Development of an American Sign Language Recognition System using Canny Edge and Histogram of Oriented Gradient

I. A. Adeyanju[1*], O. O. Bello[2], M. A. Azeez[1]

[1]Department of Computer Engineering, Federal University, Oye-Ekiti, Nigeria
[2]Department of Computer Engineering, Ekiti State University, Ado-Ekiti, Nigeria

ABSTRACT: Sign language is used by people who have hearing and speaking difficulties, but not understood by many without these difficulties. Therefore, sign language recognition systems are developed to aid communication between hearing impaired people and others. This paper developed a static American Sign Language Recognition (ASLR) system using canny-edge and histogram of oriented gradient (HOG) for feature extraction with K-Nearest Neighbour (K-NN) as classifier. The sign language image datasets used consist of English alphabets from both Massey University and Kaggle, and numbers (0-9) from Massey University. Median filter was used to remove noise after images were converted to grayscale. Otsu algorithm was used for segmentation while edges in the images were preserved using canny edge detection technique with HOG parameters tuning to obtain feature vectors. The extracted features were used by K-NN for classification. An average recognition accuracy and computational testing time of 97.6% and 0.39s respectively were obtained based on experiments with the Massey University dataset. Similarly, an average recognition accuracy and computational testing time of 99.0% and 0.43s respectively were obtained based on experiments with the Kaggle dataset. The developed system successfully recognized static English alphabets and numbers and outperformed some existing systems.

KEYWORDS: American Sign Language, Recognition system, Otsu algorithm, Feature extraction, Histogram of oriented gradient, K-Nearest Neighbour (K-NN)

[Received Feb. 9, 2022; Revised May 25, 2022; Accepted May 26, 2022] Print ISSN: 0189-9546 | Online ISSN: 2437-2110

I. INTRODUCTION

The World Health Organization (WHO) estimated that there are over a billion people living with disability in the world, which is about 15% of the world's population. About 466 million of these people living with different types of hearing impairment; this represents over 5% of the world's population (World Health Organizations, 2021). In Nigeria, the percentage of people having hearing impairment is estimated to be 23.76% of the country's population (Treat, 2016). People with hearing impairment use sign language as means of communication in the society which is typically done through an interpreter. The major challenge is that there are few and inadequate numbers of expert sign language interpreters thereby increasing the communication gap between the hearing impaired and others in the society. Several assistive devices have been proposed for hearing impaired to communicate with others which complement conventional sign language interpreters. Such assistive devices are either used for demonstrating sign language given a text or automatically recognizing a sign language. Few works have been done on sign language demonstration mainly using robot-like machines(Alabi, 2019; Maliki *et al*., 2017). Also many researchers have worked on sign language recognition system and different approaches have been proposed which can be broadly categorised as data glove based or vision based (Cheok *et al*., 2019). The data glove-based approach used sensors attached to a glove and worn by a user to convert finger movements into electrical signals for determining the hand posture for recognition. On the other hand, the vision-based approach uses digital image processing techniques. The use of these approaches produced good results; however, there are still limitations such as low accuracy, usability issues, high computational time and space complexity. Hence, this proposed system intends to address accuracy and high computational time limitations with canny edge and histogram of oriented gradient techniques with K-nearest neighbour classifier. The rest of this paper is organized as follows. Related work is presented in Section II. Section III describes design methodology. Section IV presents experimental setup. Result and discussion are presented in section V. Finally, Section VI presents conclusion and future work.

II. RELATED WORKS

Sign language recognition system was developed using artificial neural network (ANN) (Hassan *et al*., 2018; Kulkarni and Lokhande, 2010). Kulkarni and Lokhande (2010) used Otsu's segmentation and Canny edge detection techniques for feature extraction. It achieved 92.3% recognition accuracy for

*Corresponding author: ibrahim.adeyanju@fuoye.edu.ng
doi: http://dx.doi.org/10.4314/njtd.v19i3.2

the 26 English alphabets. Hasan *et al*., (2017) system is based on Luminance, Chromaticity Blue, Chromaticity Red (YCbCr) colour space segmentation. The boundary edge of the hand sign area is extracted through Canny edge detector and Freeman Chain Code was used for feature extraction. The system had an accuracy of 96.5% for 20 Bangla alphabets. However, both systems (Hassan *et al*., 2018; Kulkarni and Lokhande, 2010) were limited to recognition of static alphabets with high computational cost and achieved low recognition rate without uniform background. The major drawback of a colour-based approach is that it is more sensitive to the colour and intensity of the light source, which affects recognition accuracy.

Sign language recognition system was proposed using Hue, Saturation, Intensity (HSV) colour space to extract useful features and Euclidean distance for template matching (Hartanto *et al*., 2014; Huong *et al*., 2016). Hartanto *et al*. (2014) extracted feature from sign images of 26 Indonesian alphabets using SURF with a recognition accuracy of 63% while Huong *et al*. (2016) extracted features from sign images of 26 Vietnamese characters using PCA with a recognition accuracy of 91.5%. Similar approach by Jimoh *et al*. (2020) based on OpenCV template matching on android phone was developed for sign language recognition with selected English vocabularies. Features were extracted using Oriented Fast and Rotated Brief (ORB) algorithm with Principal Component Analysis. The system achieved an accuracy of 87% on the test data of hand gestures. They argued that the accuracy of their systems can be improved upon using other feature extraction techniques and by increasing the dataset size.

Yasir *et al*. (2016) developed Bengal sign language recognition using support vector machine. The features were extracted using shift-invariant feature transform (SIFT) and k-means clustering to create bag of words from video sequence of signs achieving a significant accuracy for Bangal vowels and few selected words signed with one hand or two hands. A similar research by Jin *et al*. (2016) developed an American Sign Language recognition for 16 alphabets using support vector machine and implemented on a mobile application. Their system used Canny edge detection and seeded region growing to segment the hand gesture from its background. Features were extracted from the edge detected image using speed up robust features (SURF) algorithm and K-means clustering. The system achieved an accuracy of 97.1% but it has misclassified signs with high similarity. Both systems recognized limited alphabets of their chosen sign language.

Work has also been done on Nigerian local sign language recognition. Hassan *et al*. (2018) segmented Hausa sign language images using thresholding techniques and median filter for removing unwanted noise. Edges of the segmented images were obtained using Prewitt edge detection techniques and Fourier descriptor while Particle Swarm Optimization (PSO) were used for feature extraction. The system used artificial neural network (ANN) as the classification algorithm for recognition of 21 static Hausa sign languages with a recognition accuracy of 93.9%. In a related work, an offline static gesture recognition system for Yoruba numeral counting was proposed by Jimoh *et al*. (2018) which considered hand gesture for Yoruba numeral from one to ten (1-10) as input image. Canny edge detection and histogram of the oriented gradients were used as feature extraction techniques with support vector machine (SVM) and achieved a recognition accuracy of 95%. Both systems were developed with a limited dataset which might affect their recognition accuracy.

Mahmud *et al*. (2019) developed ASL alphabets recognition system; the images were pre-processed to obtain region of interest. The features needed for recognition were obtained with Canny edge and Histogram of Oriented Gradient (HOG) techniques. The techniques extracted 20736 features vectors for each image of 200 x 200-pixel size with block of 2x2 cells. The system achieved recognition accuracy of 94.2% with K-Nearest Neighbour (K-NN). In the same research, features were extracted using bag of features (BoF) and k-means clustering with support vector machine (SVM) achieved recognition accuracy of 86%. Similar approach was reported by Sharma *et al*. (2020) using HOG and extracted 720 feature vectors. The system was classified using K-NN with recognition accuracy of 94.1%.

Masood *et al*. (2018) developed a sign language recognition system with Massey dataset of English alphabets and digits (0-9). The images in the dataset were augmented to produce large dataset. The model used VGG16 based on convolution neural network (CNN) architecture with 4 epochs and achieved recognition accuracy of 96%. Also, Tolentino *et al*. (2019) developed a real-time system using a convolution neural network (CNN) to learn sign-language for beginners. This system is based on a skin-colour technique, where the range of skin-colour is predetermined to extract the hand region from the background. The system achieved an average testing accuracy of 93.7%; with 90.0% attributed to ASL alphabets, 93.4% for number and 97.5% for static word recognition respectively.

Dudhal *et al*. (2019) proposed CNN based approach for Indian sign language recognition system. Features were extracted by hybridized adaptive thresholding and Gaussian blur with shift-invariant feature transform (SIFT) algorithm. The extracted features were fed into CNN for classification with recognition accuracy of 92.8%. The research also used adaptive thresholding with CNN and achieved accuracy of 91.8%. Their findings show that better performance is achieved using hybridization of SIFT and adaptive thresholding with gaussian blur compared to only adaptive thresholding with CNN. Wadhawan & Kumar (2020) proposed a similar approach for a robust modelling of static signs language recognition using deep learning-based CNN. The developed system achieved the highest training accuracy of 99.7% and 99.9% on coloured and grayscale images respectively. In a similar approach, Brahmankar *et al*. (2021) developed a system using canny edge detection with CNN to recognize Indian sign language of 35 signs containing alphabets and numbers. Their system achieved recognition accuracy of 98%.

Das *et al*. (2020) used a convolutional neural network to create static American Sign Language. From the Massey dataset, the system recognized 26 English alphabets. The model is composed of four groups of two convolutional layers followed by a max-pool layer and a dropout layer, as well as two groups of fully connected layers followed by a dropout

layer and one final output layer, with a recognition accuracy of 94.3%. Similarly, Jain *et al*. (2021) developed a model using support Vector Machine (SVM) and Convolutional Neural Network (CNN) for American Sign Language (ASL). Their research achieved recognition accuracy of 98.6% for double layer convolutional neural networks. The experiment also shown that optimal filter size was obtained at 8 x 8 for both single and double layer convolutional neural network. They argued that accuracy of CNN model can be improved by altering the filter size.

Poor image background and illumination, space complexity, low accuracy, and high computational time are some of the observed gaps and challenges facing existing sign language recognition models. Several researchers have developed models using various datasets. However, comparing their works becomes difficult due to the datasets used not being available. Therefore, this research is aimed to improve the recognition accuracy and computational time of existing models using benchmark datasets used by other researchers.

III. DESIGN METHODOLOGY

The stages involved in design sign language recognition system are image acquisition, image pre-processing, image segmentation, feature extraction, classification and recognition. These stages are further explained in section below. A block diagram description of developed sign language recognition system for static American Sign Language alphabets (A-Z) and numbers (0-9) is shown in Figure 1. The figure comprises the various stages to achieve the research aim and objectives.

A. Image Acquisition

Image acquisition is the first step in image processing techniques. In this paper, static single hand sign images of ASL are acquired from publicly available datasets. Two publicly available datasets were used for training and testing the model. The Kaggle dataset by Akash (2018) consists of an alphabet (A-Z) with 3000 images per class. The second dataset from Massey University by Barczak *et al*. (2011) consists of 36 classes of alphabets (A-Z) and numbers (0-9) with 70 images per class except for the sign labelled T, which has 65 images.

B. Image Pre-processing

In this stage, the input RGB (Red Green and Blue) colour image was converted into grayscale image. The median filter technique was used to eliminate noise from the images while keeping relevant edges. Figures 2(a) and 2(b) show the input image and its grayscale equivalent from the Kaggle dataset, respectively. Figures 3(a) and 3(b) show the input image and its grayscale equivalent from the Massey dataset, respectively. The equation used to transform input coloured image to grayscale, GY is given as:

$$GY = 0.56G + 0.33R + 0.11B \quad (1)$$

C. Image Segmentation

Image segmentation is one of the most important stages in image processing techniques. It is done by partitioning an image into region of interest from the entire image. In achieving image segmentation in this research, Otsu algorithm technique was used to separate an image into background and foreground. Histogram and probabilities of each intensity level of input image were computed to find the class mean and variance. The threshold value used for the separation is determined by iterating through all possible threshold values in an image to maximize the image's between-class variance. This technique creates background and foreground in an image, as shown in Figures 4 and 5 for both datasets.

D. Feature Extraction Techniques

Feature extraction techniques are used to obtain the most distinctive features from the input image. It is a form of dimensionality reduction which is widely used in image processing applications. The technique reduced the entire features of an image into a smaller collection of features that represent the entire image. Hence, effectively reducing the required computational time without compromising the efficiency. In this research, Canny edge detection and Histogram of oriented gradient (HOG) techniques were used to extract features needed for recognition of static signs of ASL alphabets (A-Z) and (0-9).

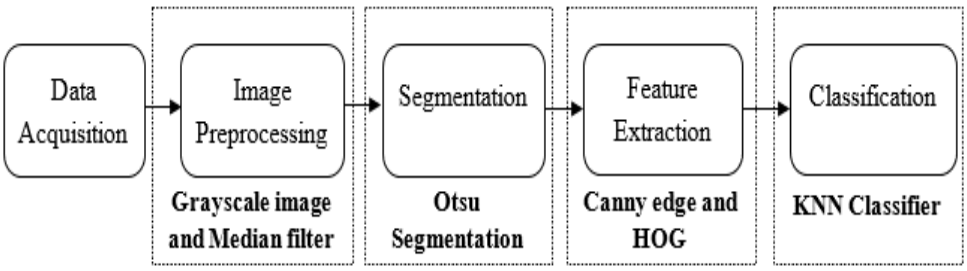

Figure 1: Block diagram of the developed system.

Figure 2(a): RGB colour image for sign 'A' from Kaggle dataset.

Figure 2(b): Grayscale image for sign 'A' from Kaggle dataset.

Figure 3(a): RGB colour image for sign 'A' from Massey dataset.

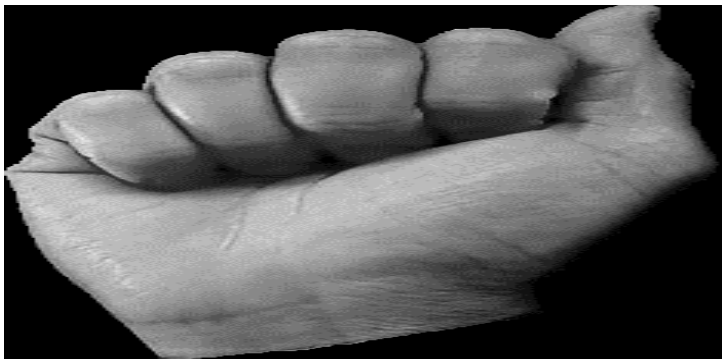

Figure 3(b): Grayscale image for sign 'A' from Massey dataset.

Figure 4: Otsu thresholding for Kaggle sign image.

Figure 5: Otsu thresholding for Massey sign image.

1) Canny Edge Detection

This technique extracts features that have clearly identified edges in an image. It has proven as a robust method to detection edges in an image when appropriate thresholds are applied (Adeyanju *et al*., 2021; Lawend *et al*., 2005; Shah *et al*., 2020). Canny edge detection was used in this research to detect a range of edges in the images. The result from edge detection was used to extract features using HOG. Canny edge detection implementation is shown in Algorithm listing 1 and output images for a sample from both datasets used in this research are shown in Figures 6 and 7.

2) Histogram Oriented Gradient (HOG)

Histogram of oriented gradient (HOG) is widely used feature descriptor in image processing to extract useful features from an image (Dalal and Triggs, 2005). The parameters that determine the type and numbers of features to be extracted are cell size (pixel per cell), block size (cell per block), orientation bin and image size (Mahmud *et al*., 2019; Mohammed and Melhum, 2020). In this paper, the output of canny edge detection was used as an input to generate features needed for sign language recognition using Histogram oriented gradient (HOG) which described the property of a given sign. The intensity distributions of local edges in each region of interest are counted and described using histograms. This is accomplished by partitioning the image into cells and creating single-dimensional histograms for the edge orientations of pixels in each cell. In this paper, HOG parameters used to obtain feature vectors are; Image size: 200 x 200 pixel, pixels per cell of 24 x 24, 16 x 16, and 8 x 8, cells per block of 2x2, orientation bin of 9 and normalization scheme of L2-Norm.

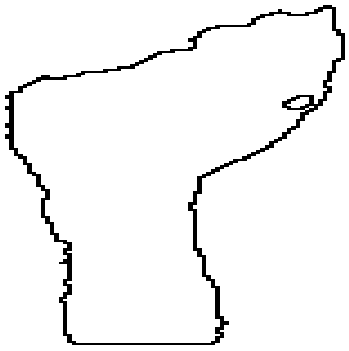

Figure 6: Canny edge for Kaggle sign image.

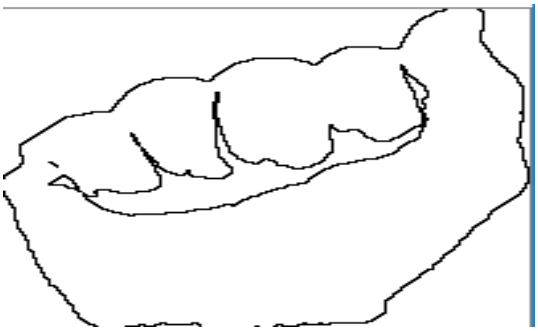

Figure 7: Canny edge for Massey sign image.

Algorithm listing 1: Canny edge detection (Jimoh *et al*., 2018)

Step1: Read the input image I
Step 2: Remove noise from the input grayscale image using median filter with kernel size of 3.
Step 3: Compute the gradient and edge direction representations of the image.
Step 4: Apply non-maxima suppression to the gradient magnitude of the image.
Step 5: The acquired gradient is compared with the set threshold value to understand if the taken point is an edge or not. Any gradient value G >Upper threshold value is an edge. While any gradient value G < lower threshold is taken not to be an edge.
Step 6: Store the edge detected image.
Step 7: Steps 1 to 6 were applied to all the images in the dataset by looping through it

Algorithm listing 2: HOG feature extraction (Mahmud *et al*., 2019)

Step 1: The input image, I of 200 x 200 image size was divided into cells
Step 2: Gradient of image are computed in both x and y direction of the cell image.
Step 3: Magnitude and orientation of the gradient image were computed as shown in Eqns (2) and (3).

The magnitude of the gradient, $|G| = \sqrt{{I_X}^2 + {I_Y}^2}$ (2)

The orientation of the gradient, $\theta = \tan^{-1}\frac{I_X}{I_Y}$ (3)

Step 4: Weighted votes for gradient and orientation are obtained for every cell to creating the cell histogram.
Step 5: Histogram of gradient are normalized for a set of cells to avoid been affected by scaling and lighting variation. *L*2-Norm technique was used and the equation is given as:

$$L2 - norm := \frac{v}{\sqrt{{[v]_k}^2 + e^2}} \quad (4)$$

v is non-normalized vector containing all histograms in each block and is *L*2-norm value for $k = 1, 2, 3\ldots n$, *n* is the number of features, e is a small normalization constant to avoid division by zero.
Step 6: Steps 1 to 5 are repeated for all the images and features were stored.

HOG implementation algorithm used for feature extraction are shown in Algorithm listing 2.

E. Classification

Classification is a significant tool for the analysis of statistical problems and identifying whether an object belongs to a particular class based on trained model. Image classification is the process of assigning pixels to an ordered set of related categories in which features are categorized according to its similarities. In this paper, K-Nearest Neighbours (K-NN) algorithm was used for classification of different classes of static ASL image. The distance between test features of unknown labelled are compared with all feature vectors in the training dataset of known label for prediction of the class it belongs. This is done by calculating the frequency of all classes out of K closest neighbours and labels the test data with the class having the highest frequency. Euclidean distance metrics was used to determine the distance between the data points. The Euclidean distance (ED) equation is given as:

$$ED = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \quad (5)$$

where;

x_i = Test data, y_i = Training data and k = Number of neighbour

IV. EXPERIMENTAL SETUP

This research was developed using python 3.7 as a programming language which is an open source with Panda, Seaborn, Scikit-learn and OpenCV libraries to enable it make image manipulation and recognition. The model was

implemented on Intel(R) Core (TM) i5-4310U CPU @ 2.00GHz, 8GB of RAM and Windows 10 operating system.

The two datasets Kaggle and Massey University of static sign images were used in this research. 4680 sign images of 26 classes (A-Z) from Kaggle dataset were randomly selected from the entire dataset, and all the 2515 sign images of 36 classes (A-Z and 0-9) from Massey University dataset were used.

Using the percentage split method, two–third (67%) of selected sign images from Kaggle dataset were used for training, which represents 3120 sign images (120 images per class), and the remaining one–third (33%) used for testing, which represents 1560 sign images (60 images per class). Similarly, for the Massey University dataset of 2515 sign images, two–third (67%) of the sign images were used for training, which represents 1653 sign images (46 images per class except sign 'T' of 43 images), and the remaining one–third (33%) were used for testing, represent 862 sign images (24 images per class except sign 'T' of 22 images). Table 1 shows size of images used for training and testing for the two datasets considered.

Table 1: Size of images used for training and testing sets for the two datasets.

Dataset	Dataset Size	Training Size	Testing Size
Kaggle Dataset	4680	3120	1560
Massey Dataset	2515	1653	862

Table 2 shows the features vector length obtained with chosen values of HOG parameters. The HOG parameters that determine the features to be extracted from an input image are cell size (pixel per cell), block size (cells per block), orientation bin, and image size. The values of block size (cell per block), orientation bin and image size were fixed while cell size (pixel per cell) were tuned to obtained good features.

Table 2: 200 x 200 image with HOG parameters and their features length.

Pixel per cell	Cells per block	Orientation bin	Feature length
8 x 8	2 x 2	9	20736
16 x 16	2 x 2	9	4356
24 x 24	2 x 2	9	1764

For image size of 200 x 200-dimension, pixel per cell of 24 x 24, 16 x 16 and 8 x 8, cells per block of 2 x2 and orientation bin (bin size) of 9. The developed system was evaluated using accuracy metric and computational time. The accuracy of the system with testing portion of the dataset is computed using Eqn. (6). The average computational testing time was achieved after three successful predictions of the sign image.

$$Accuracy\ (\%) = \frac{Correctly\ predicted\ signs}{Total\ no.of\ testing\ \ samples} \times 100 \qquad (6)$$

V. RESULTS AND DISCUSSION

A. Kaggle Dataset with HOG Parameters and K-NN Classifier

The performance of parameters experimented are computed using accuracy metric and computational time.

Table 3 shows recognition accuracy of developed system on each class of testing sign image for block size of 24x24 with k = 1. Figure 8 shows the confusion matrix of the system. Developed system achieved overall recognition accuracy of 99.0% on Kaggle testing dataset. This accuracy result was tested against other k- values of 3, 5, 7 and 9 to check for optimal result. Figure 9 represents accuracy chart of k = 1, 3, 5, 7 and 9 with better accuracy obtained when k = 1.

Table 3: Performance of developed system with HOG parameters (pixel per cell of 24x24) with *K*-nearest neighbour, *k* = 1 on Kaggle Dataset.

Sign	Test Sample	Correctly Classify Sign	Wrongly Classify Sign	Accuracy (%)
A	60	60	0	100
B	60	60	0	100
C	60	60	0	100
D	60	60	0	100
E	60	60	0	100
F	60	60	0	100
G	60	60	0	100
H	60	60	0	100
I	60	60	0	100
J	60	60	0	100
K	60	60	0	100
L	60	56	4	93.3
M	60	60	0	100
N	60	60	0	100
O	60	58	2	96.7
P	60	59	1	98.3
Q	60	60	0	100
R	60	60	0	100
S	60	58	2	96.7
T	60	59	1	98.3
U	60	60	0	100
V	60	60	0	100
W	60	60	0	100
X	60	55	5	91.7
Y	60	60	0	100
Z	60	59	1	98.3
			Average	**99.0**

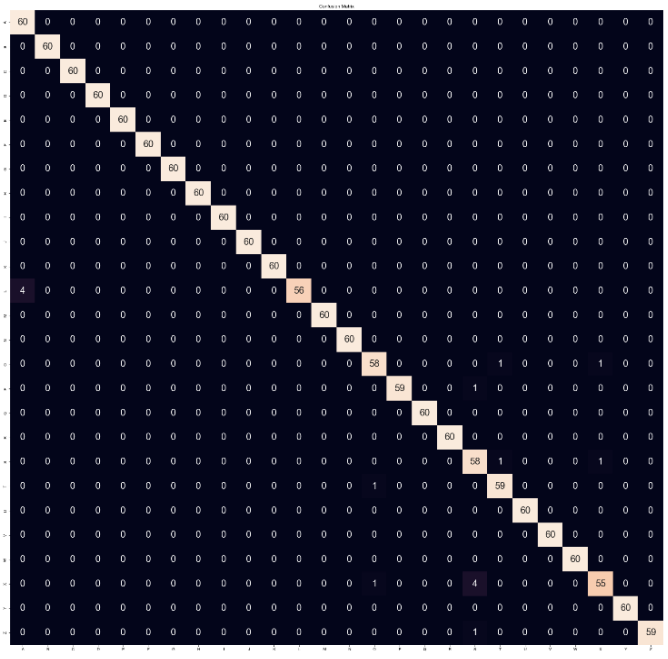

Figure 8: Confusion matrix of developed system.

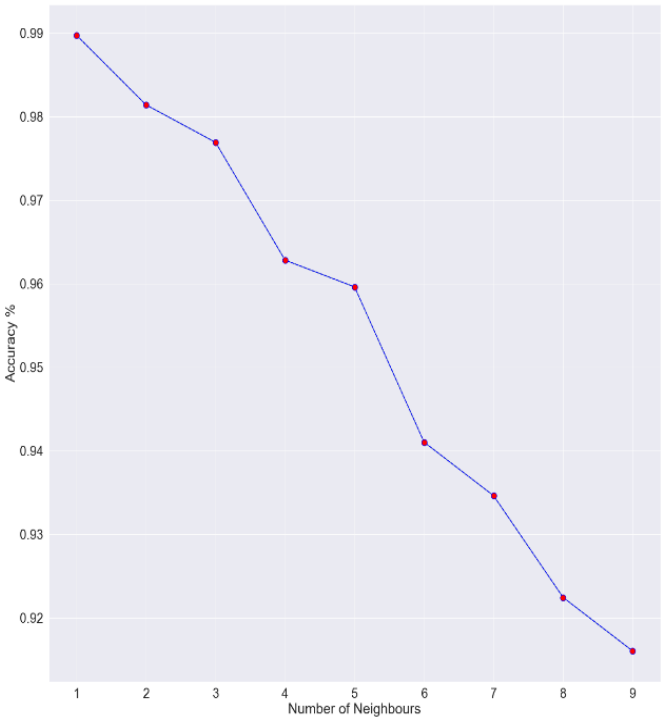

Figure 9: Accuracy of other k values.

An average computational time (CT) on testing image was determined by calculating the time taken to predict an image. The predictions were performed on testing image after five attempts and an average computational testing time of 0.43 seconds were computed as shown in Table 4 for developed system with pixels per cell of 24x24 at and k =1.

The experiment was also carried out for HOG parameters of pixel per cell of 16x16 and 8x8 with K-NN of k = 1,3,5,7 and 9 respectively. The performance of the experiment when k = 1 achieved high average recognition accuracy of 99.0% for 16x16 pixel per cell and 98.2% for 8x8 pixel per cell respectively, compared to other values of k. Figures 10 and 11 show the accuracy chart of other k values on Kaggle dataset.

B. Massey Dataset with HOG Parameters and K-NN Classifier

Similar to Section IV.A, the same experiment was conducted on the Massey dataset. The performance of developed system with HOG parameter of 24 x 24 pixel per cell and when k = 1 achieved an average recognition accuracy of 97.6%. Table 5 shows recognition accuracy of developed system on each class of testing sign image for block size of 24x24 with k = 1.

Figure 12 shows the confusion matrix of the developed system classification. The accuracy of the system developed was tested against other K values of 3, 5, 7 and 9 to check for optimal performance as shown in Figure 13. It further shows that an increase in K values reduce the accuracy of the system. Similarly, an average computational time (CT) on testing image was determined by calculating the time taken to predict an image. The predictions were performed on testing image of Massey dataset after five attempts and an average computational testing time of 0.39 seconds were computed as shown in Table 6 for developed system with pixels per cell of 24x24 at and k = 1.

The experiment was also carried out for HOG parameters of pixel per cell of 16x16 and 8x8 with K-NN of k = 1,3,5,7 and 9 respectively. The performance of the experiment when k = 1 achieved high average recognition accuracy of 97.3% for16x16 pixel per cell and 97.3% for 8x8 pixel per cell respectively, compared to other values of k. Figures 14 and 15 show the accuracy chart of other k values on Massey dataset.

Table 4: Computational time of developed HOG with K- nearest neighbour of k = 1 on Kaggle dataset after five tries.

Computational Time (CT)						
Pixel per cell	CT 1 (sec)	CT 2 (sec)	CT 3 (sec)	CT 4 (sec)	CT 5 (sec)	Average (sec)
24 x 24	0.39	0.42	0.45	0.46	0.44	0.432

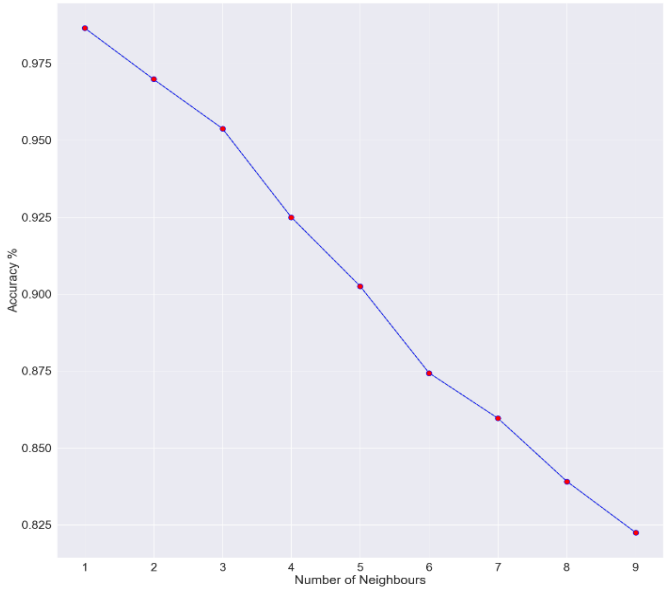

Figure 10: Accuracy of 8x8 pixel per cell with k values.

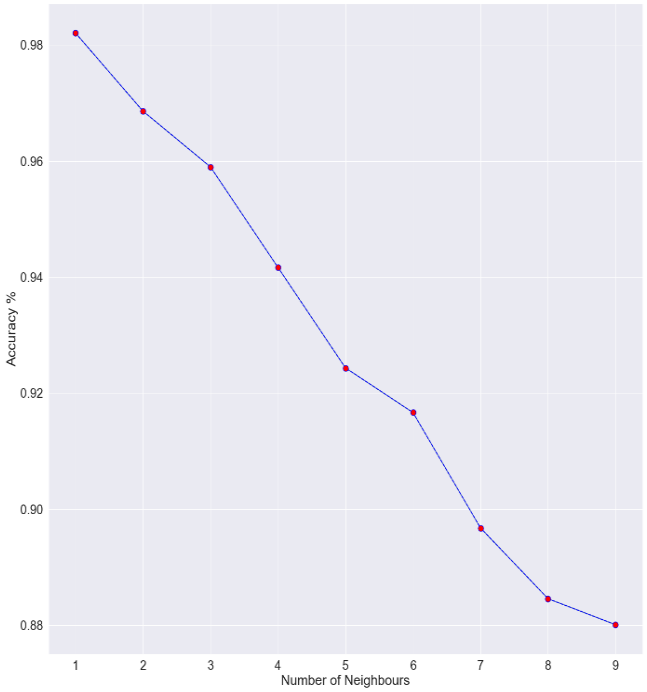

Figure 11: Accuracy of 16x16 pixel per cell with k values.

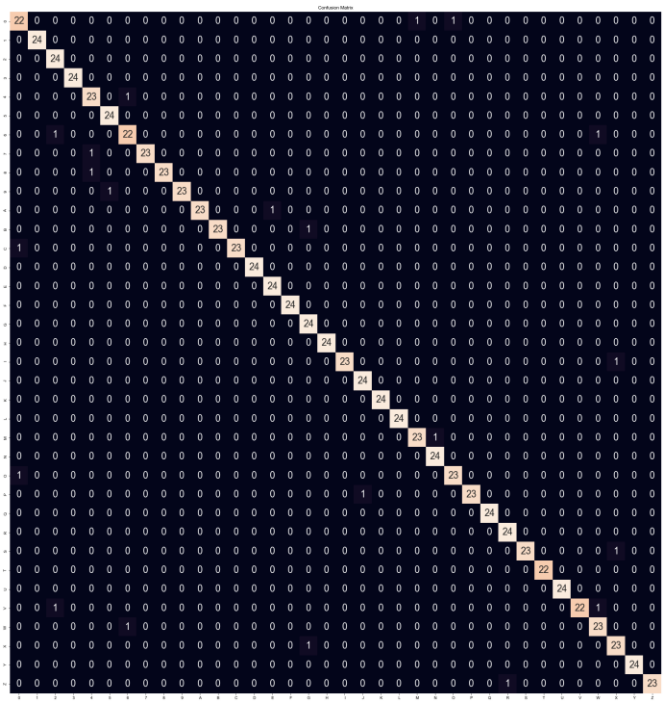

Figure 12: Confusion matrix results.

Table 5: Performance of developed system with HOG parameters (pixel per cell of 24x24) with K-nearest neighbour, *k* = 1 on Massey Dataset.

Sign	Test Sample	Correctly Classified	Wrongly Classified	Accuracy (%)
0	24	22	2	91.7
1	24	24	0	100
2	24	24	0	100
3	24	24	0	100
4	24	23	1	95.8
5	24	24	0	100
6	24	22	2	91.7
7	24	23	1	95.8
8	24	23	1	95.8
9	24	23	1	95.8
A	24	23	1	95.8
B	24	23	1	95.8
C	24	23	1	95.8
D	24	24	0	100
E	24	24	0	100
F	24	24	0	100
G	24	24	0	100
H	24	24	0	100
I	24	23	1	95.8
J	24	24	0	100
K	24	24	0	100
L	24	24	0	100
M	24	23	1	95.8
N	24	24	0	100
O	24	23	1	95.8
P	24	23	1	95.8
Q	24	24	0	100
R	24	24	0	100
S	24	23	1	95.8
T	24	22	0	100
U	24	24	0	100
V	24	22	2	91.7
W	24	23	1	95.8
X	24	23	1	95.8
Y	24	24	0	100
Z	24	23	1	95.8
		Average		**97.6**

Table 6: Computational time of developed HOG with K- nearest neighbour of k = 1 on Massey dataset after five tries.

	Computational Time (CT)					
Pixel per cell	CT 1 (sec)	CT 2 (sec)	CT 3 (sec)	CT 4 (sec)	CT 5 (sec)	Average (sec)
24 x 24	0.34	0.4	0.42	0.37	0.43	0.392

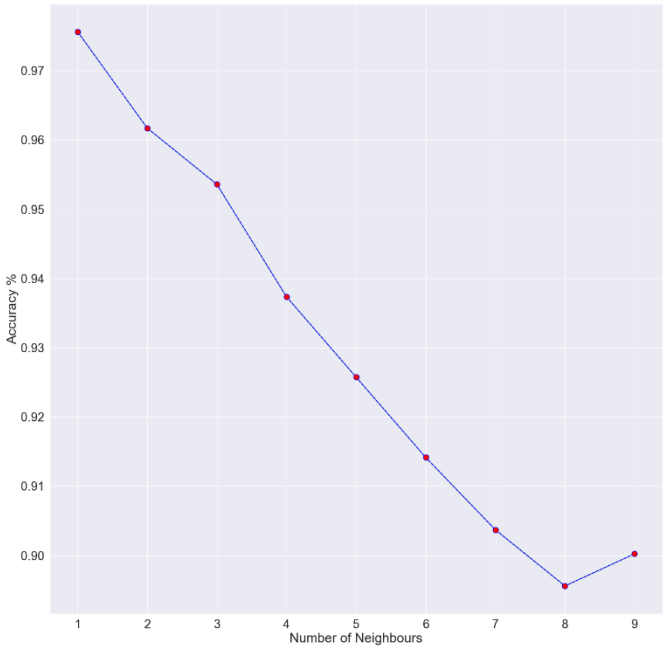

Figure 13: Accuracy of other *k* values.

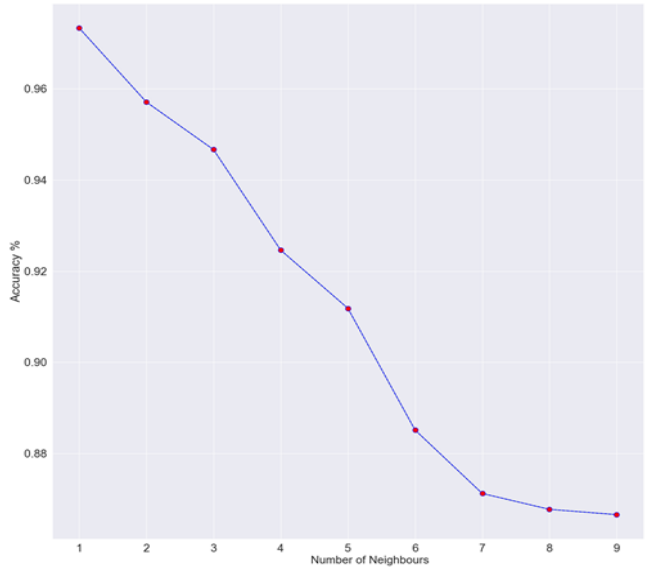

Figure 14: Accuracy of 8x8 pixel per cell with *k* values.

The prediction output of selected samples of testing images of the signs '8', '9', 'K', '4', 'M', 'V', 'U', 'C', 'E', 'F', and 'P' is shown in Figure 16.

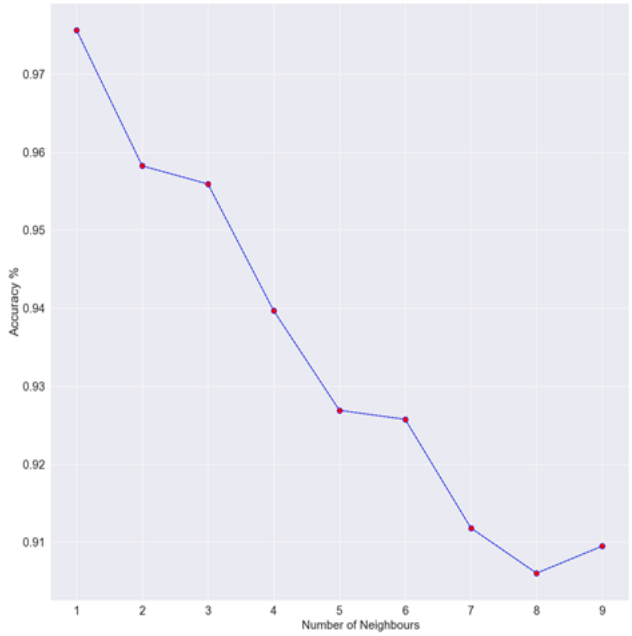

Figure 15: Accuracy of 16x16 pixel per cell with *k* values.

C. Comparison of Developed System with the Existing Systems

Table 7 shows the comparison of existing system with developed system in this research. The comparison is based on existing system with similar techniques or using same dataset to validate the performance of their system. This paper compared the performance of existing HOG based sign language recognition system with the developed HOG based system using same dataset or similar techniques.

It was observed that the developed ASLR system in this research outperformed Mahmud *et al.* (2019) system in terms of recognition accuracy and computational time based on number of features extracted with same dataset. Developed ASLR system also outperformed (Masood *et al.*, 2019; Das *et al.*, 2020) in term of average recognition accuracy with Massey dataset.

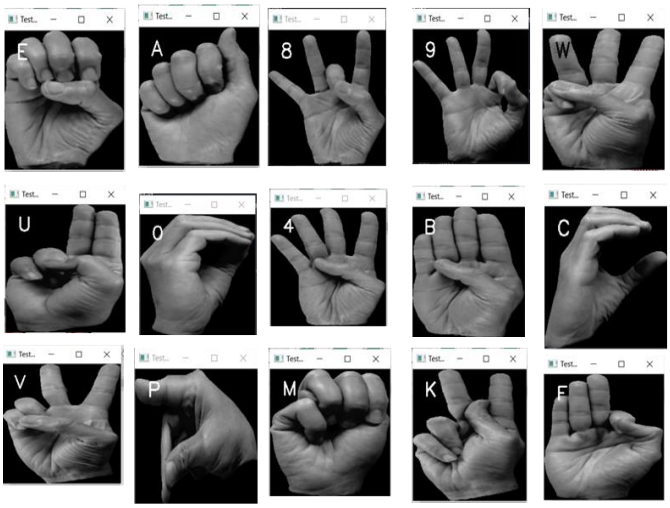

Figure 16: Samples of predicted sign image on testing dataset.

Table 7: Comparison of techniques with some literatures.

Author(s)	Dataset	Techniques	Accuracy %	CT Per Sign (Seconds)
Mahmud *et al*. (2019)	Kaggle Alphabets (A-Z)	HOG with K-NN	94.2	-
Masood *et al*. (2018)	Massey University Dataset Alphabet (A-Z and 0-9)	VGG16	96.0	-
Das *et al*. (2020)	Massey University Alphabet (A-Z)	CNN	94.3	-
Proposed system	Massey University Alphabet (A-Z and 0-9)	HOG parameters of 24 x 24 pixel per cell with K-NN	97.6	0.39
Proposed system	Kaggle Dataset Alphabet (A-Z)	HOG parameters of 24 x 24 pixel per cell with K-NN	99.0	0.43

VI. CONCLUSION

In this study, histogram of oriented gradient (HOG) based on parameters turning was developed for the extraction of feature useful for sign languages recognition. The technique uses HOG parameters of pixel per cell, cell per block, orientation bin and image size to reduce the feature vector length with K-nearest neighbour as classifier algorithm. This research was carried out on two publicly available datasets and the findings have indicated that the developed ASLR system with selected features achieved good recognition accuracy. The performance of the developed system achieved recognition accuracy and computation testing time of 99.0% and 0.43sec respectively on Kaggle dataset. Also, recognition accuracy and computational testing time of 97.3% and 0.39sec were achieved on Massey dataset. The experiments also revealed that higher accuracy was obtained with reduce K values of K-nearest neighbours compare to high K values. The developed system outperformed some of existing system with high accuracy and computational time. In future research, signs which are continuous and involves the movement of hands should be considered, the use of deep learning approach should be implemented to improve the performance of sign language recognition and furthermore, implementation of sign language recognition system on mobile phones should be considered for easy access and convenience.

REFERENCES

Adeyanju, I.A.; O. O. Bello and M. A. Adegboye. (2021). Machine learning methods for sign language recognition: A critical review and analysis. Intelligent Systems with Applications, 12, (2021): 1-36. https://doi.org/10.1016/j.iswa.2021.200056.

Akash, N. (2018). Image dataset for Alphabets in the American Sign Language. Available online at: https://www.kaggle.com/grassknoted/asl-alphabet. Accessed on May 2. 2021.

A**labi, S. (2019)** Design and Implementation of a Robotic Hand for Sign Language Communication. Unpublished Undergraduate Project, Department of Computer Engineering, Federal University Oye-Ekiti (FUOYE), Nigeria.

Barczak, A. L. C.; N. H. Reyes; M. Abastillas; A. Piccio and T. Susnjak. (2011). A New 2D Static Hand Gesture Colour Image Dataset for ASL Gestures. Res. Lett. Inf. Math. Sci.15: 12–20

Brahmankar, V.; N. Sharma; S. Agrawal; A. Saleem and P. Borse. (2021). Indian Sign Language Recognition Using Canny Edge Detection. International Journal of Advanced Trends in Computer Science and Engineering, 10 (3): 1576–1583. https://doi.org/10.30534/ijatcse/2021/131032021

Cheok, M. J.; Z. Omar and M. H. Jaward. (2019). A review of hand gesture and sign language recognition techniques. International Journal of Machine Learning and Cybernetics 10 (1): 131–153 https://doi.org/10.1007/s13042-017-0705-5

Dalal, N. and Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. Presented at the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 886–893, IEEE. https://doi.org/10.1109/CVPR.2005.177

Das, P.; T. Ahmed and Md. F Ali. (2020). Static Hand Gesture Recognition for American Sign Language using Deep Convolutional Neural Network, in: 2020 IEEE Region 10 Symposium (TENSYMP). Presented at the 2020 IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 1762–1765, IEEE. https://doi.org/10.1109/TENSYMP50017.2020.9230772

Dudhal, A.; H. Mathkar; A. Jain; O. Kadam and M. Shirole. (2019). Hybrid SIFT feature extraction approach for Indian sign language recognition system based on CNN. Presented at the 2018 International Conference on ISMAC in Computational Vision and Bio-Engineering, 727-738, Springer, Cham. https://doi.org/10.1007/978-3-030-00665-5_72

Hartanto, R.; A. Susanto and P. I. Santosa. (2014). Real time static hand gesture recognition system prototype for Indonesian sign language. Presented at the 6th International Conference on Information Technology and Electrical Engineering (ICITEE), Yogyakarta, Indonesia, 1-6, IEEE. https://doi.org/10.1109/ICITEED.2014.7007911.

Hasan, M. M.; M. Khaliluzzaman; S. A. Himel and R. T. Chowdhury. (2017). Hand sign language recognition for Bangla alphabet based on Freeman Chain Code and ANN, in:

4th International Conference on Advances in Electrical Engineering, ICAEE 2017. Institute of Electrical and Electronics Engineers Inc., 749–753. https://doi.org/10.1109/ICAEE.2017.8255454

Hassan, S. T.; J. A. Abolarinwa; C. O. Alenoghena; S. A Bala; M. David and P. Enenche. (2018). Intelligent Sign Language Recognition Using Image Processing Techniques: A Case of Hausa Sign Language. ATBU Journal of Science, Technology and Education 6 (2):

Huong, T. N. T.; T. V.; Huu; T. L. Xuan and S. V. Van. (2016). Static hand gesture recognition for Vietnamese sign language (VSL) using principal components analysis. Presented at International Conference on Computing, Management and Telecommunications (ComManTel), DaNang, Vietnam, 138–141, IEEE https://doi.org/10.1109/ComManTel.2015.7394275.

Jain, V.; A. Jain; A. Chauhan; S. S. Kotla and A. Gautam. (2021). American Sign Language recognition using Support Vector Machine and Convolutional Neural Network. International Journal of Information Technology 13 (3): https://doi.org/10.1007/s41870-021-00617-x

Jimoh, K.; T. M. Adepoju; A. A. Sobowale and O. A. Ayilara. (2018). Offline Gesture Recognition System for Yorùbá Numeral Counting. Asian Journal of Research in Computer Science 1(4): 1–11. 1–11. https://doi.org/10.9734/ajrcos/2018/v1i424753

Jimoh, K.O.; A. O. Ajayi and I. K. Ogundoyin. (2020). Template Matching Based Sign Language Recognition System for Android Devices. FUOYE Journal of Engineering and Technology, 5(1): 42-48.
https://doi.org/10.46792/fuoyejet.v5i1.465

Jin, C. M.; Z. Omar and M. H. Jaward. (2016). A mobile application of American sign language translation via image processing algorithms, in: Proceedings Presented at the 2016 IEEE Region 10 Symposium (TENSYMP), Bali, Indonesia, 104–109. https://doi.org/10.1109/TENCONSpring.2016.7519386

Kulkarni, V. S. and S. D. Lokhande. (2010). Appearance Based Recognition of American Sign Language Using Gesture Segmentation. International Journal on Computer Science and Engineering (IJCSE), 2 (3): 560–565.

Lawend, H. O.; A. M. Muad and A. Hussain, A. (2005). Robust Edge Detection Based on Canny Algorithm for Noisy Images. Journal of Theoretical & Applied Information Technology, 95(19): 5104- 5114.

Mahmud, I.; T. Tabassum; M. P. Uddin; E. Ali.; A. M. Nitu and M. I Afjal. (2019). Efficient Noise Reduction and HOG Feature Extraction for Sign Language Recognition, in: 2018 International Conference on Advancement in Electrical and Electronic Engineering, ICAEEE 2018. Institute of Electrical and Electronics Engineers Inc., 1–4. https://doi.org/10.1109/ICAEEE.2018.8642983

Maliki, R.; D. Alhaidar; K. Attallah; S. Alsalem; M. Morris and S. Tosunoglu (2017). Robotic Hands to Teach Sign Language 7. Presented at 30th Florida Conference on Recent Advances in Robotics, Boca Raton, Florida, 1-7.

Masood, S.; H. C Thuwal and A. Srivastava. (2018). American Sign Language Character Recognition Using Convolution Neural Network, in: Satapathy, S.C., Bhateja, V., Das, S. (Eds.), Smart Computing and Informatics, Smart Innovation, Systems and Technologies. Springer Singapore, Singapore, 403–412. https://doi.org/10.1007/978-981-10-5547-8_42

Mohammed, M. G. and Melhum A. I. (2020). Implementation of HOG Feature Extraction with Tuned Parameters for Human Face Detection. International Journal of Machine Learning and Computing, 10 (5): 654-661. https://doi.org/10.18178/ijmlc.2020.10.5.987

Shah, B. K.; V. Kedia; R. Raut; S. Ansari and A. Shroff. (2020). Evaluation and Comparative Study of Edge Detection Techniques. IOSR Journal of Computer Engineering (IOSR-JCE) 22 (5): 6–15.

Sharma, A.; A. Mittal; S. Singh and V. Awatramani. (2020). Hand Gesture Recognition using Image Processing and Feature Extraction Techniques. Procedia Computer Science 173:181–190 181–190. https://doi.org/10.1016/j.procs.2020.06.022

Tolentino, L. K. S.; R. O. Serfa Juan; A. C. Thio-ac; M.A. B. Pamahoy; J. R. R. Forteza and X. J. O. Garcia. (2019). Static sign language recognition using deep learning. International Journal of Machine Learning and Computing 9 (6): 821–827. https://doi.org/10.18178/ijmlc.2019.9.6.879.

Treat, S. (2016). Deaf Education. Available online at: https://prezi.com/ckdvqq0rv5cx/deaf-education/ (accessed 9.1.21).

Wadhawan, A. and Kumar, P. (2020). Deep learning-based sign language recognition system for static signs. Neural Computing and Applications 32 (12): 7957–7968. https://doi.org/10.1007/s00521-019-04691-y

World Health Organizations. (2021). Deafness and hearing loss - Key facts. Available online at: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss Accessed on May 12.2022.

Yasir, F.; P. W. C. C Prasad; A. Alsadoon and A. Elchouemi. (2016). SIFT based approach on Bangla sign language recognition. Presented at 2015 IEEE 8th international workshop on computational intelligence and applications (IWCIA), 35-39, IEEE.
https://doi.org/10.1109/IWCIA.2015.7449458