = 2 * (precision * recall)/ (precision + recall)

The following table summarizes these measures for all alphabets.

Table 2: Measure of Precision, Recall and F-Measure the technique deployed

Measure	Maximum Value	Minimum Value	Median Value
Precision	1.00	0.76	0.91
Recall	1.00	0.81	0.94
F-Measure	1.00	0.79	0.93

C. *End Results*

The below figure 11 shows the User Interface of the application. It shows the construction of the word 'ILL' and various options like 'Add', 'Back', 'Clear' and 'Speech'.

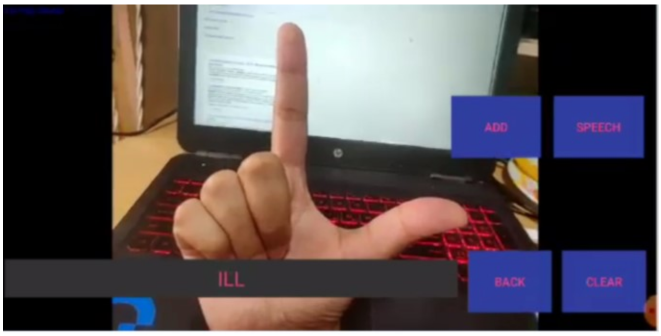


Figure 11 Construction of the word 'ILL'

ACKNOWLEDGMENT

We thank the institute BMS College of Engineering for the wonderful learning opportunity. We also express our gratitude to the department of information science and engineering for facilitating the process.

CONCLUSION

This paper compares different techniques and chooses the most optimal approach for creating a vision-based application for sign language to text/speech conversion for deaf/dumb people. The proposed system could efficiently recognize the alphabets from images using a customized SVM model. This project is aimed at societal contribution.

REFERENCES

[1] Patwary, Muhammed J. A. & Parvin, Shahnaj & Akter, Subrina. (2015). Significant HOG-Histogram of Oriented Gradient Feature Selection for Human Detection. International Journal of Computer Applications. 132. 20-24. 10.5120/ijca2015907704.

[2] ASL Reverse Dictionary - ASL Translation Using Deep Learning Ann Nelson Southern Methodist University, alnelson@mail.smu.edu KJ Price Southern Methodist University, kjprice@mail.smu.edu Rosalie Multari Sandia National Laboratory, ramulta@sandia.gov.

[3] Dumitrescu, & Boiangiu, Costin-Anton. (2019). A Study of Image Upsampling and Downsampling Filters. Computers. 8. 30. 10.3390/computers8020030.

[4] Saeed, Khalid & Tabedzki, Marek & Rybnik, Mariusz & Adamski, Marcin. (2010). K3M: A universal algorithm for image skeletonization and a review of thinning techniques. Applied Mathematics and Computer Science. 20. 317-335. 10.2478/v10006-010-0024-4.

[5] Mohan, Vijayarani. (2013). Performance Analysis of Canny and Sobel Edge Detection Algorithms in Image Mining. International Journal of Innovative Research in Computer and Communication Engineering. 1760-1767. M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[6] Tzotsos, Angelos & Argialas, Demetre. (2008). Support Vector Machine Classification for Object-Based Image Analysis. 10.1007/978-3-540-77058-9_36.

[7] Mishra, Sidharth & Sarkar, Uttam & Taraphder, Subhash & Datta, Sanjoy & Swain, Devi & Saikhom, Reshma & Panda, Sasmita & Laishram, Menalsh. (2017). Principal Component Analysis. International Journal of Livestock Research. 1. 10.5455/ijlr.20170415115235.

[8] Evgeniou, Theodoros & Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-7_12.

[9] Banjoko, Alabi & Yahya, Waheed Babatunde & Garba, Mohammed Kabir & Olaniran, Oyebayo & Dauda, Kazeem & Olorede, Kabir. (2016). SVM Paper in Tibiscus Journal 2016.

[10] Pradhan, Ashis. (2012). Support vector machine-A survey. IJETAE. 2

[11] Apostolidis-Afentoulis, Vasileios. (2015). SVM Classification with Linear and RBF kernels. 10.13140/RG.2.1.3351.4083.

[12] Kumar, Pradeep & Gauba, Himaanshu & Roy, Partha & Dogra, Debi. (2017). A Multimodal Framework for Sensor based Sign Language Recognition. Neurocomputing. 259. 10.1016/j.neucom.2016.08.132.

[13] Trigueiros, Paulo & Ribeiro, Fernando & Reis, Luís. (2014). Vision Based Portuguese Sign Language Recognition System. Advances in Intelligent Systems and Computing. 275. 10.1007/978-3-319-05951-8_57.

[14] Singh, Sanjay & Pai, Suraj & Mehta, Nayan & Varambally, Deepthi & Kohli, Pritika & Padmashri, T. (2019). Computer Vision Based Sign Language Recognition System..

[15] M. Khan, S. Chakraborty, R. Astya and S. Khepra, "Face Detection and Recognition Using OpenCV," *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Greater Noida, India, 2019, pp. 116-119