Trajectory Based Hand Gesture Recognition

DANIEL POPA, GEORGIANA SIMION, VASILE GUI, MARIUS OTESTEANU
Faculty of Electronics and Telecommunications
"Politehnica" University of Timisoara
Bd. V. Parvan, Nr. 2, 300223 Timisoara
ROMANIA

Abstract: The recognition of hand gestures from image sequences is an important and challenging problem. This paper presents a robust solution to track and recognize a list of hand gestures from their trajectory. The CamShift algorithm is used for hand tracking and the resulting trajectory is segmented into strokes. The trajectory of recognized gestures consists of at least 2 strokes. The gestures are classified based on the number of strokes, the strokes' angle sequence and, eventually, strokes proportionality. The low computational cost of the algorithm allows implementation on low-cost processing systems.

Key-Words: video tracking, CamShift, robust methods, gesture recognition, human machine interface.

1 Introduction

Human gestures [1] are expressive human body motions, which generally contain spatial and temporal variation. To handle these variations, an appropriate representation must be chosen.

A vast amount of work in gesture recognition has been performed in the area of computer vision, and is reviewed in [2]. These works can be divided into two categories: trajectory-based and dynamics model-based analysis. The trajectory-based approach matches curves in configuration space to recognize gestures [3]. The dynamics model-based approach learns a parametric model of gestures.

Gesture recognition systems in general are composed of three main [4] components: image preprocessing, tracking, and gesture recognition. In individual systems some of these components may be merged or missing, but their basic functionality [5] will normally be present.

Image preprocessing is the task of preparing the video frames for further analysis by suppressing noise, extracting important clues about the position of the object of interest (for example hands) and bringing these on symbolic form. This step is often referred to as feature extraction.

Tracking – on the basis of the preprocessing, the position and possibly other attributes of the object (hands) must be tracked from frame to frame. This is done to distinguish a moving object of interest from the background and other moving objects, and to extract motion information for recognition of dynamic gestures.

Gesture recognition decides if the user is performing a meaningful gesture based on the collected position, motion and pose clues.

The classical algorithms from the field of pattern recognition are Hidden Markov Models (HMM), correlation, and Neural Networks. Especially the first two have been used successfully in gesture recognition while the Neural Networks often have the problem of modeling non-gestural patterns [6]. HMM is a typical dynamics model and was proven to be robust in its recognition of gestures [7]. The HMM model has been extended to a more general model named Dynamic Bayesian Networks [8].

Black and Jepson extended the CONDENSATION algorithm [9], to recognize gestures and facial expressions in which human motions were modeled [10], [11], [12] as temporal trajectories of some estimated parameters (which describe the states of a gesture or an expression) over time.

Many gesture recognition methods used colored gloves or markers to track hand movements. Fels and Hinton used data gloves and Polhemus sensors to extract 3D hand location, velocity, and orientation [13]. Bobick and Wilson [14] extract 3-D location of hands using stereo cameras and skin color to recognize a set of 32 size gesture using parameterized hidden Markov Model and data glove.

Recognition of human gestures is important for human-computer interfaces, automated visual surveillance, video library indexing [1], remote control of home appliances, such as TV sets and DVD players [15], which in the future could be extended to the more general scenario of ubiquitous computing in everyday situations, control of a videogame etc.

In this paper hand gestures and a trajectory based approach are used. Our goal is to develop a low computational cost real-time gesture recognition

algorithm for a human-machine interface (HMI), able to run on low-complexity hardware systems. The system must be able to learn from the user a small number of gestures in order to recognize them later. The trajectory based approach was chosen as skin detection is quite well developed and robust [16] and relatively robust low computational cost tracking algorithms are also available [17], [18].

2 Problem Formulation

Our purpose is to track and discriminate some hand gestures composed of multiple quasi-linear segments (strokes). The algorithm must be able to distinguish the gestures based on the number of segments, the angles between segments and the horizontal axis, the angles between consecutive segments and segments proportionality.

Using these parameters it is possible to define a variety of gestures which can be easily discriminated: square, wide/tall rectangle, triangle, Z, N vertical/horizontal multiple strokes, cross etc. Each gesture may have a different meaning depending on the movement direction. For example one action can be associated with drawing a square clockwise, and another (possibly opposite) action can be associated with drawing a square counterclockwise.

The Hue, Saturation, Value (HSV) color space is usually preferred [17] instead of the Red, Green, Blue (RGB) color space in human skin tracking applications because hue is less sensitive to different skin colors and because it is more robust to illumination changes.

As the overall computational cost must be low, complex tracking solutions like those based on particle filters are not appropriate, and a tracking algorithm based on target representation and localization must be used. Such a solution is offered by the CamShift (Continuously Adaptive Mean Shift) algorithm introduced by Bradski [17]. An implementation of this algorithm is available in the OpenCV library [19].

The CamShift Algorithm mainly consists of the following steps:

1. Choose the initial location of the search window;
2. Mean Shift (one or many iterations); store the zeroth moment;
3. Set the search window size equal to a function of the zeroth moment found in Step 2;
4. Repeat Steps 2 and 3 until convergence.

The algorithm outputs the center, size and orientation of the tracked object. The object trajectory can be obtained from the positions of the center of the object in successive frames.

3 Problem Solution

The method we propose, presented in figure 1, uses a CamShift tracker to track the hand of the user and saves a trajectory obtained from the centers of the tracked region. The saved trajectory is then segmented into strokes. Considering all the gestures used are composed of a reduced number of strokes, information like number of strokes, average angles with horizontal axis, angles with neighboring segments and segments proportionality can be easily derived from the segmented trajectory. Finally, based on the extracted features the gesture is uniquely identified.

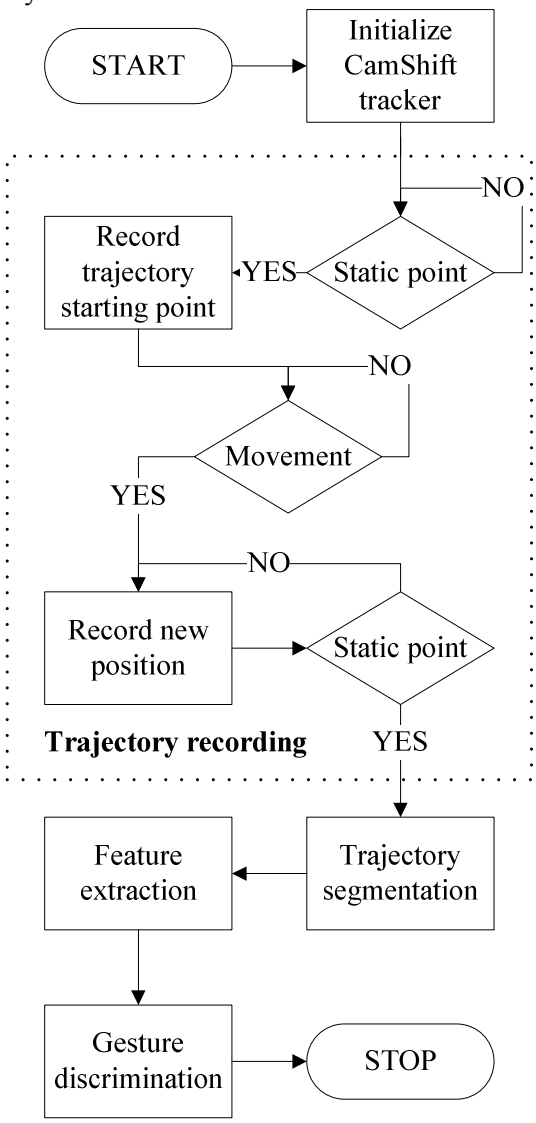


Fig. 1. Flow diagram of the trajectory based recognition algorithm

3.1 Tracking

The region containing the hand to track must be firstly selected. Then, a mask is applied on the HSV image in order to eliminate the pixels which have too small

saturation and also pixels with too small or too high value. Hue is too noisy for these removed pixels.

A model of the desired hue of the object to track is created for the first frame using a color histogram. The CamShift algorithm is then used to track the object in the next frames.

3.2 Trajectory recording

The consecutive centers of the tracked region define a relatively rough (noisy) trajectory, increasing the difficulty of strokes detection. Therefore the trajectory is smoothed so that each new recorded trajectory point, $t[i]$, is obtained as a weighted average of the new measured point, $m[i]$, and the previous trajectory point, $t[i-1]$:

$$t[i] = \alpha m[i] + (1-\alpha) t[i-1]. \quad (1)$$

Our practical tests revealed that a value of 0.4 for the weighting parameter, α, produces a relatively smooth trajectory. Smaller values tend to oversmooth the trajectory, while larger values lead to relatively rough trajectories.

The recording of a new gesture trajectory is triggered by a movement of the user's hand occurring after a short interval (1-2 seconds) of static position. Minimum thresholds are imposed to the amplitude and speed of the movement in order to avoid false triggering due to tracking noise or hand trembling. The gesture trajectory recording ends when the movement speed falls below the imposed threshold for at least 2 seconds.

A set of angles with the horizontal axis is computed over the recorded trajectory. Computing the angle for each small segment determined by two consecutive points of the trajectory may result in a very noisy angle set, with many false angle discontinuities. This noise is caused by angles between trajectory points that are relatively close to each other, because the image is sampled on a rectangular grid. Selecting a reduced number of trajectory points using a fixed step (e.g. choosing each second or third point of the trajectory) results in a relatively smoothed angle set. Improved results can be obtained by adaptively selecting trajectory points based on a threshold distance. Even with the fixed step selection, a distance threshold must be imposed in order to avoid computing the angle if the two points have the same position.

3.3 Trajectory segmentation

In order to split the trajectory into strokes, the ends of these strokes must be detected. The starting point of the trajectory is also the starting point of the first segment, and the end point of the trajectory is the end point of the last segment. All other strokes' end points are detected as angle discontinuity points, the starting point of each segment being the end of the previous segment. A stroke discontinuity is detected as a point between two small segments which have significantly different angles with the horizontal axis. For this purpose a derivative over the angles set is computed:

$$\frac{d\theta}{di}[i] = \frac{\theta[i] - \theta[i-1]}{i-(i-1)} = \theta[i] - \theta[i-1]. \quad (2)$$

All the maximums of the absolute value of this derivative which exceed an imposed threshold indicate an important angle discontinuity and correspond to stroke ends. Usually a threshold of 30° is enough to reject the small segments angle noise. When computing the derivatives, special care must be taken due to the circular definition of angle, so the angle difference must also be computed on a circular domain (e.g. the difference between angles of –179° and 178° is of 3° not –357°). Therefore, the angle difference in (2) is computed using the relation:

$$\frac{d\theta}{di}[i] = \begin{cases} \theta[i] - \theta[i-1] + 360, & \theta[i] - \theta[i-1] < -180 \\ \theta[i] - \theta[i-1], & |\theta[i] - \theta[i-1]| \le 180 \\ \theta[i] - \theta[i-1] - 360, & \theta[i] - \theta[i-1] > 180 \end{cases} \quad (3)$$

Another threshold must be imposed on the minimum length of a stroke, in order to avoid detection of false strokes. A reasonable value for this threshold is 1/10 of the image height.

3.4 Feature extraction

After the stroke ends are detected, the parameters of the trajectory are extracted. The useful parameters which must be extracted are: the number of strokes, the strokes' angle sequence and the strokes' lengths.

The number of strokes can be easily obtained from the number of stroke ends. Also the strokes' lengths can be easily obtained using the coordinates of the stroke ends.

The analysis of the angle sequence is done sequentially, at each step being possible to stop the algorithm, if an angle value can not be classified.

Each gesture known by the system is represented by a codified angle sequence. The angles of the 8 directions allowed and the associated codes are presented in table 1. Opposite directions have complementary codes.

Table 1

Angle [°]	0	45	90	135	–45	–90	–135	180
Code	0	1	2	3	4	5	6	7

A mean angle is computed over each stroke, and then the angles between consecutive strokes may also be computed. The mean angle of the segment must be

classified and assigned to one of the 8 classes. In order to assign the angle to a class, its value must fit a ±20° window around the standard value. A 5° guard space is left between consecutive windows as shown in figure 2. If a stroke's angle falls within a guard space, it is not possible to classify it and the analysis is stopped, invalidating the current gesture.

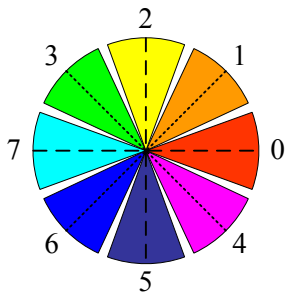


Fig. 2. Angle classes

The first angle of the sequence is the angle between the first stroke and the horizontal axis, while the next angles can be either the angles made by each stroke in the sequence with the horizontal axis or with the previous stroke. The second solution may increase the robustness to small global trajectory rotations.

3.5 Gesture discrimination

A gesture consists of a minimum of 2 strokes. Each gesture is uniquely identified based on:

- number of strokes (>1),
- angle sequence and
- strokes proportionality.

The discrimination process is done sequentially, based on the 3 parameters, as shown in figure 3. As all the gestures that reach this processing step were already validated from the number of strokes and angle sequence point of view, these parameters are strict, while the stroke proportionality is allowed to vary within an error window.

The first parameter to take into account when discriminating between gestures is the number of strokes. This step removes from further analysis all the gestures with different number of strokes.

In the next step the angle sequence is decoded. In most cases this step is enough to uniquely identify the gesture and terminate the analysis.

Some 4-stroke gesture codes (e.g. square/rectangle codes) need further classification based on strokes proportionality. The strokes proportions are allowed to vary within an error window. The error window must be chosen large enough to accommodate the imperfections of the hand drawn trajectory, but small enough to allow correct discrimination between different gestures.

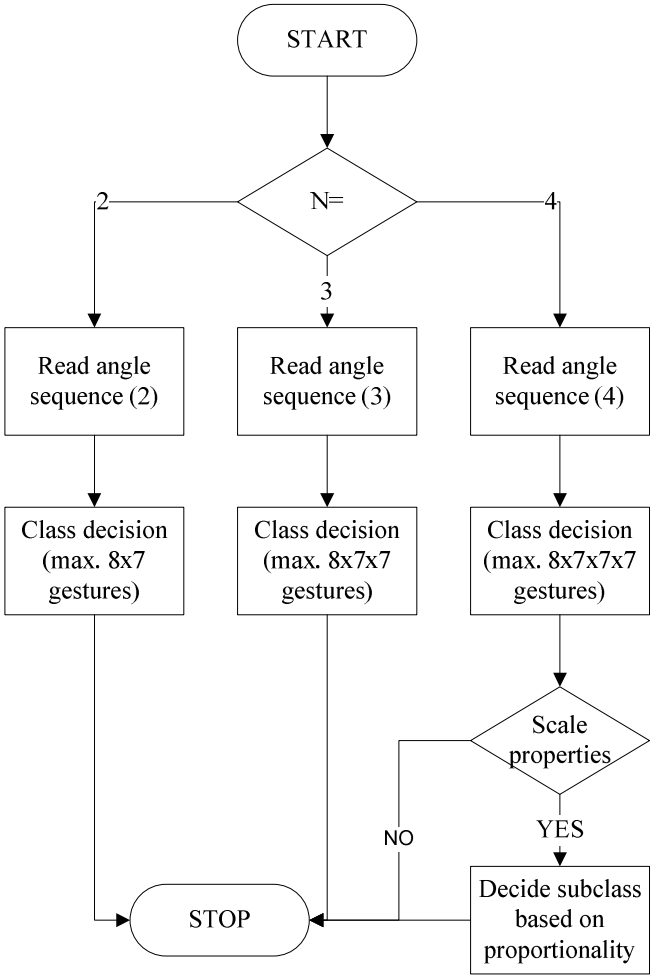


Fig. 3. Sequential gesture discrimination

4 Results

The hardware processing system we used to implement and test the solution was a PC with a 1.6 GHz AMD Athlon XP processor and 768 MB of RAM. Video acquisition was realized using a commercial USB webcam with 352×288 video resolution. The software application was implemented in Visual C++ and some functions from the OpenCV library were used.

Fig. 4. The CamShift tracker.

Our application uses a tracker based on the CamShift function of the OpenCV library. This algorithm is able to track well a hand based on the hue, assuming saturation and value thresholds are relatively well calibrated and no occlusions with objects of similar color occur. Figure 4 presents the tracker focused on a hand.

The recorded trajectory for a 3-stroke gesture is presented in figure 5.

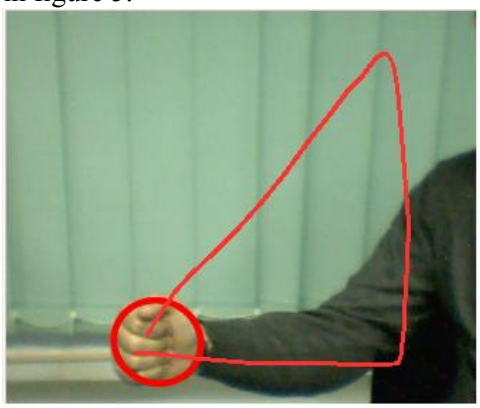
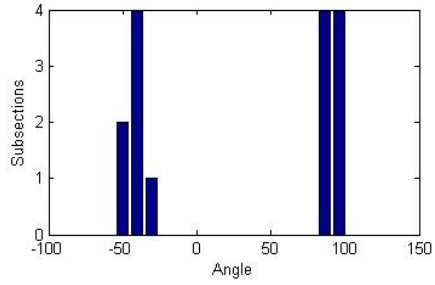


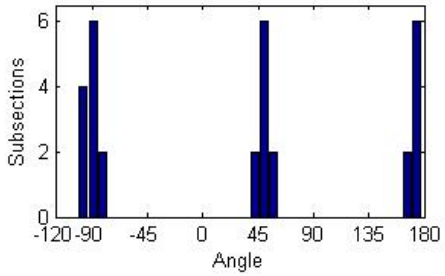
Fig. 5. Trajectory of a 3-strokes gesture.

Histograms for 2, 3 and 4-strokes gestures are presented in figure 6. The 3 histograms of subsection angles indicate that the subsection angles are clustered, allowing easy separation of the strokes.

a)



b)



c)

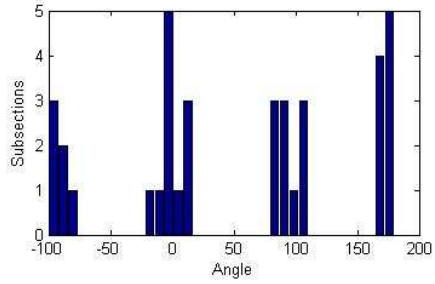


Fig. 6. Angle histogram for: a) 2-stroke gesture, b) 3-stroke gesture, c) 4-stroke gesture.

The resulting mean stroke angle sequence for the gesture in figure 5 is: 49.9°, -92.1°, 174.1°, values which easily fit the tolerance window around the standard directions. The resulting coded sequence for this example is "1, 5, 7".

The total number of possible gestures composed of a certain number, n, of strokes using the 8 directions is $8 \times 7^{n-1}$. For the first stroke, all the 8 directions are possible. For each subsequent stroke only 7 possible directions are available (the direction of the previous stroke is excluded). Over 3000 gestures are possible even with a maximum of only 4 strokes, and the total number of possible gestures increases exponentially with the number of strokes.

Not all of the possible gestures can be used in practical applications (e.g. successions containing more than 2 strokes with positive Δy, without at least one stroke with negative Δy are very likely to exceed the frame space captured by the camera).

We tested the solution for a small number of gestures composed of 2, 3 and 4 strokes, with sharp and/or right angles between consecutive strokes and the results are promising. Our algorithm runs in real time on the specified hardware system, being able to recognize successfully the gestures we tested so far.

5 Conclusions

Other gesture recognition solutions, [20], [21], rely on processing different successions of shape and positions of the hand and achieve relatively good successful recognition rates at the price of significantly higher computational complexity.

Our solution has a reduced computational complexity, using a mean-shift based tracker, and syntactic, trajectory based gesture recognition. Our project is still in an early phase and the tests we were able to make so far are not very extensive, but the obtained results are promising, entitling us to believe it is possible to achieve good successful recognition rates at a reduced computational cost.

The low computational cost allows the use of the algorithm to implement HMIs on relatively cheap systems. The user can train the system to associate each valid gesture to a specific action.

Further developments include better definition of valid gestures, in order to exclude the gestures which would normally exceed the area of the frame captured by the camera, being also possible to relax the angle restrictions for some classes of gestures (e.g. allow equilateral triangles). An audio feedback to the user would also be useful, in order to allow it to focus its sight on its main activity (e.g. driving).

References:

[1] T. S. Wang, H. Y. Shum, Y. Q. Xu and N. N. Zheng, Unsupervised Analysis of Human Gestures, *IEEE Pacific Rim Conference on Multimedia*, 2001, pp. 174-181.

[2] Y. Wu, T. Huang, Vision-Based Gesture Recognition: A Review, *Proceedings of the International Gesture Recognition Workshop*, 1999, pp. 103-115.

[3] L. Campbell and A. Bobick.: Recognition of Human Body Motion Using Phase Space Constraints, Proceedings of the Fifth International Conference on Computer Vision, 1995, pp. 624-630.

[4] T. Moeslund and L. Nrgaard, *A Brief Overview of Hand Gestures used in Wearable Human Computer Interfaces*, Technical Report CVMT 03-02, Computer Vision and Media Technology Laboratory, Aalborg University, DK, ISSN: 1601-3646, 2003.

[5] V.I. Pavlovic, R. Sharma, and T.S. Huang, Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, 1997, pp. 677–695.

[6] H.K. Lee and J.H. Kim, An HMM-based Threshold Model Approach for Gesture Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 10, 1999, pp. 961–972.

[7] T. Starner and A.Pentland.: Visual Recognition of American Language Using Hidden Markov Models, Proceedings of the International Workshop on Automatic Face and Gesture Recognition, Zurich, 1995, pp. 189-194.

[8] V. Pavlovic, J. M. Rehg and J. MacCormick, Impact of Dynamic Model learning on Classification of Human Motion, *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2000, pp. 788-795.

[9] M.J. Black and A.D. Jepson, A Probabilistic Framework for Matching Temporal Trajectories: CONDENSATION-Based Recognition of Gestures and Expressions, *Proceedings of the Fifth European Conference on Computer Vision*, 1998, pp. 909-924.

[10] M. H. Yang, N. Ahuja and M. Tabb, Extraction of 2D Motion Trajectories and Its Application to Hand Gesture Recognition, *IEEE Transactions on pattern analysis and machine intelligence*, Vol. 24, No. 8, 2002, pp. 1061- 1074.

[11] L. Nørgaard, *Probabilistic Hand Tracking for Wearable Gesture Interfaces*, Master's thesis, Laboratory of Computer Vision and Media Technology, Aalborg University, Denmark, 2003.

[12] J. Lin, Y. Wu, and T.S. Huang. Capturing human hand motion in image sequences. *Proceedings of IEEE Workshop on Motion and Video Computing*, 2002, pp. 99-104.

[13] S.S. Fels and G.E. Hinton, Glove-Talk II: A Neural Network Interface which Maps Gestures to Parallel Format Speech Synthesizer Controls, IEEE Transactions on Neural Networks, Vol. 9, No. 1, 1997, pp. 205-212.

[14] A.D. Wilson and A.F. Bobick, Parametric Hidden Markov Models for Gesture Recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 9, 1999, pp. 884-900.

[15] S. Lenman, L. Bretzner and B. Thuresson, *Computer Vision Based Recognition of Hand Gestures for Human-Computer Interaction*, Technical report TRITANA -D0209, CID-172, ISSN 1403-0721, Department of Numerical Analysis and Computer Science, 2002.

[16] M. J. Jones and J. M. Rehg, Statistical Color Models with Application to Skin Detection, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1999, 274-280.

[17] G. R. Bradski, Computer vision face tracking as a component of a perceptual user interface, *Intel Technology Journal Q2 1998*, http://developer.intel.com/technology/itj/archive/1998.htm.

[18] D. Comaniciu, V. Ramesh, P. Meer, Kernel-Based Object Tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 5, 2003.

[19] http://opencvlibrary.sourceforge.net.

[20] L. Bretzner, I. Laptev and T. Lindeberg, Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering, *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition,* 2002, pp 405-410.

[21] J.H. Shin, J.S. Lee, S.K. Kil, D.F. Shen, J.G Ryu, E.H. Lee, H.K. Min and S.H. Hong, Hand Region Extraction and Gesture Recognition using entropy analysis, *IJCSNS International Journal of Computer Science and Network Security*, Vol. 6 No. 2, 2006, pp. 216-222.