

Hand Gesture Recognition System For Multimedia Applications

Neha S. Rokade¹, Harsha R. Jadhav², Sabiha A. Pathan³, Uma Annamalai⁴

¹BE(Student) Department Of Computer Engineering, GESRHSCOE, Nashik, Maharashtra, India

²BE(Student) Department Of Computer Engineering, GESRHSCOE, Nashik, Maharashtra, India

³BE(Student) Department Of Computer Engineering, GESRHSCOE, Nashik, Maharashtra, India

⁴BE(Student) Department Of Computer Engineering, GESRHSCOE, Nashik, Maharashtra, India

Abstract –

Hand gesture recognition in aspect to human-machine interface is being developed vastly in recent days. Because of the disturbance of lighting and background being not plain, many visual hand gesture recognition systems operate or show successful results only in restricted background. To recognize the various hand gestures, we will build a non complex and with greater speed motion history image related system. In our system, we mainly focus on applying pointing behavior for the human machine interface. Now days, the gesture recognition has been a new developmental and experimental thing for most of the human related electronics. This system allows people to operate electronic products more conveniently. In our system, a gesture recognition method is to be build which will be an interface between human machine interaction i.e. HMI. In our system we propose some non-complex algorithm and hand gestures to decrease the hand gesture recognition complexity and would be more easy and simple to control real-time computer systems.

Key Words:

Human-Computer interaction, Gesture Recognition.

1. Introduction

With the development in Computer Vision and Human Machine Interaction the Computer holds most important role in our daily life. Human Computer Interaction can provides several advantages with the introducing the different natural forms of device free communication. Gesture recognition is one of the several types of them to interact with the humans gestures are the natural form of action which we often used in our day to day life. But in computer application to interact humans with machine the interaction with devices like keyboard, mouse etc. must be requires. As the various hand gestures are frequently used by humans so the aim of this project is to reduce external hardware interaction which is required for computer application, and hence this causes system more reliable for use with ease. This paper implements gesture based recognition technique to handing multimedia application. In this system, a gesture recognition scheme is been proposed as an interface between human and machine. In our system we represent some low-complexity algorithm and some hand gestures to decrease the gesture recognition

complexity and which becomes easier to control real-time systems.

2. Existing System

The various hand gesture recognition systems are developed for various applications. The systems are based on vision, facial gestures, hand gestures etc.

In 2015, proposed the system which uses kinect depth camera. It is based on a compact representation in the form of super pixels, which efficiently capture the shape, texture and depth features of the gestures. Since this system uses kinect depth camera, the cost of system is more.

In 2014, the proposed systems focus on using pointing behaviors for a natural interface to classify the dynamic hand gesture, they developed a simple and fast motion history image based method. This paper presents low complexity algorithm and gestures recognition complexity and more suitable for controlling real time computer system. It is applicable only for the application of power point presentation.

In 2014, this system uses various hand gestures as input to operate the windows media player application. This system uses single hand gestures and its directional motion which defines a particular gesture for the above mentioned application. In this system decision tree has been used for classification. This system only supports windows media player application and not any others.

3. System Architecture

The Figure 3.1 shows overall architecture of our system. First the system will take the input from the user by using webcam i.e., the original image. The image will be converted into HSV scale in parallel approach. The HSV image is then transformed into threshold image. In thresholding the biggest contour is found out to matches the hand border. After that we calculate the center of gravity by using some formulae. Then we numbered that detected fingertips as output shown in figure.

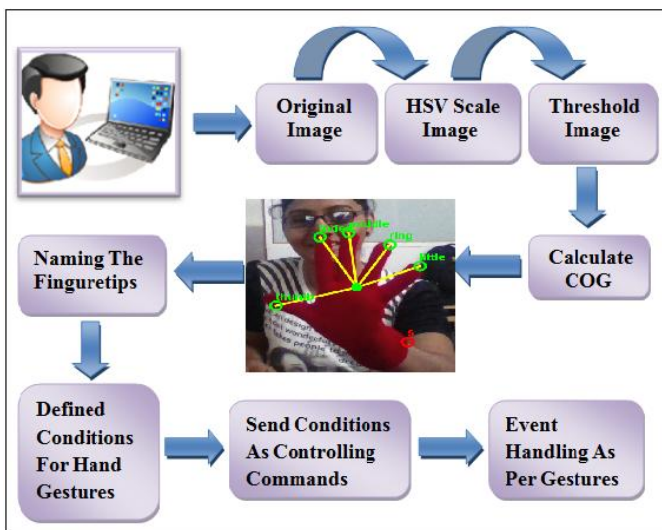


Figure3.1:- System Architecture

In next stage we give naming to the fingertips. After that we set the conditions like next, previous, play, pause etc. as per hand gestures. Then set that condition as command and load the native libraries of windows for perform the events such as Power-point, Media Player and PDF Reader etc. Then event handling is done as per hand gestures.

4. Proposed System

4.1. Image Processing

As per architecture discussed above we first capture the hand gestures images through web camera. The image will be converted into HSV scale i.e; the HSV scale required for RED colour is configured in parallel approach. HSV defines a type of color space. 'Value' is defined as brightness. In HSV, hue represents a color. In this system we have considered Hue for red colour in a range from 160 to 179. Saturation indicates the range of grey in the color space. We have considered saturation range from 100 to 255. Value is brightness of the color and varies with color saturation. In this system we have considered a range from 100 to 255 for Value, when the value is 0 the color space will be totally black. Then we obtained threshold image using HSV ranges of colour detection. In thresholding we have find the biggest contour which is a bounding box of white pixels.

After that we extract the noisy pixel in contour, this process is called as filtering of image. The next step is finding center of gravity (COG). We find the centroid of the contour by applying spatial moments as,

$$m(p, q) = \sum_{i=1}^n I(x, y)x^p y^q$$

The function takes two arguments, p and q, which are used as powers for x and y. The I() function is the intensity for a pixel defined by its (x, y) coordinate. n is the number of pixels that make up the shape. If we consider a contour like the one in Figure 4.1 then θ is the angle of its major axis to the horizontal, with the +y-axis pointing downwards.

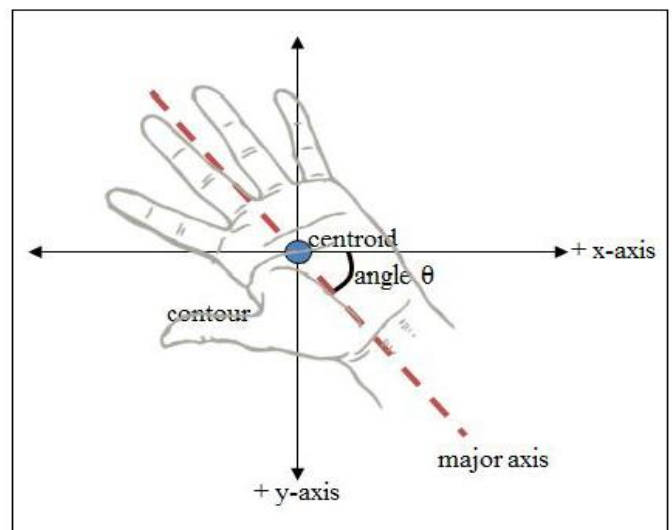


Figure 4.1 :- Contour And it's Major Axis Line

In terms of the $m()$ function, it can be shown that:

$$\tan 2\theta = \frac{2 \cdot m(1,1)}{m(2,0) - m(0,2)}$$

The extractContourInfo() method shown below uses spatial moments to obtain the contour's centroid, and cvGetCentralMoment() to calculate the major axis angle according to the above equation; these results are stored in the globals cogPt and contourAxisAngle for use later.

Identifying the fingertips is carried out in the first row of Figure 3; in the code, a convex hull is wrapped around the contour by JavaCV's cvConvexHull2() and this polygon is compared to the contour by cvConvexityDefects() to find its defects. Hull creation and defect analysis are speeded up by utilizing a low-polygon approximation of the contour rather than the original. The fingertips are stored in a tipPts[] array, the finger folds (the indentations between the fingers) in foldPts[], and their depths in depths[].

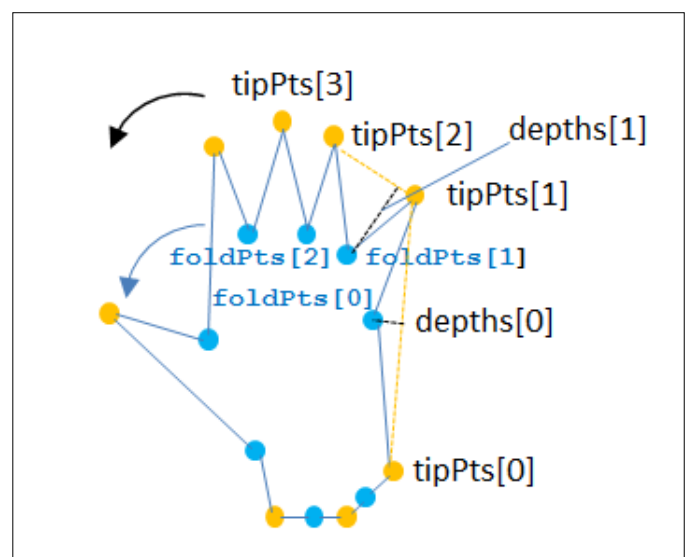


Figure 4.1(a):- Fingertips, Folds, Depth

As Figure 4.1(a) suggests, the analysis often generates too many defects, and so reduceTips() is called at the end of findFingerTips(). It applies two simple tests to filter out defects that are unlikely to be fingertips – it discards points with shallow defect depths, and coordinates with too great an angle between their neighboring fold points. Examples of both are shown in Figure 4.1(b).

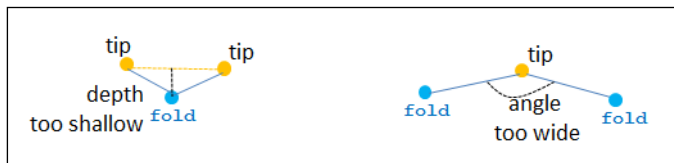


Figure 4.1(b):- Shallow Depths And Wide Angles

nameFingers() uses the list of fingertip coordinates, and the contour's COG and axis angle to label the fingers in two steps. First, it calls labelThumbIndex() to label the thumb and index finger based on their likely angles relative to the COG, assuming that they are on the left side of the hand. nameFingers() attempts to label the other fingers in labelUnknowns(), based on their known order relative to the thumb and index finger. One of the possible finger names is UNKNOWN, which is used to label all the fingertips prior to the calls to the naming methods. labelThumbIndex() attempts to label the thumb and index fingers based on the angle ranges illustrated in Figure 4.1(c).

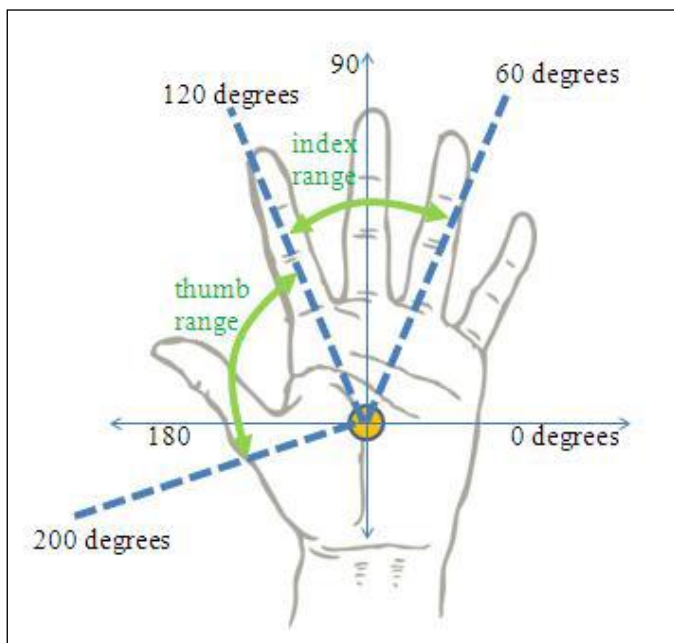


Figure 4.1(c):- Angle Ranges For The Thumb And Index Fingers

The index finger can turn between 60 and 120 degrees around the COG, while the thumb can move between 120 and 200 degrees. We arrived at these angles through trial-and-error, and they assume that the hand is orientated straight up.

labelThumbIndex() also assumes that the thumb and index fingers will most likely be stored at the end of the

fingerTips list, since the contour hull was built in a counter-clockwise order. It therefore increases its chances of matching against the right defects by iterating backwards through the list.

labelUnknowns() is passed a list of finger names which hopefully contains THUMB and INDEX at certain positions and UNKNOWNs everywhere else. Using a named finger as a starting point, the UNKNOWNs are changed to finger names based on their ordering in the FingerName enumeration.

labelPrev() and labelFwd() differ only in the direction they move through the list of names. labelPrev() moves backwards trying to change UNKNOWNs to named fingers, but only if the name hasn't already been assigned to the list.

The analysis performed by update() will result in a list of fingertip points (in the fingerTips global), an associated list of named fingers (in namedFingers), and a contour COG and axis angle. All of these, apart from the angle, are utilized by draw() to add named finger labels to the webcam image, as shown in Figure 4.1(d) below,

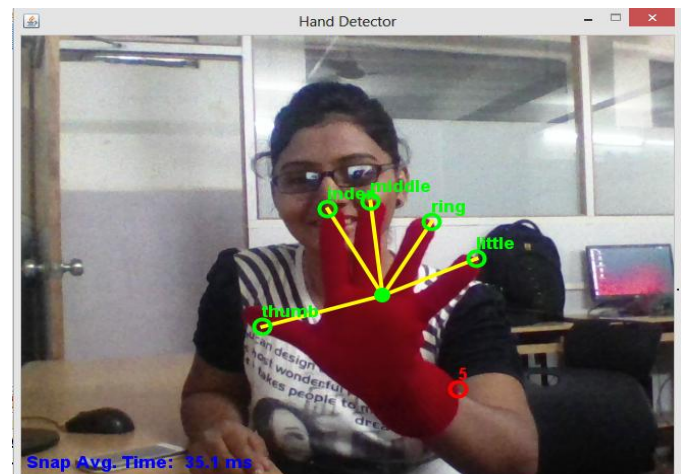


Figure4.1(d):- Named Fingers And Unknown

4.2. Application Control

Once the hand detection is completed we move towards the applications control part, that means event handling as per the hand gestures. We set some hand gesture for controlling multimedia applications such as Power Point Presentation, PDF Reader, and Windows Media Player etc. These hand gestures perform the operations like next, previous, play and pause of multimedia applications. The number of active fingers recognized by webcam and performs the respective operations.

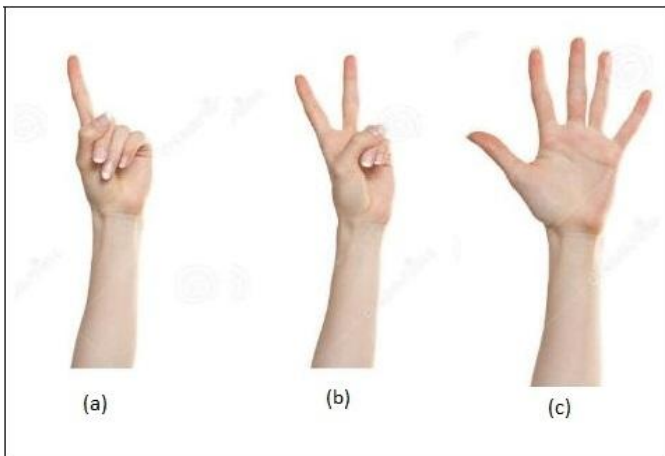


Figure 4.2:- Hand Gestures

As shown in figure 4.2 gestures are set for previous, next, play and pause. The figure 4.2(a) for previous operation, figure 4.2(b) for next operation and figure 4.2(c) for play and pause both operations. These hand gestures are detected by webcam then we set the commands of keyboard for controlling operations to the hand gesture through the robot class of java.

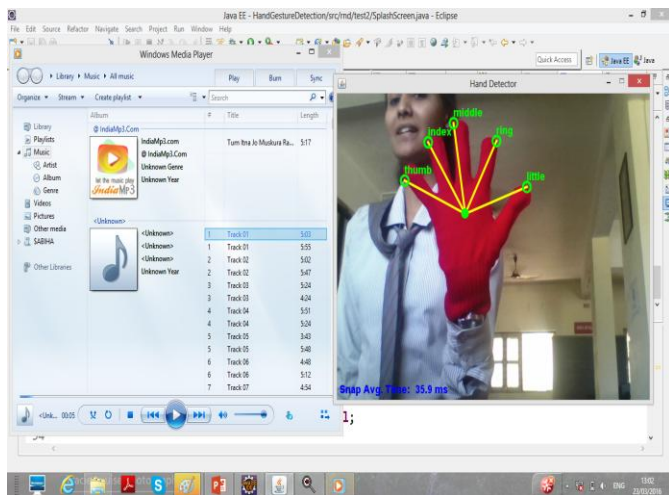


Figure 4.2(A):- Application Control

The hand gesture is recognized using the number of active fingers. Each hand gesture is mapped to a particular action as mentioned above. The actions are performed by using OpenCV functions. These functions and commands control the above mentioned multimedia applications. We have implemented this system only for the windows applications.




4.3 Performance Analysis & Results

For the accuracy of hand detection we are using the red colour gloves. If distance between camera and user is below 2 meters then the application run successfully with accurate results. If distance between camera and user is in between 2 to 4 meters then the application run partially. If

distance between camera and user is in 4 meters then the hand gesture cannot detect by webcam and the application cannot run properly.




a) Power Point Presentation

In this we open the Power Point file manually from system. Through hand gesture we perform the applications as follows:

	<p>This gesture is used for previous operation. After recognizing this gesture the current slide moves to previous slide.</p>
	<p>This gesture is used for next operation. After recognizing this gesture the current slide moves to next slide.</p>
	<p>This gesture is used for start and returns back from slideshow i.e. after recognizing this gesture the slideshow is started and return back to application.</p>



b) Windows Multimedia Player

The windows media is open through hand gesture application. We have to create playlist of songs and then start the application. The application performed the operations as follows:

	<p>This gesture is used for previous operation. After recognizing this gesture the current song moves to previous song in playlist.</p>
	<p>This gesture is used for next operation. After recognizing this gesture the current song moves to next song in playlist.</p>
	<p>This gesture is used for play and pause operation. After recognizing this gesture if the song is in pause mode it goes to play mode and if the song is in play mode is goes to the pause mode.</p>

c) PDF Reader

After launching the PDF reader application we have to manually select PDF file from system. Then we perform operations as follows:

	<p>This gesture is used for page up operation. After recognizing this gesture the page up i.e., the previous page turns over.</p>
	<p>This gesture is used for page down operation. After recognizing this gesture the page down i.e., the next page turns over.</p>

5. Limitations

- The proposed system is only applicable to windows applications.
- For hand detection, the white background is required.

6. Conclusion

The proposed system is used to control the multiple applications like PDF reader, multimedia player and power point presentation by avoiding the physical interfaces like mouse and keyboard. By using the gesture as commands any one can use the system to operate different applications. In some previously implemented system the costly 3-D sensors like kinect are used for gesture recognition. To reduce the cost we are using simple web camera. The separate training set is not require to recognize the gestures, so there is no need to maintain any database for storing the frames of images. Our focus of future work is on the extending the gestures for the gaming and voice for mute applications.

References

- [1] Chong Wang, Zhong Liu and Shing-Chow Chan "Superpixel-Based Hand Gesture Recognition With Kinect Depth Camera", *IEEE Trans. Multimedia*, Vol. 17, No. 1, Jan. 2015.
- [2] Swapnil D. Badgular, Gourab Talukdar, Omkar Gondhalekar and Mrs. S.Y. Kulkarni "Hang Gesture Recognition System", *International Journal of Scientific and Research Publications*, Vol. 4, Issue 2, Feb. 2014, pp. 2250-3153.
- [3] Viraj Shinde, Tushar Bacchav, Jitendra Pawar and Mangesh Sanap "Hand Gesture Recognition System Using Camera", *International Journal of Engineering Research & Technology (IJERT)*, Vol. 3, Issue 1, January - 2014.

- [4] N.Krishna Chaitanya and R.Janardhan Rao "Controlling of windows media player using hand recognition system", *The International Journal Of Engineering And Science (IJES)*, Vol. 3, Issue 12, Pages 01-04, 2014.
- [5] Swapnil V.Ghorpade, Sagar A.Patil, Amol B.Gore and Govind A.Pawar, "Hand Gesture Recognition System for Daily Information Retrieval", *International Journal of Innovative Research in Compute and Communication Engineering*, Vol. 2, Issue 3, March 2014.
- [6] Ram Rajesh J., Sudharshan R., Nagarjunan D. and Aarthi R. "Remotely controlled PowerPoint presentation navigation using hand gestures", *Proc. of the Intl. Conf. on Advances in Computer, Electronics and Electrical Engineering*, 2012.
- [7] Ruize Xu, Shengli Zhou, and Wen J. Li "MEMS Accelerometer Based Nonspecific-User Hand Gesture Recognition", *IEEE International Journal*, vol. 12, no. 5, May 2012.
- [8] Anupam Agrawal and Siddharth Swarup Rautaray "A Vision based Hand Gesture Interface for Controlling VLC Media Player", *International Journal of Computer Applications*, Vol. 10, No. 7, Nov. 2010.
- [9] Asanterabi Malima, Erol Ozgur, and Mujdat Cetin "A Fast Algorithm for Vision-Based Hand Gesture Recognition for Robot Control", Turkey, 2006.
- [10] Ahmed Elgammal, Vinay Shet, Yaser Yacoob, Larry S. Davis. Learning Dynamics for Exemplar-based Gesture Recognition. *In Proceedings of the Fifteenth International Conference on Computer Vision and Pattern Recognition*, pages 571{578. IEEE, June 2003.
- [11] Lars Bretzner, Ivan Laptev, Tony Lindeberg. Hand Gesture Recognition using Multi-Scale Color Features, Hierarchical Models and Particle Filtering. *In Proceedings of the Fifth International Conference on Automatic Face and Gesture Recognition*, pages 423-428. IEEE, May 2002.
- [12] Neha S. Rokade, Harsha R. Jadhav, Sabiha A. Pathan and Uma Annamalai "Controlling Multimedia Applications Using Hand Gesture Recognition", *International Research Journal of Engineering and Technology (IRJET)*, Vol. 2, Issue 7, Oct. 2015.

Biographies



Miss. Neha S. Rokade is currently a student of Gokhale Education Society's R. H. Sapat College of Engineering, Nashik Under the University of Savitribai Phule Pune.



Miss. Harsha S. Jadhav is currently a student of Gokhale Education Society's R. H. Sapat College of Engineering, Nashik Under the University of Savitribai Phule Pune.



Miss. Sabiha A. Pathan is currently a student of Gokhale Education Society's R. H. Sapat College Of Engineering, Nashik Under the University of Savitribai Phule Pune.



Miss. Uma Annamalai is currently a student of Gokhale Education Society's R. H. Sapat College Of Engineering, Nashik Under the University of Savitribai Phule Pune.