**SPEAKING SYSTEM FOR MUTE PEOPLE USING IMAGE PROCESSING**

**Author**

Mohammed Hanzala – 4NM15EC088 Email: [hanzalas1997@gmail.com](mailto:hanzalas1997@gmail.com)

K Vignesh Kamath – 4NM15EC075 Email:[Vigneshkamath242av@gmail.com](mailto:Vigneshkamath242av@gmail.com)

Karthik U K – 4NM15EC078 Email: [karthikuk8@gmail.com](mailto:karthikuk8@gmail.com)

Gautham Rao B – 4NM15EC059 Email: [gautham457@gmail.com](mailto:gautham457@gmail.com)

# *Abstract*

*The project aims at helping mute people in communicating with others feasibly using sign language The project uses image processing to identify, especially English alphabetic sign language used by mute people to communicate and converts them into speech so that normal people can understand. The image is taken from the Raspberry Pi camera and then is converted to grayscale. Further grayscale is converted into binary image. Set a threshold so that the pixels that are above certain intensity are set to white and those below are set to black. From binary image,* *we generate the coordinates of the image, these coordinates are then compared with the stored coordinates in the database for the purpose of output generation using pattern matching technique. Here pattern matching algorithm is used. If the pattern is matched, the alphabet’s letter audio corresponding to image given to a speaker.*

**Keywords :**

***Binary image,Database,Grayscale image,Image processing,OpenCV,Pattern matching algorithm,Raspberry Pi***

**1.Introduction**

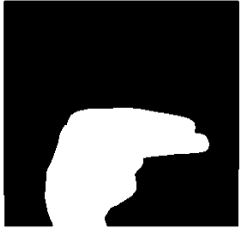
Sign languages  are [languages](https://en.wikipedia.org/wiki/Language) that use [manual communication](https://en.wikipedia.org/wiki/Manual_communication) to convey meaning. This can include simultaneously employing hand gestures, movement, orientation of the fingers, arms or body, and [facial expressions](https://en.wikipedia.org/wiki/Facial_expressions) to convey a speaker's ideas. Sign languages often share significant similarities with their respective spoken language, such as [American Sign Language](https://en.wikipedia.org/wiki/American_Sign_Language) (ASL) . This project focuses on the recognition of letters from the American Sign Language (ASL) . Recognition is done using Contour Analysis and Feature Extraction techniques. The project involves various algorithms in finding contours and OpenCV techniques. The project is coded in Python, using OpenCV 3.2.0 on Raspberry Pi 3 model B.

**2. Methods**

The main aim of this project is to design a program for an errorless extraction of human hand properties based on OpenCV.The algorithm used in this project is loosely based around visually based interfaces.The human hand gestures made by the user is recognized by the lens of the camera,which is interfaced to Raspberry Pi ,the gestures are then converted to it’s digital representation.The image is a matrix of scalar or vector values,which can be processed in numerous ways.This project uses OpenCV to do the processing of these values.The Contour of object is defined by set of points, which describe the edge of object, which is the outline.OpenCV library offers very efficient implementation of contour finding algorithm, which contains additional features like extraction of contours in hierarchy and approximation of found contours. Approximation of line contour with set of points is very handy feature and significantly speeds up further processing of contour. Such contour, represented as set of points, can be enclosed into n-dimensional polygon, also known as hull.

**2.1 The Threshold Image**

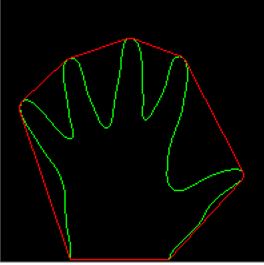
The creation of the threshold image is very important for the detection of the hand gestures.The essential part is to separate the foreground and the background because our area of interest is only the features of the hand.



**Figure 1. Threshold image of letter “H”**

**2.2 Obtaining Contours**

The Hand is identified using an inbuilt function that finds Contours,the libraries which OpenCV provides.The function later ,then returns an array of co-ordinates of the formation of the Contour.



**Figure 2. Contour of Hand**

**2.3 Convex Hull and Convexity Defects**

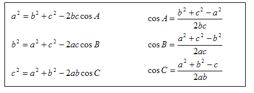
Hull, as the geometrical shape, can be concave or convex polygon. We can say that the hull is convex, when it is not possible to draw a line inside polygon which would intersect its border. If it is possible, then the polygon is not convex and therefore contains convexity defects. The data from the Contour Analysis is manipulated to obtain convexity defects.Based on the value of this,we can Identify how many fingers are present.This is valuable information as it provides information as to which Letter is corresponding. The number of contour defects is calculated by the following process .

We compute a triangle . Let the sides be “a” ,”b” and “c” . This triangle is formed by the starting point of the contour , the ending point of the contour and the farthest point of the contour . (a , b , c respectively ) . “a” is computed as follows

a = math.sqrt((end[0] - start[0])\*\*2 + (end[1] - start[1])\*\*2) [7]

Similarly , b and c are also calculated .

Now, using the Cosine rule ,



**Figure 3. Cosine Rule**

The angle A is calculated .

If the angle A is less than or equal to 90 degrees , it means that there is a convexity defect . Once there is a convexity defect recognised , a variable by the name “cnt” increments by one .

So , by this algorithm we can efficiently identify how many convexity defects there are.

**2.4 Identification of Letters**

Letter A: For identifying A , we computed the difference between the area of a circle and the area of the contour . The circle is obtained by bounding the contour . The reason this method is adopted for A is that there is very little difference between the two areas which makes the Letter A stand out from the other letters . Hence this algorithm was found to be very efficient .

Letter B: For the letter B , we computed the contour area . This method is adopted because the Letter B has the largest area among the other letters .

Letters V, C, L, Y: This part gets executed when the Letter A fails . If the number of convexity defects are equal to 1 , the following algorithm is employed .

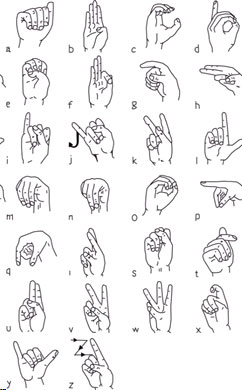
The “angle” is calculated . This entity is obtained by an OpenCV inbuilt function that calculates the overall figure’s orientation , giving us an angle . Based on the values of the angle , Letters V , C, L and Y are identified .

Letters F and W: Letters F and W are the only alphabets in the American Sign Language to have 2 convexity defects . Once 2 convexity defects are identified , the angle is compared . Hence the Letters F and W are identified in this manner.

Letters D, J, H, I, U: A combination of parameters are computed to identify these letters ,

Solidity , Aspect ratio and Angles are computed .

The figure 4 shows the standard signs of letters in American Sign Language(ASL)

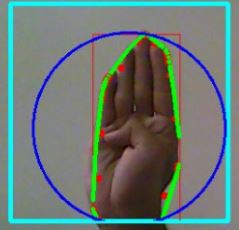


**Figure 4. Letter Signs Of American Sign Language**

**2.5 Contour Properties**

Many contour properties are analyzed and processed to identify the letter made by the user’s hand gestures,shown in figure 5.

1. Aspect Ratio
2. Perimeter of Contour
3. Area of Contour
4. Equivalent Diameter of Contour Area
5. Solidity
6. Number of Convexity Defects
7. Angle
8. Bounding Rectangle Area
9. Moments



**Figure 5. Analysis of Hand Gesture with various methods**

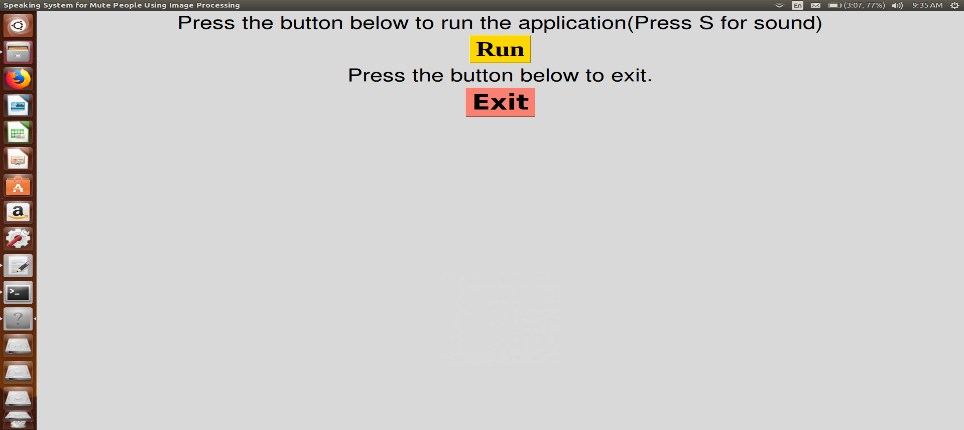
**2.6 Audio Output Of The Identified Letter**

After the identification of a Letter,the identified letter’s audio is played by using an audio file,this is implemented using a bash script ,For example, The letter “U” is identified, a directory is made and we navigate into that directory . Then a “.txt” file is made containing the letter “U”. If U already exists , it is overwritten .Next,A Shell Script is used which goes into the directory and finds the last modified file , using its corresponding time stamp . Once the last modified file is found , The script plays the corresponding audio file .

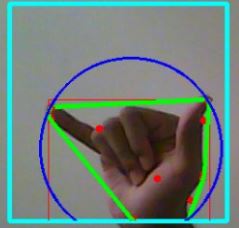
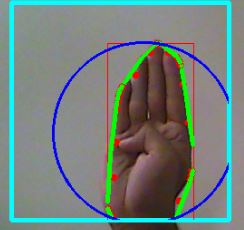
**3. Results and Discussion**

The system has been implemented using Raspberry Pi board with Linux operating system and ARM CORTEX processor.The python programming is used to implement the system.The camera system is connected to the board using USB.

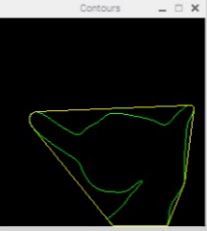
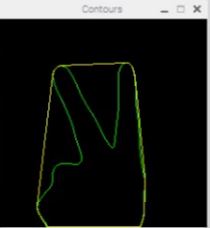
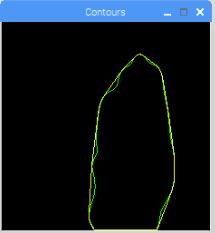
When the code is run,we get the graphical window shown in figure 6,this graphical interface is actually an application,that startups on the successful running of the code.On clicking the “Run” button ,the camera is turned on,the threshold and the contour images of the objects detected are formed.The input to the system is shown in figure 7.The signs of letters B,V and Y are shown in figure 7.The contour of the image then obtained is shown in figure 8.The binary images are shown in figure 9.The generated coordinates of the image,are then compared with the stored coordinates in the database for the purpose of output generation using pattern matching technique. Here pattern matching algorithm is used. If the pattern is matched, the corresponding letter’s text is obtained.the output images of letters B,V and Y are shown in figure 10. And the audio corresponding to image and the letter is given to a speaker.



**Figure 6.Graphical Interface**



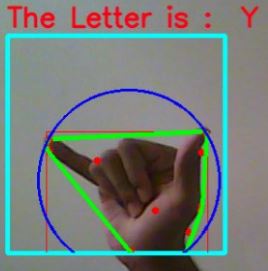
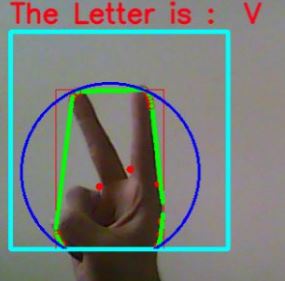
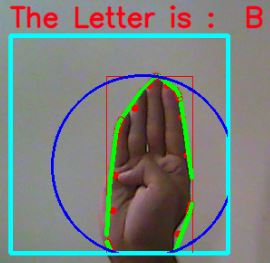
**Figure 7.Input hand signs for letters B,V and Y respectively**

****

**Figure 8.Contours of signs for letters B,V and Y respectively**

****

**Figure 9.Threshold images of signs for letters B,V and Y respectively**

****

**Figure 10.Output image with text display of the letters B,V and Y respectively**

**4. Conclusion**

The principal goal of this project was to design the algorithm for tracking and extraction of features of object, specifically human hand in video based on Open CV. Few Letters are currently being able to be recognised . Further research into this project can be done. The advantage of this solution is that it does not require any special inputs. Ordinary digital image of human hand is sufficient and algorithm can process stream of such images in real time.

**References**

1. Russ, John C. *The image processing handbook*. CRC press, 2016.
2. Bradski, Gary, and Adrian Kaehler. "OpenCV." *Dr. Dobb’s journal of software tools* 3 (2000).
3. Parker, Jim R. *Algorithms for image processing and computer vision*. John Wiley & Sons, 2010.
4. Bradski, Gary, and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. "O'Reilly Media, Inc.", 2008.
5. Jensen, John R., and Kalmesh Lulla. "Introductory digital image processing: a remote sensing perspective." (1987): 65-65.
6. Bovik, Alan C. *Handbook of image and video processing*. Academic press, 2010.
7. Prashanth, H. S., H. L. Shashidhara, and Balasubramanya Murthy KN. "Image scaling comparison using universal image quality index." *Advances in Computing, Control, & Telecommunication Technologies, 2009. ACT'09. International Conference on*. IEEE, 2009.
8. Kriegman, David J., and Jean Ponce. "On recognizing and positioning curved 3-D objects from image contours." *IEEE Transactions on Pattern Analysis and Machine Intelligence*12.12 (1990): 1127-1137.
9. Pansare, Jayashree R., Shravan H. Gawande, and Maya Ingle. "Real-time static hand gesture recognition for American Sign Language (ASL) in complex background." *Journal of Signal and Information Processing* 3.03 (2012): 364.
10. <http://docs.opencv.org/3.0-beta/doc/tutorials/tutorials.html>
11. <https://docs.scipy.org/doc/numpy/>
12. <https://docs.python.org/2/library/tkinter.html>
13. <https://www.pygame.org/docs/>
14. <http://effbot.org/imagingbook/pil-index.html>
15. <https://pyglet.readthedocs.io/en/pyglet-1.2-maintenance>
16. <https://docs.python.org/2/library/math.html>
17. <https://www.pyimagesearch.com/2017/09/04/raspbian-stretch-install-opencv-3-python-on-your-raspberry-pi/>