Hand Gesture Recognition System

Mohamed Alsheakhali, Ahmed Skaik, Mohammed Aldahdouh, Mahmoud Alhelou

Computer Engineering Department, The Islamic University of Gaza
Gaza Strip , Palestine, 2011

msali@iugaza.edu.ps, ahmskaik@gmail.com, mhmd1986@hotmail.com,
mmh.1989@hotmail.com

Abstract—**The task of gesture recognition is highly challenging due to complex background, presence of nongesture hand motions, and different illumination environments. In this paper, a new technique is proposed which begins by detecting the hand and determining its canter, tracking the hands trajectory and analyzing the variations in the hand locations, and finally recognizing the gesture. The proposed technique overcomes background complexity and distance from camera which is up to 2.5 meters. Experimental results show that the proposed technique can recognize 12 gestures with rate above 94%.**

***Keywords*— Hand gesture recognition, skin detection, Hand tracking.**

I. Introduction

Human gestures constitute a space of motion expressed by the body, face, and/or hands. Among a variety of gestures, hand gesture is the most expressive and the most frequently used. Gestures have been used as an alternative form to communicate with computers in an easy way. This kind of human-machine interfaces would allow a user to control a wide variety of devices through hand gestures. Most work in this research field tries to elude the problem by using markers, marked gloves or requiring a simple background [1-6]. Glove-based gesture interfaces require the user to wear a cumbersome device, and generally carry a load of cables that connect the device to a computer. A real-time gesture recognition system which can recognize 46 ASL letter spelling alphabet and digits was proposed [7]. The gestures that are recognized by [7] are static gestures without any motion.

This paper introduces a hand gesture recognition system to recognize 'dynamic gestures' of which a single gesture is performed in complex background. Unlike previous gesture recognition systems, our system neither uses instrumented glove nor any markers. The new barehanded proposed technique uses only 2D video input. This technique involves detecting the hand location, tracking the trajectory of the moving hand, and analysing the hand-position variations. Then the obtained motion information is been used in the recognition phase of the gesture.

The present paper is organized as follows: Section I introduces an overview of system components. Hand gesture detection and tracking the trajectory of the moving hand approach are presented in Sections II and III. Section IV demonstrates the proposed recognition method. The results of the proposed system are discussed in Section V. Conclusion and future work are given in Section VI.

II. SYSTEM COMPONENTS

A low cost computer vision system that can be executed in a common PC equipped with USB web cam is one of the main objectives of our approach. The system should be able to work under different degrees of scene background complexity and illumination conditions. Fig 1 shows an overview of our hand gesture detection, tracking and recognition framework.

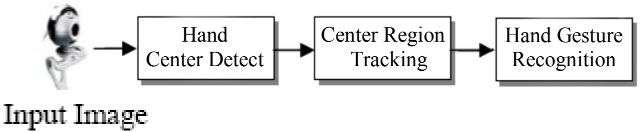


Fig. 1. Proposed System Framework.

The proposed technique depends on the following approach:

1) *Acquisition:* a frame from the webcam is captured.
2) *Segmentation and detection:* the image is segmented into two parts, both of them are manipulated simultaneously before analysing the resultant data, skin pixels and moving patterns are detected. A new image is created containing the location of the center of the moving hand.
3) *Tracking:* 10 latest consecutive frames are tracked continuously, in each frame the centers of the moving hands are detected.
4) *Pattern Recognition:* through the user's hands motion, the features are compared with those stored in the database, the maximum likelihood correspondence is chosen.

III. Hand detection

The technique is built around the idea that Splits the input video screen into two parts and processes each part separately, so that the processing in the two parts is similar and simultaneous. The operators used for image processing must be kept low time consuming in order to obtain the fast processing rate needed to achieve real time speed. The detection steps are:

1) *Skin tone detection:* Skin can be easily detected using the color tone information. RGB-based color is classified as skin if:

$$0.08 < \frac{3 * B * R^2}{(\mathrm{R} + \mathrm{G} + \mathrm{B})^3} < 0.12 \quad AND$$

$$1.5 < \frac{R * B * G^2}{G * B} < 1.8 \quad AND$$

$$\frac{\mathrm{R} + \mathrm{G} + \mathrm{B}}{3 * \mathrm{R}} + \frac{\mathrm{R} + \mathrm{G} + \mathrm{B}}{\mathrm{R} - \mathrm{G}} < 1.4$$

2) *Motion detection:* The used skin formula may include a wide range of colors. Therefore, many regions will appear other than the skin tone regions. However, these non-skin regions will be excluded by considering moving objects only. The moving objects can be detected by subtracting two consecutive frames [8] and thresholding the output.

3) *Combination of motion and skin color:* The hand gestures information consist of movement and skin pixels, therefore the logic 'AND' is applied to combine them:

$$P_i(x, y) = M_i(x, y) \wedge S_i(x, y)$$

Where $M_i(x, y)$ and $S_i(x, y)$ indicate the moving and skin tone pixels. Figure 2 illustrate the idea.

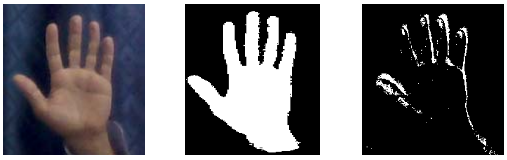

(a) (b) (c)

Fig. 2. The hand detection: (a) Original image, (b) skin color region, (c) combination of motion and skin pixels.

4) *Region identification:* The center of the desired hand is determined as follows:

i. Midpoint of each movement and skin color row will be calculated as shown in Eq. 1, and stored in a collection (e.g. Array), so that the row's number is stored in this array as an address. Midpoint manipulation process will be illustrated in Figure 3.

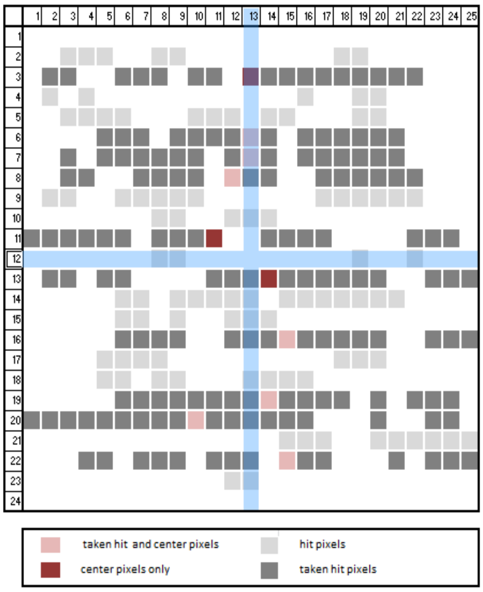
Light and dark grey and pink pixels are all satisfy skin tone and movement pixels (hit pixels).

Rows that contain more than 15 skin tone and moving pixels, the pixels in it are colored dark grey (taken hit pixels).

The appendant left table consists of two columns. Right column shows number of pixels satisfying skin tone and motion in each row, and left column shows the center of each rows that contains taken hit pixels.

The pink and red pixels refer to the center of row in which these pixels are found.

Centres	No of Hit
0	0
0	7
13	16
0	5
0	11
13	15
13	16
12	15
0	13
0	5
11	16
0	4
14	16
0	14
0	6
15	16
0	7
0	8
14	17
10	19
0	9
14	15
0	2
0	0



X-center	Y-center
12.9	12.5

Fig.3 Hand center determination

ii. Summing prestored midpoints values and then dividing the result by number of rows that are stored in the array which gives the evenhanded lines, as shown in Eq. 2 and Eq. 3.

$$Row's\ center = \frac{\sum X\ position\ of\ taken\ hit\ pixels}{Number\ of\ taken\ hit\ pixels} \qquad \text{Eq(1)}$$

$$Vertical\ evenhanded\ line = \frac{\sum Centers_{(i)}}{number\ of\ hit\ rows} \qquad \text{Eq(2)}$$

$$Horizontal\ evenhanded\ line = \frac{\sum Rows\ addresses}{Rows\ count} \qquad \text{Eq(3)}$$

Algorithm I shows the details of hand center detection algorithm.

ALGORITHM I. Hand center detection

```
for i←0 to imageHeight
   for j←0 to imageWidth
      if Px(i,j) satisfiy Motion
      AND inSkinRange   then
            TruePixel ← TruePixel + 1
          XTruePx ← XTruePx + j
     end if
   end for
   if  TruePixel > 15 then
      MPArray[i] ← ( XTruePx /
 TruePixel )
   else
      MPArray[i] ← 0
   end if
 end for
```

```
for i←0 to ImageHeight
    if MPArray[i] ≠ 0 then
       MPSum ← MPSum + MPArray[i]
       TrueRow ←TrueRow + 1
      YTrueRow ←YTrueRow + i
    end if
 end for

if TrueRow > 0 AND YTrueRow > 0
  AND MPSum > 0 then
   X_Center ← MPSum / TrueRow
   Y_Center ← YTrueRow / TrueRow
  end if
```

Figure 4 shows the results of hand position detection using different backgrounds.

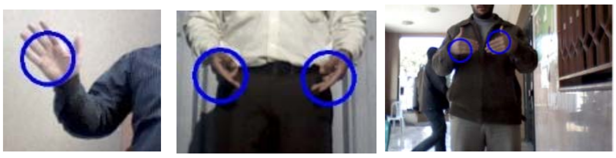
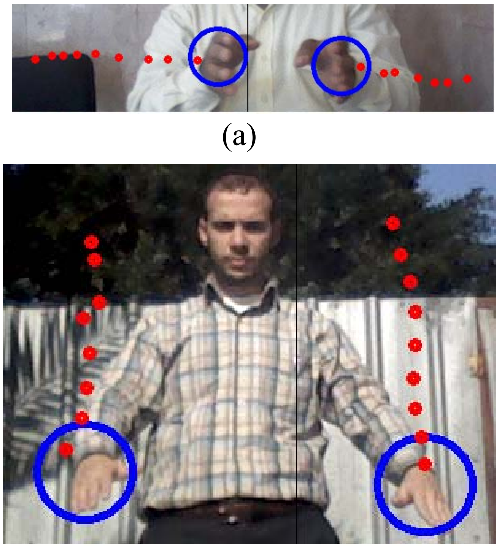


Fig. 4. Hand region detection images.

IV. Hand gesture trajectory TRACKING

The overall system for hand region tracking has two stages: the first stage focuses on the motion information and stores the latest 10 transitions of the predefined central points on queue as shown in Figure 5, whereas the second stage is focused on analysing and processing these points as follows:

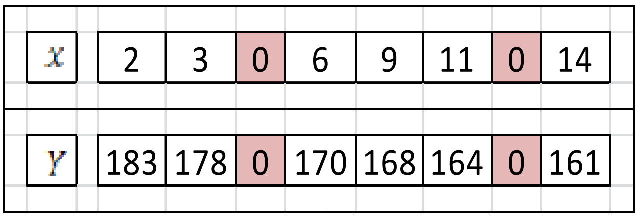
Select $x - position$ from centroid $P(x, y)$ and store it in X_{array} and repeat this work for Y as Y_{array} with neglecting the value of the point at $P(0,0)$ as shown in Table 1.



(a)

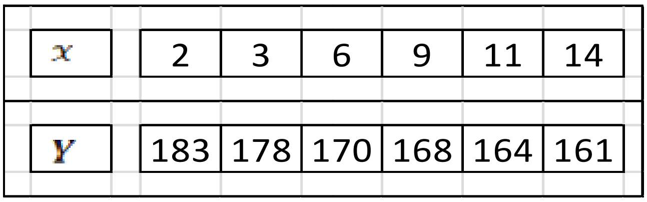
(b)

Fig. 5 (a),(b) illustrate the latest manipulated points.



x	2	3	0	6	9	11	0	14
Y	183	178	0	170	168	164	0	161

(a)



x	2	3	6	9	11	14
Y	183	178	170	168	164	161

(b)

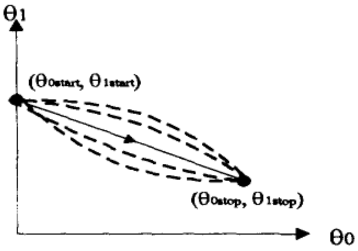
Table 1,(a) The original array (b) the processed array with $P(0,0)$ *elimination*

Being impossible for a person to make more than 25 gestures in a second [9], it is feasible to consider only 10 frames in a second; a tracking system should be implemented in order to avoid the complete analysis of each frame. Anyway, the use of the system currently would ensure a real time processing, which allows the use of the proposed approach in real time video applications.

To measure the distance and the angle between each consecutive center in the previous midpoints array, we use Pythagoras's formulas as shown in Figure 6.

(a) $$d_{(a,b)} = \sqrt{\Delta x^2 + \Delta y^2}$$

(b) $$\theta_{(a,b)} = \tan^{-1}\frac{\Delta y}{\Delta x}$$



(c)

Fig. 6 (a),(b) Equations used to find the angle and distance between each two consecutive point, (c) variable position of two consecutive points with different angles.

Then we specify four variables associated with the four major directions

variables → {
var_0, variable associated with 0°.
var_90, variable associated with 90°.
var_180, variable associated with 180°.
var_270, variable associated with 270°.

In order to avoid erroneous results and make the desired movement more accurate we attach every angle with ±15° as shown in Fig. 7.

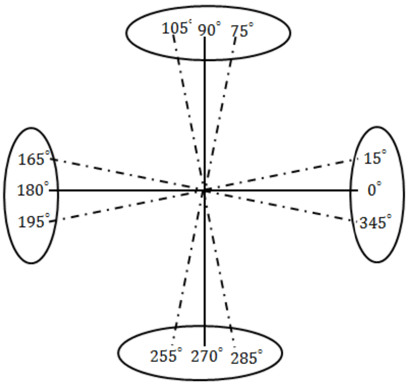


Fig. 7. Four major angles with $\pm 15^{\circ}$

After that, we ignore the movements that exceeded the angles latitude and those that did not relate to any of the specified ranges, then counting the distances for each predefined variables that belongs to the same extent.

For the last step in hand tracking phase, we need to create two variables for each angle as follows, $angleCounter$ to evaluate number of movements related to the given angle, $sumDistance$ to sum up the distances that that travelled by the hand at a given angle.

If the angle between each prestored consecutive midpoints belongs to one of the predefined four basic direction extents, then increment $angleCounter$ for the given angle and add the distance between the two consecutive midpoints to $sumDistance$.

Thus, we have obtained sufficient information about every movement and in which direction through the related $angleCounter$ and $sumDistance$ variables. Algorithm II clarify the hand tracking method.

Algorithm II. Hand movement tracking

```
for i ← 0 to array.Length
  $d = distance\ between\ C_i\,(x,y) and\ \ C_{i-1}(x,y)$
  $\theta = Angle\ \ between\ C_i\,(x,y) and\ \ C_{i-1}(x,y)$
  if -15 < θ <15 then
    rtl  ← rtl + 1
    Sum_rtl ← Sum_rtl + d
  else if  75 < θ <105 then
    dtu  ← dtu + 1
    Sum_dtu ← Sum_dtu + d
  else if  165 < θ <195 then
    ltr ← ltr + 1
    Sum_ltr ← Sum_ltr + d
  else if 255 < θ < 285 then
    utd ← utd + 1
    Sum_utd ← Sum_utd + d
  end if
end for
```

V. Hand gesture recognition

Since the considered gestures are processed dynamically, we need a mechanism to recognize gestures using their axial movement. The recognition scheme must be robust and should accommodate variations in the attributes of a gesture.

At this stage, we divide the analysis phase into two modules. (a) One hand movement analysis, (b) Both hands analysis. The angle and distance of the hand centroid, and the counters of the movements in each direction were used to recognize the gestures as shown in the following two algorithms.

Algorithm III shows one hand movements recognition method:

Algorithm III. One Hand recognition

```
if Sum_rtl > 150 AND rtl > 4
  AND ltr* < 2  then
    DoEvent(Right To Left )
else if Sum_ltr > 150 AND ltr > 4
  AND rtl* < 2  then
    DoEvent(Left to right )
else if Sum_utd > 200 AND utd > 4
  AND utd* < 2  then
    DoEvent(Up to down )

else if Sum_dtu > 200 AND dtu > 4
  AND dtu* < 2  then
    DoEvent(down to up )
```

Algorithm IV. shows both hands movements recognition method:

Algorithm IV. Both Hand recognition

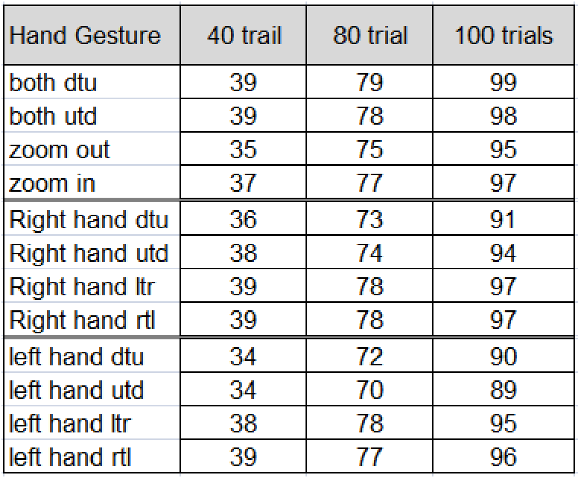
```
if Sum_utd > 200 AND Sum_utd * > 200
AND utd > 3 AND utd * > 3 then
    DoEvent(Both hand up )

else if Sum_dtu > 200
AND Sum_dtu  * > 200 AND dtu > 3
AND dtu* > 3  then
    DoEvent(Both hand Down )

else if Sum_ ltr > 150
AND Sum_rtl * > 150 AND ltr > 3
AND rtl* > 3  then
    DoEvent(Zoom In )

else if Sum_ rtl > 150
AND Sum_ltr * > 150 AND rtl > 3
AND ltr* > 3  then
    DoEvent(zoom Out )
```

VI. Experimental results

The proposed method was implemented using an optimized C# code and without any auxiliary library. The experimental results are illustrated in Table 2.

We have tested 12 different hand gestures. Each gesture is tested 40, 80 and 100 times with three different persons. There are 12 different gestures, and 2640 image sequences used. The recognition rate of this system is 94.21%.

Table 2. Experimental results of the gestures recognition system



Hand Gesture	40 trail	80 trial	100 trials
both dtu	39	79	99
both utd	39	78	98
zoom out	35	75	95
zoom in	37	77	97
Right hand dtu	36	73	91
Right hand utd	38	74	94
Right hand ltr	39	78	97
Right hand rtl	39	78	97
left hand dtu	34	72	90
left hand utd	34	70	89
left hand ltr	38	78	95
left hand rtl	39	77	96

* dtu = down to up; utd = up to down; rtl = right to left; ltr = left to right.

VII. Conclusions

A new technique has been proposed to increase the adaptability of a gesture recognition system. We have implemented a real-time version, using an ordinary workstation with no special hardware beyond a video camera input. The technique works well under different degrees of scene background complexity and illumination conditions with more than 94% success rate.

References

[1] T. Baudel, M. Baudouin-Lafon, Charade: remote control of objects using free-hand gestures, Comm. ACM 36 (7) (1993) 28–35.

[2] D.J. Sturman, D. Zeltzer, A survey of glove-based input, IEEE Computer Graphics and Applications 14 (1994) 30–39.

[3] J. Davis, M. Shah, Visual gesture recognition. IEE Proc. Vis. Image Signal Process, 141(2) :101-106, 1994.

[4] A. Bobick, A. Wilson, A state-based technique for the summarization and recognition of gesture. In Proc. IEEE Fifth Int. Conf. on Computer Vision, Cambridge, pp. 382-388, 1995.

[5] E. Hunter, J. Schlenzig, and R. Jain.Posture Estimation in Reduced- Model Gesture Input Systems. Proc. Int´l Workshop Automatic Face and Gesture Recognition, pp. 296-301, 1995.

[6] C. Maggioni. Gesturecomputer. New Ways of Operating a Computer, *Proc.* Int´l Workshop Automatic Face and Gesture Recognition, 1995.

[7] R. Lockton, A.W. Fitzgibbon, Real-time gesture recognition using deterministic boosting, Proceedings of British Machine Vision Conference (2002).

[8] R. Gonzales, and E. Woods, “Digital Image Processing,” Prentice Hall, Inc, New Jersey, 2002.

[9] E. Sanchez-Nielsen, L. Anton-Canalis, M. Hernandez-Tejera, Hand gesture recognition for human machine interaction, Journal of WSCG, Vol.12, No.1-3, 2003.