Convexity Local Contour Sequences for Gesture Recognition

Pablo V. A. Barros, Nestor T. M. Junior, Juvenal M. M. Bisneto,
Bruno J. T. Fernandes, Byron L. D. Bezerra, Sergio M. M. Fernandes
Polytechnics School
University of Pernambuco
Recife, Brazil
{pvab, ntmj, jmmb, bjtf, byronleite, smmf}@ecomp.poli.br

ABSTRACT

Algorithms for hand feature extraction used in gesture recognition systems have some problems such as unnecessary information gathering. This paper proposes a novel method for feature extraction in gesture recognition systems based on the Local Contour Sequence (LCS). It is called the Convexity Local Contour Sequence (CLCS) and represents the hand shape only with the most significant information. This generates a smaller output result, but capable to model an entire dynamic gesture. It is used to classify dynamic gestures with an Elman Recurrent Network and Hidden Markov Model and presents a better result compared to regular LCS.

Categories and Subject Descriptors

I.4 [**Image Processing and Computer Vision**]: Feature Measurement,Segmentation,Scene Analysis

General Terms

Algorithms

Keywords

Image Processing, Gesture Recognition, Hidden Markov Model, Recurrent Neural Network, Local Contour Sequence

1. INTRODUCTION

Nowadays, human-computer interaction based on gesture recognition systems is becoming more usual. There are a lot of applications for this technology, such as video games, televisions, systems for impaired people, augmented reality, medical applications, among others. These systems can be categorized in three different approaches. One is glove-based analysis, where sensors are attached to a glove. Look-up table software is usually provided with the glove to be used for hand gesture recognition [3]. The next one is the recognition of drawing gestures that is basically a character recognition problem. This problem was presented in the work of Schlölmer et al. [20] where it was used a Wii controller to drawn the gestures to be recognized. The last category is analysis based on computational vision that uses video cameras to capture the movement of the hands without gloves. This category uses a set of image processing techniques to identify and extract features for gesture recognition as in the work of Li et al. [14] where he used image processing techniques for gesture recognition applied to games.

Techniques such as fuzzy decision trees, transition movement models, spatio-temporal, or common sense context were applied with remarkable success in gesture recognition [5, 11]. Yuan et al. [22] designed a gesture recognition system using a Jordan Recurrent Neural Network. Florez et al. [6] presented a structure capable of gesture recognition by hand posture and its movement. Topology of a self-organizing neural network determines posture, whereas its adaptation dynamics throughout time determines gesture. Murakami et al. [17] developed a posture recognition system using an Elman Recurrent Neural Network which could recognize a finger alphabet of 42 symbols of Japanese sign language.

Using a Hidden Markov Model, Gaus [1] has developed a single gesture recognition system for Malaysian Sign Langue. He obtained 83% success rate to recognize 112 signs. Meena [16] proposed a feature extraction algorithm, called Local Contour Sequence (LCS), that represents the hand shape by it contour. Gupta [8] uses the LCS to generate an output vector to be classified to a Support Vector Machine. He got an elevated classification rate for simple gestures using LCS, but it generates a sizeable output vector, resulting in a long time of training and in the curse of dimensionality which says that the numerical approximation of a function will require more computation as the number of active variables grows. Thus, problems are generated such as the presence of irrelevant features and the correlations between subsets of features have a strong influence on learning.

Taking in consideration such problems we propose the Convexity Local Contour Sequence (CLCS) that calculates the LCS of a selected sequence of points, chosen dynamically by the CLCS algorithm. It uses an arrangement of image processing techniques, such as image threshold, contour detection and convexity hulls to identify the features. It generates an output vector containing a limited number of features, capable to model the gesture better than the reg-

SAC'13 March 18-22, 2013, Coimbra, Portugal.

```
List<double> GestureCLCS = new List<double>();

foreach (Image frame in Gesture.ImageSequence)
{
    double[][] result = BackgroundRemoval(frame);

    result = FindContour(result);

    result = Douglas_PeuckerAlgorithm(result);
    result = SklankyAlgorithm(result);

    double[] LCS = doLCS(result);

    GestureCLCS.Add(LCS);

}
```

Figure 1: CLCS code segment representing the complete feature extraction algorithm for a image sequence.

ular LCS. This vector is used as input in a Hidden Markov Model and an Elman Recurrent Network to be classified.

This paper is structured as follows. Section II describes the CLCS algorithm. Section III presents the experimental results. Finally, in Section IV, the conclusions and some future work are given.

2. CONVEXITY LOCAL CONTOUR SEQUENCE

Convexity Local Contour Sequence (CLCS) is a method for hand feature extraction that can be applied in dynamic and static gesture recognition. It uses the hand shape variations in the gesture movement to generate a descriptor of the gesture that can be used in classifications techniques. This is possible through the dynamic selection of the minimal amount of points capable to represent the hand shape in each movement. This generates a representative model that is used to calculate the LCS and results in a gesture representation output vector. The CLCS is composed by three steps: background removal, hand contour identification and feature extraction. Figure 1 shows the code segment of the CLCS algorithm. The input of the algorithm is composed by an image set representing the hand gesture. Each frame is used as input and has its owns CLCS vector. The vector descriptor of all image set is used to represent the entire gesture.

The input is composed by raw images containing the entire hand and the background, the CLCS algorithm needs to identify and remove the latter. To accomplish this, the Otsu Threshold Technique [18] is used, resulting in a separated image containing only the hand shape in a binary representation. Even after applying Otsu, the image still has some noise, especially in the edges. A Median Filter [7] is used and results in a smoothed image.

The binary image containing only the hand shape information is used as input for step two, responsible for find the hand contour. The Canny edge detection algorithm [2] is applied and is capable to find the hand contour precisely. This creates a reduced image data that is more representative than the entire hand for CLCS.

The hand contour is used as input for the third step, feature extraction. First, the minimal number of points that can represent the gesture is selected. This is performed dynamically for each different hand position as the hand shape changes through gesture. This is accomplished using the Douglas-Peucker Algorithm [10] to create an approximation curve of the external points, forming a polygon that represents the gesture.

The Douglas-Peucker algorithm recursively divides the points sequence. Initially it is given all the points between the first and last point of the hand shape. It automatically marks the first and last point to be kept in the new geometrical model. then, it finds the point that is furthest from the line segment formed by the two given points. If the point is closer than and given measure, called E, to the line segment then any points not currently marked to be kept can be discarded. If the point furthest from the line segment is greater than E that point must be kept. The algorithm recursively calls itself with the first point and the new point and then with the new point and the last point. When the recursion is completed a new polygon can be generated consisting of all (and only) those points that have been marked as kept.

After that it is used the Sklansky [21] Algorithm to minimize the geometrical model, excluding all the redundant information for CLCS calculation. The sklansky algorithm consists in an algorithm to find the convex hull of the polygon. The algorithm consists in five steps: The first step is find the convex vertex of the polygon. The second step is to label the remaining n-1 vertices in a clockwise order, starting at P0. The third one, is to select the P0, P1 and P2 vertices and call then "Back", "Center" and "Front" respectively. The fourth step is represented in the follow algorithm:

If "Back", "Center" and "Front" forms a right turn or if they are collinear vertex, change "Back" to the vertice ahead of "Front". Relabel "Back" to "Front", "Front" to "Center" and "Center" to "Back". If they form a left turn, change "Center" to the vertex behind "Back", Remove the vertex and associated edges that "Center" was on and relabel "Center" to "Back" and "Back" to "Center". Do this until "Front" is on vertix P0 and "Back", "Center" and "Front" form a right turn. The generated model defines a position in such a way that even if there is a similar hand shape position in the gesture they could be differentiated by the classifier. Figure 2 illustrates the results of each algorithm used so far.

Now the distance calculation is applied to extract the main features of the hand. The CLCS consists in an adaptation of the regular LCS, designed by Gupta [8]. The LCS calculates the distance between each previous selected point of the hand contour with a line made by the points $(x - w)$ and $(x + w)$, where w is a window previously defined and returns a vector of distances that represents each hand position in the gesture.

The CLCS does not use the standard concept of window to calculate the value of the points. All the selected points are used sequentially to perform LCS distance calculation, generating an output vector.

Each image produces a not known number of distance calculations as features. In order to normalize the number of features, two steps have to be followed. First, the number of distances that all the images will be normalized for is defined. Thus, for all the images that have fewer points than the previously determined length, it is added a '0' at the

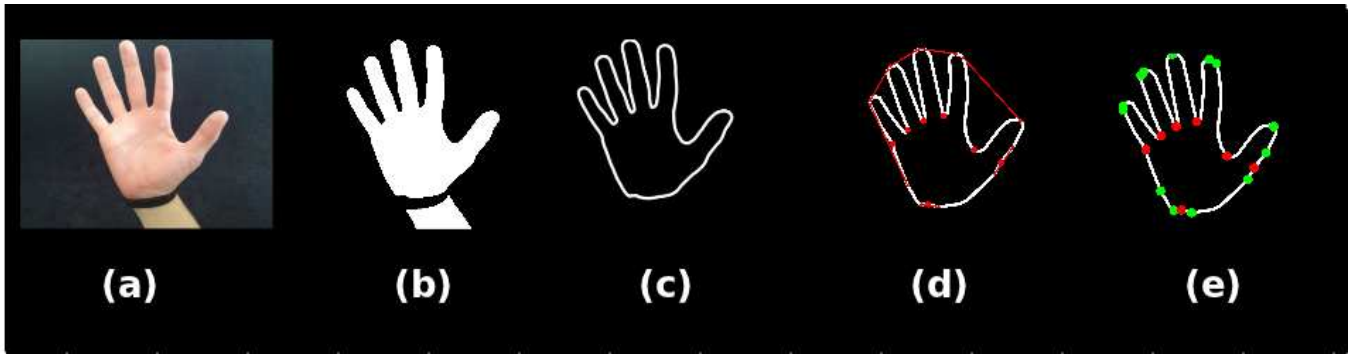

Figure 2: Feature extraction algorithm sequence, starting at (a) as one of the images in the gesture image sequence. (b) Shows result image after the application of the first step, Background Removal using Otsu Threshold. (c) Shows the result image after the application of the second step, Find Contour using the Canny Edge Detector. (d) Shows the result after the application of the third step, Feature Extraction, and it shows the minimized hand contour convex hull. (e) Shows the selected points to calculate the LCS.

end of the vector, until it matches the desired length. This does not lead to any modification in the CLCS calculation, because the main distance distribution is not affected.

The outputs with more points than the desired length are normalized using a selection algorithm. This algorithm consists in calculate a window, W, through the division of the output length for the desired length. The vector of outputs is traversed, and each W position, it value is added to the new vector of outputs.

If the new output vector is smaller than the desired length, the remaining positions are randomly visited and used to compose the new output vector until the desired length is achieved. During the feature extraction there are no restrictions as hand position, distance, and orientation of gesture in front of camera. CLSC is invariant if there is a change in position of a gesture. This is possible by the dynamical points selection. Even if the number of pixels changes in each gesture position, the CLCS is capable of represent the entire gesture.

3. EXPERIMENTAL RESULTS

This paper uses a recognition system to test the CLCS capability to recognize dynamical gestures. These results are compared against the same system results using the LCS outputs and discussed in the conclusion session. A new dynamical gesture database was created and used as CLCS and LCS input to achieve the results.

3.1 Methodological Protocol

To test the CLCS algorithm it was developed a hand gesture database called Pattern Recognition and Digital Image Processing Group of the University of Pernambuco, Brazil (RPPDI) Gesture Database[1]. It contains a set of four different gestures: Click, Grasp, No and Goodbye. For each gesture, several sequences were recorded with a smartphone camera. Figure 3 shows one of the sequences of the Click gesture.

After recorded, each sequence was separated in an image sequence set with 14 samples each one with 640x480 pixels. The hand has a marker in the wrist, to aid the hand shape recognition step during the Feature extraction step.

[1]Available at http://rppdi.ecomp.poli.br/gesture/database/

Figure 3: Example of the Gesture 1, click, contained in the RPPDI database.

Table 1: RPPDI Database Hand Gesture Sequences

Gesture	Amount of Sequences
Gesture 1	24
Gesture 2	24
Gesture 3	31
Gesture 4	18

Table 1 shows the amount of sequences recorded for each gesture. To test the CLCS capability of modelling a dynamic gesture it was compared against the LCS as input for a gesture recognition system. This was made using the RPPDI Gesture Database as input. This system contains two modules: a Feature Extraction Module and a Classification module. Figure 4 shows the system architecture. The system receives as input a set of 14 images. All the images are inserted into the Feature Extraction module, and the result is used as input in the Classification Module. The Feature Extraction Module is implemented with both the CLCS and LCS algorithms, generating two different outputs.

Using the LCS in each image generates a large amount of data to represent a gesture, so, for classification proposes, it is used a limitation of 400 features per image resulting a total of 5600 features in a gesture. The CLCS has a shorter output vector, containing a total of 5 to 15 features extracted

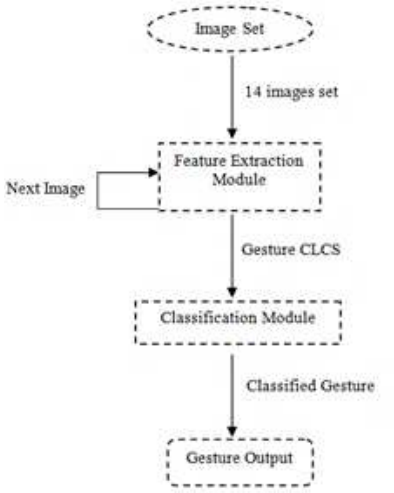

Figure 4: Recognition System Architecture.

Table 2: Classification techniques, HMM and Elman RNN, best initial setup for the tests

Technique		LCS	CLCS
Elman RNN	Neurons in Input Layer	5600	140
	Hidden Layers	75	12
	Neurons in Output Layer	4	1
HMM	States	3	3
	Baum-Welch Iterations	100	100
	Features	5600	140

per image. To use this output in the Classification Module this length was normalized to 10 features. This results in a total of 140 features to represent a gesture.

Two classification techniques, Hidden Markov Model (HMM) [19] and Elman Recurrent Neural Network(Elman RNN)[12], were implemented in the classification module. They use the two outputs generated by the Feature Extraction module as input. To obtain the best classification rate, both the HMM and Elman RNN are trained using a raining routine. This routine is able to identify the best database division ratio, initial setup and training for each feature extraction algorithm.

The routine is executed in three steps: find best database division ratio, find best initial setup parameters and execute the training algorithms. To find the best database division, each technique was trained and tested 30 times in one of these database ratio divisions: 1/3 for training and 2/3 for testing, 1/2 for training and 1/2 for testing, and for last 2/3 for training and 1/3 for testing. The Mean Square Error of each ratio and of each technique was measured and compared.

To find the ideal initial setup, a similar approach was executed: some parameters are selected and tested in the training/testing database. Then these values are modified to more or to less in a fixed ratio, and tested. This was done 30 times and the parameters were selected in the best one. The founded results have been proved to be a good solution to the initial parameters for the System. These results are showed are Table 2.

The training algorithm selected for the Elman RNN was

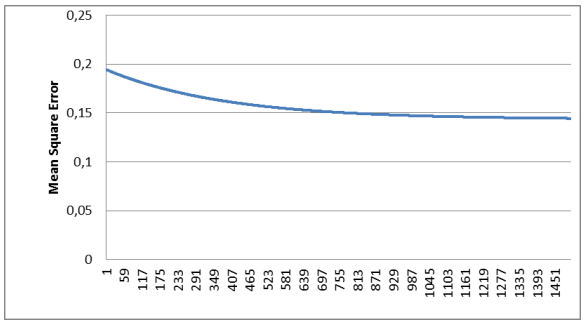

Figure 5: Mean square error for LCS and Elman RNN versus training iteration.

Table 3: Confusion Matrix using LCS as feature extraction technique and Elman RNN as classification technique

Gestures	Gest. 1	Gest 2.	Gest. 3	Gest. 4
Gest. 1	14	0	10	0
Gest. 2	0	24	0	0
Gest. 3	0	6	25	0
Gest. 4	0	2	0	16

the Backpropagation[15]. This algorithm did not converge in a way that the classification rate can be trustful. This problem was caused because the Backpropagation cannot handle the recurrent context of the Elman RNN since it does not take in consideration the context layers.

To solve this problem a hybrid training technique was used, mixing the Backpropagation with the Simulated Annealing [23] method. It increased the classification rate, because the capability of the Simulated Annealing to avoid local minima. This is done by verifying each training epoch and, if the mean square error is not decreasing, the Simulated Annealing is called to find a better solution. If it is founded, the Backpropagation is called again to continue the training. The sigmoid function is used in all the neurons of the network, to calculate the neuron output.

The Hidden Markov Model technique uses a K-Means Clustering [9] to find the best initial approximation; this improved the final recognition rate. The Baum-Welch algorithm [13] is used to train the HMM resulting in a fast training process.

3.2 Results

To choose the best database ratio division for test and algorithm training we executed the algorithm in three different database ratio: 1/2 for training and 1/2 for test; 1/3 for training and 2/3 for test; and 2/3 for training and 1/3 for test. The one the showed the lowest mean square error, being executed 30 times, was the 2/3 for training and 1/3 for test.

After selecting the division ratio, the next step is to train the Elman RN with the data of the LCS algorithm. Figure 5 shows the training iterations versus the mean square error. The mean square error decrease very slowly and took almost 1450 iterations to achieve the value of 0.14. Using

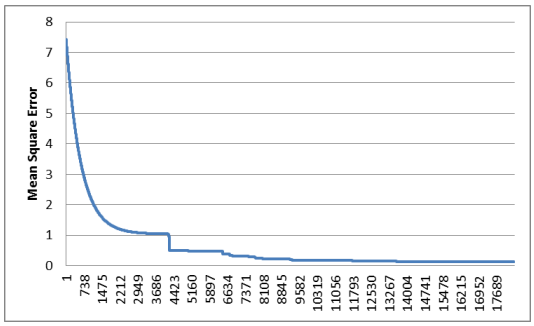

Figure 6: Mean square error for CLCS and Elman RNN versus training iteration.

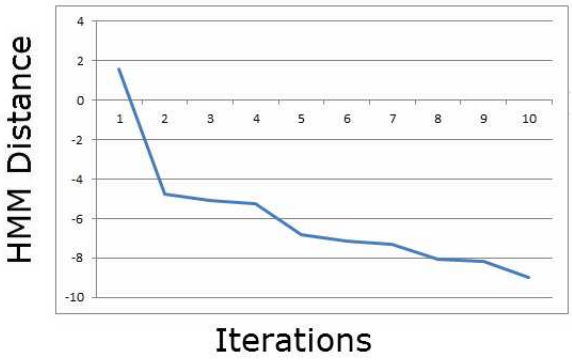

Figure 7: HMM distances for CLCS, applying logarithm, versus training iteration.

Table 4: Confusion Matrix for CLCS as feature extraction technique and Elman RNN as classification technique

Gestures	Gest. 1	Gest 2.	Gest. 3	Gest. 4
Gest. 1	21	0	3	0
Gest. 2	0	24	0	0
Gest. 3	0	0	31	0
Gest. 4	0	0	0	18

the standard LCS as input for the Elman RN, 77.13% of the sequences where successfully recognized. The confusion matrix is shown in Table 3.

Figure 6 shows the mean square error convergence using the hybrid training algorithm through the training epochs. Using the CLCS as input for the Elman RN, 96.91% of the sequences where successfully recognized. The confusion matrix is shown in Table 4. Using HMM the standard LCS was unable to classify due to large amount of features to be classified. The 5600 features were too excessive to calculate the probabilities, and a NAN is generated by the algorithm.

Using the Convexity LCS, the HMM reached a better result, as show in the confusion matrix in Table 5. The recognition rate was 95,78%. One advantage of the HMM was that in a few iterations, the distance between it, calculated using the Kullback-Leibler Distance [4], was below e-15 very early. This means that the HMM can infer the gesture even without ending all the states, predicting which gesture is being classified without reaches the end state. Figure 7 shows the probabilities distances, applying logarithm, for the gesture 3, and illustrate this behaviour. For each new calculation, the distance between the past values is smaller. Table 6 shows the final comparison results between the two feature extraction techniques and the two classification techniques.The recognition using HMM with LCS was unable to classify, as explained in the previous session, so was not represented.

Table 5: Confusion Matrix using CLCS as feature extraction technique technique and HMM as classification technique

Gestures	Gest. 1	Gest 2.	Gest. 3	Gest. 4
Gest. 1	20	0	4	0
Gest. 2	0	24	0	0
Gest. 3	0	0	31	0
Gest. 4	0	0	0	18

Table 6: Classification Rate for Gesture Recognition System in the RPPDI Database

Techniques	Recognition Rate
HMM + CLCS	95.87%
Elman RNN + LCS	77.31%
Elman RNN + CLCS	96.91%

4. CONCLUSIONS

The Convexity LCS calculates fewer points than the regular LCS through the selection of a preferential points mapping. This results in a smaller input vector compared to the latter for the classification techniques which reflects in a faster classification convergence. In the HMM case, the standard LCS could not even be used, because of the amount of input data causing a leak of precision in the algorithm. In the CLCS the recognition rate is on par with the Elman RNN.

The RPPDI Database was created as a solution for dynamic gesture database because of the need of a good quality set of examples, which also needed to be highly extensible. As the classification rates are on par for both of the classification technique using the CLCS algorithm is concluded that both of the classification technique them can be used to classify its output reaching a good recognition rate.

The Future Works can be listed as: Improve of the extraction algorithm to work with dynamic background removal, improve the recognition rate with other training or classification techniques and improve the database with more gestures.

Acknowledgments

This work was partially supported by Brazilian agencies: CNPq, Facepe and CAPES.

5. REFERENCES

[1] Y. bte Abdul Gaus, F. Tze, and K. Kin. Feature extraction from 2d gesture trajectory in malaysian

sign language recognition. In *Mechatronics (ICOM), 2011 4th International Conference On*, pages 1 –6, may 2011.

[2] J. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, June 1986.

[3] J. Eisenstein, S. Ghandeharizadeh, L. Golubchik, C. Shahabi, D. Yan, and R. Zimmermann. Device independence and extensibility in gesture recognition. In *Proceedings of the IEEE Virtual Reality 2003*, VR '03, pages 207–, Washington, DC, USA, 2003. IEEE Computer Society.

[4] M. Falkhausen, H. Reininger, and D. Wolf. Calculation of distance measures between hidden markov models. In *In Proc. Eurospeech*, pages 1487–1490, 1995.

[5] G. Fang, W. Gao, and D. Zhao. Large vocabulary sign language recognition based on fuzzy decision trees. *Trans. Sys. Man Cyber. Part A*, 34(3):305–314, May 2004.

[6] F. Flórez, J. M. García, and J. García. Hand gesture recognition following the dynamics of a topology-preserving network. In *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, FGR '02, pages 318–, Washington, DC, USA, 2002. IEEE Computer Society.

[7] R. Gonzales. *Processamento Digital de Imagens*. Pearson Education, 3rd edition, 2010.

[8] L. Gupta and S. Ma. Gesture-based interaction and communication: automated classification of hand gesture contours. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 31(1):114 –120, feb 2001.

[9] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. In *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, volume 28, pages 100 –108, 1979.

[10] P. S. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms, 1997.

[11] I. Infantino, R. Rizzo, and S. Gaglio. A framework for sign language sentence recognition by commonsense context. *Trans. Sys. Man Cyber Part C*, 37(5):1034–1039, Sept. 2007.

[12] L. Jain. *Recurrent Neural Networks*. CRC Press, 1st edition, 2001.

[13] T. L. Baum, G. Peterie, and N. W. Souled. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. In *Proceedings of Annals of Mathematical Statistics*, volume 41, pages 164 –171, 1995.

[14] K. Li, Y. Du, and Z. Fu. Treeheaven: a table game using vision-based gesture recognition. In *Proceedings of the 2011 ACM symposium on The role of design in UbiComp research & practice*, RDURP '11, pages 11–14, New York, NY, USA, 2011. ACM.

[15] M. Maraqa and R. Abu-Zaiter. Recognition of arabic sign language (arsl) using recurrent neural networks. In *Applications of Digital Information and Web Technologies, 2008. ICADIWT 2008. First International Conference on the*, pages 478 –481, aug. 2008.

[16] S. Meena. A Study on Hand Gesture Recognition Technique. Master's thesis, National Institute Of Technology, Rourkela,India, 2011.

[17] K. Murakami and H. Taguchi. Gesture recognition using recurrent neural networks. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, CHI '91, pages 237–242, New York, NY, USA, 1991. ACM.

[18] N. Otsu. A threshold selection method from gray-level histograms. *Systems, Man and Cybernetics, IEEE Transactions on*, 9(1):62 –66, jan. 1979.

[19] L. R. Rabiner. Readings in speech recognition. chapter A tutorial on hidden Markov models and selected applications in speech recognition, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[20] T. Schlömer, B. Poppinga, N. Henze, and S. Boll. Gesture recognition with a wii controller. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, TEI '08, pages 11–14, New York, NY, USA, 2008. ACM.

[21] J. Sklansky. Finding the convex hull of a simple polygon. *Pattern Recogn. Lett.*, 1(2):79–83, Dec. 1982.

[22] Y. L. Y. Yuan and K. Barner. Tactile gesture recognition for people with disabilities. 5:461–464, Mar. 2005.

[23] H. Zhang, Y. Wang, and C. Deng. Application of gesture recognition based on simulated annealing bp neural network. In *Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on*, volume 1, pages 178 –181, aug. 2011.