IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 10, October 2016

An Open CV Framework of Automated Sign language to Text translator for Speech and Hearing Impaired

Anchit Biswas[1], Shreedeep Gangopadhay[2]

PG, Department of Electronics and Communication, Techno India Salt Lake, Kolkata, India[1]

Assistant Professor, Department of Electronics and Communication, Techno India Salt Lake, Kolkata, India[2]

Abstract: Generally hearing impaired people use sign language for communication, but they find difficulty in communicating with others who don't understand sign language. This project aims to lower this barrier in communication. It is based on the need of developing an electronic device that can translate sign language into text in order to make the communication take place between the mute communities and the general public as possible. Computer recognition of sign language is an important research problem for enabling communication with hearing impaired people. This project introduces an efficient and fast algorithm for identification of the number of fingers opened in a gesture representing text of the Binary Sign Language. The system does not require the hand to be perfectly aligned to the camera and any specific back ground for camera. The project uses image processing system to identify, especially English alphabetic sign language used by the deaf people to communicate. The basic objective of this project is to develop a computer based intelligent system that will enable the hearing impaired significantly to communicate with others using their natural hand gestures. The idea consisted of designing and building up an intelligent system using image processing, machine learning and artificial intelligence concepts to take visual inputs of sign language's hand gestures and generate easily recognizable form of outputs. Hence the objective of this project is to develop an intelligent system which can act as a translator between the sign language and the spoken language dynamically and can make the communication between people with hearing impairment and normal people both effective and efficient. The system is we are implementing for Binary sign language but it can detect any sign language with prior image processing.

Keywords: Image Processing, Sign language, Pattern Reorganization, Colour Detection, Shape Detection, Text Generation, Open CV, C++.

I. INTRODUCTION

Hearing impaired people are usually deprived of normal communication with other people in the society. It has been observed that they find it really difficult at times to interact with normal people with their gestures, as only a very few of those are recognized by most people. Since people with hearing impairment cannot talk like normal people so they have to depend on some sort of visual communication in most of the time. Sign Language is the primary means of communication in the deaf and hearing impaired community and 2013 edition of **Ethnologue** lists 137 sign languages. As like any other language it has also got grammar and vocabulary but uses visual modality for exchanging information.

The problem arises when hearing impaired or deaf people try to express themselves to other people with the help of these sign language grammars. This is because normal people are usually unaware of these grammars. As a result it has been seen that communication of a hearing impaired person are only limited within his/her family or the deaf community. The importance of sign language is emphasized by the growing public approval and funds for international project. At this age of Technology the demand for a computer based system is highly demanding for the hearing impaired community. However, researchers have been attacking the problem for quite some time now and the results are showing some promise. Interesting technologies are being developed for speech recognition but no real commercial product for sign recognition is actually there in the current market.

The idea is to make computers to understand human language and develop a user friendly human computer interfaces (HCI). Making a computer understand speech, facial expressions and human gestures are some steps towards it. Gestures are the non-verbally exchanged information. A person can perform innumerable gestures at a time. Since human gestures are perceived through vision, it is a subject of great interest for computer vision researchers. The project aims to determine human gestures by creating an HCI. Coding of these gestures into machine language demands a complex programming algorithm. In our project we are focusing on Image Processing and Template matching for better output generation.

II. LITERATURE REVIEW

Maximum works are on glove using accelerometer. But these systems are very bulky as well as very costly and power consuming as they need many accelerometers. There are some latest works on image processing by detecting the hand using the color of the hand. But they all have done this in white background that they can detect the color of the hand properly. It means these techniques are error prone and difficult to apply in our real world because in real world all time white background is not possible. So it is very necessary to develop an electronic device that is not bulky, not costly and less error prone and that can be possible using good image processing technique.

III. IMPLEMENTATION

To archive the aim five different markers of different colors and different shapes have been used for the five fingers of a hand. The colors are ‘Triangle of Red’, ‘Pentagon of Blue’, ‘Hexagon of Green’, ‘Heptagon of Orange’, and ‘Decagon of Yellow’. Open CV (Open Source Computer Vision: http://opencv.org)) 3.00 library with Visual Studio Ultimate 13 in Windows 7 OS has been used.

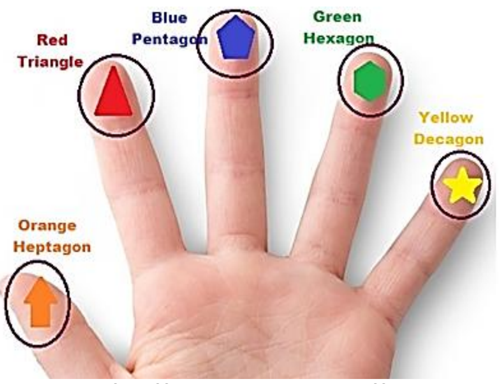


Fig.1: Markers of Different Color and Different Shape in a palm

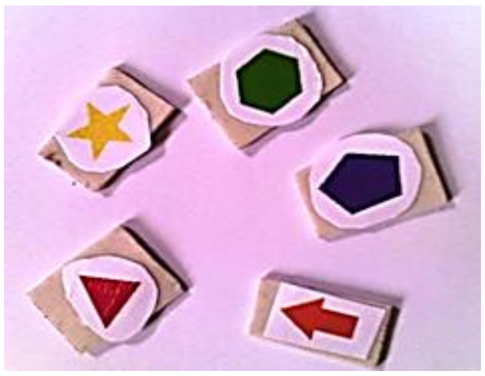


Fig. 2: Markers

The proposed system diagram consists of 3 main modules, one is hardware and other two are software modules. The first module is the ‘Image Capture Module’ which consists of a webcam which is mainly a hardware part; it captures the image of palm along with fingers. The second module, the most important part of the system diagram, is the ‘Image Processing Module’. It mainly consists of two parts, first is ‘Color Detection Module’ and next is ‘Shape Detection Module’. The ‘Image Processing Module’ takes the captured image m the Webcam and then it detects 5 different colors: Red, Blue, Green, Yellow and Orange and sets it into a single frame as a part of Color Detection Module. The Shape Detection Module then detects the shape from that new frame created by color detection module and marks the Triangle, Pentagon, Hexagon, Heptagon and Decagon. After detection of shape the ‘Pattern Generation Module’ the third and last module of the proposed system diagram, a software module, compares shape(s) combination(s), the newly generated pattern detected by the shape detection module, with the predefine patterns that is the combinations of shape(s). Then display the Text corresponding with the newly generated pattern which was created by the shape detection module actually the gesture of the palm using markers.

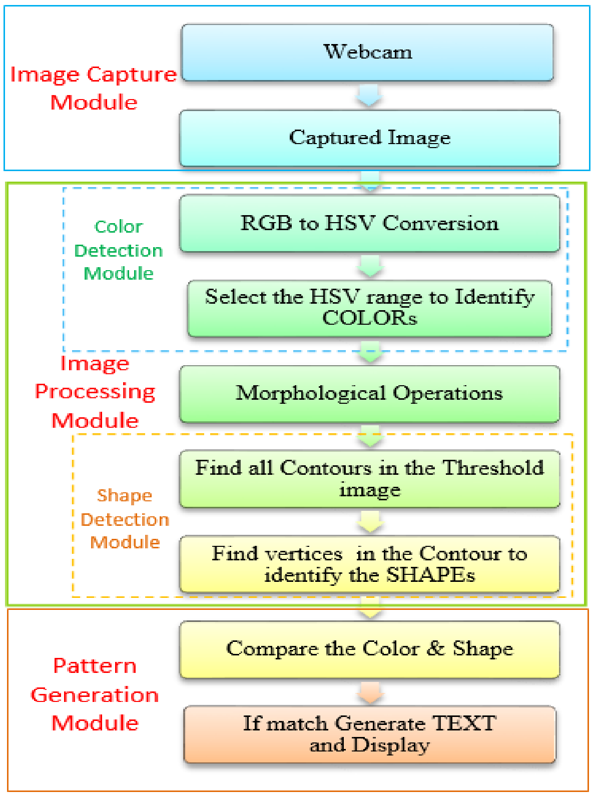


Image Capture Module
Webcam
Captured Image
Color Detection Module
RGB to HSV Conversion
Select the HSV range to Identify COLORs
Image Processing Module
Morphological Operations
Shape Detection Module
Find all Contours in the Threshold image
Find vertices in the Contour to identify the SHAPEs
Pattern Generation Module
Compare the Color & Shape
If match Generate TEXT and Display

Fig.3: System Flow Diagram

4.1 Image Capture Module:
The Camera Interface block is the hardware block that interfaces and provides a standard output that can be used for subsequent image processing. Thus to ensure the progress of the work, the camera (webcam) was interfaced with Open CV (Open Source Computer Vision: http://opencv.org)) 3.00 library with Visual Studio Ultimate 13 in Windows 7 OS.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

4.1.1Camera Specifications
An iball CHD20.0 Night Vision 16 MP Webcam with frame rate up to 30fps was used. The iball CHD20.0 120 MP Webcam comes with Night Vision feature. This webcam gives clear video imaging and can work even in the darkness. The night vision gives good images in the dark also. The upper portion of the webcam is movable depending on the need.

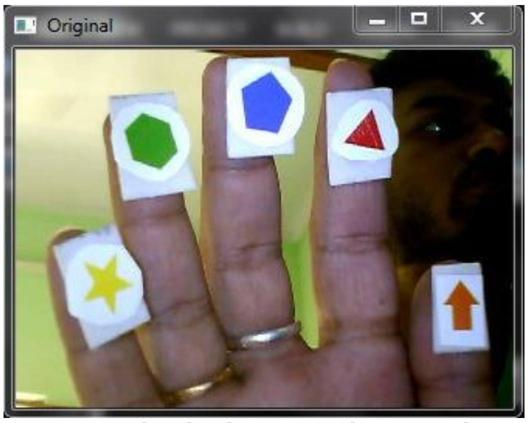


Fig.4: Image taken by the camera the Original Image

4.2 Image Processing Module:
This modules will analysis the colors, eliminate the noise and detect the shapes of the markers from the image captured by the webcam in real time.

4.2.1 Color detection Module:
In this module of the application will detect the colors of markers: Red, Green, Blue, Orange and Yellow.

4.2.1.1 RGB to HSV Conversion:
Open CV captured images are in BGR format. In other words, captured images can be considered as 3 matrices, BLUE, GREEN and RED with integer values ranges from 0 to 255. Usually, one can think that BGR color space is more suitable for color based segmentation. But HSV color space is the most suitable color space for color based image segmentation.

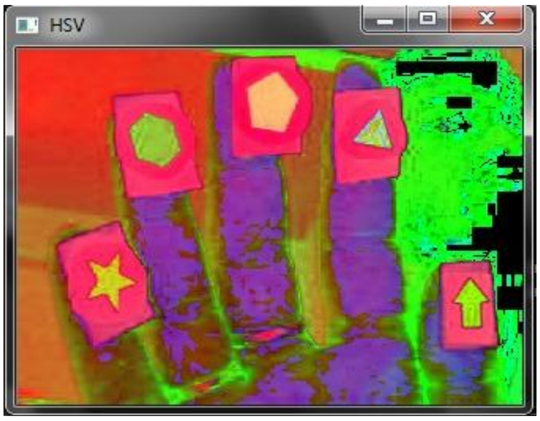


Fig.5: BGR Image (Fig.4)) Converted into HSV Image (Open CV cannot show the proper HSV Image as it converted from BGR not from RGB)

So, in this application, the color space of original image of the video has been converted from BGR to HSV image.

Because the RGB components of an object’s color in a digital image are all correlated with the amount of light hitting the object, and therefore with each other, image descriptions in terms of those components make object discrimination difficult. Descriptions in terms of hue/lightness/chrome or hue/lightness/saturation are often more relevant.

HSV color space is consists of 3 matrices, 'hue', 'saturation' and 'value'. In Open CV, value range for 'hue', 'saturation' and 'value' are respectively 0-179, 0-255 and 0-255. 'Hue' represents the color, 'saturation' represents the amount to which that respective color is mixed with white and 'value' represents the amount to which that respective color is mixed with black.

4.2.1.2 Identification of COLORs:
In order to detect the colors, each pixel of HSV Image (src) is compared with the corresponding predefine lower (lowerb) and upper (upperb) HUE, SATURATION and VALUE values. If the value in HSV Image (src) is greater than or equal to the value in lower (lowerb) and also less than the value in upper (upperb), then the corresponding value in Output Image (dst) that is Blue/Green/Yellow/Orange/Red (Fig: 7, 8, 9, 10, 11) will be set to 0xff (white); otherwise, the value in Output Image (dst) will be set to 0 (black).

This flow will repeat for 5 times, as in this application five different colors would be detected.

Parameters:

- src – first input array(HSV Image).
- lowerb – inclusive lower boundary array or a scalar(Different for each Color).
- upperb – inclusive upper boundary array or a scalar(Different for each Color).
- dst – output array of the same size as src.

The function checks the range as follows:

$$dst = lowerb \leq src \geq upperb$$

In this application, the coonsidered Hue values (empirical over standard) are:
- Orange 5-18
- Yellow 28- 38
- Green 45-65
- Blue 105-120
- Red 160-179

HUE is unique for that specific color distribution for that objects.
But SATURATION and VALUE may vary according to the lighting condition of that environment.

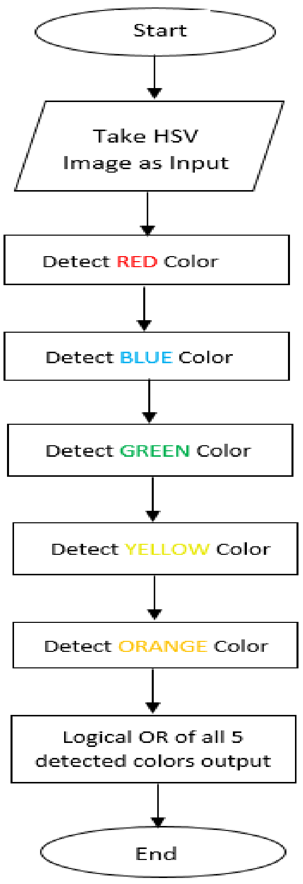


Fig.6 Flow chart of Color Detection

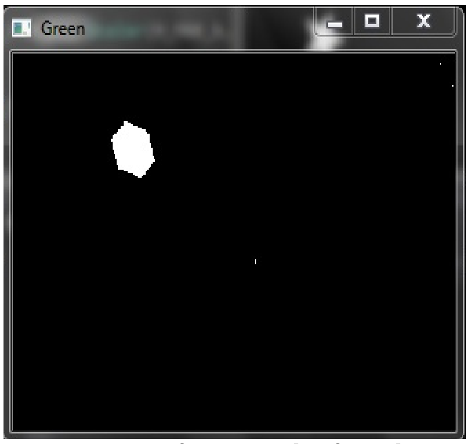


Fig.9: Detection of Green Color from the Fig.5

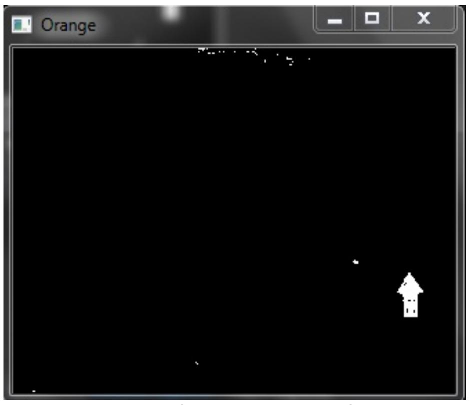


Fig.10: Detection of Orange Color from the Fig.5

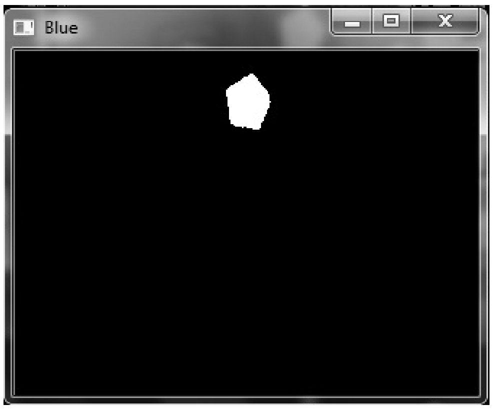


Fig.7: Detection of Blue Color from the Fig.5

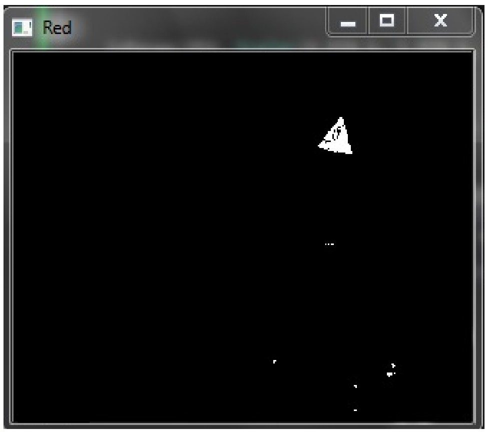


Fig.11: Detection of Red Color from the Fig.5

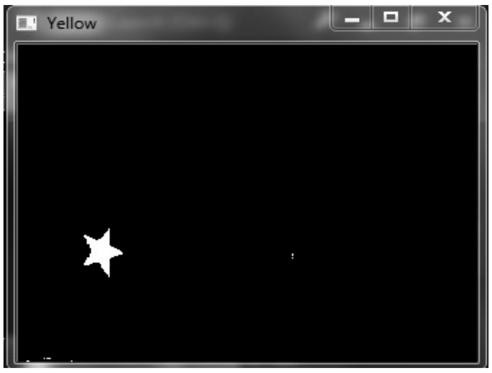


Fig.8: Detection of Yellow Color from the Fig.5

After detection of all the colors there are five different images for the five different detected colors (all are in black and white). All these images are composed into a single image (Fig.12) using logical OR operator, which can be an image that shows all the desired detected color into a single image frame. As the output of all the detected colors are in black (0) and white (0xff) so the OR of those will show all white part (detected parts) of all output image in a single image after using OR operator. Because OR of 0 and 1 always produce 1, in the black and white image the value of the black part is 0 and the value of white part is 1.

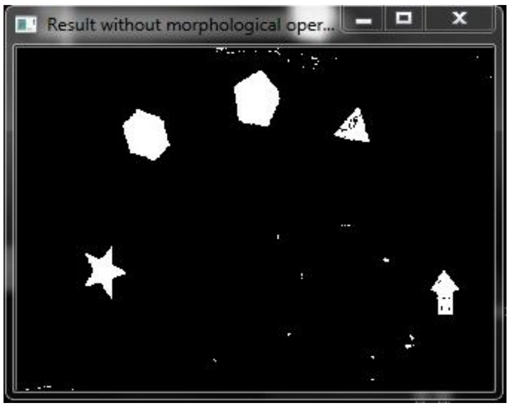


Fig.12: All the Detected Colors are in a same Frame after doing logical OR operation

4.2.2 Morphological Operation

In order to obtain a noise-free image, morphological image cleaning algorithm has been used. Morphological Operation needs to remove the noise from the foreground and from the main object. This morphological algorithm is implemented by logic operations. In this application morphological opening has been done followed by morphological closing.

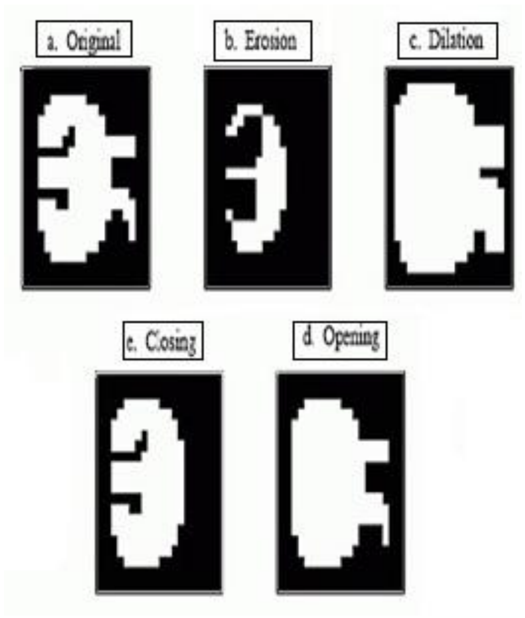


Fig.13: Description of Morphological operation

Figure (a) of Fig.13 is an example of binary image like Fig.12 Each pixel in the background is displayed as back, while each pixel in the object is displayed as white. The image is changed by the two most common morphological operations, erosion and dilation. In erosion, every object pixel that is touching a background pixel is changed into a background pixel. Erosion makes the objects smaller, and can break a single object into multiple objects. The erosion operation examines the value of a pixel and its neighbors and sets the output value equal to the minimum of the input pixel values. In dilation, every background pixel that is touching an object pixel is changed into an object pixel.

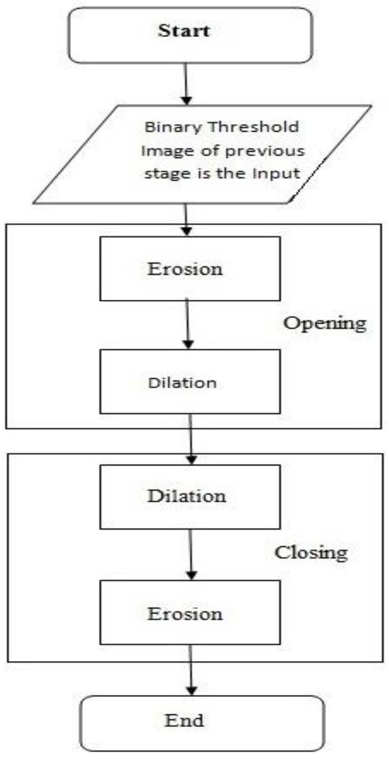


Fig.14: Flow chart of Morphological operation

Dilation makes the objects larger, and can merge multiple objects into one. Actually in dilation operation examines the value of a pixel and its neighbors and sets the output value the maximum of these pixels. The number of pixels added or removed from the objects in an image depends on the size and shape of the structuring element used to process the image. As shown in Figure (d) of Fig.13, closing is defined as erosion followed by dilation. Figure (e) of Fig.13 shows the opposite operation of opening, defined as dilation followed by erosion. Opening removes small islands and thin filaments of object pixels. Likewise, closing removes islands and thin filaments of background pixels. In each step it is required to evaluate the neighborhood of each pixel according to the specific operation.

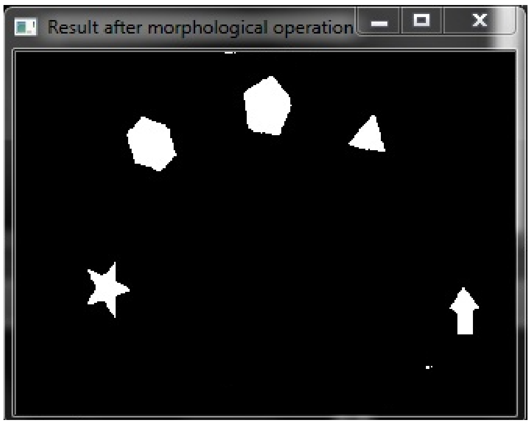


Fig.15: All the Detected Colors are in a same Frame after Morphological Operation

The Fig.15 is showing the noise free image after the morphological operation which had taken the noisy image Fig.12 as input of the morphological stage.

4.2.3 Identification of SHAPEs:
This module will detect the shapes of the detected colors that the applications can properly identify the Markers.

4.2.3.2 Find contours:
A contour is a list of points that represent, in one way or another, a curve in an image. This representation can be different depending on the circumstance at hand. There are many ways to represent a curve. Contours are represented in OpenCV by sequences in which every entry in the sequence encodes information about the location of the next point on the curve. A contour is represented in OpenCV by a CvSeq sequence that is, one way or another, a sequence of points. The function FindContours() computes contours from binary images. Contour tracing is one of many pre-processing techniques performed on digital images in order to extract information about their general shape.
Here the contour is drawn using the in the uncultured input image (webcam input) which have been detected from Morphological Operated Threshold Image (Fig.15). Here I have used RGB (0, 0, 0) which is Black to draw the contour.

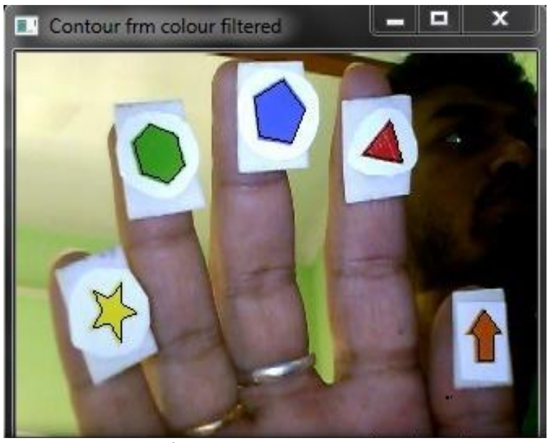


Fig.16: Contour from Color Filter Image (Fig.4.n) on Original Image on (Fig.4.c) using DrawContours()[24]

4.2.3.2 Identify Shapes:
Using contours with Open CV, we can get a sequence of points of vertices of each white patch (White patches are considered as polygons). For example, there will be 3 points (vertices) for a triangle, and 4 points for quadrilaterals. So, any polygon can be identified by the number of vertices of that polygon. Using the OpenCV function ApproxPoly().

In this application the Triangle, Pentagon, Hexagon, Heptagon and Decagon will be identified. I'll draw a line along the perimeter of every identified polygon with colors Green (RGB(0, 200, 0)) for Triangle, Yellow Ochre (RGB (255, 200, 0)) for Pentagon and Red (RGB(0, 0, 255)) for Hexagon, Aqua Blue (RGB (0, 200, 200)) for Heptagon, Pink (RGB (255, 0, 200)) for Decagon.

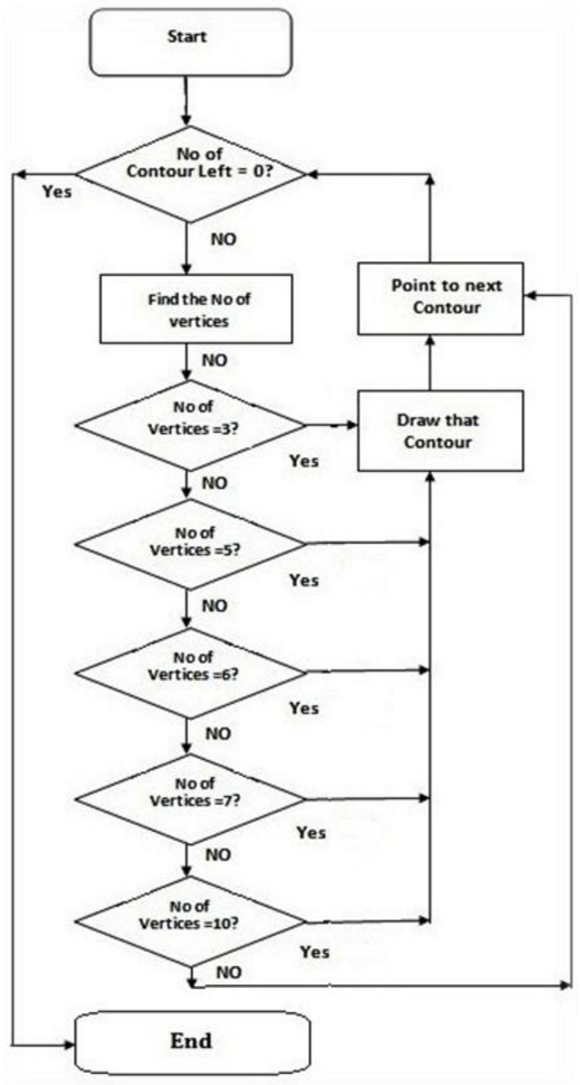


Fig.17: Flow chat for shape detection

And all the desired shapes have been in the image and marked with the corresponding colors.

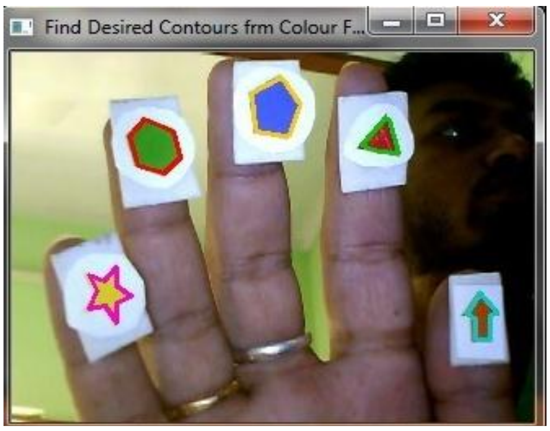
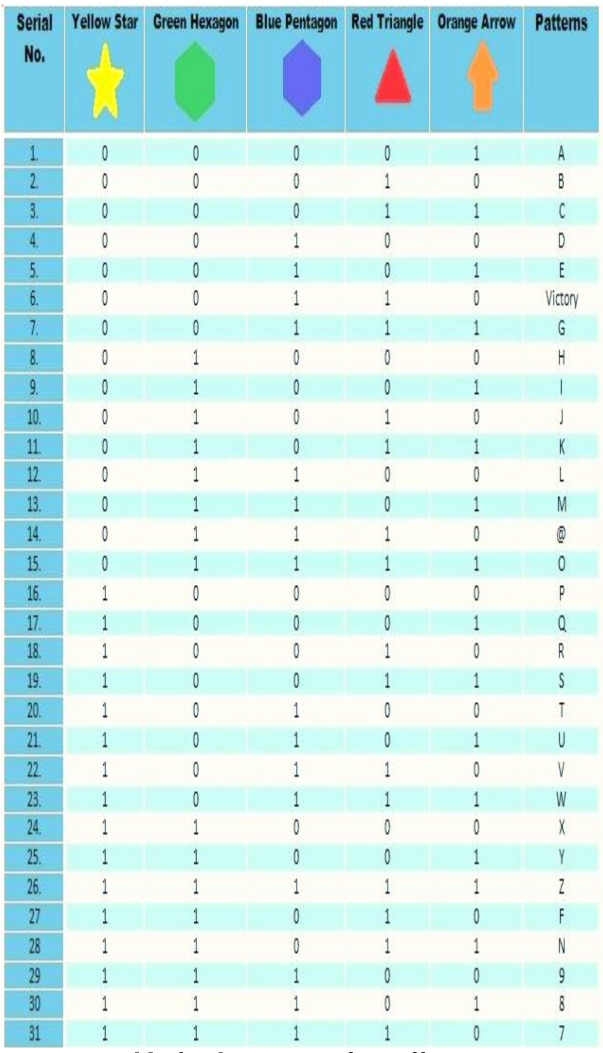


Fig.18: Detection of Shape from Contour by different Color

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 10, October 2016

4.3 Pattern Generation Module :
This module is responsible for comparing of markers and generation of pattern. Primarily this module compares the patterns of markers using the table (Fig.19).



Serial No.	Yellow Star ★	Green Hexagon ⬢	Blue Pentagon ⬟	Red Triangle ▲	Orange Arrow ⬆	Patterns
1.	0	0	0	0	1	A
2.	0	0	0	1	0	B
3.	0	0	0	1	1	C
4.	0	0	1	0	0	D
5.	0	0	1	0	1	E
6.	0	0	1	1	0	Victory
7.	0	0	1	1	1	G
8.	0	1	0	0	0	H
9.	0	1	0	0	1	I
10.	0	1	0	1	0	J
11.	0	1	0	1	1	K
12.	0	1	1	0	0	L
13.	0	1	1	0	1	M
14.	0	1	1	1	0	@
15.	0	1	1	1	1	O
16.	1	0	0	0	0	P
17.	1	0	0	0	1	Q
18.	1	0	0	1	0	R
19.	1	0	0	1	1	S
20.	1	0	1	0	0	T
21.	1	0	1	0	1	U
22.	1	0	1	1	0	V
23.	1	0	1	1	1	W
24.	1	1	0	0	0	X
25.	1	1	0	0	1	Y
26.	1	1	1	1	1	Z
27	1	1	0	1	0	F
28	1	1	0	1	1	N
29	1	1	1	0	0	9
30	1	1	1	0	1	8
31	1	1	1	1	0	7

Fig.19: Table for the Patterns for Different gestures

It is possible to display total (2^5-1) i.e.31 gestures' using the 5 different shapes in the 5 fingers of a single hand, and ($2^{10}-1$) i.e. 1023 gestures if both hands and 10 different shapes are used. The gestures can be for alphabets, numbers, text or special characters.

In this application, 5 different markers have been used for a single hand and only 31 gestures have been taken for 31 patterns. It takes the data from the previous module that 'Identify Different Shapes' and then compare it with the following table. If a particular shape is found then it assign the value for the shape is binary '1' if not found then assign it binary '0'. Depending upon the values from the previous stage it assigns all the values for all the shapes either '1' or '0', then compare those values after completion of all the contours of the frame and then it display the predefine Text (patterns) for the corresponding values only if present in the table(Fig.19).

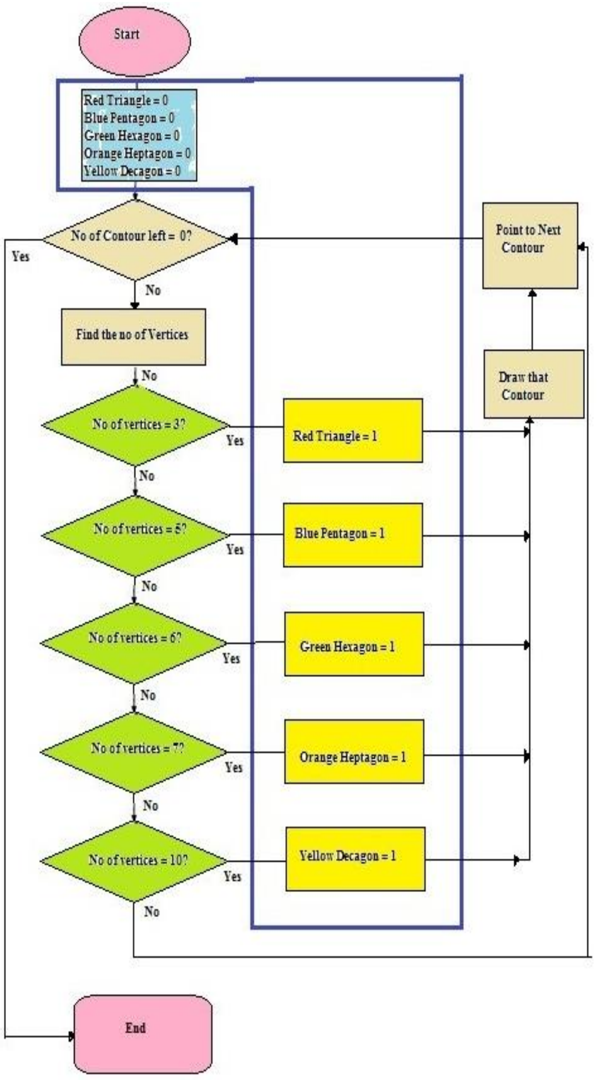


Fig.20: Flow chat for pattern detection table generation after shape detection. The blue box is only done in this module and the rest done shape detection module

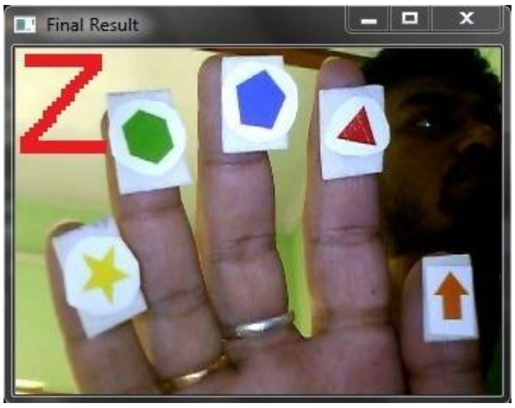


Fig.21: Displaying of Text

As an example if all the markers are detected then it displays 'Z' as per previous table (Fig.4.q). Here Fig.4r is displaying 'Z' as all the markers have shown here and then previous modules also detected all the colors and shapes of the markers.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 10, October 2016

IV. RESULTS

5.1 Results of Color Detection
The following images (Fig.22) detect Color(s) which is then shown in 'Result Box' after all morphological operations. 'Original Image Box' is the image taken by the webcam real time; 'HSV Box' is the image after conversion of original image into HSV for the color detection module, 'Red Box', 'Green Box', 'Yellow Box', 'Blue Box' and 'Orange Box' showing the detection of respective colors Red, Green, Yellow, Blue, and Orange. 'Result Box' shows the final result after adding all the detected colors into single frame after all morphological operations.

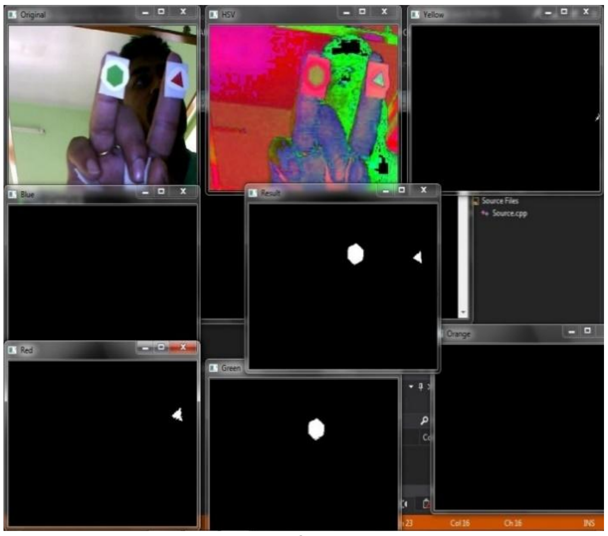


Fig 4.22: Detection of Green and Red Color

5.2 Results of Shape Detection and Pattern Generation:
The following image (Fig.23) showing the result of shape detection and generate predefined Alphabet. The 'Original Image Box' showing the real time captured image. On the other hand 'Color Filtered Threshold Image Box' shows only the detected colors' threshold image.

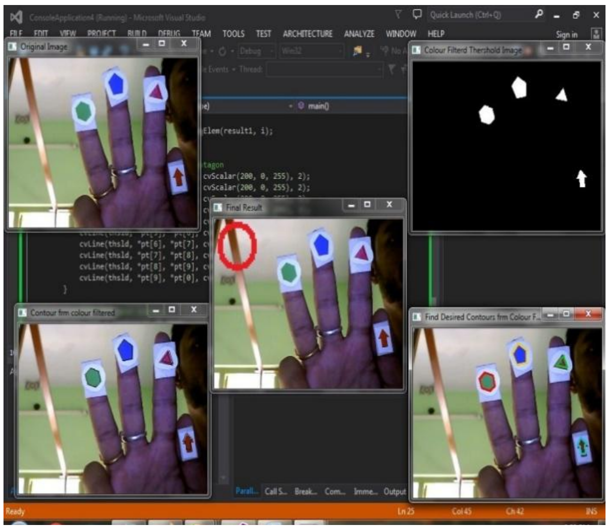

ConsoleApplication4 (Running) - Microsoft Visual Studio
Original Image
Colour Filtered Thershold Image
Final Result
Contour frm colour filtered
Find Desired Contours frm Colour F...

Fig. 23: Displaying O

The 'Contour from color filtered Box' shows the contour of the image of 'Color Filtered Threshold Image Box' which has been drawn on original Image using Black (RGB (0, 0, 0)). Then draw the desired shape using desired colors these are Green (RGB (0, 200, 0)) for Triangle, Yellow Ochre (RGB (255, 200, 0)) for Pentagon and Red (RGB (0, 0, 255)) for Hexagon, Aqua Blue (RGB (0, 200, 200)) for Heptagon, Pink (RGB (255, 0, 200)) for Decagon colors on original image in 'Find the Desired Contour from the Color filtered Box'. 'Final Result Box' displays the desired predetermined alphabet.

V. CONCLUSION

An Open CV Framework of Automated Sign language to Text translator for Speech and Hearing Impaired is based on the need of developing an electronic device that can translate sign language into text which works properly as it detects the colors of markers first then detects the shapes of the markers using image processing in order to properly identify the markers and generates predefined pattern.

It is tested with different gestures and is able to classify and display every pattern correctly and it does not require any special background, at any background it is capable to display proper pattern. But to detect the colors of markers properly in the color detection module we need to adjust saturation and value according to the environment. As saturation and value may be vary according to the lighting condition of that environment. In order to perform this application the markers should not be very far from the webcam; in that case it cannot detect the shapes of markers properly.

VI. FUTURE SCOPE

1. In this application text has been generated only from sign language. In future if it can generate sound then the speech and hearing impaired can communicate with visually impaired also.
2. The application is PC based but it can also be created in Android Smart Phone as Open CV can integrate with Android as well.
3. The application can have the capability of sharing, hosting and downloading the detected sign language on cloud that any one from anywhere can easily communicate with mute community at real time.
4. Secured Communication using Encryption can be done for better security.

REFERENCES

[1] "Intelligent Sign Language Recognition Using Image Processing", Sawant Pramada1, Deshpande Saylee 2, Nale Pranita3, Nerkar Samiksha4 Mrs.Archana S. Vaidya5 IOSR Journal of Engineering (IOSRJEN) e-ISSN: 2250-3021, p-ISSN: 2278-8719 Vol. 3, Issue 2 (Feb. 2013), ||V2|| PP 45-51
[2] "Gesture Controlled Robot using Image Processing, Harish Kumar Kaura1", Vipul Honrao2 , Sayali Patil3 , Pravish Shetty4, (IJARAI) International Journal of Advanced Research in Artificial Intelligence, Vol. 2, No. 5, 2013

[3] "Sign Language Translation", Juniar Prima Rakhman1), Nana Ramadijanti, S.Kom, M.Kom2), Edi Satriyanto S.Si, M.Si3) Jurusan Teknik Informatika, PENS –ITS Surabaya Kampus ITS Sukolilo,
[4] "Colour Object Tracking On Embedded Platform Using Open CV", Krutika A Veerapur, Ganesh V. Bhat, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-2, Issue-3, July 2013M. Wegmuller, J. P. von der Weid, P. Oberson, and N. Gisin, "High resolution fiber distributed measurements with coherent OFDR," in Proc. ECOC'00, 2000, paper 11.3.4, p. 109.
[5] "Gesture Recognition using Image Processing and conversion of Text and speech", Kajal B. Borole, Bhagyashree D. Patil, H. D. Gadade, International Journal of Exploring Emerging Trends in Engineering (IJEETE) Vol. 03, Issue 02, Mar-Apr, 2016 Pg. 57-66 (2002) The IEEE website. [Online]. Available: http://www.ieee.org/
[6] "Hand-talk Gloves with Flex Sensor: A Review", Ambika Gujrati1, Kartigya Singh2, Khushboo3, Lovika Soral4, Mrs. Ambikapathy, International Journal of Engineering Science Invention ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726/
[7] "Remote accelerometer based hand gesture recognition ", Jing Pang, Department of Electrical and Electronic Engineering California State University, USA, International Conference and Exhibition on Biosensors & Bioelectronics May 14-16, 2012 Embassy Suites Las Vegas, USA "PDCA12-70 data sheet," Opto Speed SA, Mezzovico, Switzerland.
[8] http://docs.opencv.org/2.4/genindex.html
[9] Special Imaging Techniques / Morphological Image Processing (http://www.dspguide.com/ch25/4.htm)
[10] Book : OReilly Learning OpenCV , Gary Bradski and Adrian Kaehler

BIOGRAPHIES

Anchit Biswas received his B.Tech degree in Electronics and Communication from JIS College of Engineering, Kalyani, West Bengal in 2011 and M.Tech in VLSI and Micro-Electronics from Techno India, Salt Lake, Kolkata, West Bengal in 2016. He have more than 3 years' of experience of college level teaching. His area of interest includes Image Processing, Embedded System, IOT, C, and C++.

Shreedeep Gangopadhay, currently Assistant Professor, Department of Electronics and Communication, Techno India Salt Lake, Kolkata, West Bengal. He received his B.Tech from Heritage Institute of Technology, Kolkata, West Bengal in 2006 and M.E in Digital Systems & Instrumentation from Indian Institute of Engineering Science and Technology, Shibpur in 2008. His area of interest includes FPGA, Embedded System, VLSI and IOT.