Hand Gesture Recognition using Gaussian Threshold and different SVM kernels

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Information Security

Submitted by

Shifali Sharma
(801533013)

Under the supervision of

Mr. Shatrughan Modi
Lecturer

Dr. Jhilik Bhattacharya
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA - 147004

July 2017

Certificate

I hereby certify that the work, which is being presented in the thesis, entitled *Hand Gesture Recognition using Gaussian Threshold and different SVM kernels*, in partial fulfillment of the requirements for the award of the degree of *Masters of Engineering* in *Information Security* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Mr. Shatrughan Modi* and *Dr. Jhilik Bhattacharya* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Shifali Sharma
(801533013)

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Mr. Shatrughan Modi
Lecturer
Computer Science and Engineering Department
Thapar University

Dr. Jhilik Bhattacharya
Assistant Professor
Computer Science and Engineering Department
Thapar University

i

Acknowledgement

First, I would like to express my deep gratitude to my supervisor **Mr. Shatrughan Modi and Dr. Jhilik Bhattacharya** for their invaluable advice and encouragement at every step of my M.E. program. Without their unfailing support and belief in me, this thesis would not have been possible. Their contribution to this thesis goes well beyond their role as an academic supervisor and includes constant support on a personal level without which this journey may never have been completed. And for this, I am truly grateful. They are great mentor for my life as well.

I would like to express my gratitude to **Dr Prashant Singh Rana** for his constant motivation and encouragement.

Finally, I would like to express my sincere and deep gratitude to my parents and family members for their love, encouragement, care and support. Finally thanks to my husband Rahul Sharma for having faith on me and supporting me at every step, without his support, I could not have completed my M.E. program.

Shifali Sharma

Abstract

Hands play an important part in expressing one's actions and ideas thus Hand Gesture Recognition (HGR) is very significant in computer vision based gesture recognition for Human Computer Interaction (HCI). In our work, the dataset has been generated for five static hand gestures (Close Hand, Open Hand, Victory Hand, Thumb Down and Thumb Up), by making videos of 10 different users doing the gestures with all possible variations resulting in total 16,240 images extracted from videos. The objective of this thesis is to develop an efficient hand gesture method using image processing and machine learning algorithms and methods.

Background Subtraction is important for extracting hands from a static background. After extraction of the hand, the image processing algorithms like Bilateral Filter and Median Blur have been used for smoothing the images. Gaussian Threshold removes the most of the noise in the images and give the clear outline of the hand gesture. Convex hull points are calculated for each hand gesture and number of points in convex hull are taken as feature.

After pre-processing the images, the extracted features are given as input to the Support Vector Machine classifier. Comparison of the performance of different SVM kernels i.e. rbfdot, vanilladot, polydot, tanhdot, laplacedot and besseldot, for HGR is done. The accuracy achieved with different SVM kernels varied from 24.17% to 85.07% with training-testing ratio of 70-30% for 16,240 entries in the dataset. The 10-fold cross validation is performed to prove the robustness of the kernel with SVM.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mechanical devices such as Keyboards, Mouse, Joysticks etc can be a hindrance to Human Computer Interaction (HCI). The more natural the interaction is, more easily the interaction can take place between the humans and the computers. In recent years, gesture recognition has gained popularity in research field. Gestures can refer to a particular movement or a posture made using body parts (head, face, hands, arms etc). One important research field direction in gesture recognition is the hand gesture recognition, in which gestures are made using hands.

There are two classes of hand gestures: Static hand gesture and Dynamic hand gesture. Static hand gesture can be represented using a single image with static configuration and location of hand, without any movements (for example a fist) whereas Dynamic hand gesture is represented using a series of frames of static hand gestures connected by a continuous movement in some specified time-span (for example waving a hand).

Hand gestures are always needed to communicate. Also, in some situations the preferred way is the silent communication for example in an operation theatre the doctor can use hand gestures for communicating with the nurse for any assistance.

Gestures are considered the most natural way for communication and its applications are not limited to HCI. Hand Gestures can be used along with some real time algorithm for interacting with video game. A system can be created for controlling an industrial robot using hand gestures. For better two-way communication with deaf and dumb people a system can be developed which can take hand gestures as an input and give the meaning of the hand gesture in speech format. Human hands have many connected parts and joints, thus making it an articulated complex structure with roughly 27 degree of freedom when in motion [2]. Also, Different humans have different hand structures varying in shape and size. The problem with gesture recognition is how to make computer understand gestures. Extra hardware like sensors or glove based devices maybe used to transmit hand

configuration and movement data to the computer. These extra hardware do make the process of gesture recognition easier but these devices can be expensive and cumbersome in nature.

In recent times, Computer-vision based gesture recognition techniques have evolved for overcoming these restrictions. Computer-Vision techniques involve capturing gestures using one or more cameras and then recognising them. They can provide more natural communication between human and computer without any external dedicated hardware, which makes them more feasible. They are considered non-intrusive and thus can be used for real-time applications. Earlier, gesture recognition using Computer-Vision techniques required colored gloves or markers for making image processing easier. Nowadays, the focus is on tracking the bare hand and recognising the gestures, after extracting features, using different techniques like image processing, machine-learning models etc.

Fast recognition of the hand gestures is important for real-time applications. A real-time method should be, such that it can be used for robust and fast hand tracking even in complex background, motion blur and skin color variation. The gestures recognised by the method can be used for functional inputs and can interface with various applications and games.

1.1 Thresholding

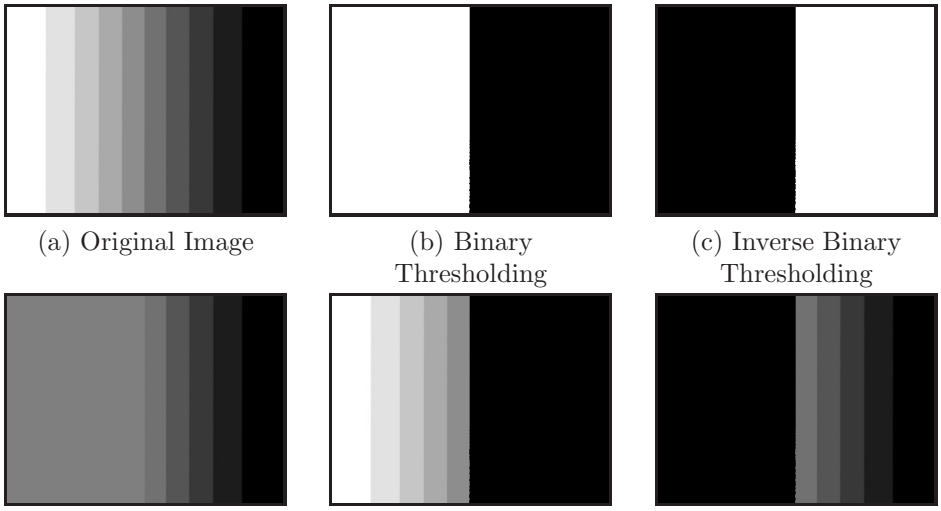
Thresholding can be defined as a simple method of image segmentation using which we can create a binary image from a grayscale image. In the simplest thresholding the main objective is to select a constant, such that any pixel value in the image less than that constant is changed to a black pixel value and any pixel value in the image greater than that constant is changed to a white pixel value.

Thresholding can be of two types:

- Simple Thresholding: In simple thresholding, we set a threshold value, if the value of the image pixel is less than the threshold value the pixel value is changed to black (or white) and if the value of the image pixel is grater than the threshold value the pixel value is changed to white (or black). Various parameters which are considered while simple thresholding are:
  - Thresholding can be done on grayscale images only. So the image should be converted to a grayscale image before applying thresholding.

- A threshold value must be selected on the basis of which each pixel value will be classified.
- A maximum value which is to be replaced with the pixel value when the pixel value is greater (or maybe less in some cases) than the threshold value.
- Different styles are provided by OpenCV for thresholding which are:
    * THRESH_BINARY: Binary Thresholding is the simplest of the thresholding styles. The pixel value is changed to the maximum value when the original pixel value is greater than the threshold value. In case the original pixel value is less than the the threshold value it is changed to zero(black).
    * THRESH_BINARY_INV: In Inverse Binary Thresholding the pixel value is changed to the maximum value when the original pixel value is less the threshold value and zero if the original pixel value is greater than the threshold value. We can say that Inverse Binary Thresholding is the opposite case of the binary thresholding.
    * THRESH_TRUNC: In Truncate Thresholding the original pixel value is replaced with the threshold value when the original pixel value is greater than the threshold value. In case it is less than the threshold value, the pixel value is not changed and remains as it is. The maximum value parameter is not considered in this style of thresholding.
    * THRESH_TOZERO: In Threshold to Zero style of thresholding, the original pixel value is not changed to any other value when the original pixel value is greater than the threshold value. If the original pixel value is less than the threshold value it is changed to zero. The maximum value parameter is not considered in this style of thresholding.
    * THRESH_TOZERO_INV: In Inverted Threshold to Zero, the original pixel value is not changed to any other value when the original pixel value is less than the threshold value. If the original pixel value is greater than the threshold value it is changed to zero. The maximum value parameter is not considered in this style of thresholding.

  Fig 1.1 shows the original image and the results of different styles of thresholding an image.



(a) Original Image (b) Binary Thresholding (c) Inverse Binary Thresholding

(d) Truncate Thresholding (e) Threshold to Zero (f) Inverted Threshold to Zero

Figure 1.1: Results of different styles of thresholding

– Adaptive Thresholding: In adaptive thresholding, the value of the threshold depends upon the neighboring pixel values of the original pixel and not on the user defined threshold value as in the case of simple thresholding. For calculating the threshold T(a,b), which is the threshold value at image pixel location (a,b), the following steps can be followed:

  * A region of q × q (q is user defined) size, around the original pixel location is chosen.

  * After selecting the q × q region its weighted average is calculated. There are two options for calculating this weighted average:

    · Mean Weighted Average: In this, adaptive thresholding is done using the mean weighted average

    · Gaussian Weighted Average: In this, adaptive thresholding is done using the gaussian weighted average.

  * The final step is to calculate the Threshold T(a,b). T(a,b) is calculated as follows:

$$T(a,b) = W(a,b) - c$$

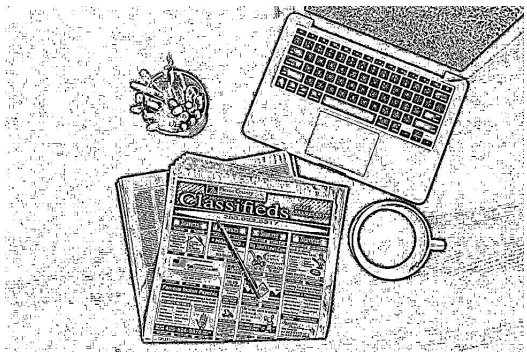
    where W(a,b) is the weighted average calculated in the previous step and c is constant parameter. From the weighted average of a pixel location we subtracted a constant to get the threshold value of that pixel location.

The following parameters are considered for adaptive thresholding:

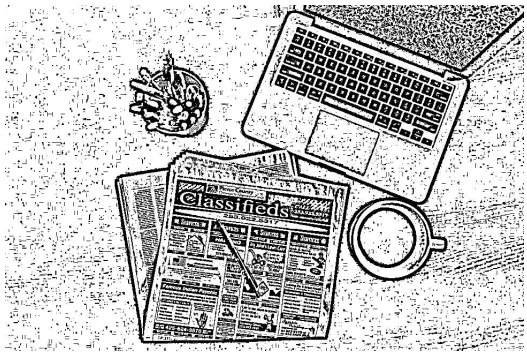
* Maximum value: A value which can be assigned to the pixel value after the application of adaptive thresholding
* Adaptive method: The method which is to be chosen for calculating weighted average. It can be either mean weighted average or the gaussian weighted average.
* Threshold type: The type of thresholding which is to be applied. It can be any one of the discussed in simple thresholding.
* Block Size: It is the q × q region which is to be selected around the original pixel location.
* Constant: A constant whose value is to be subtracted from the weighted average to get threshold value.

Fig 1.2 shows the original image and the results of the Gaussian Adaptive Threshold and Median Adaptive Threshold.

(a) Original Image



(b) Gaussian Adaptive Threshold



(c) Mean Adaptive Threshold

Figure 1.2: Results of Gaussian Adaptive Threshold and Mean Adaptive Threshold

1.2 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a machine learning algorithm is used for supervised learning. Though it can be used for both regression and classification problems, it is popularly used for classification data. This algorithm plots a point in n-dimensional space where n is the number of features in your data and each co-ordinate represents the value of the feature. The classification of the data by differentiating the two classes is done by finding the hyper plane as shown in Fig 1.3.

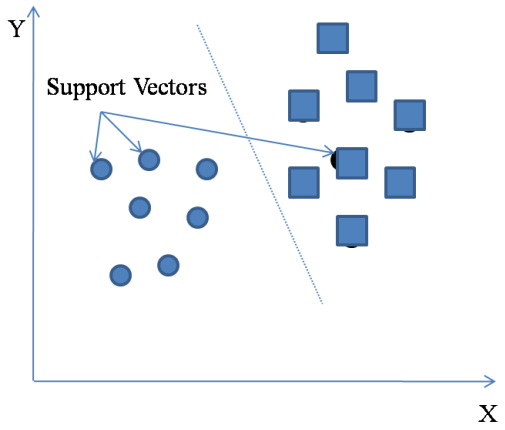


Figure 1.3: Classification of data done by finding Hyper Plane

Individual observations represented as a co-ordinate which segregates the classes from each other defines SVM. SVM thus segregates two classes in the best and easiest possible way since the two classes are distinctly defined.

1.2.1 Finding SVM for case in hand

Fig 1.4 shows the three possible frontiers for classification of data in hand. By finding the minimum distance between the closest support vector of any class from frontier the SVM's objective function can be interpreted. For example, red frontier is closest to blue circles and the closest blue circle is 2 units away from the frontier. When all the distances for all the frontiers is calculated, the frontier with the maximum distance from the closest support vector is chosen. In Fig 1.4 we can see that out of the three frontiers, green frontier is farthest at 11 units from the nearest support vector.

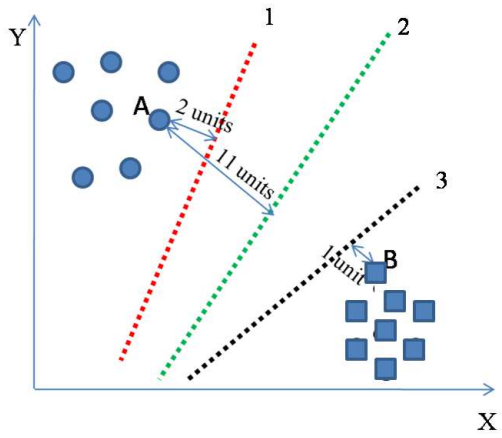


Figure 1.4: Possible frontiers for classification of data in hand

1.2.2 Case when clean frontier is not found which segregates the classes

Fig 1.5 shows the case in which we cannot see directly a straight line frontier in current plane which would segregate the given data into classes and do the job of SVM.

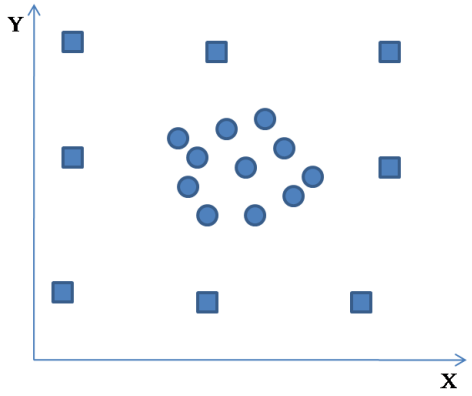


Figure 1.5: Case when clean frontier is not found

No straight line frontier is found directly in current plane which serves as a SVM, in these cases. The vectors are to be then considered in higher dimension different plane so that they get segregated from each other. Visually, such a transformation will result into a SVM shown in Fig 1.6

Each square vector is mapped on a different higher scale which segregates the two classes. Such transformations can be done using various algorithms.

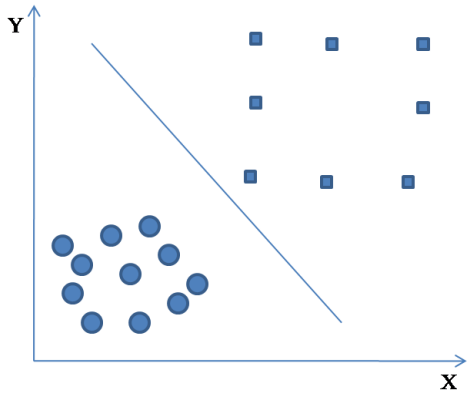


Figure 1.6: Vectors considered in Higher Dimension Plane

1.2.3 Working of SVM: Identifying right Hyper-Plane

Identifying the right hyper plane which segregates the two classes is explained using scenarios below.

- Scenario 1: Fig 1.7 shows the Scenario 1, with 3 hyper planes and two classes. In such case, the thumb rule to follow is selecting the hyper plane

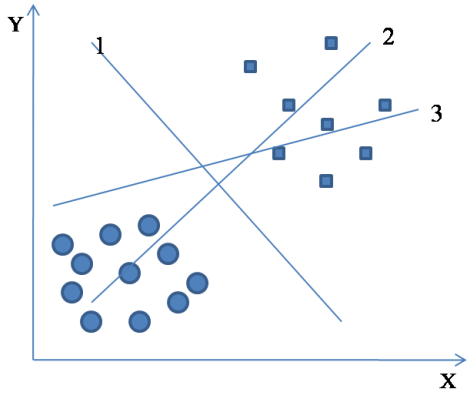


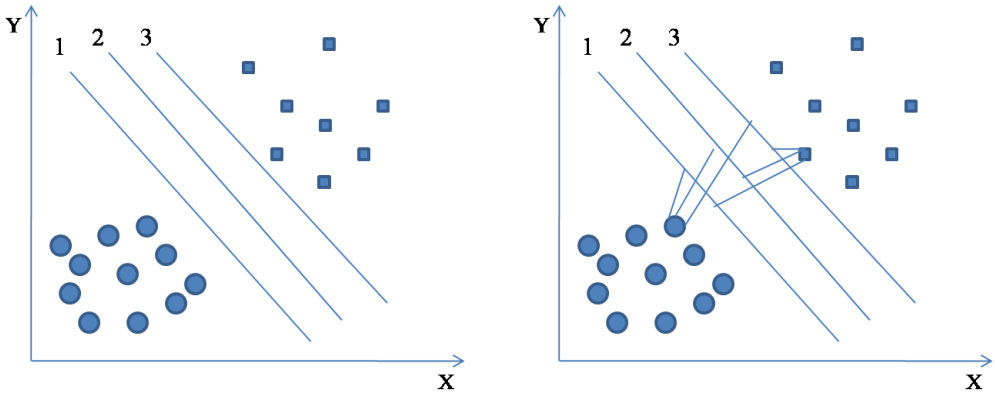
Figure 1.7: Scenario 1

  which segregates the classes in the best way. In our scenario, hyper plane 1 does the job and segregates the two classes better than hyper plane 2 and hyper plane 3.

- Scenario 2: Fig 1.8a shows the Scenario 2, in which all three hyper planes are segregating the classes in the best way possible.

  In such a scenario, maximizing the distance, called the margin, between a class vector and hyper plane. Margins of Scenario 2 are shown in Fig 1.8b.

  In Fig 1.8b we see that the margin for hyper plane 2 is highest as compared to both hyper plane 1 and 3. Hence, hyper plane 2 is the right choice. Selecting



(a) Without Margins (b) With Margins

Figure 1.8: Scenario 2

the hyper plane with highest margin also results in robustness since chances of miss-classification are higher in case of hyper plane having low margin.

- Scenario 3: Fig 1.9 shows the scenario 3 where hyper plane 1 has the higher margin than the hyper plane 2.

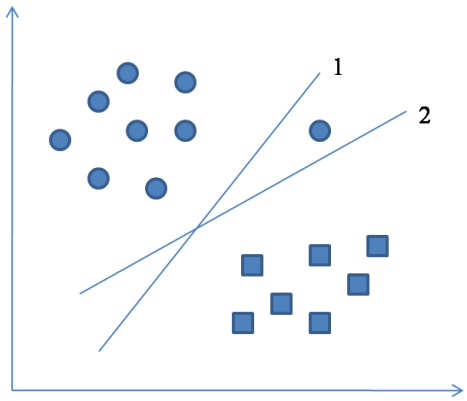
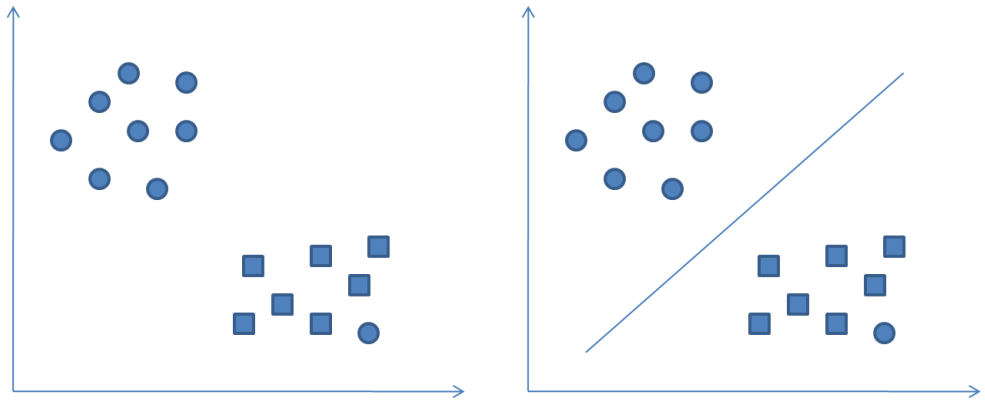


Figure 1.9: Scenario 3

Even if the hyper plane 1 has higher margin than hyper plane 2, the catch is that the SVM selects the hyper plane which segregates the classes accurately prior to selecting the hyper plane with maximum margin. Thus, hyper plane 1 has a miss classification case and hyper plane 2 classifies all the classes accurately. Hence, hyper plane 2 is the right hyper plane.

- Scenario 4: In Fig 1.10a the segregation of the two classes is not possible with straight line since one circle vector lies in the area of square vector as an outlier.

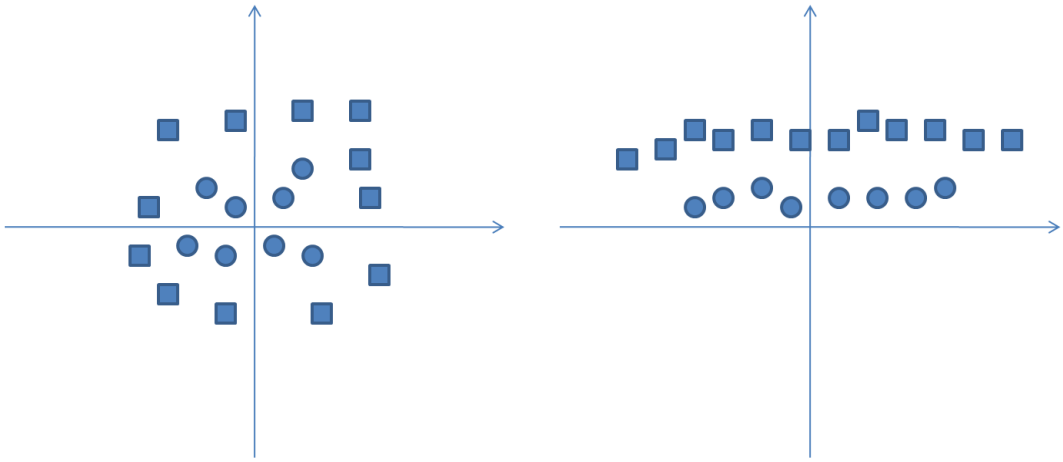
  SVM ignores outliers and selects the hyper plane which has maximum margin, thus proving that SVM is robust to outliers as shown in Fig 1.10b



(a) Without Hyper Plane (b) With Hyper Plane

Figure 1.10: Scenario 4

- Scenario 5: In Fig 1.11a we cannot draw a linear hyper plane to classify the two classes.



(a) Without adding additional feature (b) With additional feature

Figure 1.11: Scenario 5

SVM solves the problem easily by adding an additional feature

$$z = x^2 + y^2 \tag{1.1}$$

which makes the scenario plot look like as shown in Fig 1.11b

1.2.4 SVM kernels

- rbfdot: rbfdot kernel is basically a radial basis kernel function (Gaussian) or RBF kernel which is used in kernelized learning algorithms [3]. In some given input space, two feature vectors a and b, the RBF kernel is defined as [4]:

$$\text{K(a,b)} = \exp(\text{-}\frac{(\|a-b\|)^2}{2\sigma^2})$$

or alternatively as

$$\text{K(a,b)} = \exp(\text{-}\gamma \parallel a - b \parallel^2) \text{ where } \gamma = \frac{1}{2\sigma^2}$$

- vanilladot: vanilladot is a linear kernel function. This kernel function is the simplest and can be defined as the inner product ⟨a,b⟩ with an additional optional constant c as given below:

$$\text{K(a,b)} = \text{ab} + \text{c}$$

- polydot: polydot is a polynomial kernel function which is known to be non-stationary kernel. For the normalized training data, the polynomial kernel function is best option to be chosen from other kernel functions. Polynomial kernel function with varying values of the constant c, slope α and the polynomial degree d can can be defined as:

$$\text{K(a,b)} = (\alpha ab + c)^d$$

- tanhdot: tanhdot is the hyperbolic tangent kernel or the sigmoid kernel or the multilayer perceptron (MLP) kernel origins from the neural network theory. When it is used with SVM it is equal to the two layered perceptron neural network. It is defined with varying values of the constant c and the slope α as:

$$\text{K(a,b)} = \tanh(\alpha\text{ab} + \text{c})$$

- laplacedot: laplacedot is the laplacian kernel function. For the changes in the values of σ it is less sensitive, otherwise it is fully equivalent to the exponential kernel function. It can be defined as:

$$\text{K(a,b)} = \exp(\text{-}\frac{(\|a-b\|)}{\sigma})$$

- besseldot: besseldot is the bessel kernel function which origins from the theory of function spaces of fractional smoothness. It is defined as:

$$\text{K(a,b)} = \frac{J_{v+1}(\sigma\|a-b\|)}{\|a-b\|^{-n(v+1)}}$$

where J is the Bessel function of first kind. But in kernlab package for R documentation is defined as:

$$\text{K(a,b)} = \text{-}Bessel^n_{nu+1}(\sigma|a-b|^2)$$

1.2.5 Thesis Outline

In **Chapter 2**, the previous work and different methods and algorithms used for hand gesture recognition have been discussed. **Chapter 3** states the problem and the objectives to be achieved. **Chapter 4** describes our proposed method and the various steps of implementation in detail. The various results of the implementation process, accuracy and robustness of the model have been shown in **Chapter 5**. **Chapter 6** tells the conclusion of the work and the work to be done in future.

Chapter 2

Literature Survey

This Chapter describes the work that has been done in past in the field of Hand Gesture Recognition. It discusses various methods and algorithms that have been used in the past.

Ionescu et al. [1] propose a dynamic hand gesture recognition algorithm which is based on 2D representation of the skeleton of hand. Dynamic gesture signatures are created by superimposing the hand skeleton of different postures of the gesture to give a single image. These signatures are then used for gesture recognition by comparing them with the gesture alphabet and the dissimilarities in model parameters are measured using Baddeley's distance. Fig 2.1 shows the dynamic signatures for the two different hand gestures.

Figure 2.1: Dynamic Signature for two different gestures[1]

Kim et al. [5] proposed a gesture recognition system which combines the fuzzy max-min composition along with Relational Database Management System to be implemented on Post-PC platform. The combination can provide improved data acquisition efficiency. 19 dynamic hand gestures were tested for the proposed gesture recognition system using 5DT Company's 5th Data Glove System as an input device.

Manresa et al. [6] presented a real time algorithm which can be used to interact with video game using hand gestures. The algorithm uses the color cue for hand segmentation and hand morphological features are extracted from the tracking process for which constant velocity is assumed and pixel labeling approach is followed. The finite state classifier takes the extracted features as input and classifies

the gesture according to its hand configuration. Experiments performed using the system under varying lighting and background conditions have also shown satisfactory results. Four different hand gestures and four different hand movement directions are considered. A total of 1600 gestures collected by 40 different users were tested using the algorithm.

Temburwar et al. [7] developed a system for two-way communication between deaf and dumb people. The user of the desktop application will use the camera to register different signs to create a database. When in recognition mode the application will recognise the sign using the database and announce the recognised gesture in speech format. The accuracy varies when the gestures are performed in varying light conditions and also depends upon the distance between the camera and the user.

Osimani et al. [8] present an experimental prototype for gesture recognition. It uses the background subtraction process, which uses the skin color detection method, to remove the background in order to extract the hand from the images captured by the standard webcam. In calibration process, the face of the user is detected in order to locate the hands since the hands will be close to the face. Then the face is removed as the background to get the hand's position. Lastly, in Hand Posture Recognition process, for continuous tracking of the hand and its position Lucas-Kanade algorithm is used. Then the contour of the hand and a few significant points in the fingers are extracted using geometry feature extraction to identify the gesture. Two hand gestures, open hand and close hand, and two hand movements, horizontal motion with close hand and vertical motion with close hand, are recognised. The prototype is tested on 180 videos with total 1800 movements.

Thang et al. [9] study the effectiveness, feasibility and comparison of Simplification of Support Vector Machine (SimpSVM) and Relevance Vector Machine (RVM) classification methods for Hand Gesture Recognition problem. SimpSVM is better than RVM in gesture recognition. Also, SimpSVM requires less training time during training phase. Performance changes with amount of data and number of features.

Yeo et al. [10] propose a real-time method that can be used for robust and fast hand tracking even in complex background, motion blur and skin color variation. The gestures recognised by the method can be used for functional inputs and can interface with various applications and games. The writers have developed various applications for testing. Three cameras have been used by the method. Nine hand gestures are considered by the method. Ten members tested the prototype system

and were asked to test the accuracy twice.

Siby et al. [11] have introduced an effective system for interaction with a dumb person. The user uses white woolen gloves with each finger having red, blue and green pattern. The camera captures the gestures and the changes in the intensity of the colors due to gestures are detected. For gesture recognition it uses image processing techniques like skin color and outline identification. The accuracy depends on the intensity of light in the surroundings.

Sykora et al. [12] have compared Scale-invariant feature transform (SIFT) and Speeded up robust features (SURF) feature extraction methods on the set of depth maps of ten left hand gestures. Microsoft Kinect has been used for capturing images and Support Vector Machine (SVM) for classifying the images. Best result was obtained with SURF features. Since, SURF and SIFT features never change with changed orientation they are not fit for recognition of hand gestures in two images with different features. The method is implemented on 1500 images for classification of ten hand gestures.

Shukla et al. [13] have presented a method which uses modelling along with learning approach for hand gesture recognition. Images are captured using Microsoft Kinect with which dense and three dimensional scans of the user can be captured in real time. Background is subtracted from the images using depth feature of the Kinect to get hand gesture images. Image processing techniques are used on these images to find contour. Convex hull and convexity defects of the contour are calculated which are used as features by Naive Bayes classifier for classifying five hand gestures. The performance is tested on 75 images.

Wan et al. [14] have proposed a new approach for one-shot learning gesture recognition to obtain good performance. The new approach derives a new framework from Bag of Feature (BoF) model. The approach also proposes a spatio-temporal feature (3D EMoSIFT) which does not change on scaling and rotation and is insensitive to slight motion. In the coding stage Simulation Orthogonal Matching Pursuit (SOMP) is used instead of Vector Quantisation (VQ). The approach is evaluated on ChaLearn gesture database and has obtained high ranking in ChaLearn gesture challenge.

Ganapathyraju [15] has shown that an industrial robot can be controlled using hand gestures. The image is captured using a webcam. The image firstly undergoes the skin detection process so that only skin region of the hand is captured. Then, convex hull and convexity hull algorithms are applied for getting the outline of the hand and determining the number of fingers respectively. The determined count of fingers, received from hand gesture recognition module, is then used to

move the robot in one of the four directions. The accuracy of the system was affected by factors like background noise because of presence of skin color in background, poor lighting, hand tilted at different angle and hand having different orientation. However these drawbacks were rectified.

Feng et al. [16] in their paper have shown that Gradient Direction Histogram (HOG) feature which is mostly used pedestrian detection can be effectively used for gesture recognition. HOG feature can alleviate the impact of illumination variations and rotation of the hand for the recognition rate. After extracting the HOG features of the hand gestures, Support Vector Machine (SVM) is used to train the feature vector. The decision about the gesture is taken using previously learned SVMs. Results are compared for different lighting conditions. The system is tested on ten hand gestures with 1000 images for training set and 100 images for test set.

Shrivastava [17] focussed more on the speed with which the gesture is recognised rather than the accuracy rate of gesture recognition, by proposing a Hidden Markov Model (HMM) for the recognition. L$\alpha\beta$ color space is applied for hand detection. Hu invariant moments and hand orientation are combined for the feature extraction process. Baun-Welch algorithm using Left-Right Banded (LRB) topology is applied for training and forward algorithm for recognition. The proposed system was tested with five gestures and ten training videos of each gesture.

Oprisescu et al. [18] have used the depth and intensity data given by the Time-of-Flight (ToF) camera for obtaining reliable hand segmentation using the modified region growing algorithm. Gesture classification is carried out with decision tree which exploits the structural descriptions of partitioned contour segments. Nine static hand gestures and 450 test images were considered for the proposed automatic algorithm.

Hasan et al. [19] have explored the usage of neural network based approach for hand gesture recognition. Two set features are extracted from the gesture image: Hand Contour and Hand Complex Moments. For recognition a supervised back-propagation multi-layer feed-forward neural network is used. Hand Complex Moments features with neural network gave better results than Hand Contour features with neural network. The algorithm was tested on six static hand gestures. 56 hand gesture images were used to test Hand Contour features with neural network and 84 hand gesture images to test Hand Complex Moments features with neural network.

Gupta et al. [20] have used 15 local Gabor filters instead of 40 Gabor filters so that complexity is reduced with increased accuracy. After that combination of

Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are used for feature extraction and feature reduction. One against one multiclass Support Vector Machine (SVM) is used for classification. Ten Hand Gestures are taken in the dataset.

Igorevich et al. [21] in their method are using grayscale histogram for defining the depth threshold of calculated disparity map. Simple stereo camera is used to detect hand motion. The algorithm is fully independent of the previous frames for computations of current frame to reduce calculation time. The distance between the camera and the user can vary.

Starner et al. [22] have used Hidden Markov Models(HMM) and presented two SLR systems which are vision based. The first one used a camera mounted on the desk and gave a second-person view and in the second the user wears a hat and the camera is mounted on that hat to get the first-person view. The continuous motion tracking was done using HMM. The skin color matching algorithm was used by both the systems for hand tracking.When a skin colored pixel was found, eight nearest neighbouring pixels were searched for similar color areas. It was assumed that the face remains stagnant while doing hand movements, hence it is not considered. Since the camera used gave 2D video, two hands overlapping separation was not achieved. Hence, wherever the occlusion happened the whole area was simply assigned by them. Both the systems were developed for recognising American Sign Language (ASL) sentences. In comparison first-person view system was better in terms of word accuracy than the second-person view system.

Ong et al. [23] proposed an approach in which in order to track hands, hand shapes were detected in gray scale images using a boosted cascade of classifiers. For this they have collected images of hands from numerous different people since different people have different shapes and sizes of hands. After collecting the images, the grouping of the images was done using the K-medoid clustering algorithm. Then, a tree structure with two layers is formed where each layer is the weak hand detector. One representative image for a group of images is in the first layer of the tree structure. After this the classifiers which are the representative images choose all the participant images which may contain hand shapes. Then they are passed on to the second layer for comparison with the corresponding clusters of images. For finding weak classifiers the FloatBoost algorithm is used. The approach was performed on 5,013 hand images, 98% accuracy was achieved at the first layer of hand detection and 97.4% accuracy was achieved at the second layer of hand shape detection.

Hernandez-Rebollar et al. [24] proposed a real-time system which was glove-based and recognised 26 ASL letters and two additional signs for the space and enter. The interface was designed for the users, using which words or phrases were formed and then passed on to the speech synthesizer. The proposed system was composed of five MEMS (Micro Electro Mechanical System) dual axis accelerometers and an Accele Glove which is a micro controller. During the training phase of the system, the positions of the fingers collected by the Accele Glove are read by the micro controller and 10 bytes packets are sent to a PC. The PC extracts a collection of posture features and a classifier with three-level hierarchy is created. Since a trained classifier is programmed in the micro controller, it recognises the spellings of the fingers. The recognised finger-spelling's corresponding ASCII is sent to the voice synthesizer which would voice out the letters. The system's accuracy for 21 letters was 100% and the worst case being that of 'U', its accuracy was 78%.

Raheja et al. [25] in their paper have presented a method which keeps a track of the fingertips and the palm center using Kinect. For segmentation of the hand, thresholding was applied on the depth of the region of the hand. After this, to get only the fingers of the hand in the image, the palm of the hand is filtered and then subtracted from the image. In most of the situations when the user's hand is in front of him, the fingers must have shallowest depth i.e. they should be at minimum distance from the Kinect. Hence, fingertips are found by calculating the minimum depth and the center of palm is found by calculating the maximum distance in the hand's image. In case of extended fingers, the fingertips were detected with an accuracy of 100% and palm with an accuracy of 90%.

Singha et al. [26] have discussed about how self co-articulation can be used as a feature for improving the performance of hand gesture recognition system. The self co-articulation features were detected from the trajectory of the gestures by adding speed to the information and adding a pause when a gesture is spotted. Other features that were used are hand's position, distance and ratio features. The ANN and the SVM models were used to test the feature's accuracy and performance for the hand gesture recognition.

Table 2.1 summarises some of the data of literature survey.

Table 2.1: Literature Survey Summary

S.No	Authors	Type of Gesture	No. of Classes	No. of gestures in Dataset
1	Ionescu et al. [1]	Dynamic	10	40 (4 test for each gesture)
2	Kim et al. [5]	Dynamic	19	For 20 reagents, repeated 10 times
3	Manresa et al. [6]	Static and Dynamic	4 Hand Gestures, 4 Movement Direction	1600
4	Temburwar et al. [7]	Dynamic	26 combinations of Indian characters	Varies according to user
5	Osimani et al. [8]	Dynamic	4	180 videos with 1800 movements
6	Yeo et al. [10]	Dynamic	9	180(10 users tested accuracy twice)
7	Siby et al. [11]	Static	-	-
8	Shukla et al. [13]	Static	5	75
9	Ganapathyraju [15]	Static	4	-
10	Feng et al. [16]	Static	10	Training- 1000 for each gesture, Testing- 100
11	Shrivastava [17]	Dynamic	5	10 training videos
12	Oprisescu et al. [18]	Static	9	450
13	Gupta et al. [20]	Static	10	-
14	Igorevich et al. [21]	Dynamic	-	-
15	Starner et al. [22]	Static	26	-
16	Ong et al. [23]	Static	-	5,013
17	Hernandez-Rebollar et al. [24]	Static	28	-

Chapter 3

Problem Statement

In this chapter, the gaps in the current work; the problem statement; the objectives of the work to be done and the methodology for achieving the objectives are discussed.

3.1 Gap Analysis

In the Literature Review, we have discussed the different ways in which hand gestures can be recognised which are different from one another in many aspects like hardware used, static or dynamic gestures, number of entries in dataset etc. In the existing work following gaps were noticed:

- All variations of the hand gestures that can be given by the different users were not considered for a particular hand gesture in the HGR approaches.
- The process of capturing the hand gestures required high-end devices like Kinect, 5DT Company's 5th Data Glove System etc.
- Different methods proposed in Literature Survey, have used small dataset of images for training and testing the methods.

3.2 Problem Statement

Various existing methods of HGR have been already presented which effectively recognise different hand gestures be it static or dynamic hand gesture, using different input devices for capturing images like Kinect, 5DT Company's 5th Data Glove System, ToF camera etc, with different image processing and machine learning techniques. But in most of the techniques different variations of a gesture that a human can give for a gesture have not been considered.

In order to overcome these problems a simple approach using image processing and machine learning algorithms, can be introduced and applied on the videos captured from normal camera for hand gesture recognition.

3.3 Objectives

The objectives to be achieved from our work are:

- To study the literature for existing HGR systems and techniques.
- To propose a method for HGR using Image Processing and Machine Learning.
- To implement the technique using the proposed method.
- To test and validate the output of proposed method.

3.4 Methodology

The methodology to be followed is as follows:

- Literature Review.
- Propose the technique using different image processing techniques and various kernels of SVM.
- Implementation of the desired method is as follows:
  - Making of the videos with different persons giving variations of the hand gestures using a mobile camera.
  - Background subtraction from the video so as to remove the static background from the videos and to get just the moving hand in the video.
  - Extraction of the frame after applying Bilateral and Median filters.
  - Application of the Gaussian Threshold so as to get a proper outline of the hand gesture.
  - Classification of the hand gestures using SVM classifier with different kernels.
- Test and validate the robustness of the system against different sample of inputs.

Chapter 4

Proposed Methodology

In this chapter, the methodology used for HGR in our proposed work has been discussed. It discusses the various Image Processing and Machine Learning algorithms and methods used for implementation.

4.1 OpenCV Library and its functions used in methodology

OpenCV was originally developed by Intel and has various functions for performing image processing for real time computer vision. Its various functions used in our methodology are:

- Bilateral Filter: The Bilateral filter is a smoothing filter which is non-linear, noise-reducing and edge preserving. The weighted average of intensity values, which can be based on Gaussian distribution and can depend on both Euclidean distance of pixels and radiometric differences like range differences such as depth distance, color intensity etc, is calculated and the intensity value at each pixel of the image is replaced by this value. Thus, sharp edges are preserved by visiting each pixel systematically in a loop and accordingly adjusting the values of weights with adjacent pixels.
- Median Blur: Salt and pepper noise also known as impulse noise which are presented in the form of black and white pixels sparsely spread on the image. It is caused by sudden and sharp disturbances in the image signal. Median filter is used to effectively reduce such type of noise. The median filter function replaces the central pixel of the kernel area with the median value of all pixels under the kernel area. Since the central pixel value is replaced by some pixel value which may be in the image, the noise is removed effectively. The kernel size considered for median blur is a positive odd integer.
- cvtColor(): It is a function which is used to convert an image's color space to

another color space. When the transformation is to/from RGB color space, the order of the channels should be mentioned explicitly. The default color format in OpenCV is BGR rather than RGB. In a standard 24-bit image, first 8 bits refer to the blue component, next 8 bit the green component and the last 8 bit the red component. The default ranges for R, G and B channels are:

  - CV_8U images: 0 to 255
  - CV_16U images: 0 to 65535
  - CV_32F images: 0 to 1

  The range does not matter in case of linear transformations, but in case of non-linear transformations the input RGB image has to be normalised to proper value ranges so as to get correct results.

- BackgroundSubtractorMOG2: This background subtraction algorithm is a gaussian mixture based and is used for Background/Foreground segmentation [27][28]. One of the important features of this algorithm is that for each pixel it selects an appropriate number of gaussian distribution. Shadow detection is optional and can be removed or shown with gray color.

4.2 R Studio and its functions used in methodology

R Studio desktop application is an open source Integrated Development Environment (IDE) for R language, which is used for statistical computing and graphics.

Following functions of R Studio have been used in our methodology:

- kernlab Package [29]: kernlab package is used for kernel based machine learning methods in R. This extensible package provides a framework with the help of which we can create and use kernel based algorithms, by taking advantage of S4 object model in R. The package contains Dot Product Primitives (kernels). It also contains the implementations of SVMs and the relevance vector machine, kernel PCA, kernel CCA, kernel feature analysis, online kernel methods, Gaussian processes, a ranking algorithm and a spectral clustering algorithm. Also, it provides methods for an incomplete Cholesky decomposition and a general purpose quadratic programming solving.

- ksvm function [29]: ksvm function is the kernlab's implementation of SVMs. SVMs are an excellent tool for classification, regression and novelty detection. The ksvm function along with parameters used in methodology is:

  ksvm(x,y,kernel=" ",prob.model=True)

  - x is the formula which is a symbolically describes the model to be fit.
  - y is the optional data frame which contains the training dataset when a formula is used
  - kernel tells about the kernel function which is used in training of the model and prediction. This parameter's value can be one of the following functions of the class kernel:
    - rbfdot Radial Basis kernel "Gaussian"
    - vanilladot Linear kernel
    - polydot Polynomial kernel
    - tanhdot Hyperbolic tangent kernel
    - laplacedot Laplacian kernel
    - besseldot Bessel kernel
  - prob.model is set to true, thus building a model for calculating class probabilities.

- predict function: It is a generic function which uses the results of various model fitting functions to give prediction values for the new data. The predict function with its parameters used in our methodology is:

  predict(x,y)

  where x is the model and y is the dataset on which the predictions are to be made.

4.3 Proposed Method

For our work we have used a Mobile Camera for making videos of 720×1280 pixels at 30fps. Five different static hand gestures shown in Fig 4.1 were displayed in front of the camera with different variations. Fig 4.2 shows the important steps to be carried out in our proposed methodology.

(a) Close Hand (b) Open Hand (c) Victory Hand

(d) Thumb Down (e) Thumb Up

Figure 4.1: Types of Gestures

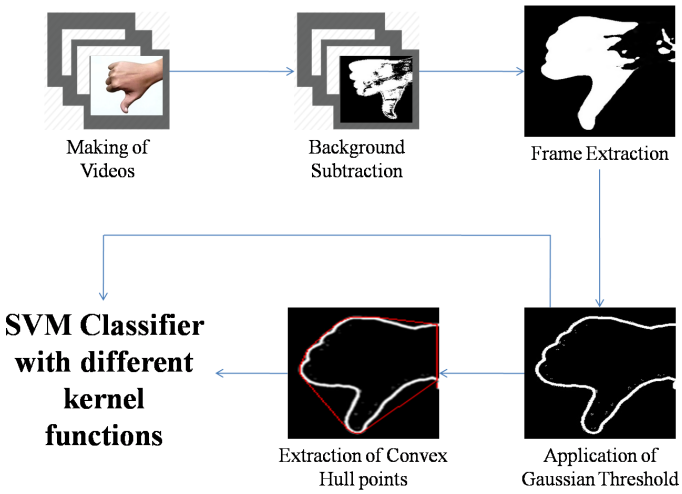


Figure 4.2: Methodology Used

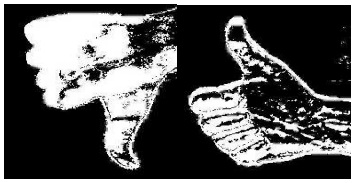
4.3.1 Background Subtraction

Background subtraction is used for extracting moving foreground from static background. It is an important preprocessing step. If image of the static background is available background subtraction can be done by just subtracting the background image from the new image to get foreground objects alone. Such images are not available in most of the cases and we have to adjust with the available images. Shadows can also complicate the process.

For background subtraction we have used BackgroundSubtractorMOG2 gaussian mixture-based Backgroud/Foreground Segmentation OpenCV algorithm [27],[28]. The essential feature of this algorithm is that for each pixel it selects an appropriate number of gaussian distribution and also in varying scenes due to illumination

changes (etc) it provides better adaptibility. Fig 4.3 shows the images after background subtraction from video.

(a) Close Hand (b) Open Hand (c) Victory Hand



(d) Thumb Down (e) Thumb Up

Figure 4.3: Hand Gestures after Background Subtraction

4.3.2 Frame Extraction

Since we have extracted the moving foreground from static background, we have got our Region of Interest (ROI) i.e. the hand gesture. We can now extract frames from the video which will be part of our dataset after applying some image processing algorithms. For smoothing the images, after extracting frames we apply bilateral filter and median blur on the frames. Image smoothing involves convolving the image using a low-pass filter kernel. It is helpful in removing high frequency content like noise, edges from the image which results in blurring of the edges too in the process.

Bilateral filter effectively keeps the edges sharp while removing noise. Bilateral filter takes two gaussian filters, one a function of space which makes sure that during filtration, for blurring, only neighbourhood pixels are considered and the other a function of pixel difference which makes sure that for blurring only the pixels with similar intensity to central pixel are considered.

Median filter is effective in removing salt and pepper noise. The median blur function always replaces the central element by some pixel in the image, calculated by taking median of all the pixels under kernel area. The size of the kernel should be odd and it must be a positive integer. Fig 4.4 shows the frame after applying firstly bilateral filter and then median blur on the extracted frame from video.

(a) Close Hand (b) Open Hand (c) Victory Hand

(d) Thumb Down (e) Thumb Up

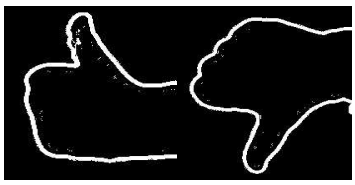
Figure 4.4: Hand Gestures after applying Bilateral Filter and Median Blur

The dataset is created by applying gaussian threshold and resizing the images to 50×50 pixels.

4.3.3 Application of Gaussian Threshold

Thresholding a grayscale image can create a binary image. In simple thresholding, if the pixel value is greater than the given threshold value the pixel value is set to one of the two values else it is set to the other value.

(a) Close Hand (b) Open Hand (c) Victory Hand



(d) Thumb Down (e) Thumb Up

Figure 4.5: Hand Gestures after applying Gaussian threshold

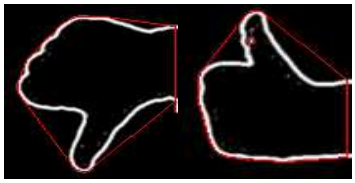
In adaptive thresholding, the threshold value is calculated for small regions in the image, such that the threshold value of the pixel is dependent on the pixel

intensity of its neighbouring pixels. In gaussian threshold for threshold value, the weighted sum of the neighbouring pixel values is calculated, where weights are a gaussian window. Fig 4.5 shows the frames after applying gaussian threshold. Noise is removed at a good rate after applying Gaussian threshold.

4.3.4 Extraction of Convex Hull points

After applying gaussian threshold to the images, we calculated the convex hull of the hand and also the contour of the hand. A set, K, of points say k_0, k_1, k_2, k_3, k_4,..., k_n, is said to be convex if a line segment joining a pair of points in the set, should also be a part of the set. The convex hull of the set K will be the smallest convex polygon containing all the points in the set K [30]. Fig 5.4 shows the hand gestures with convex hull.

(a) Close Hand (b) Open Hand (c) Victory Hand



(d) Thumb Down (e) Thumb Up

Figure 4.6: Hand Gestures with Convex Hull

4.3.5 Using SVM for classification

Features extracted during previous step and the convex hull values of each image are used to create a single Comma Separated Values (CSV) file for the dataset. The data is then used to train and test the SVM classifier with different kernels in different ratios for HGR. SVM is one of the most used machine learning algorithm used for classification which builds a model and predicts in which class the new example falls into. The ability to generalize a problem is higher in SVM. SVMs are a set of related supervised learning methods used for classification, which means it is machine learning task of inferring the results from previously supervised

trained data. SVM kernels used for classification in our work are rbfdot, polydot, vanilladot, tanhdot, laplacedot and besseldot.

Chapter 5

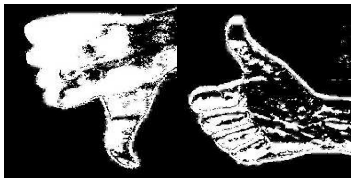
Experiments and Results

This section analyzes the prediction results of SVM machine learning classification model with different kernels on the testing dataset. Total dataset has 16,240 entries which has five classes of different hand gestures with different variations collected from 10 users in different backgrounds. Each user has given different variations of the 5 gestures such that all possible ways to perform the gesture are covered.

5.1 Results of Background Subtraction

The resultant images after Background Subtraction from the videos are shown in Fig. 5.1.

(a) Close Hand (b) Open Hand (c) Victory Hand

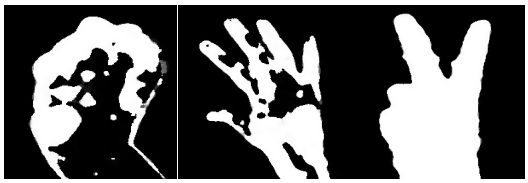


(d) Thumb Down (e) Thumb Up

Figure 5.1: Hand Gestures after Background Subtraction

5.2 Results of Frame Extraction

The output images after extracting frames from the videos and application of Bilateral filter and Median filter are shown if Fig. 5.2.

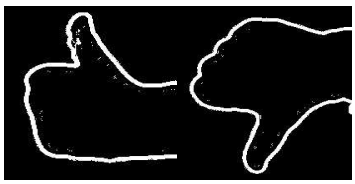


(a) Close Hand (b) Open Hand (c) Victory Hand

(d) Thumb Down (e) Thumb Up

Figure 5.2: Hand Gestures after applying Bilateral Filter and Median Blur

(a) Close Hand (b) Open Hand (c) Victory Hand



(d) Thumb Down (e) Thumb Up

Figure 5.3: Hand Gestures after applying Gaussian threshold

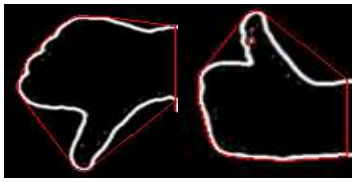
5.3 Results of Application of Gaussian Threshold

The images after applying Gaussian Threshold on the previous images in which Background Subtraction was done are shown in Fig. 5.3.

5.4 Results of Extraction of Convex Hull Points

The hand gestures with convex hull are shown in Fig. 5.4

(a) Close Hand (b) Open Hand (c) Victory Hand



(d) Thumb Down (e) Thumb Up

Figure 5.4: Hand Gestures with Convex Hull

5.5 Results of SVM model with different kernel functions

The accuracy is calculated for the SVM model on 50-50, 60-40, 70-30 and 80-20 training-testing partitions and is shown in Table 5.1 for each SVM kernel. To determine the accuracy a confusion matrix is created which shows the information about actual and predicted classifications done by SVM. The diagonal elements of the confusion matrix represent the number of predicted entries whose values are equal to the actual values where as the off diagonal represent the number of predicted entries whose values are not equal to the actual values. The higher the value of the diagonal elements the higher is the accuracy. It is evident that

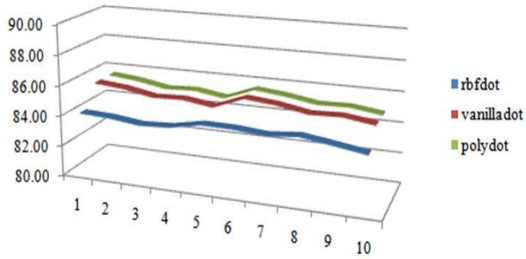
the results obtained by using SVM kernels vanilladot and polydot with different training-testing partitions are satisfactory and as the training dataset increases the result tends to get better.

Table 5.1: Performance comparison of SVM model with different kernels on different training-testing data

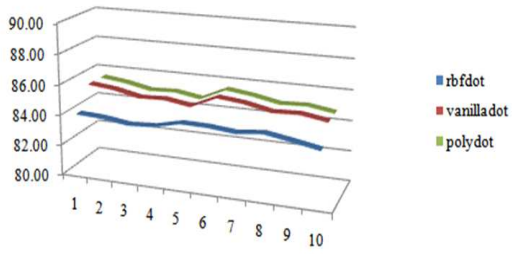
	Accuracy with Training - Testing Partition			
	50-50%	60-40%	70-30%	80-20%
rbfdot	**79.32**	**82.09**	84.11	86.13
vanilladot	75.00	80.35	**85.07**	**90.09**
polydot	75.00	80.35	**85.07**	**90.09**
tanhdot	24.18	24.12	24.17	24.28
laplacedot	38.50	41.58	42.99	44.70
besseldot	59.18	65.91	71.24	75.48

5.6 Results of Cross Validation

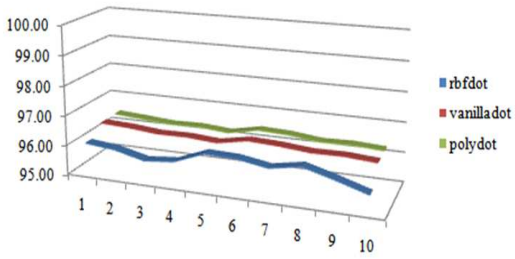
Cross validation is a predictive model evaluation technique to measure the robustness of the model in which some part of dataset is used to train the model and the other to evaluate it. In K-fold cross validation The original dataset is randomly partitioned into K equal size sub-datasets. From these sub-datasets one sub-dataset is used for the validation of the model and the other K-1 sub-datasets are used for the training of the model. The cross validation process is repeated K times (the folds) such that each of the K sub-dataset is used exactly once for validation of the model. The advantage of cross-validation is that all the entries of the dataset are used for both training and testing of the model and each entry is used exactly once for testing the model. Here, we have used 10-fold cross vali-



(a) Cross Validation of Accuracy



(b) Cross Validation of Sensitivity



(c) Cross Validation of Specificity

Figure 5.5: Results of 10-fold cross validation of the rbfdot, vanilladot and polydot SVM kernels and 70-30% training-testing partition

dation to measure the robustness and performance of SVM model with vanilladot kernel on our dataset. Figure 5.5 shows the results of 10 fold Cross Validation for accuracy, sensitivity and specificity respectively calculated on training-testing dataset(70-30%). The results of cross-validation show a uniform performance in accuracy, sensitivity and specificity. Thus, validating the performance of SVM vanilladot kernel on our dataset.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this work, we have proposed a HGR method for recognising 5 static hand gestures (Open Hand, Close Hand, Victory Hand, Thumb Up and Thumb Down). No extra hardware as been used for detecting hand while making videos. We have used various image processing algorithm like Bilateral Filter, Median Blur, Gaussian Adaptive threshold and Convex Hull points to pre-process the images. SVM classifier with different kernels i.e. rbfdot, vanilladot, polydot, tanhdot, laplacedot and besseldot is used for classifying of the images into 5 static hand gestures. The dataset used for this work has 16,240 images used for training and testing of SVM model for HGR. The maximum accuracy 90.09% is obtained by SVM kernels vanilladot and polydot by training-testing partition of 80-20%. The results obtained from 10-fold cross validation confirms the robustness of our method using SVM vanilladot kernel.

6.2 Future Work

In future work, we will work on dynamic HGR since they will provide better natural interaction. We will also work on more number of static hand gestures for recognition. We will try for other machine learning models and improve the accuracy of our work.

References

[1] B. Ionescu, D. Coquin, P. Lambert, and V. Buzuloiu, "Dynamic hand gesture recognition using the skeleton of the hand," *EURASIP Journal on Advances in Signal Processing*, vol. 2005, no. 13, p. 236190, 2005.

[2] Y. Wu and T. S. Huang, "For vision-based human computer interaction," *studies*, vol. 5, p. 22, 2001.

[3] Y.-W. Chang, C.-J. Hsieh, K.-W. Chang, M. Ringgaard, and C.-J. Lin, "Training and testing low-degree polynomial data mappings via linear svm," *Journal of Machine Learning Research*, vol. 11, no. Apr, pp. 1471–1490, 2010.

[4] J.-P. Vert, K. Tsuda, and B. Schölkopf, "A primer on kernel methods," *Kernel Methods in Computational Biology*, pp. 35–70, 2004.

[5] J.-H. Kim, D.-G. Kim, J.-H. Shin, S.-W. Lee, and K.-S. Hong, "Hand gesture recognition system using fuzzy algorithm and rdbms for post pc," *Fuzzy Systems and Knowledge Discovery*, pp. 487–487, 2005.

[6] C. Manresa, J. Varona, R. Mas, and F. J. Perales, "Hand tracking and gesture recognition for human-computer interaction," *ELCVIA Electronic Letters on Computer Vision and Image Analysis*, vol. 5, no. 3, pp. 96–104, 2005.

[7] S. Temburwar, P. Jaiswal, S. Mande, and S. Patil, "Design of a communication system using sign language aid for differently abled peoples.," 2017.

[8] C. Osimani, J. A. Piedra-Fernandez, J. J. Ojeda-Castelo, and L. Iribarne, "Hand posture recognition with standard webcam for natural interaction," in *World Conference on Information Systems and Technologies*, pp. 157–166, Springer, 2017.

[9] P. Q. Thang, N. D. Dung, and N. T. Thuy, "A comparison of simpsvm and rvm for sign language recognition," in *Proceedings of the 2017 International Conference on Machine Learning and Soft Computing*, pp. 98–104, ACM, 2017.

[10] H.-S. Yeo, B.-G. Lee, and H. Lim, "Hand tracking and gesture recognition system for human-computer interaction using low-cost hardware," *Multimedia Tools and Applications*, vol. 74, no. 8, pp. 2687–2715, 2015.

[11] J. Siby, H. Kader, and J. Jose, "Hand gesture recognition," *IJITR*, vol. 3, no. 2, pp. 1946–1949, 2015.

[12] P. Sykora, P. Kamencay, and R. Hudec, "Comparison of sift and surf methods for use on hand gesture recognition based on depth map," *AASRI Procedia*,

vol. 9, pp. 19–24, 2014.

[13] J. Shukla and A. Dwivedi, "A method for hand gesture recognition," in *Communication Systems and Network Technologies (CSNT), 2014 Fourth International Conference on*, pp. 919–923, IEEE, 2014.

[14] J. Wan, Q. Ruan, W. Li, and S. Deng, "One-shot learning gesture recognition from rgb-d data using bag of features.," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 2549–2582, 2013.

[15] S. Ganapathyraju, "Hand gesture recognition using convexity hull defects to control an industrial robot," in *Instrumentation Control and Automation (ICA), 2013 3rd International Conference on*, pp. 63–67, IEEE, 2013.

[16] K.-p. Feng and F. Yuan, "Static hand gesture recognition based on hog characters and support vector machines," in *Instrumentation and Measurement, Sensor Network and Automation (IMSNA), 2013 2nd International Symposium on*, pp. 936–938, IEEE, 2013.

[17] R. Shrivastava, "A hidden markov model based dynamic hand gesture recognition system using opencvshrivastava2013hidden," in *Advance Computing Conference (IACC), 2013 IEEE 3rd International*, pp. 947–950, IEEE, 2013.

[18] S. Oprisescu, C. Rasche, and B. Su, "Automatic static hand gesture recognition using tof cameras," in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, pp. 2748–2751, IEEE, 2012.

[19] H. Hasan and S. Abdul-Kareem, "Static hand gesture recognition using neural networks," *Artificial Intelligence Review*, pp. 1–35, 2014.

[20] S. Gupta, J. Jaafar, and W. F. W. Ahmad, "Static hand gesture recognition using local gabor filter," *Procedia Engineering*, vol. 41, pp. 827–832, 2012.

[21] R. R. Igorevich, P. Park, D. Min, Y. Park, J. Choi, and E. Choi, "Hand gesture recognition algorithm based on grayscale histogram of the image," in *Application of Information and Communication Technologies (AICT), 2010 4th International Conference on*, pp. 1–4, IEEE, 2010.

[22] T. Starner, J. Weaver, and A. Pentland, "Real-time american sign language recognition using desk and wearable computer based video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, pp. 1371–1375, 1998.

[23] E.-J. Ong and R. Bowden, "A boosted classifier tree for hand shape detection," in *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pp. 889–894, IEEE, 2004.

[24] J. L. Hernandez-Rebollar, R. W. Lindeman, and N. Kyriakopoulos, "A multi-class pattern recognition system for practical finger spelling translation," in *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*, p. 185, IEEE Computer Society, 2002.

[25] J. L. Raheja, A. Chaudhary, and K. Singal, "Tracking of fingertips and centers of palm using kinect," in *Computational intelligence, modelling and simulation (CIMSiM), 2011 third international conference on*, pp. 248–252, IEEE, 2011.

[26] J. Singha and R. H. Laskar, "Self co-articulation detection and trajectory guided recognition for dynamic hand gestures," *IET Computer Vision*, vol. 10, no. 2, pp. 143–152, 2016.

[27] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 2, pp. 28–31, IEEE, 2004.

[28] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern recognition letters*, vol. 27, no. 7, pp. 773–780, 2006.

[29] A. Zeileis, K. Hornik, A. Smola, and A. Karatzoglou, "kernlab-an s4 package for kernel methods in r," *Journal of statistical software*, vol. 11, no. 9, pp. 1–20, 2004.

[30] M. De Berg, O. Cheong, M. Van Kreveld, and M. Overmars, *Computational Geometry: Introduction*. Springer, 2008.

Video Presentation

https://www.youtube.com/channel/UChIIVnJ-jfN6dbEhW7DrFcQ