American Journal of Intelligent Systems 2017, 7(3): 90-94
DOI: 10.5923/j.ajis.20170703.11

Hand Gesture Recognition using Webcam

Athiya Marium[*], **Deepthi Rao, Divina Riya Crasta, Kavya Acharya, Rio D'Souza**

Department of Computer Science and Engineering, St Joseph Engineering College, Mangaluru, India

Abstract The communication between the user and the computer can be done through various input devices such as the keyboard, mouse etc. However, this paper presents a more natural and intuitive form of communication i.e. via hand gestures. The gesture performed by the user is recognised and the action specific to it is performed, thus eliminating the use of any of the hardware input devices completely. This method makes use of image processing to recognise the gestures and has built in functions for every gesture provided by the PyAutoGUI module. It is coded in Python and uses the OpenCV library. Experiments show that the implementation is reliable enough for practical use.

Keywords Image processing, Gesture Recognition, PyAutoGUI module, OpenCV, Numpy

1. Introduction

Gesture is a form of nonverbal communication which involves movement of a part of the body, especially the hand usually to express an idea. In a human interaction, we make use of speech, gestures and body movements to convey information. The human-computer interaction which makes use of input devices such as keyboard or mouse for communication lacks natural communication between humans and machines. For this purpose, it is important to develop applications or devices that support intuitive communication.

As computer technology continues to grow, the need for natural communication between humans and machines also increases. Although our mobile devices make use of the touch screen technology, it is not cheap enough to be implemented in desktop systems.

Although the mouse is very useful for device control, it could be inconvenient to use for physically handicapped people and people who are not accustomed to use the mouse for interaction.

The method proposed in this paper makes use of a webcam through which gestures provided by the user are captured, processed and the function related to that gesture is carried out. For example, a gesture "V" i.e. two fingers, could be predefined in the system to perform a click operation.

The system has four phases namely, image acquisition, image pre-processing, feature extraction and gesture recognition, as described by Babu et.al [1]. Image acquisition involves capturing images frame by frame using a webcam.

The captured images go through the image pre-processing process which involves color filtering, smoothing and thresholding. Feature extraction involves extracting features of the hand image such as hand contours. Gesture recognition involves recognising hand gestures with the help of extracted features.

2. Related Work

There are many gesture recognition techniques developed for tracking and recognizing various gestures and these are surveyed by Madhuri and Kumar [2]. Gesture recognition using webcam is an appealing option for replacing human computer interaction using a mouse.

Back in 2009, Bayazit et.al [3] implemented a GPU-based system for gesture recognition which runs in real-time. While this system demonstrated the principle, it was too bulky and unwieldy for practical use. Advancements in development of optic modules, have allowed for 3-D image sensors to emerge as a viable alternative to stereo and structured light imaging for capturing range information [4]. Due to this gesture recognition has the potential to emerge as an alternative input device in the near future.

A system using pointing behaviours for a natural interface to classify the dynamic hand gesture has been proposed by Badgujar et.al. Though this system is suitable for controlling real-time computer systems, it is applicable only for the powerpoint presentations [5]. To overcome the challenges of gesture recognition using color detection and their relative position with each other, gesture recognition using contour analysis is implemented.

The algorithm implemented in this paper detects the gesture based on the number of contours that are visible and thereby performs the necessary operation related to the gesture. This paper brings out an innovative idea to use the camera instead of mouse.

* Corresponding author:
divinacrasta@gmail.com (Athiya Marium)
Published online at http://journal.sapub.org/ajis

3. System Overview

The cursor movement by hand gestures is done using OpenCV [6, 7] library, uses Python programming language, which provides an ease to understand code through its simplicity. Python modules and packages used here are PyAutoGUI [8, 9] and NumPy [10, 11].

The captured video is broken down into continuous image frames using functions defined in OpenCV. The image frames are processed in order to detect any valid gestures being performed by the user.

The basic architecture for the system is shown below.

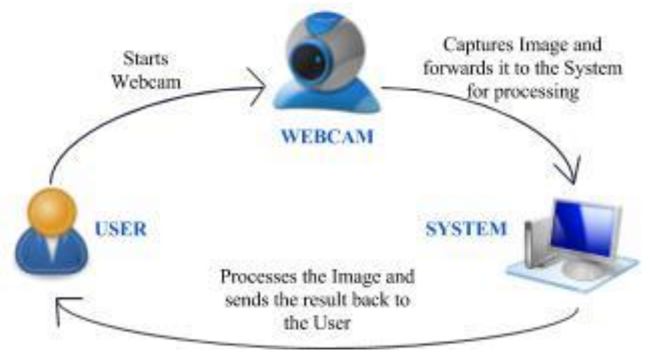

Starts Webcam
Captures Image and forwards it to the System for processing
WEBCAM
USER
SYSTEM
Processes the Image and sends the result back to the User

Figure 1. Basic architecture of the system

3.1. Module Description of Proposed System

3.1.1. Module for Finding the Number of Convexity Defects

The convexity defects for the hand contour are first calculated by using the OpenCV inbuilt function "cvConvexityDefects()". After the convexity defects are obtained, we perform steps for identifying the fingertips and the fingers.

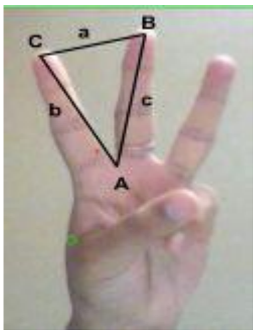


Figure 2. Finding convexity defect

Convexity defects obtained, is a structure that returns four values, start point, end point, farthest point and approximate distance to farthest point, out of which three have been used. The figure 2, denotes for one of the contours the start, the end and the far point. C represents the start point, B represents the end point and A is the farthest point. The angle made by the two fingers must be found to correctly determine if a finger is held up. This is done using the triangle formed by A, B and C. The length of each line is found using the distance formula as

$$\boldsymbol{a} = \sqrt{(\boldsymbol{start}[0] - \boldsymbol{end}[0])^2 + (\boldsymbol{start}[1] - \boldsymbol{end}[1])^2}$$

$$\boldsymbol{b} = \sqrt{(\boldsymbol{start}[0] - \boldsymbol{far}[0])^2 + (\boldsymbol{start}[1] - \boldsymbol{far}[1])^2}$$

$$\boldsymbol{c} = \sqrt{(\boldsymbol{end}[0] - \boldsymbol{far}[0])^2 + (\boldsymbol{end}[1] - \boldsymbol{far}[1])^2}$$

Once the length of each have been found, using the Cosine rule,

$$\boldsymbol{a}^2 = \boldsymbol{b}^2 + \boldsymbol{c}^2 - 2\boldsymbol{bc}\ \boldsymbol{cos}\ \boldsymbol{A}$$

the angle A is found using

$$\boldsymbol{A} = \boldsymbol{cos}^{-1}(\boldsymbol{b}^2 + \boldsymbol{c}^2 - \boldsymbol{a}^2/2 * \boldsymbol{b} * \boldsymbol{c})$$

If the angle A is less than 70°, it is considered that two fingers are held up. The same technique is applied on all the defects to find the number of fingers held up.

The approach can only be used to find the number of fingers, if the number of fingers held up are more than two.

3.1.2. Module for finding the center of the hand

Before finding the gesture for one finger held up, the center of the hand should be calculated. For this, first we considered a bounding rectangle that bounded the entire hand. From this we took an approximation of the palm of the hand. Every point in the palm of the hand was put to a pointPolygonTest of openCV, which returned the distance of the point to the nearest contour edge. A selection sort returned the point with the maximum distance. This point was considered to be the center. The red circle in Figure 3 marks the centre of the hand.

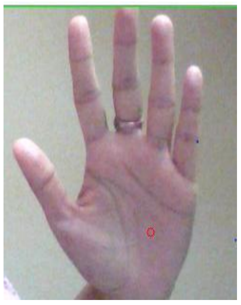


Figure 3. Finding the center of the hand

3.1.3. Module for Identifying the Gesture

To find a single finger held up, here, we have taken the index finger, and a selection sort is performed to find the farthest distance from the center of the hand, given by (x,y). This farthest distance is compared against a range of distances that can be considered for the index finger. Another parameter for determining if the finger held up is the index, is the angle it makes with the center, given by (h,k). Assuming that every finger makes a different angle with the centre, we found the angle made by the finger using the formula,

$$\boldsymbol{A} = \boldsymbol{tan}^{-1}((\boldsymbol{k} - \boldsymbol{y})/(\boldsymbol{h} - \boldsymbol{x}))$$

If the angle and farthest distance lie within the range specified, we have concluded that the finger held up is the index finger. The Figure 4 shows the recognition of index finger by finding the distance from the centre of the hand (h,k), represented by the green circle, and the farthest point (x,y), in the hull.

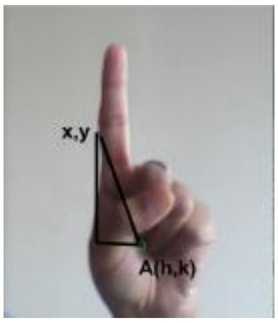


Figure 4. Identifying gesture

The next gesture we worked on needed to recognise two fingers held up in the way shown in Figure 5. The same technique of finding angle between two fingers was used here. The angle to which it was compared was however different from the one we specified for two consecutive fingers held up. In addition to this, the angle made by the pinky finger with the centre was also found and compared. The distance between the two fingers was found using the distance formula used above.

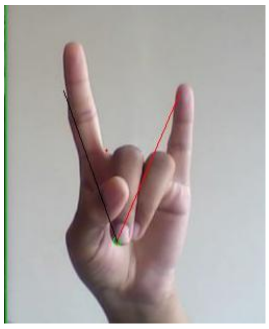


Figure 5. Identifying gesture

This way any gesture can be defined and recognised. We found our approach to be simple and easy to understand and manipulate.

3.2. Algorithm for the System

The following algorithm shows the basic steps performed by the system when gestures are detected.

```
START: Start the webcam
STEP 1: Detect the user's hand
STEP 2: Capture the image
STEP 3: Identify the specific hand gesture
STEP 4: If the gesture for cursor movement is detected, go to STEP 10
STEP 5: If the gesture for single click is detected, go to STEP 11
STEP 6: If the gesture for double click is detected, go to STEP 12
STEP 7: If the gesture for drag is detected, go to STEP 13
STEP 8: If the gesture for left wave is detected, go to STEP 14
STEP 9: If the gesture for right wave is detected, go to STEP 15
STEP 10: Detect the coordinates of the mouse and perform cursor movement
STEP 11: Using coordinates from the mouse perform the selection, go to STEP 3
STEP 12: Perform selection or opening actions, go to STEP 3
STEP 13: Perform dragging action, go to STEP 3
STEP 14: Decrease the speed of the cursor
STEP 15: Increase the speed of the cursor
STEP 16: Stop
```

4. Experimental Results

The experimental setup requires a web camera in order to capture the gestures performed by the user, preferably with a plain background. The web camera should be at a fixed position to help in capturing images more efficiently.

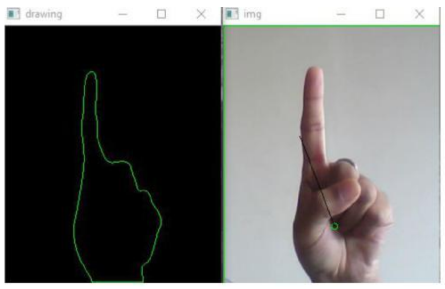


Figure 6. Gesture will perform a click operation

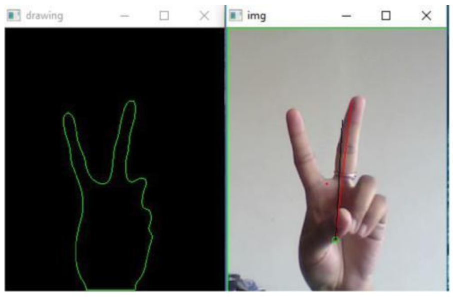


Figure 7. Gesture will move the cursor towards the right

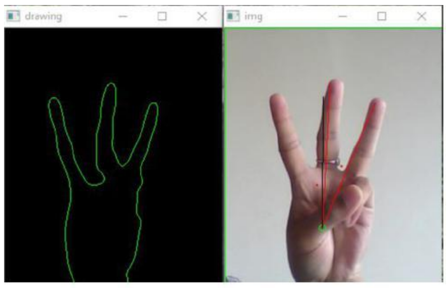


Figure 8. Gesture will move the cursor downwards

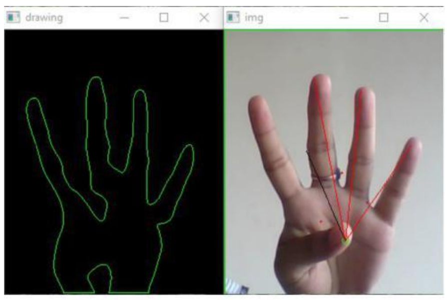


Figure 9. Gesture will move the cursor towards the left

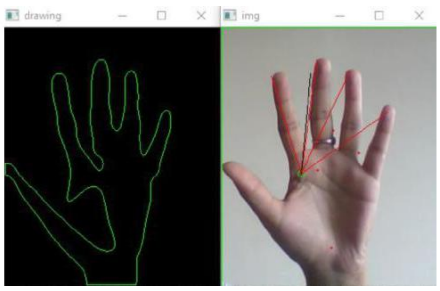


Figure 10. Gesture makes the cursor move upwards

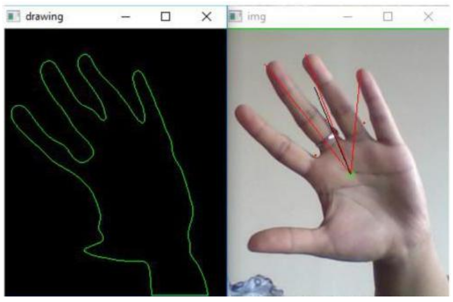


Figure 11. The gesture will decrease the speed of movement of cursor

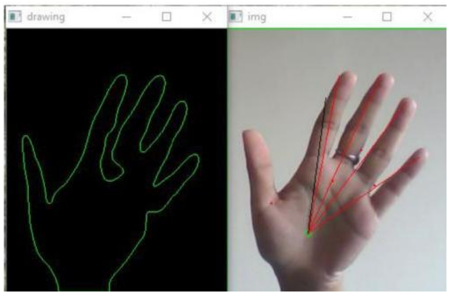


Figure 12. Gesture increases the speed of the cursor movement

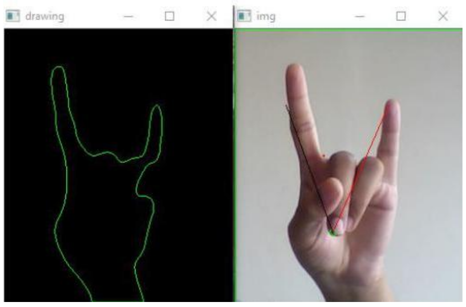


Figure 13. Gesture makes all the cursor movements as drag

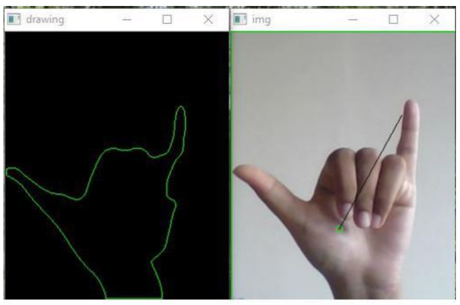


Figure 14. The gesture makes all cursor movements move

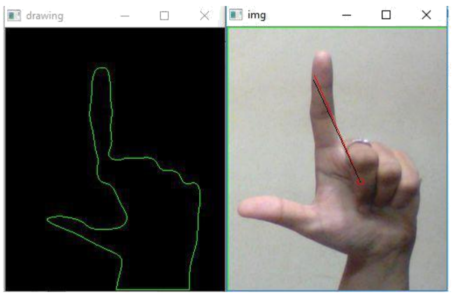


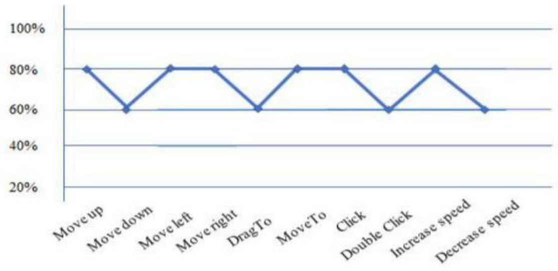
Figure 15. The gesture performs double click operation

The experimental results are tabulated in Table 1.

Table 1. Analysis table for gesture recognition

Events	No. of input samples	No. of recognized samples	Recognition rate
Move the cursor up	5	4	0.8
Move the cursor down	5	3	0.6
Move the cursor left	5	4	0.8
Move the cursor right	5	4	0.8
DragTo	5	3	0.6
MoveTo	5	4	0.8
Click	5	4	0.8
Double click	5	3	0.6
Increase speed of cursor	5	4	0.8
Decrease speed of cursor		3	0.6

The graph for Recognition rate for gesture events is given below.



Graph 1. Performance Results

5. Conclusions

This paper describes a system that controls computer applications with the help of hand gestures. The method proposed here successfully created a hand gesture recognition system, that is able to recognise which gesture is performed by the user and accurately perform the functionality associated with it.

Presently, the webcam, microphone and mouse are an integral part of the computer system. Our product which uses only webcam would completely eliminate the mouse. Also this would lead to a new era of Human Computer Interaction (HCI) where no physical contact with the device is required.

6. Future Work

The current system gives best results in a plain background and hence puts certain constraints on the user for successful working. The future work will include implementation of additional gestures which will enable the user to perform more functions with ease. Furthermore, background subtraction algorithm can be used for a more effective performance. The proposed system uses only the right hand to perform gestures. Hence, enhancement of the technique proposed, is possible using both hands for performing different computer operations. Experiments need to be done on a larger scale so that results can be more accurate.

ACKNOWLEDGEMENTS

As we present before you the paper "Hand Gesture Recognition Using Webcam" we take this opportunity to thank those who have helped and encouraged us. Developing an idea into action is known as team effort. But behind the work done by the team of people are the invisible help, support and guidance of a large number of well wishers.

REFERENCES

[1] Ann Abraham Babu, Satishkumar Varma, Rupali Nikhare, “Hand Gesture Recognition System for Human Computer Interaction Using Contour Analysis”, International Journal of Research in Engineering and Technology, eISSN: 2319-1163, pISSN: 2321-7308.

[2] K. Madhuri, L. Praveen Kumar, “Cursor Movements Controlled By Real Time Hand Gestures”, International Journal of Science and Research (IJSR), India Online ISSN: 2319-7064.

[3] Mark Bayazit, Alex Couture-Beil, and Greg Mori, ‘Real-time motion based gesture recognition using the gpu’, in IAPR Conference on Machine Vision Applications (MVA), (2009).

[4] D. I. Ko and G. Agarwal, Gesture Recognition: Enabling Natural Interactions with Electronics. Dallas, TX, USA: Texas Instruments, Apr. 2012.

[5] Swapnil D. Badgujar, Gourab Talukdar, Omkar Gondhalekar and S. Y. Kulkarni “Hand Gesture Recognition System”, International Journal of Scientific and Research Publications, Vol. 4, Issue 2, Feb. 2014, pp. 2250-3153.

[6] Itseez, Open Source Computer Vision Library - itseez2015opencv, 2015, accessed at https://github.com/itseez/opencv on 01/2/2017.

[7] Itseez, The OpenCV Reference Manual - itseez2014theopencv 2.4.9.0, April 2014, accessed at http://opencv.org/ on 01/2/107.

[8] Python Software Foundation (PSF), PyAutoGUI package, PyAutoGUI 0.9.36 accessed at https://pypi.python.org/pypi/PyAutoGUI on 21/4/2017.

[9] Al Sweigart, PyAutoGUI Documentation, Release 1.0.0, accessed at https://media.readthedocs.org/pdf/pyautogui/latest/pyautog ui.pdf on 21/4/2017.

[10] Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation, Computing in Science & Engineering 13, 22-30 (2011) DOI:10.1109/MCSE.2011.37.

[11] NumPy Developers, numpy 1.13.0rc2, accessed at https://pypi.python.org/pypi/numpy on 01/2/107.