2012 1st International Conference on Emerging Technology Trends in Electronics, Communication and Networking

INDIAN SIGN LANGUAGE RECOGNITION

Divya Deora[1], Nikesh Bajaj[2]
[1]divyadeora930@gmail.com
[2]Department of Electronics Engineering, Z.H.College of Engineering and Technology
Aligarh Muslim University, Aligarh 202002; bajaj.nikkey@gmail.com

Abstract-**Understanding human motions can be posed as a pattern recognition problem. Applications of pattern recognition in information processing problems are diverse ranging from Speech, Handwritten character recognition to medical research and astronomy. Humans express time-varying motion patterns (gestures), such as a wave, in order to convey a message to a recipient. If a computer can detect and distinguish these human motion patterns, the desired message can be reconstructed, and the computer can respond appropriately. This paper represents a framework for a human computer interface capable of recognizing gestures from the Indian sign language. The complexity of Indian sign language recognition system increases due to the involvement of both the hands and also the overlapping of the hands. Alphabets and numbers have been recognized successfully. This system can be extended for words and sentences Recognition is done with PCA (Principal Component analysis). This paper also proposes recognition with neural networks. Further it is proposed that number of finger tips and the distance of fingertips from the centroid of the hand can be used along with PCA for robustness and efficient results.**

I. INTRODUCTION

A sign language is a language which, instead of acoustically conveyed sound patterns, uses visually transmitted sign patterns to convey meaning – simultaneously combining hand shapes, orientation and movement of hands, arms or body, and facial expression to fluidly express a speaker's thought. Deaf people exist in all parts of the world. Wherever the communities of deaf people exist, sign languages automatically develop. Regionally different languages have been evolved as ASL (American Sign Language) in America or GSL (German sign language) in Germany or ISL (Indian sign language) in India. There are mainly two different motivations for developing sign language recognition model. The first aspect is the development of an assistive system for the deaf people. For example development of a natural input device for creating sign language documents would make such documents more readable for deaf people. Moreover hearing people have difficulties in learning sign language and likewise the majority of those people who were born deaf or who became deaf early in life, have only a limited vocabulary of accordant spoken language of the community in which they live. Hence a system of translating sign language to spoken language would be of great help for deaf as well as for hearing people. A second aspect is that sign language recognition serves as a good basis for the development of gestural human-machine interfaces. This paper is organized as follows:

Section II shows the challenged in Indian Sign Language recognition system. Section III gives an idea about data acquisition and database. Section IV represents Experimental approach followed. In the end section VI represents Conclusion and future work. Signs of Indian sign language are shown in figure 1.

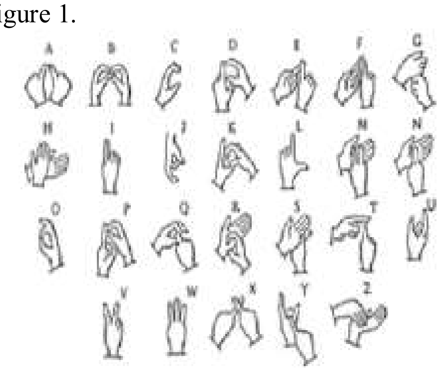


Fig. 1 Indian sign language

II. CHALLENGES IN INDIAN SIGN LANGUAGE RECOGNITION SYSTEM

The major challenges that are faced in Indian sign language recognition system are:

- Unlike other sign languages (American Sign Language, German Sign language) Indian Sign language uses both hands to make signs.
- Some signs involve overlapping of both the hands. This overlapping of hands poses difficulty in segmentation and recognition which is explained in steps later.
- Recognition for static signs is easy to implement but some signs involve motion in them due to which their recognition becomes more difficult. For example – signs h, j, v, and y have motion in them.

III. DATA ACQUISITION AND DATABASE

Data is acquired through a 3 Mega pixel camera. The intensity for capturing images is maintained throughout the experiment with the help of a table lamp having a uniform light intensity. For simplicity the person wears red and blue colored gloves. For each sign 15 images are taken and stored in the database. Therefore total number of images in the database is 510 which include 25 alphabets and 9 numbers as the system is made only to recognise alphabets and numbers.

978-1-4673-1627-9/12/$31.00 ©2012 IEEE

IV. Experimental Approach

Experimental approach explains the methods used for the accomplishment of the recognition of signs of Indian Sign Language. The steps are (1) segmentation, (2) finger tip finding algorithm and (3) PCA.

A. *Segmentation*

In computer vision, segmentation refers to the process of partitioning a digital image into multiple segments (sets of pixels, also known as super pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyse. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. In this experiment color base segmentation is done. Color of the gloves i.e. red and blue is detected.
Following are the steps to segment out red and blue color

Step1*: Separating all the components*

Fig. 2 shows the RGB image. Every RGB image is composed of three components: Red, Green and Blue. All the three components (bands) i.e. red, green, and blue are separated from the image from which the red and blue color is to be segmented.

Step 2*: Setting threshold levels*

For red color to be segmented threshold level high is 255 and threshold level low is 0. For the other two colors threshold levels are decided using Otsu’s method which chooses the threshold to minimize the intraclass variance of the threshold black and white pixels.

Step 3*: Creating red, green, and blue masks*

Red, green and blue masks are created from the components (bands) separated in the step 1. Red mask is a binary image in which wherever red component is present, it will indicate one and otherwise it represents zero where red component is not present. Same masks are for green and blue color. Fig. 3 shows Red, green and blue masks. Equation (1) shows how to create a mask for red color.

$$RM = (RB>=RTL) \wedge (RB<=RTH) \qquad (1)$$

Where RM=Red Mask
RB=Red Band
RTL=Red Threshold low
RTH=Red Threshold High

Step 4*: Creating the object mask*

Object mask simply means, if we want to detect red color the binary image should contain ones wherever red color is present and zeros where red color is not present. This is done by simply applying ‘and’ operation between all the masks created in the previous step. The output of this image is a binary image.

Step 5*: Obtain masked component images*

Masked component image is the component masked with the mask created in the previous step. When red component image is multiplied with the object mask, it just represents value in the red component where red object is present. Same operation is done for the green and the blue component. Fig. 4 shows masked component images.

Step 6*: Concatenating all the components*

After getting all the masked component images, all these images are combined to form the final image with red objects segmented out. Fig. 5 shows the image with red color segmented out.

Fig. 2 RGB image

Fig. 3(a) Red Mask

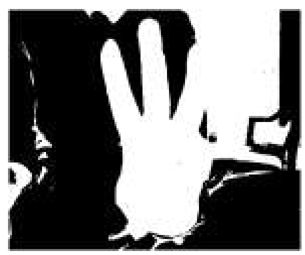


Fig. 3(b) Green Mask

Fig. 3(c) Blue Mask

Fig. 4(a) Red Masked Component image Image

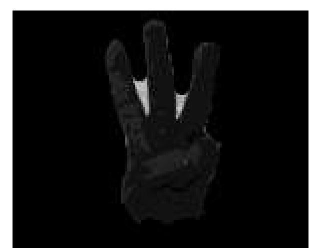


Fig. 4(b) Green Masked Component Image

Fig. 4(c) Blue Masked Component Image

Fig. 5 Segmented image after concatenating masked component images

B. *Fingertip finding algorithm*

Selection of features is probably the single most important factor in achieving high recognition performance. The feature selected in this paper to reduce the recognition time and

978-1-4673-1627-9/12/$31.00 ©2012 IEEE

increase the efficiency is the number of finger tips in the image. After finding fingertips using this algorithm some problems still arise like fingertips at wrong places which are discussed and solved later. Following steps shows the method of finding finger tips.

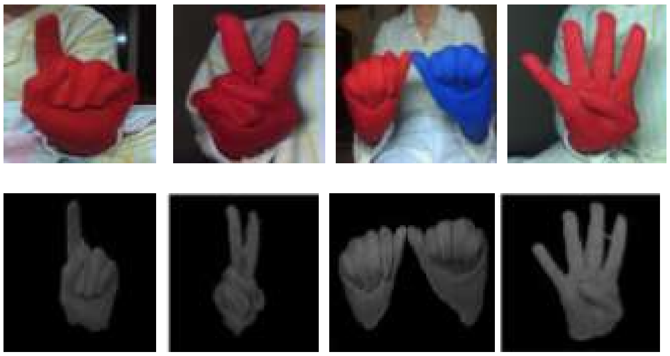


Fig. 6 Segmentation results

Step 1*: Thinning using Distance Transform*

Distance transform is the map of distance from any point to nearest point of some type. Imagine that foreground regions in the binary image are made of some uniform slow burning inflammable material. Consider a fire starting simultaneously on the boundary of foreground region and letting the fire burn its way into the interior. Label each point in the interior with the amount of time that the fire took to first reach that point. Fig. 7 (b) shows the result after applying distance transform

Step 2: *Finding perimeter pixels of the image*

After thinning, a binary image is found containing only the perimeter pixels of objects in the image. A pixel is the part of the perimeter if it is non zero and it is connected to at least one zero valued pixels. Fig. 7 (c) shows the result after finding perimeter pixels.

Step 3: *Finding Corner points*

For finding corner points, the first step is to find out perimeter pixels of the thinned image that are less than value 18. Corners are found for the fig 5 using Harris Corner detection. The Harris corner detector algorithm relies on a central principle: at a corner, the image intensity will change largely in multiple directions. This can alternatively be formulated by examining the changes of intensity due to shifts in a local window. Around a corner point, the image intensity will change greatly when the window is shifted in an arbitrary direction. It examines the similarity of the image function I(x, y) at a point (x, y) to itself when shifted by (Δx, Δy). This is given by autocorrelation function given in (2). It is assumed that there can be no corner points at a distance of 40 pixels from each edge. Fig. 8 shows the result finding perimeter pixels that are less than 18.

$$C(x, y; \Delta x, \Delta y) = \sum_{(u,v \in W(x,y))} \omega (I(u,v) - I(u+\Delta x, v+\Delta y))^2 \quad (2)$$

Where W(x, y) is a window centered at point (x, y)

ω (u, v) is window function.

Figure 9 shows the result after finding corner points using Harris corner detection.

Step 4: *Eliminating Corner Points*

It is clearly seen in figure 9 that many of the corner points are not satisfying the criteria for finger tips. So these points should be eliminated. They are eliminated according to the following algorithm:

- Consider a region around every corner point that is at a distance of 40 pixels in each direction and perimeter pixels are calculated for this region.
- Calculate area and eccentricity of this region.
- The region around a corner is selected if the eccentricity is less than .85 and area is greater than 100, otherwise it is rejected.

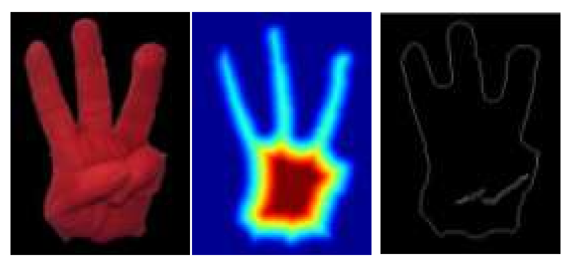
After eliminating many of the points, we are still left with many extra points. From fig 10(b) it is clearly seen that instead of one tip on each finger, we are getting multiple points. Clustering algorithm is used to solve this problem.

1) Clustering Algorithm

i. Calculate distance of each point from every other point and put these distances into a matrix as given in figure 12 where (1, 2) means distance of point 1 from point 2.

ii. Each value in this matrix is examined and if the value is less than 30, it is removed from the matrix. So if after this operation the cell with non zero value is (1, 4) and (4, 1) (say), this means there are two points left 1 and 4.

iii. If all the cells have zero value but still we have points, it means there is only one cluster i.e. only one finger tip is there.



(a) (b) (c)

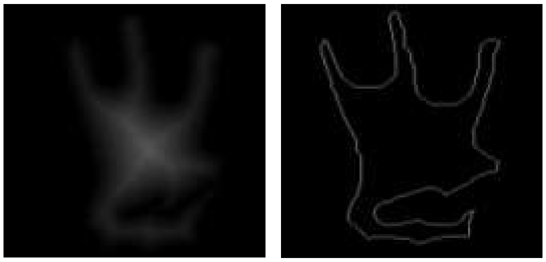
Fig 7. (a) Segmented Image (b) Result after thinning of image using distance transform (c) Binary image after finding perimeter pixels

C. Problem in finger tip algorithm

A problem that arises while finding finger tips algorithm is that when the images are taken wearing gloves, the images are not smooth at the ends after segmentation. The problem that is found after finding finger tips is shown in figure 12. There are some areas in the image that are still satisfying the criteria for the finger tip which are not correct. To fix this problem we have proposed an algorithm. For this we find the orientation of the binary image. Using the orientation and the centroid of the hand, a line is drawn, which passes through this centroid at an angle that is given by orientation. A line perpendicular to the

978-1-4673-1627-9/12/$31.00 ©2012 IEEE

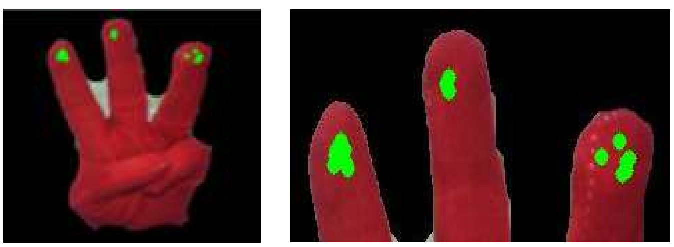
line shown in figure 13 (a) is drawn as shown in figure 13 (b). According to the algorithm, we check for every finger tip its x coordinate value. For that x coordinate value, we check a point on the line drawn that has the same x coordinate value for any value of y coordinate. Then we check if the y coordinate value



(a) (b)

Fig. 8 Binary image after finding perimeter pixels that are less than 18

Fig. 9 Corner points



(a) (b)

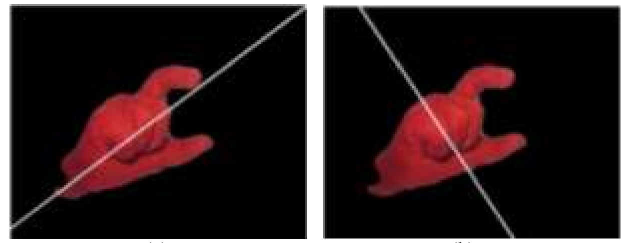
Fig. 10 (a) Results after eliminating Corner points (b) Zoomed view of the corner points.

(1,1)	(1,2)	(1,3)	(1,4)	(1,5)
(2,1)	(2,2)	(2,3)	(2,4)	(2,5)
(3,1)	(3,2)	(3,3)	(3,4)	(3,5)
(4,1)	(4,2)	(4,3)	(4,4)	(4,5)
(5,1)	(5,2)	(5,3)	(5,4)	(5,5)

Fig. 11 Matrix that contain distances between the corners

Fig. 12 Extra finger tips

of the finger tip is less than the y coordinate value of the corresponding point on the line, that finger tip is removed. Figure 14 shows the unwanted finger tips removed.



(a) (b)

Fig. 13 (a) Line drawn using centroid and orientation. (b) Line drawn perpendicular to the direction of orientation

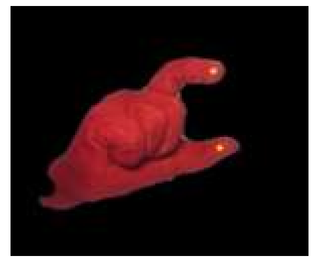


Fig. 14 Unwanted finger tips removed

D. *Recognition with PCA*

Principal Component Analysis is one of the most popular tools for high dimensional data analysis where dimensionality reduction is necessary to reduce the number of input variables in order to simplify the problems. Commonly, in PCA, one tries to find out a set of projections that maximize the variance of give data, or equivalently, that minimize the residuals of the projections. Let $X_{(d,n)} = \{x_1, x_2 \ldots x_n\}$ be the training data matrix with d the dimension of the data and n the number of training samples. Assume that the data has been centred by removing the mean. Covariance matrix is calculated as $C_x = X*X'$. C_y is the diagonal matrix whose elements along the main diagonal are the Eigen values, λ of C_x given by (3).

$$C_y = \begin{matrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_n \end{matrix} \qquad (3)$$

PCA is obtained by calculating the Eigen vectors of the covariance matrix of the current frame; the Eigen vectors are stored according to their corresponding Eigen values in decreasing order called as the feature vector as in (4). Finally the new data set is derived by multiplying the old data set by the feature vector which gives us the old data solely in terms of vectors we choose as in (5). For the recognised alphabet or number its image is also displayed and audio is also played. The results for still images using PCA are shown in figure 16. For live images, frames of a live video have been given as input. Every 20^{th} frame is given as input to be recognised. When the input alphabet or number is recognized, the image and audio of corresponding alphabet or number is introduced, so that it becomes easy for the common people to understand the meaning of the sign made. The result for recognition of the frames is 94%.

$$\text{FeatureVector} = (\text{Eigen1 Eigen2 Eigen3... EigenN}) \tag{4}$$

$$\text{Final data} = X' * \text{Feature vector} \tag{5}$$

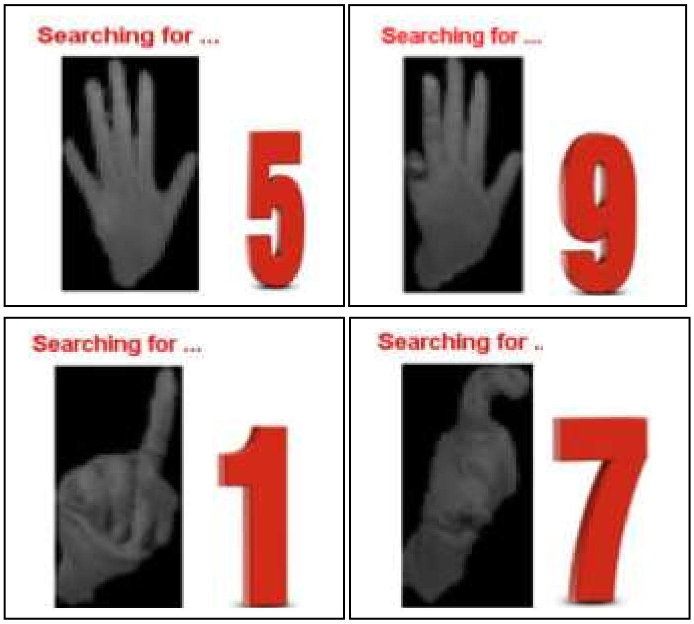


Fig. 15 Recognized signs with PCA.

V. Result

We presented a framework for Indian Sign Language Recognition system. Segmentation is color based i.e. Color the gloves (red and blue) are segmented. Recognition is done using PCA for live data frames where every 20^{th} frame is given as input to be recognized. The result of recognition is 94% i.e. the system can recognise 94% of the signs made correctly. The signs in which both the hands overlap each other and the signs which involves motion in them poses a problem in recognition. Finger tip algorithm is used to make database on the basis of number of finger tips. This can be combined with PCA in such a way that on the basis of number of finger tips in an image, PCA compares the input image with the new database formed with the help of finger tips.

VI. Conclusion and Future Work

This paper presents a system for recognition of Indian Sign Language. Recognition is successfully done for static signs using Principal Component Analysis. Further it has been proposed that recognition can be done using neural networks. Combining finger tip algorithm and PCA can prove to be a very efficient method to make the system more robust as

References

[1] N. Tanibata, N. Shimada, and Y. Shirai, *Extraction of hand features for recognition of sign language words*, In Proc. Intl Conf. Vision Interface, pages 391–398, 2002.

[2] S. Tamura and S. Kawasaki, *Recognition of sign language motion images*, In Pattern Recognition, volume 21, pages 343–353, 1988.

[3] Stephen Liwicki, Mark Everingham (2009), *Automatic recognition of finger spelled words in British sign language.*

[4] http://www.deafsigns.org/

[5] J. Ma, W. Gao, C. Wang, and J. Wu, *A continuous Chinese Sign Language recognition system*, International Conference on Automatic Face and Gesture Recognition, pages 428–433, 2000.