Gesture To Speech Translation

Vivek Vivian, Mohammad Metwally, Arslan Hameed, Raheel Ali and Jatin Anand
Embedded Systems, Uppsala University

***Abstract*—People with the inability to speak, use different modes to communicate with others, one of them being sign language.[1]. New technologies of Human Computer Interaction(HCI) are being developed to command robots. Since hand gestures are able to express enriched information, the hand gesture recognition is widely used in robot control, intelligent furniture and other aspects.[2] In this paper we describe the concept of landmark extraction using background subtraction followed by gesture recognition based on convolutional neural network which is then followed by Gesture Classification using SVM and this data is finally fed into a Kotlin server using TCP from which the Gesture gets converted to speech and emotions visually displayed on a Furhat social Robot. The results of this paper demonstrate the effectiveness of the proposed interactive method.**

***Index Terms*—Machine Learning, Hand Recognition, Convolution Neural Network(CNN), Image Classification, Data Classification, Background Subtraction, Landmark Extraction, Contours, Convex Hull, Convexity Defects, Support Vector Machine(SVM), Logistic Regression, Linear Discriminant Analysis, Decision Tree Classifier, GaussianNB, Furhat, Kotlin, Keras, OpenCV, Tensorflow, Scikit Learn, Socketserver, Socket Programming.**

I. Introduction

With the development in Artificial Intelligence, the interaction between human and machine is improving greatly. Among them, Hand Gesture recognition has gained alot of interest as the applications are endless. Since there are various hand gestures and enriched information contained in them, recognition of hand gesture has been greatly used in many fields, such as UAV, somatosensory game, sign language recognition and so on.[3][4]In order to improve the accuracy of recognition, many researchers have deployed methods such as HMM, Artificial Neural Networks, and Kinect platform. Effective algorithms for segmentation, classification, pattern matching and recognition have evolved.[5]
The Interaction sysytem in this paper is further divided into three stages as Landmark Extraction, Gesture Recognition and Responsive Behaviour.

II. System overview

We have divided the pipeline functions in three subsystems. Namely, Landmark extraction, Gesture recognition and Responsive behaviour.

Main function of sub-system 1 is to detect a hand and save the extracted landmarks of the hand in a .csv file. Several methods were tried and tested to fulfil this functionality. Several convolutional neural network models were also designed with different parameters(epochs and batch size of training) to find the best model suitable for our project specifications.

Sub-system 2's functionality is to recognise a gesture. This was fulfilled by training a SVC Machine Learning model using the landmarks dataset of all gestures for classification. Support Vector Machines(SVM's) is a technique that uses statistically methods to classify data. The objective of this algorithm is to find the optimal separating hyper plane between two classes by maximizing the margin between them [6]. Faria et al. [7],[8] used it to classify robotic soccer formations and the classification of facial expressions, Ke et al.[9] used it in the implementation of a real-time hand gesture recognition system for human robot interaction, Maldonado-Báscon[10] used it for the recognition of road-signs and Masaki et al used it in conjunction with SOM (Self-Organizing Map) for the automatic learning of a gesture recognition mode

The function of subsystem 3 is to generate an output via the furhat application in response to the input it receives from the subsystem 2. The connection between the two environments of kotlin and python is done via socket programming. 6 gestures are used to represent the 26 English alphabets to reduce the amount of training required by associating each letter with a unique gesture.

III. Sub-system 1: Landmark extraction

Several methods were tried and tested to achieve a successful sub-system 1. This system includes a landmark extraction program and a Convolutional Neural Network (CNN) machine learning model for gesture recognition.

A. Methodology

a) Background Subtraction: One of our approaches was using Background subtraction method to detect a hand on a part of the screen(or full screen). Background subtraction is a common and widely used technique for generating a foreground mask (namely, a binary image containing the pixels belonging to moving objects in the scene) by using static cameras.[11] On the detected hand, we find its contours followed by convexHull and Convexitydefects that help us find the x and y coordinates of the finger tips and the joints as shown in Figure 1. These are converted into landmarks further.

b) Haar-cascade: Another approach was to detect the hand using haarcascade method. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect

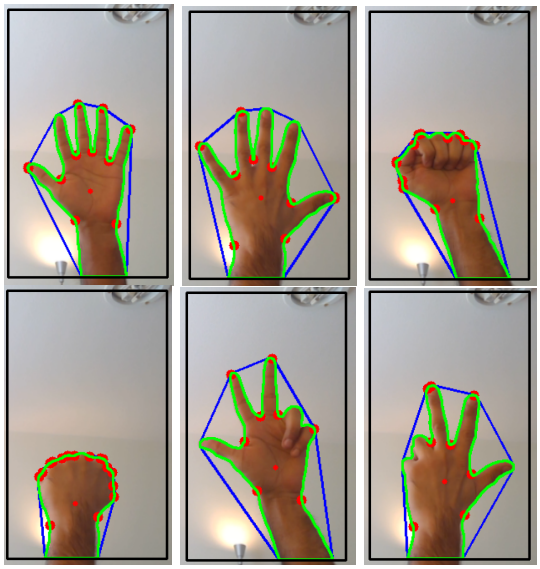

Figure 1. 6 Hand Gestures

objects in other images. Initially, the algorithm needs a lot of positive images (images of the hand) and negative images (images without a hand) to train the classifier. Then we need to extract features from it. For this, Haar features shown in figure 2 are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle [12]

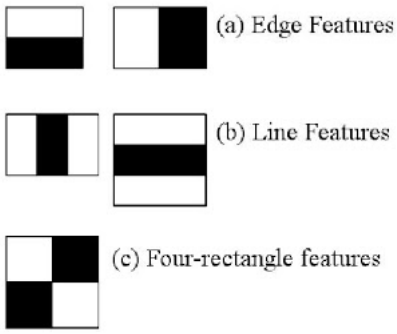

Figure 2. Haar Features.

c) CNN model: A machine learning CNN model was also designed using frames from video dataset of 6 hand gestures of 14 people for training the model. CNN is used to do images recognition, images classifications. Objects detection, recognition faces etc. Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernals), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1. [13][14]

B. Evaluations and Discussion

a) Background subtraction: In our first approach, the foreground mask(thresholded image) as shown in Figure 3 is the hand in our case. *cv2.findContours()* is used to find the contours points along with hull and defect points found using *cv2.convexHull()* and *cv2.convexityDefects()*. Hull points are the points on the finger tips of the hand and defects points are the intersections of the hand.

These opencv functions help us plot the coordinates on a moving hand as shown in 1. To plot the centroid of the hand *cv2.moments()* was used with x and y coordinates as follows:

$$cx = int(\frac{M['m10']}{M['m00']})$$

$$cy = int(\frac{M['m01']}{M['m00']})$$

These coordinates are drawn on each frame of the webcam live stream as shown in 1.
The landmarks of the last frame are saved in a .csv file and used as sub-system 2's input. Pressing 'q' will save the frame landmarks to a csv file. Pressing 'r' will reset the background mask and make the current frame as the background mask.

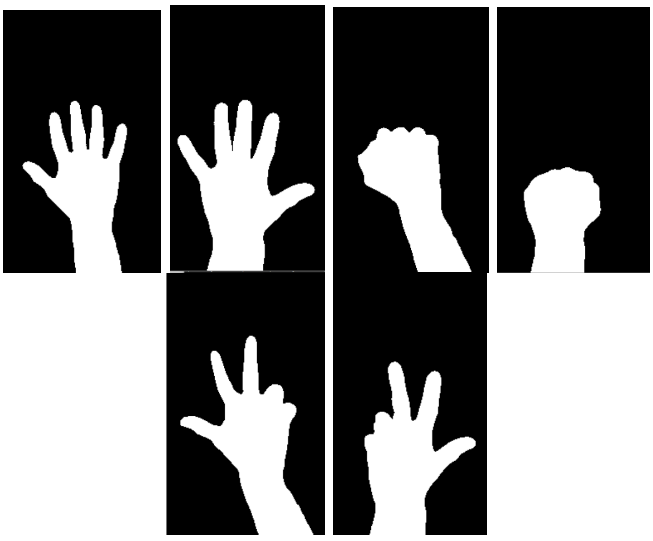

Figure 3. Thresholded images after Background Subtraction

b) Haar-cascade: Haar cascade classifier allows better detection of objects with static features such as balloons, boxes, faces, eyes, mounts, noise, etc. But a hand in motion has few static features because its shape and fingers can change as well as its orientations. So, Haar cascade classifier allows detection of only basic hand poses, which are not suitable to recognize a hand in motion with a long mount of different poses [15]. There was two methods used for training the classifier:

- A GUI developed by Amin Ahmadi, "Cascade Trainer GUI" that is using OpenCV [16].

- Manually training it using OpenCV and a dedicated Linux VPS.

By using the first method the training took quite some time because the computer that was used did not have enough processing power. The second method was more faster but a lot of issues was faced with setting the parameters that should be used in the function "**opencv_creatsamples**" and "**opencv_traincascade**", unlike the GUI most of these parameters were preset. Other issues that was faced was that, the latest version of opencv did not have these functions, so an older version (2.4) was used. There was also a lot of things about training the cascade that was unclear like how many positive and negative images should be used and how many training stages should be performed to get an acceptable classifier and there was not enough of information on the web of how to train the classifier for hands detection. However, with all of these challenges and after a prolonged research, tests were performed successfully on the open palm gesture. The optimum training was performed using 2000 negative images and 1000 positive images with an acceptable acceptance ratio of 0.0001 for 10 training stages and in 32 minutes to be precise. The result for detecting an open palm is shown in figure 4.

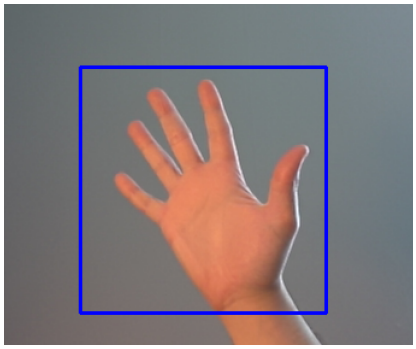

Figure 4. Detection of open palm gesture using Haar Classifier.

When the testing was carried out with different gestures the haar cascade proved to fail. Mostly noticed, It did not distinguish between the palm and dorsal. As mentioned in the beginning, the haar cascade is not a robust method for hand detection. We believe combined with different methods, the haar cascade classifier can prove its efficiency in detecting the hand.

c) CNN model: While designing the CNN model, we learnT how different parameters of the function are used and applied that knowledge to make several models to test our data. Models with different number of iterations and batch sizes were made to test the accuracy of our model. Initially, having iterations of 5 and batch size as 64 did not give a perfect result. Other models were made with various values and we came to a conclusion of having more epochs and a smaller batch size that increased the accuracy to 99.96%. We tested this by giving random hand gestures (which were not included in the training dataset) as input and the output of the model would give the detected hand gesture label.

IV. Sub-system 2: Gesture recognition

Overall Nine different methods of data classification was used and it's performance and accuracy was checked. Here we decided to go ahead with the multiclass Support Vector Machine to distinguish between the 6 different hand gestures we have using the csv file which contains all the landmarks points of a hand. The coordinates or landmarks of the hand have been generated into a csv. These landmarks are then fed into the Support Vector Machine as an input which will train and classify the same so that the next time when a test data is given it would get classified accordingly[17]. Six gestures representing the gestures open palm, open dorsal, open fist, closed fist, three fingers palm and three fingers dorsal are trained and recognized as shown below in the figure.

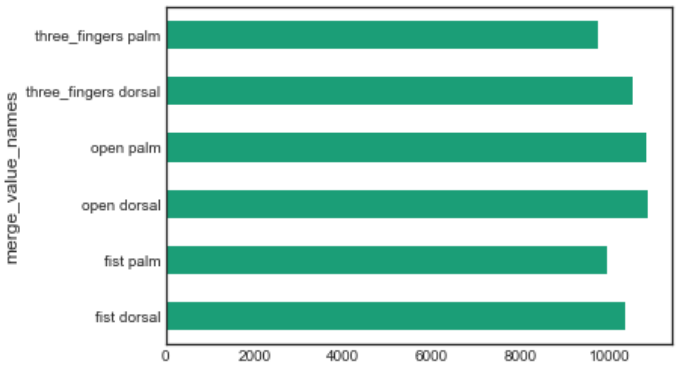

Figure 5. Total Number of Data Sets for each Gesture

A. Methodology

After the Landmarks are generated, it is feasible to classify the 6 hand gestures. Eight different methods were implemented in order to achieve Data Classification.

1) SVM RBF Model
2) k-Nearest Neighbors (k-NN)
3) Naive Bayes
4) Linear SVM
5) Decision Tree
6) Random Forest
7) Quadratic Discriminant Analysis
8) Multi-layer Perceptron

Out of these above different models which were tested, SVM using RBF model was selected. Linear Support Vector Classification is a supervised learning model and has been used here to distinguish between the different gestures. Similar to SVC with parameter kernel='linear', but implemented in terms of liblinear rather than libsvm, so it has more flexibility in the choice of penalties and loss functions and should scale better to large numbers of samples. This class supports both dense and sparse input and the multiclass support is handled according to a one-vs-the-rest scheme.[18] It does this by drawing a

hyperplane which has the largest distance to functional margin is a good separation, since in general the larger the margin the lower the generalization error of the classifier.[19] The dataset used here for training and testing the model consisted of 62219 rows of data which consisted of 82 different columns which had (X,Y) co-ordinates of different hands and gestures. The data was cleaned and scaled in order to train the model. The targets were the different gestures and the data consisted of the various landmarks. The data was split by 30% which are 43553 samples in the training set and 18666 samples in the test set. The parameters used for the linear model were ('gamma': 1.0, 'C': 100.0)

B. Evaluations and Discussion

Here we see the Confusion matrix of all the 8 systems that were trained with. The accuracy, Precision and Recall score of each model is as shown.

Score Table			
Classification Model	Accuracy	Precision	Recall Score
SVC using RBF	100%	100%	100%
Logistic Regression	100%	100%	100%
KNeighbors Classifier	100%	100%	100%
GaussianNB	82.70%	73.85%	82.70%
Decision Tree Classifier	99.95%	99.95%	99.95%
Random Forest Classifier	100%	100%	100%
Quadratic Discriminant Analysis	100%	100%	100%
MLP Classifier	100%	100%	100%

Table I
SHOWS ACCURACY, PRECISION & RECALL SCORE

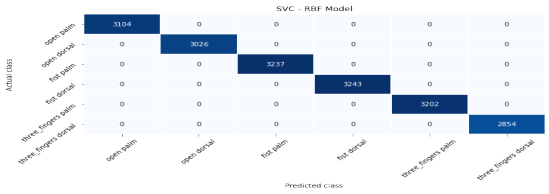

Figure 6. SVC using RBF

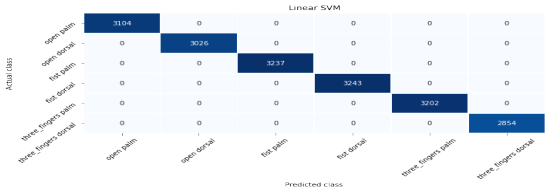

Figure 7. SVC using Linear

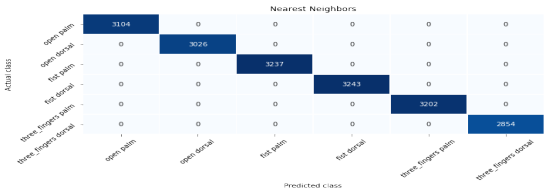

Figure 8. K Nearest Neighbors

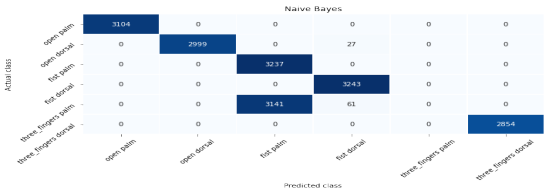

Figure 9. GaussianNB

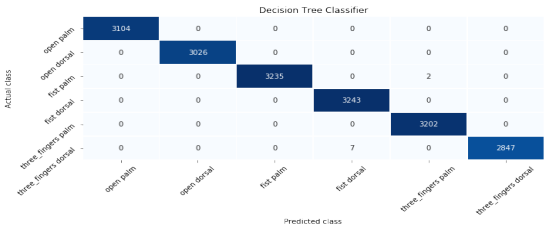

Figure 10. Decision Tree Classifier

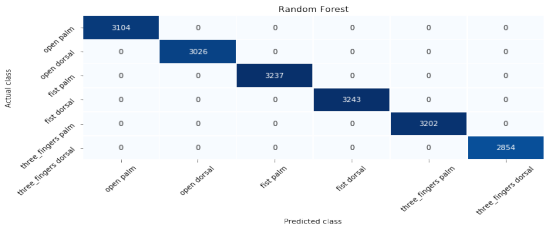

Figure 11. Random Forest Classifier

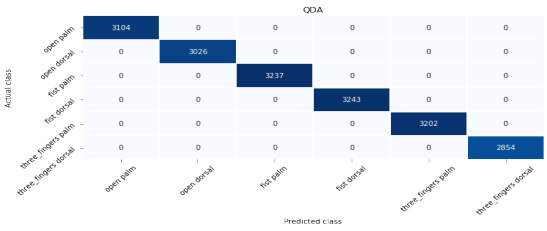

Figure 12. Quadratic Discriminant Analysis

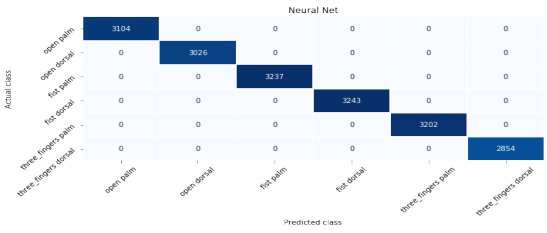

Figure 13. Multi-layer Perceptron

Here we see that most of our models yielded a 100% accuracy except GaussianNB and Decision Tree Classifier.

V. Sub-system 3: Responsive behaviour

A. Methodology

We have used Furhat SDK [20] for the functionality of subsystem 3. All of Furhat's components interact together using an elegant and extendable events system. The function of this subsystem is to take the detected gesture as an input from subsystem 2 and provide a letter associated with the detected gesture as an output. The output is in the form of voice and facial expressions through the Furhat application. First step in this sub system was to connect the subsystem 2 and 3 which are basically two different environments (i.e python and kotlin)[21]. This could be achieved by socket programming in Python. Socket programming is basically a method used widely in the computer science field for communication between two nodes connected to a network. A good example of this is the internet we use daily which we connect to using sockets. We tried to host a server and send the results from subsystem 2 to subsystem 3 by making the subsystem 2 as a host and subsystem 3 as client. However there were frequent disconnections between the server and client and most of the times the messages were not transmitted or delayed leading to error in the results or slow responses from the furhat application.

Our model detects 6 different hand gestures and has been trained on these gestures to provide an output to subsystem 3 in the form of an index value of a six element array where each element represents one gesture. We wanted our system to be able to communicate to the user all 26 letters of the English alphabet using only these six gestures on which we trained our model. In order to achieve this we use a delay function in subsystem 3 with a combination of the six gestures representing the remaining 20 alphabets. Subsystem 3 after getting the input from subsystem 2 saves the index value as input1 and waits for some seconds and reads the output from the subsystem 2 again saving it in input 2. If both values are equal the output is from the first 6 letters of the English alphabet and if the the value is different it outputs the letter based on the combination of the two index values received in these few seconds which is shown in the table 2. After the robot welcomes the user, the user can interact via gestures and the robot outputs the letter associated with that gesture by pronouncing it and also displaying it in the application as shown in figure 12. [20] [21]

Combination Table		
Input 1	Input 2	Letter
0	0	A
1	1	B
2	2	C
2	2	D
0	1	E
0	2	H
1	0	L
1	2	M
4	0	Yes
5	0	No

Table II
COMBINATIONS FOR DIFFERENT LETTERS ACCORDING TO THE 2 INPUTS RECEIVED

Gesture Table	
Gesture name	Index
open palm	0
open dorsal	1
fist palm	2
fist dorsal	3
three fingers palm	4
three fingers dorsal	5

Table III
TAB:GESTURE TABLE

When the user enters the interface, the robot welcomes the user with a smiling gesture as shown in the figure 13.

Figure 14. Robot

B. Evaluations and Discussion

Since the sockets for python were not working properly and giving satisfactory results hence affecting the overall efficiency of our system. We used two other methods to verify that the subsystem 3 is working or not. Instead of taking the input index from the subsystem 3 via the server we take the voice input via furhat application and through the keyboard directly from IntelliJ IDEA. Subsystem 3 correctly identified all the gestures and combinations given by user.

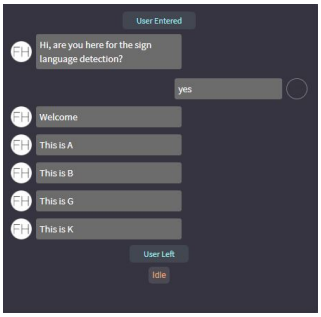

Figure 15. Furhat communication

VI. General Discussion

A. Challenges faced

Choosing the suitable algorithm for detecting the hands was one of the biggest challenges we faced. A lot of testing was also required to make an accurate ML model that could recognise hand gestures. Another challenge was connecting the sub-system 3 with sub-system 2 as the sockets for python did not give satisfactory results.

B. Ethical issues

Face and hand detection invoke a lot of criticism regarding ethics of it's appllications. Since this technology is not 100% and due to this the technology itself is discriminative as the accruracy can be as low as around 70%. Due to this most of the recognition technologies these days face a lot of privacy and government issues regarding to racial discriminatory laws. The government due to this have made policies and regulatory laws to avoid such cases. Hence in order to implement such technology it is good to Provide notice in advance so the customer knows what it is capable of and also Ask for consent as this can protect you from going against the law.[22]

VII. Conclusion

Hand Gesture Recognition is one of the most difficult problems and lots of development is taking place in this area. This paper discusses about various techniques used for Feature Extraction, Data classification and the responsive behaviour Feature. Object detection uses background subtraction followed by Landmark Extraction which was done by plotting the contours and its convex hull and defects. Another CNN model recognises a hand gesture with good results as it was trained with about 17,000 images of different hand gestures. In Data Classification we implemented 9 different techniques and then selected one which is Support Vector Machine using a Linear model as it is well suited for large data sets and multiple features.

Figure 16. Results of Predictions using CNN

The model has 100% accuracy in it's prediction of the gestures. This model then returns an array which contains a number that represents the gesture for example 0 - represents Open Palm, 1 - represents Open Dorsal and so on. This array is the fed to the Furhat robot which is the Responsive Behaviour sub system and once receiving this array it deciphers the letter based on the value present in the array. This is then converted to speech by the robot and it speaks to the user and says the letter out. For example in sub system 1 a open Palm gesture was made and these landmarks go to the subsystem 2 and the model predicts the gesture to be Open Palm, it would return an array value = 0 which is then fed to the Robot and the Robot understands that if the value is 0 the corresponding letter is A which it then recites to the user.

This way we have trained the two subsystems to detect and predict six different gestures which were Open Palm, Open Dorsal, Fist Palm, Fist Dorsal, Three Fingers Palm, and Three Fingers Dorsal which correspond to values 0,1,2,3,4 and 5 in the sub system 2 and further to the letters A,B,C,D,E, and F. For Specialization we have added few more letters like H,L,M which takes a combination of two gestures as in if the user shows Open Palm followed by Open dorsal the system returns an array [0,1] which the robot then decodes as H. Similarly all the 24 alphabets in english can be implemented which could then be used full time to convert gestures to speech. In future, we will try to build a real-time application based on the method.

References

[1] S. Rajaganapathy, B. Aravind, B. Keerthana and M. Sivagami, 'Conversation of sign language to speech with human gestures', *Procedia Computer Science*, vol. 50, Dec. 2015. DOI: 10.1016/j.procs.2015.04.004.

[2] J. Sun, T. Ji, S. Zhang, J. Yang and G. Ji, 'Research on the hand gesture recognition based on deep learning', in *2018 12th International Symposium on Antennas, Propagation and EM Theory (ISAPE)*, 2018, pp. 1–4.

[3] L.-c. Z. Li-juan Sun and C. long Guo, 'Technologies of hand gesture recognition based on vision [j], vol. 18', in. 2008, pp. 214–216.

[4] J.-h. C. Jing-guo Yi and X. shu Ku, 'Review of gestures recognition based on vision [j], vol. 43', in. 2016, pp. 103–108.

[5] C. S. C. Paranjape Ketki Vijay Naphade Nilakshi Suhas and D. K. Dhananjay., 'Recent developments in sign language recognition: A review, vol. 1', in. 2012, pp. 21–26.

[6] A. Ben-Hur and J. Weston, *A User's Guide to Support Vector Machines, in Data Mining Techniques for the Life Sciences*. 2008.

[7] B. Faria, N. Lau and L. Reis, 'Classification of facial expressions using data mining and machine learning algorithms', pp. 197–202, Jan. 2009.

[8] B. M. Faria, L. P. Reis, N. Lau and G. Castillo, 'Machine learning algorithms applied to the classification of robotic soccer formations and opponent teams', in *2010 IEEE Conference on Cybernetics and Intelligent Systems*, 2010, pp. 344–349.

[9] e. a. Ke W., *Real-Time Hand Gesture Recognition for Service Robot*. 2010, pp. 976–979.

[10] S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno and F. Lopez-Ferreras, 'Road-sign detection and recognition based on support vector machines', *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, 2007.

[11] OpenCV. (). Opencv background subtraction tutorials, [Online]. Available: https://docs.opencv.org/master/d1/dc5/tutorial_background_subtraction.html.

[12] ——, (). Cascade classifier, [Online]. Available: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html.

[13] R. Prabhu. (). Understanding of convolutional neural network (cnn) — deep learning, [Online]. Available: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148.

[14] A. Pande. (). A beginner's guide to understanding convolutional neural networks, [Online]. Available: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/.

[15] D. Núñez-Fernández, 'Development of a hand gesture based control interface using deep learning', in. Aug. 2019, ISBN: 978-3-030-46139-3. DOI: 10.1007/978-3-030-46140-9_14.

[16] A. Ahmadi. (). Cascade trainer gui, [Online]. Available: https://amin-ahmadi.com/cascade-trainer-gui/.

[17] P. H. R.O. Duda and D. Stork., *Pattern Classification*. Wiley-interscience, 2001.

[18] S. Learn. (). Scikit learn classification models, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC.

[19] R. T. T. Hastie and J. Friedman, *The elements of Statistical Learning*, pp. 371–409, 201.

[20] Furhat. (). Furhat developer docs, [Online]. Available: https://docs.furhat.io/.

[21] Kotlin. (). Kotlin documentation, [Online]. Available: https://kotlinlang.org/docs/reference/kotlin-doc.html.

[22] R. B. U. S. S. of Missouri. (). Blunt, schatz introduce bipartisan commercial facial recognition privacy act.