Int. Conf. on Signal, Image Processing Communication & Automation, ICSIPCA

Classification Approach for Sign Language Recognition

Srinath.S[1] and Ganesh Krishna Sharma[2]
[1]Sri Jayachamarajendra College of Engineering, Mysuru
srinath@sjce.mysore
[2]Sri Jayachamarajendra College of Engineering, Mysuru
ganeshkrishna.sh@gmail.com

***Abstract*— In recent years, a lot of research is being done in the field of Computer Vision and Human Computer Interaction where hand gestures play a vital role. Hand gestures or Hand signs are more powerful means of communication for hearing impaired people when they communicate to the normal people everywhere in day to day life. As the normal people find it difficult to interpret the meaning of sign language expressed by the hearing impaired, it is inevitable to have an interpreter for recognizing and translating the sign language. Computer recognition of sign language is an important research problem for enabling communication with hearing impaired people. This paper introduces a classification approach for recognizing and translating the static alphabets of American Sign Language (ASL). The images representing the alphabets are the input for the system which is obtained using digital camera. The output is the corresponding alphabet in the textual form. As the ASL uses only the palm for represent the alphabets, Segmentation technique is applied to the input image to obtain the palm and it is further processed using classification approach to recognize and translate them. This approach yields a success rate of 86.67%.**

***Index Terms*— American sign language, ASL recognition, hand extraction, palm extraction, boundary growing, classification.**

I. INTRODUCTION

A sign language is a language which uses manual communication and body language to convey meaning, as opposed to acoustically conveyed sound patterns. This can involve simultaneously combining hand shapes, orientation and movement of the hands, arms or body, and facial expressions to fluidly express a speaker's thoughts. They share many similarities with spoken languages (sometimes called "oral languages", which depend primarily on sound), which is why linguists consider both to be natural languages, but there are also some significant differences between signed and spoken languages.

Wherever communities of deaf people exist, sign languages develop. Signing is also done by persons who can hear, but cannot physically speak. While they utilize space for grammar in a way that spoken languages do not, sign languages exhibit the same linguistic properties and use the same language faculty as do spoken languages. Hundreds of sign languages are in use around the world and are at the cores of local deaf cultures. Some sign languages have obtained some form of legal recognition, while others have no status at all. A common misconception is that all sign languages are the same worldwide or that sign language is

Grenze ID: 02.ICSIPCA.2017.1.23

international. Aside from the pidgin International Sign, each country generally has its own, native sign language, and some have more than one, though sign languages may share similarities to each other, whether in the same country or another one. No one exactly knows how many sign languages there are [10].

American Sign Language (ASL) is a complete, complex language that employs signs made by moving the hands combined with facial expressions and postures of the body. It is the primary language of many North Americans who are deaf and is one of several communication options used by people who are deaf or hard-of-hearing and it is universally accepted sign language. Because of its signed modality, people often assume that ASL is merely a gestural representation of English. But ASL is a fully developed natural language, one of the world’s many sign languages. It is not a derivative of English. The static alphabets of ASL are represented in Figure 1.

While the Indian sign language use double hand notation, the ASL uses single hand notation. It is universally accepted as the standard sign language now and many schools for deaf/dumb teach ASL to their students. Hence this work is focused on translating ASL finger spelling to text since it is a language which is accepted as a standard one in most places and also the single hand notation makes the processing easier.

This paper presents an ASL recognizer that has been developed to recognize 18 alphabet of ASL using classification approach. In this recognition system, we have excluded the alphabets ‘J’ and ‘Z’ as they involve hand movements and also ‘closed fingers (a, e, m, n, s, and t)’, a class in the classification approach as these alphabets are very similar and are conflicting in nature. The recognition of other 18 alphabets yielded a success rate of 86.67%.

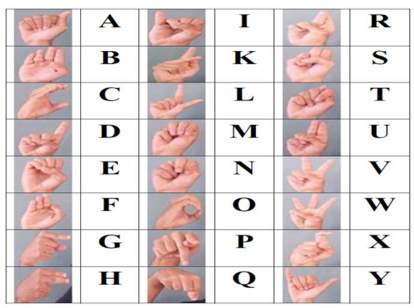

Figure 1. Static alphabets of ASL

II. Related Work

Artificial neural networks are widely used in sign language recognition research. Murakami and Taguchi [3] worked on the use of recurrent neural nets for Japanese Sign Language recognition. Although it achieved a high accuracy of 96%, their system was limited only to 10 distinct signs. Kramer and Leifer [4, 5] developed an ASL fingerspelling system using a Cyber glove, with the use of neural networks for data segmentation, feature classifier, and sign recognition. Using a tree-structured neural classifying vector quantizer, a large neural network with 51 nodes were developed for the recognition of ASL alphabets. They claimed a recognition accuracy of 98.9% for the system.

Research has been done in the area where finger detection has been accomplished via color segmentation and contour extraction [10]. But this technique requires fine-tuning every time the system switches to a new user as the color complexion varies from person to person. American Sign Language (ASL) is an efficient technique for communication among most of the 2 million deaf people in United States and Canada. ASL consists of about 6,000 signs for representing the commonly used words [6]. Wilbur [7] stated that most of signs in ASL could be considered as a combination of 36 basic hand shapes. These 36 hand shapes include most of ASL alphabets and their variations.

III. Proposed Method

In this approach the alphabets of the ASL are classified and categorized into two classes and further divided into two each sub categories is introduced. Different alphabets are oriented in different manner, so based on the orientation they are classified into different categories as shown in Figure 3.

The preprocessing involves converting the captured RGB image into YCbCr format and later to binary image. The noise is removed and the boundary image is obtained through different steps as shown using the block diagram in Figure 2.

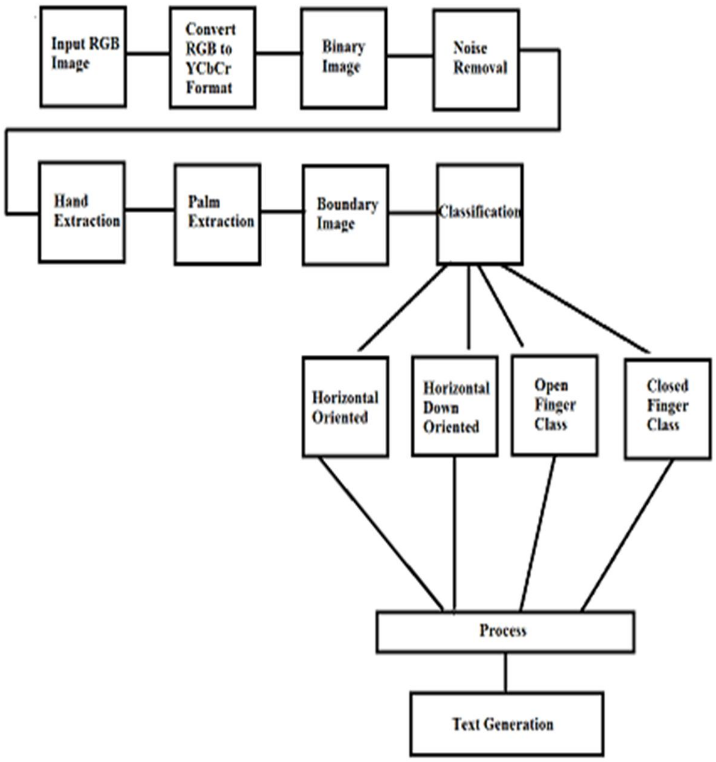

Figure 2. Block diagram of proposed method

In this approach, the closed finger class is not considered for recognition. Because they are conflicting in nature and their boundary images looks quite similar, which makes it very difficult to extract unique features and hence to distinguish between them.

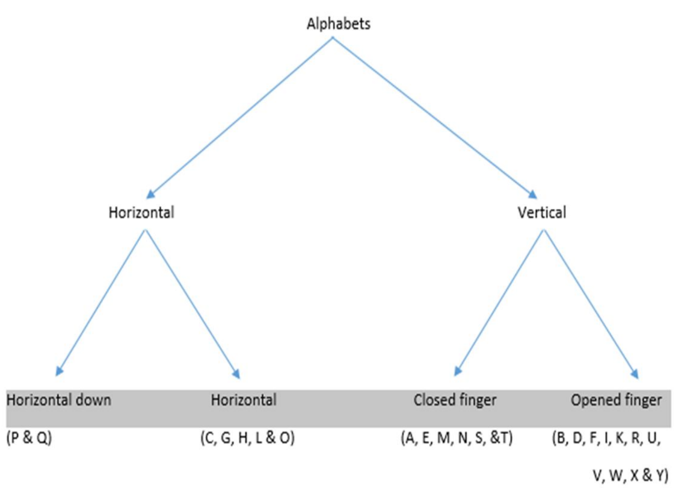

Figure 3. Tree structure of classification approach

A. Hand Extraction Algorithm

As this approach is concentrating on ASL alphabets only, which mainly depends on hand, it is aimed to extract only the portion of hand from the given image using the below algorithm.
Input: Captured RGB image.
Output: Binary image of Extracted hand.

Step 1: Start.
Step 2: Read the image.
Step 3: Convert the image to YCbCr format.
Step 4: Find the pixels that matches skin criteria using Cb and Cr plane range.
Step 5: Generate the binary image by setting the identified pixels to 1(White pixel).
Step 6: Eliminate noise by removing the smaller unwanted object.
Step 7: Retain only the hand object using its object index and Area.
Step 8: Extract the hand using BoundingBox property.
Step 9: End.

B. Palm extraction algorithm

As ASL uses only palm to represent the alphabets, only palm is extracted from the hand using the following algorithm. This algorithm extracts only the portion of palm from the hand which was extracted using the previous algorithm.
Input: Hand part of the image.
Output: Extracted and refined palm.

Step 1: Start.
Step 2: Read the extracted hand.
Step3: Initialize j = ((No_of_rows_of_hand) – (No_of_rows_of_hand/4)).
Step 4: While j > (No_of_rows_of_hand/4)
Search for a row “ROW_MIN” which contains minimum number of white pixels.
End while
Step 5: Discard all rows below the “ROW_MIN”.
Step 6: Refine the extracted palm using its BoundingBox property.
Step 7: End.

C. Classification algorithm

This algorithm classifies the alphabets of the ASL into different classes, by applying particular conditions and parameters that identifies corresponding classes.
Input: Boundary image of the refined palm. Output: Category to which the input sign belongs.

Step1: Start.
Step2: Read the boundary image of refined palm.
Step3: Calculate the x centroid.
Step4: If x centroid>33 units
Then set the horizontal flag to 1.
Calculate number of white pixels in the last row If white_pixel_count>2
Then set horizontal_down flag to 1. End if
Else
Calculate the area. If 125<area<174
Then set closed_finger flag to 1. Else
Set open_finger flag to 1. End if
End if
Step5: End.

D. Recognizing horizontally down alphabets

The algorithm recognizes and translates the alphabets of horizontally down oriented alphabets class. As the part of hand gesture representing alphabets P and Q occurs in the bottom part of the image this algorithm processes only that part to distinguish and identify these alphabets.
Input: Binary image of the refined palm.
Output: Alphabet corresponding to the input image.

Step1: Start.
Step2: Read the binary image of refined palm. Step3: Divide the image into four quadrants.
Step4: If number white pixels> number of black pixels in lower right quadrant (4th quadrant)
Then display the alphabet as ‘**Q**’.
Else
Display the alphabet as ‘**P**’. End if
Step5: End.

E. Recognizing horizontally oriented alphabets

The algorithm recognizes and translates the alphabets of horizontally oriented alphabets class. This algorithm applies different conditions to the alphabets O, (C&G) and (H&L) to recognize them. A part of this algorithm uses ‘bweuler’ function to recognize the alphabet ‘O’. The function returns total number of objects minus total no of holes in the given object and it is equal to zero only for alphabet ‘O’.
Input: Binary image of the refined palm.
Output: Alphabet corresponding to the input image.

Step1: Start.
Step2: Read the binary image of refined palm.
Step3: Apply bweuler function to the input.
Step4: If the function returns zero
Then display the alphabet as ‘**O**’. Else
Extract only upper right part
For i= (no_of_column/3) to (3*no_of_column/4) Search for white, black, white pixel sequence. If found
Then set CG_flag to 1 End if
End for End if
Step 5: If CG_flag is 1
Then calculate the number of white and black pixels in the upper half of image.
If no_white pixels are greater than the no_black pixels
Then display the alphabet as ‘**G**’. Else
Display the alphabet as ‘**C**’. End if
Else
Calculate y centroid. If ycentroid<35
Display the alphabet as ‘**H**’.
Else
Display the alphabet as ‘**L**’.
End if
End if
Step 6: End.

F. Boundary growing algorithm

This algorithm calculates the number of opened fingers by tracing the boundary image of the refined palm. It looks for considerable amount of variations between number of pixels in the upward movement and downward movement to identify opened finger.
Input: Boundary image of the refined palm. Output: Number of open fingers.

Step1: Start.
Step2: Read boundary image of refined palm.

Step3: Extract upper part of the boundary image
Step4: Find the starting point(x_min,y_min) by scanning the image from left to right and finding the column that contains at least one white pixel and the first white pixel
in that column becomes starting point.
Step5: Find the end point (x_max, y_max) by scanning from bottom to top and finding the row that contains at least two white pixels and rightmost white pixel is end point.
Step6: Initialize direction flag = 1 and finger_count to 0.
Step7: While x_min <x_max || y_min<y_max
While direction flag == 1
Search for next pixel by inspecting in the order upper right, upper, upper left pixel, next pixel.
If found
Then update x_min and y_min value to the corresponding pixel position.
Else
Set Direction flag to -1. End if
End while
While direction flag == -1
Search for next pixel by inspecting in the order bottom right, bottom, bottom left pixel, next pixel.
If found

Then update x_min and y_min value to the corresponding pixel position.
Else
If there is considerable amount of variations between number of pixels in the upward movement and downward movement
Then update the finger_count. Set the direction flag to 1.
End if End if
End while End while
Step8: End.

G. Recognizing vertically oriented open finger alphabets

This Algorithm recognizes and translates the alphabets belonging to the vertically oriented open fingers class. It also makes use of the previously mentioned Boundary growing algorithm for this purpose.

Input: Binary image of the refined palm.
Output: Alphabet corresponding to the input image.

Step1: Start.
Step2: Read binary image of refined palm
Step3: Read boundary image of refined palm.
Step4: Apply the boundary growing algorithm to binary image.
Step5: If retrace! = 1
If finger_count is 2
Then display the alphabet as ‘**V**’. Else if finger_count is 3
Then display the alphabet as ‘**W**’. Else
Extract $1/4^{th}$ upper part of the palm image
If xcentroid is less than 40% of total number of column
Then display the alphabet as ‘**I** ’.
Else if xcentroid is greater than 40% and less than 70% of total number of column
Then set BF flag to 1. Else
Set DKRU flag to 1.
End if End if
Step 6.a: If BF flag is 1
Extract upper half part of the image
If number of row with more white pixels is greater than 3/2 * rows with more black pixels
Then display the alphabet as ‘**B**’. Else
Display the alphabet as ‘**F**’. End if

End if
Step 6.b: If DKRU flag is 1
Extract 1st quadrant of the 2nd quadrant of the palm Search for rows containing at least one white pixel
If rows with no white pixel > rows with white pixel Then display the alphabet as ‘**D**’ and set DK flag to 1.
Else
Extract the 2nd quadrant of the 2nd quadrant of the palm
Search for no of row containing more no of black than white pixels
If rows with more black > rows with more white Then display the alphabet as ‘**K**’ and set DK flag to 1.
End if. End if.
If DK flag == 0
Extract the upper portion of the 2nd quadrant of the binary image.
If number of row with more white pixels > number of rows with more black pixels
Then display the alphabet as ‘**U**’. Else
Display the alphabet as ‘**R**’. End if.
End if. End if.
Else if retrace==1

If boundary growing algorithm retraces the path more than twice
Then display the alphabet as ‘**Y**’.
Else
Display the alphabet as ‘**X**’. End if.
End if.
Step 7: End .

IV. RESULTS EVALUATION

The success rate for recognition of each class in the classification approaches are represented using Table 1 to Table 3.

TABLE I. SUCCESS RATE FOR HORIZONTAL CLASS

Alphabets	Horizontal class					
	sample 1	sample 2	sample 3	sample 4	sample 5	Success rate
C	C	O	C	C	X	60
G	G	G	G	G	G	100
H	H	H	P	H	G	60
L	L	L	L	L	F	80
O	O	O	O	O	O	100

TABLE II. SUCCESS RATE FOR HORIZONTAL DOWN CLASS

Alphabets	Horizontal down class					
	sample 1	sample 2	sample 3	sample 4	sample 5	Success rate
P	P	X	P	P	P	80
Q	Q	Q	Q	Q	Q	100

TABLE III. SUCCESS RATE FOR OPENED FINGER CLASS

Alphabets	Opened finger class					
	sample 1	sample 2	sample 3	sample 4	sample 5	Success rate
B	B	B	B	B	B	100
D	D	D	D	D	D	100
F	F	F	F	F	F	100
I	I	I	I	I	I	100
K	D	K	K	K	C	60
R	R	K	R	R	F	60
U	U	U	U	U	F	80
V	V	V	V	V	V	100
W	W	W	W	W	W	100
X	X	X	X	X	X	100
Y	H	L	Y	Y	Y	60

Success Rate= (No of Recognized alphabets/No of total samples of that alphabet)*100%.

V. CONCLUSION AND FUTURE WORK

The current paper proposes a classification approach for the recognition of static ASL alphabets. This approach can be applied to the 24 alphabets (except J and Z which involve hand movements), but because of conflict raised due to similarities between the alphabets of 'closed finger' class i.e. the alphabets (a,e, m,n,s, and t) were not considered. The recognition of other 18 alphabets yielded a success rate of 86.67% which is quite satisfactory and efficient. The success rate presented in this paper work is based on 5 samples of 5 different people for each alphabet.

The approach must be extended for the remaining one class of closed fingers and it can be further processed to obtain 100% success rate. The method should be extended for recognizing and translating the hand signs involving hand movements. Also it should be able to translate videos with complex background. If an application can be developed to translate sign languages which can be installed on mobile phones, then this will reach the people very easily.

REFERENCE

[1] Real-Time Static Hand Gesture Recognition for American Sign Language (ASL) in Complex Background by Jayashree R. Pansare, Shravan H. Gawande and Maya Ingle.

[2] Point Pattern Matching Algorithm for Recognition of 36 ASL Gestures by Deval G. Patel.

[3] K. Murakami and H. Taguchi, "Gesture recognition using recurrent neural networks," in Proceedings of the Conference on Human Factors and Computing Systems, 1991, pp. 237-242.

[4] J. Kramer and L. J. Leifer, "A 'talking glove' for nonverbal deaf individuals," Technical Report No. CDR 19900312, Center for Design Research, Stanford University, U.S.A., 1990.

[5] J. Kramer, "The TalkingGlove (RTM): hand-gesture-to-speech using an instrumented glove and a tree-structured neural classifying vector quantizer," Ph.D. Thesis, Department of Mechanical Engineering, Stanford University, U.S.A., 1996.

[6] L. A. Martin and E. D. Sternberg, American Sign Language Dictionary, Revised edition, HarperCollins Publisher, New York, 1994.

[7] R. B. Wilbur, American Sign Language: Linguistic and Applied Dimensions, 2nd ed., College Hill Publication, Boston, 1987.

[8] Translation of sign language using generic Fourier descriptor and nearest neighbor by AbidatulIzzah and NanikSuciati, Informatics Department, Institute Technology Sepuluh, Indonesia.

[9] Hand Gesture Recognition Using Computer Vision by Ray Lockton, Balliol College, Oxford University.

[10] Wikipedia.

[11] www.Mathworks.com

[12] www.researchgate.net

[13] 'Digital Image Processing' by Gonzalez & Woods

[14] A Comparative Study of Different Image Features for Hand Gesture Machine Learning by Paulo Trigueiros, Fernando Ribeiro, and Luis Paulo Reis.

[15] Finger Detection for Sign Language Recognition by Ravikiran J, Kavi Mahesh, SuhasMahishi, Dheeraj R, Sudheender S, Nitin V Pujari