IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735.Volume 9, Issue 3, Ver. V (May - Jun. 2014), PP 30-35
www.iosrjournals.org

Human Computer Interaction by Gesture Recognition

Anchit.P.Narwadkar, Mhalsakant.M.Sardeshmukh
Department of Electronics and Tele-communication, Sinhgad Academy of Engineering, Kondhva Pune, India
Department of Electronics and Tele-communication, Sinhgad Academy of Engineering, Kondhva Pune, India

Abstract: *Hand Gesture Recognition to control any device is one of the new techniques identified for Human Computer Interaction. As context to that in this paper we have proposed a Real Time Hand Gesture Recognition Technique in which user can do various gestures by hand in front of camera connected to device , although there is variation in background captured video is converted into frames and by using suitable pattern recognition technique gesture is analyzed and identified and given as a command to operate the robot.*
Index Terms: *Gesture recognition, Human-computer interaction, Pattern recognition, Robotics*

I. INTRODUCTION

In our day to day life we convey messages by using different gestures. Gestures may be nodding of head, waving of hand, facial expressions, eye movements and any body part movement. Here we are specially dealing with Hand Gestures i.e. Gestures created by hand. Once gestures are recognized they can be used in various applications like sign language recognition, virtual mouse to computer, television control, human/robot manipulation etc.

To enlighten the project it has been decided that the identification would consist of counting the number of fingers that are shown by the user in the input picture and as per identified gesture command will be given to robot for its movement.

Table 1
Finger counting and associated robot movement

Number of fingers	Robot Movement
One	Forward
Two	Backward
Three	Left
Four	Right
Five	Stop

II. Related Work

The use of hand gestures provides an interface between human and device to be controlled for human computer interaction. Till now many approaches of hand gesture recognition are proposed but as we are dealing with real time hand gesture recognition following work must be analyzed.

Recent researches [1-4] in computer vision have established the importance of gesture recognition systems for the purpose of human computer interaction. An extensive survey on Gesture Recognition is presented in [5].

The main approaches for analyzing and classifying hand gestures for HCI include Glove based techniques and Vision based techniques. The data gloves method is quite expensive and forces the user to carry a load of cables which are connected to the computer and hinders the ease and naturalness of the user interaction. Computer vision based techniques are non invasive. These are based on the way human beings perceive information about their surroundings. Although it is difficult to design a vision based interface for generic usage, yet it is feasible to design such an interface for a controlled environment [6].

Hasanuzzaman et al. [7] presented a real-time Hand Gesture Recognition system using skin color segmentation and Multiple-feature based template-matching techniques.

Yin and Xie [8] created a fast and robust system that segments using color and recognizes hand gestures for human–robot interaction using a neural network.

Chen et al. [9] proposed a hand gesture recognition system to recognize continuous gestures before stationary background. The system consists of four modules: real time hand tracking and extraction, feature extraction, Hidden Markov model (HMM) training, and gesture recognition.

GRS Murthy and RS Jodan [10] presented a real time hand gesture recognition technique by using neural network as a pattern recognition technique. In this they have tested it for 10 different gestures and obtained satisfactory results.

Figure 1: Sample Gestures Under Consideration

III. Overview Of Proposed Work

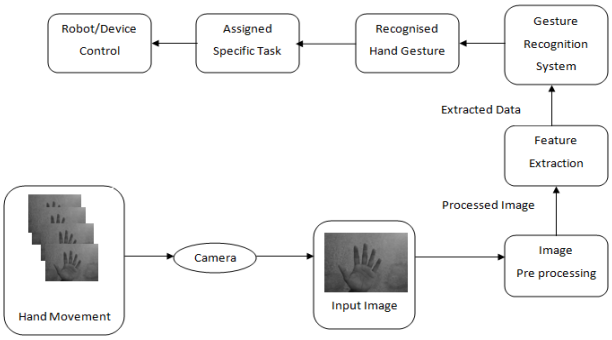

Figure 2: Block diagram of proposed software system

All videos are captured at 30 FPS in AVI format and saved to disk. We have implemented the method, using an ordinary workstation with no special hardware beyond a web camera. Our proposed system involves following steps.

Step 1:

First we captured the background comprising of one frame only as a static image without the user hands. We have taken one set of video of many frames each from different users, set we have captured five finger count gestures (one-finger, two-finger, three-finger, four finger and, five-finger). In finger counting we are not distinguishing between the finger and the thumb. For finger count gestures, user has to pose the hand in horizontal or vertical manner only.

Step 2:

Using "frame2im" function of MATLAB we have extracted frames from videos. As we have kept uniform background, it is possible to build a new image that corresponds to the difference between the current image of hand and the background. For this we have calculated their absolute difference using "imabsdiff" function of MATLAB and the resultant image contains only the hand of the user.

Step 3:

Each image extracted from a video is an uncompressed RGB image. We than converted the RGB image into grayscale by using "rgb2gray" function of MATLAB which eliminates the hue and saturation information while retaining the luminance factor and then to binary image using an experimental threshold value. The noise from the binary image is than removed using MATLAB function "bwmorph ( )" that erodes and dilates the noisy picture.

Step 4:

As our focus is on the palm area of hand and fingers, we have made a MATLAB function which takes binary image from step 3 as input and returns the binary image with the hand portion containing fingers only. This function also computes four extreme coordinates of hand (bounding box) by calculating the sum of elements in each row and columns. To find the extreme coordinates of the fingers in three sides is easy but to find the starting of hand portion from arm side is quite difficult, because the user can place his hand anywhere within the visibility of camera. To overcome this we have placed a restriction on the user that he/she has to

display the thumb compulsory in the gesture and to find the thumb of the user in the image we have created a line vector which receives the number of edge pixels from the arm side.

Step 5: We have proposed three methods for hand gesture recognition namely-

a. Pixel counting analysis- we have used simple pixel counting method based on following algorithm.

If sum < range_1
Then No fingers
If range_1 < sum < range_2
Then 1 finger
If range_2 < sum < range_3
Then 2 fingers
If range_3 < sum < range_4
Then 3 fingers
If range_4 < sum < range_5
Then 4 fingers
If range_5 < sum
Then 5 fingers

Here range term refers to assumed pixel count for number of fingers.

b. Block counting analysis-

The program has to count the number of fingers? So let's create a picture in which will remain only the fingers. It is easy to do, given that the orientation of the hand is known. Cropping the left part of the picture (including the thumb) will cause that only the fingers remain on the picture:

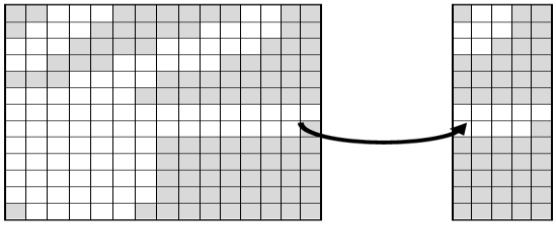

Figure 3: Block Counting Analysis

In such cases, the number of fingers is the number of blocks in the cropped picture, plus 1, because the thumb has to be considered, even if it has been cropped.

c. Edge Counting analysis-

Another method that has been tried is based on the number of edges in every column. The number of edges should be a function of the number of fingers, given that each finger adds two vertical edges more.

Figure 4: Edge Counting Analysis

The calculus of the number of fingers is quite easy: one finger is the thumb, and count two edges per finger, so finally:

$$\#\ fingers = 1 + \frac{\#edges}{2}$$

Step 6: Once proper count is obtained i.e. gesture is identified, command is passed to robot for its movement.

Step 7: If hand gesture showing one finger robot will move forward, two fingers backward, three fingers left, four fingers right and five fingers robot will stop.

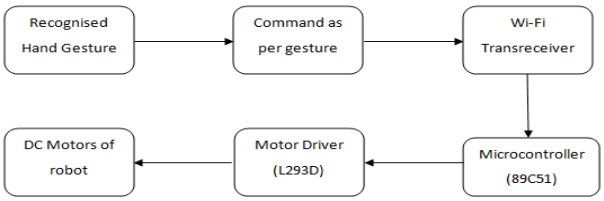

Figure 5: Block diagram of proposed hardware system

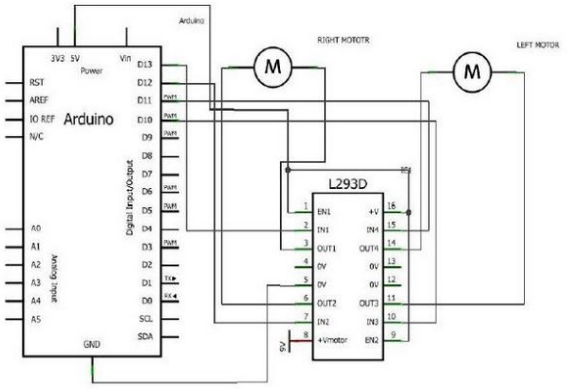

Figure 6: Hardware Setup

1. ARDUINO

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software running on a computer (e.g.MATLAB,Flash, Processing).

It uses the serial port interface in MALAB for communication with Arduino board. MALAB support package for Arduino comes with a server program adiosrv.pde that can be downloaded on the Arduino board. Once downloaded, the adiosrv.pde will:

1. Wait and listen for MATLAB commands
2. Upon receiving MATLAB command, execute and return the result

The steps in write/download the server program to Arduino board is the following:

1. Open Arduino IDE
2. Select the adiosrv.pde file by navigating through File > Open in the IDE
3. Click the upload button

2. MOTOR DRIVER IC L293D

IC L293D is used for driving the motors. L293D is a dual H-bridge motor driver integrated circuit (IC). Motor drivers act as current amplifiers since they take a low-current control signal and provide a higher-current signal. This higher current signal is used to drive the motors. L293D contains two inbuilt H-bridge driver circuits. In its common mode of operation, two DC motors can be driven simultaneously, both in forward and reverse direction. The motor operations of two motors can be controlled by input logic at pins 2 & 7 and 10 & 15. Input logic 00 or 11 will stop the corresponding motor. Logic 01 and 10 will rotate it in clockwise and anticlockwise directions, respectively. Enable pins 1 and 9 (corresponding to the two motors) must be high for motors to start operating. When an enable input is high, the associated driver gets enabled. As a result, the outputs become active and work in phase with their inputs. Similarly, when the enable input is low, that driver is disabled, and their outputs are off and in the high-impedance state. L293D has output current of 600mA and peak output current of 1.2A per channel. Moreover for protection of circuit from back EMF output diodes are included within the IC. The output supply (VCC2) has a wide range from 4.5V to 36V.

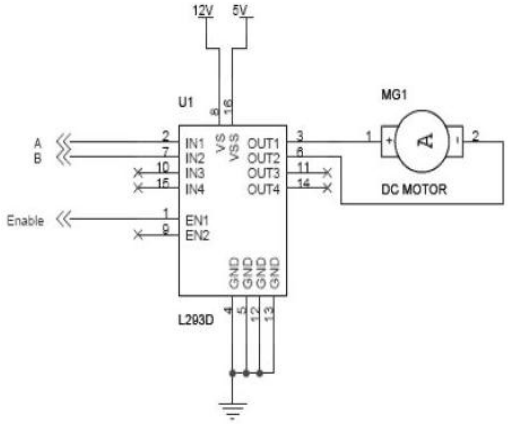

Figure 7: Schematic of Motor Driver IC L293D

Our proposed hardware system involves following steps.

Step 1:

After detecting gesture command specific signal value is generated, unique for every gesture command.

Step 2:

As soon as the command is generated on the control station, it is written in a file with tagged word with it. This file is read by WiFly after regular interval. As it is a wireless communication, so WiFly communicates with the control station using a hotspot where control station is also in the same network

Step 3:

WiFly is connected to the microcontroller through stackable pins shown in Figure 14. When the process of communication starts, WiFly tries to connect itself with the hotspot. For that it requires SSID and passphrase of the hotspot. These are provided in the burned code of the microcontroller. After forming a successful connection with the hotspot, WiFly tries to get the access of the control station, with the provided IP address of the control station and port number of HTTP port which is by default 80.

Step 4:

L293D motor driver circuit takes digital signal as an input from the Arduino and gives digital output to the DC motors of the robot.

Step 5: This is the final end product of robot consisting of all the hardware WiFly, Microcontroller and L293D motor Driver circuit on the robot chassis having power supply. Four DC motors are connected to this robot chassis to four wheels for movement. This is controlled through gestures by the user at control station.

VI. Results and Analysis

We have conducted experiments based on images we have acquired using a simple webcam. We have collected these data on uniform backgrounds. Our algorithms lead to the correct recognition of all the five gestures. Also, the computation time needed to obtain these results is very small, since the algorithms are quite simple. The output was used to drive the hardware, interfaced with the help of Arduino. The control commands were send to Arduino from MATLAB directly and on the basis of this command, motion of the ROBOT was adjusted.

We have noted that images taken under insufficient light have led to the incorrect results. In these cases the failure mainly stems from the erroneous segmentation of some background portions as the hand region. Our algorithm appears to perform well in uniform backgrounds with appropriate illumination. Overall, we find the performance of this simple algorithm quite satisfactory in the context of our motivating robot control application.

Table 2
Recognition Result of Finger counting

Number of fingers	**Correct/Total(Accuracy Rate)**		
	Pixel Counting	**Block Counting**	**Edge Counting**
One	45/50(90%)	43/50(86%)	47/50(94%)
Two	46/50(92%)	44/50(88%)	46/50(92%)
Three	43/50(86%)	44/50(88%)	45/50(90%)
Four	45/50(90%)	47/50(94%)	46/50(92%)
Five	48/50(96%)	45/50(90%)	48/50(96%)

VII. Conclusion

We have proposed a vision based gesture recognition using a simple system connected with a web camera. To detect hand, count fingers. We have used different algorithms. The hand of the user was separated from the background, the fingers region was focused and by using the pixel counting, block counting and edge counting methods as discussed in step five, our system recognized the number of fingers user was displaying. We have experimented with gesture images captured by a camera and achieved satisfactory results in operating Robot connected via serial port. Robot is moving satisfactorily in all directions as per command received by user.

We have observed that in pixel counting method size of finger decides finger count so though algorithm is simple variation in finger size can change finger count. In block counting analysis method used is simple but adjacent blocks can lead to miscounting of fingers and in edge counting analysis we will be getting maximum accuracy compared to remaining two methods.

References

[1] Pavlovic, V., Sharma, R. & Huang.T.S.: Visual interpretation of hand gestures for human-computer interaction: A review. IEEE Transaction on Pattern Analysis and Machine Intelligence, 19(7), pp 677–695,1997.

[2] Wu, Y. & Huang, T.S.: Vision-based gesture recognition: A review. In Lecture Notes in Computer Science, Gesture Workshop,1999.

[3] Konstantinos G. Derpanis. : A Review of Vision-Based Hand Gestures. Internal Report, Department of Computer Science. York University,2004.

[4] Watson, Richard.: A Survey of Gesture Recognition Techniques. Technical Report TCD-CS-93-11, Department of Computer Science, Trinity College Dublin,1993.

[5] Sushmita Mitra and Tinku Acharya: Gesture Recognition: A Survey. IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews, Vol. 37, No. 3,2007.

[6] T.S. Hunang and V.I. Pavloic: Hand Gesture Modeling, Analysis, and Synthesis. Proc. of International Workshop on Automatic Face-and gesture recognition, Zurich, pp.73-79, 1995

[7] Md. Hasanuzzaman, V. Ampornaramveth, Tao Zhang, M.A. Bhuiyan ,Y. Shirai and H. Ueno,: Real-time Vision-based Gesture Recognition for Human Robot Interaction. In the Proceedings of the IEEE International Conference on Robotics and Biomimetics, Shenyang China ,2004

[8] X. Yin and M. Xie,: Finger identification in hand gesture based human–robot interaction. J. Robot. and Auton. Syst., vol. 34, no. 4, pp. 235– 250,2001

[9] F. S. Chen, C.M. Fu and C.L. Huang,: Hand gesture recognition using a real-time tracking method and hidden Markov models. Image Vision Computer, pp. 745-758,2003

[10] G.R.S. Murthy,R.S. Jadon "Hand Gesture Recognition using Neural Networks",IEEE Conference, IEEE 2nd International Advance Computing Conference, 2010.