Gesture Interface: Modeling and Learning

Jie Yang *†, Yangsheng Xu *

C.S. Chen †

*The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

†Department of Electrical Engineering
The University of Akron
Akron, Ohio 44325

Abstract

This paper presents a method for developing a gesture-based system using a multi-dimensional hidden Markov model (HMM). Instead of using geometric features, gestures are converted into sequential symbols. HMMs are employed to represent the gestures and their parameters are learned from the training data. Based on "the most likely performance" criterion, the gestures can be recognized by evaluating the trained HMMs. We have developed a prototype to demonstrate the feasibility of the proposed method. The system achieved 99.78% accuracy for a 9 gesture isolated recognition task. Encouraging results were also obtained from experiments of continuous gesture recognition. The proposed method is applicable to any multi-dimensional signal representation gesture, and will be a valuable tool in telerobotics and human computer interfacing.

1 Introduction

A gesture interface is an interface where users specify commands by simple gestures such as drawings and actions. For example, with a gesture interface, a robot may be programmed by showing of teleoperators, and a teleoperated robot not only knows how to follow human commands but also knows the meaning of the commands [1]. To develop such an interface, the key issues are how to sense gesture information and how to recognize the gestures from sensed data. To develop a gesture interface, we need some criteria to evaluate its performance, such as: meaningful gestures; suitable sensors; efficient training algorithms; and accurate, efficient, on-line/real-time recognition.

In general, the technology for capturing gestures is expensive; e.g., a vision system or a data-glove is needed. For this reason some graphical devices, such as a mouse, light pen, joystick, trackball, touch tablet, and thumb-wheel, can be employed to provide a simple input to a gesture recognizer. Other possible devices are a foot controller, knee controller, eye tracker, data nose, and tongue-activated joystick.

The gesture recognition problem consists of pattern representation and decision making. Gestures are usually represented by various features, including templates, global transformations, zones, and geometric features. Templates are the simplest features to compute; they are simply the input data in its raw form. Global transformations, such as rotation, translation, or scaling can be employed to reduce the number of features in templates. Zoning is a simple way of deriving features from a path. Space is divided into a number of zones, and the path is transformed into the sequence of zones which the path traverses. Geometric features of a path, such as its total length, total angle, number of times it crosses itself, etc., can be used to represent the global properties of the path.

Several methods have been used for gesture recognition: template-matching [2], dictionary lookup [3], statistical matching [4], linguistic matching [5], neural network [6], and ad hoc methods. Some of the methods are suitable for only one type of feature representation, while others are more generally applicable. Template-matching systems are easy to train because the prototypes in the systems are simply example templates. However, a large number of prototypes can make the use of the template matching prohibitively expensive. When the features are a sequence of tokens from a small alphabet, lookup techniques are efficient for recognition. The drawback of dictionary lookup is that the system is not robust. Statistical matching methods employ statistics of example feature vectors to derive classifiers. The typical statistics used are average feature vector per class, per-class variance of the individual features, and per-class correlations within features. Some statistical matching methods make assumptions about the distributions of features within a class; the performance of such systems tends to be poor when the assumptions are violated. Other statistical matching methods do not have such assumptions, but they require much training data to estimate the form of the distribution. The linguistic approach tries to apply automata and formal language theory to the pattern recognition problem. The major problem with this approach is the need of supplying a grammar for each pattern class. Neural networks have been successfully applied to solve many pattern recognition problems. Their major advantage is that they are built from a large number of simple elements which learn and are able to collectively solve complicated and ambiguous problems. Unfortunately, they tend to require a large amount of processing power, especially for training.

Several gesture interfaces have been developed. Coleman [7] built a hand-drawn symbol-based text editor with a touch tablet as the input device. Buxton [8] created a musical score editor with a small amount of gesture input by a mouse. Margaret Minsky [9] implemented a system which uses gestures for selection, movement, and path specification to offer a complete Logo programming environment. Rubine [4] produced a system based on statistical pattern recognition techniques for recognizing signal-path gestures (draw with a mouse or stylus) and multiple-path gestures (consisting of the simultaneous path of multiple fingers). The Glove-talk system [6] employs a DataGlove to control a speech synthesizer. Teaching-by-showing, teleoperation [10], assembly-plan-from-observation [11] are techniques which use gestures for programming a robot.

HMM is a doubly stochastic model and is appropriate for coping with the stochastic properties in gesture recognition. HMM has been successfully applied to speech recognition

1050-4729/94 $03.00 © 1994 IEEE

[12, 13]. Recently it has been studied in force analysis and task context final segmentation [15, 16]. We have previously proposed to use HMM for modeling and learning human skill [17]. This paper presents a method for developing a gesture interface using the multi-dimensional hidden Markov model (HMM). Instead of using geometric features, gestures are converted into sequential symbols. HMMs are employed to represent the gestures, and their parameters are learned from the training data. Based on the most likely performance criterion, the gestures can be recognized by evaluating the trained HMMs. We have developed a system to demonstrate the proposed method. We defined 9 gestures and used a mouse as the gesture input device. We then employed HMM to learn and recognize these gestures. The feasibility of the method was demonstrated by the experiments in both isolated and continuous gesture recognition. The proposed method has potential applications in telerobotics and a variety of human machine interfacing problems.

2 HMM-based Gesture Recognition

Gesture is an expressive motion. It is natural to described such a motion by a sequential model. Because human performance is inherently stochastic, if one repeats a certain gesture many times or if many people try the same gesture, the measured gestures will be different. The identical case can be found in speech signals of the same word recorded at different times or from different speakers. The recorded signals are different, but there is obviously an inherent property hidden in the recordings that represents a certain word or a certain gesture. Therefore, the goal of gesture recognition is to uncover the hidden patterns associated with specific gestures from measured data. We propose the *most likely performance* as a measure of the gesture, that is, we classify the gestures based on the likelihood criterion. The likelihood criterion is appropriate in the sense of maximizing the expected average performance. The likelihood concept also makes it possible to employ stochastic methods to cope with uncertainties in both human performance and sensing processes. Based on these considerations, HMM is a feasible model for gesture recognition.

2.1 Procedure

The HMM-based gesture recognition approach can be described as follows:

1. *Defining meaningful gestures* – To communicate with gestures, we must specify the meaningful gestures first. For instance, a gesture in sign language corresponds to certain vocabulary. Gesture is generally defined as "a motion of the limbs or body made to express or help express thought". To recognize such a generalized gesture is well beyond the state of the art in current computer science. Therefore, the term "gesture" usually has a restricted connotation when used in the context of human-computer interaction. In this paper, we consider a gesture to be the paths of multiple points over time generated by a human operator. Currently the device to measure gesture limits both the number of points which may be tracked simultaneously and the space in which the points travel. Gestures limited to the motion of a single point are referred to as single-path gestures [4]. A single-path gesture can be input with a single pointer, such as a mouse, or a light pen. Some devices, such as a data-glove, are capable of tracking multi-path gestures.
2. *Describing each gesture in terms of an HMM* – We employ a multi-dimensional hidden Markov model (MHMM), which contains more than one observable symbol at each time t, to model the gesture. First, an MHMM is able to deal with multi-path gestures which are general cases in gesture recognition. Second, a single path gesture can usually be decomposed into 2D or 3D time series in Cartesian space. That is, a single path gesture $g(x, y, z, t)$ can be decomposed into $X(t)$, $Y(t)$, and $Z(t)$. Moreover, an MHMM provides the possibility of using multiple features to increase recognition rate. In the proposed approach, a gesture is described by a set of N distinct hidden states and r dimensional M distinct observable symbols. An MHMM is characterized by a transition matrix A and r discrete output distribution matrix B_i, $i = 1, \ldots, r$.
3. *Collecting training data* – Meaningful gestures may be very complex, containing simultaneous motions of a number of points. However, these complex gestures should be easily specifiable. In general, gestures can be specified either by example or by description. In the former, each application has a training session in which examples of different gestures are collected for training the models. The trained models are the representations of all gestures that the system must recognize. In the latter method of specification, a description of each gesture is written in a gesture description language which is a formal language in which the syntax of each gesture is specified. Obviously, the example method has more flexibility than the description method. In our approach, gestures are specified by examples. Raw input data are preprocessed before they are used to train the HMMs. One potential drawback of specification by example is the difficulty in specifying the allowable variation between gestures of a given class. This would not be a problem in our approach because the model parameters are determined based on the most likely performance criterion.
4. *Training the MHMMs by training data* – Training is one of the most important procedures in a HMM-based approach. The model parameters are adjusted in such a way that they can maximize the likelihood $P(O|\lambda)$ for the given training data. No analytic solution to the problem has been found so far. However, the Baum-Welch algorithm can be used to iteratively reestimate model parameters to achieve the local maximum.
5. *Evaluating gestures by the trained model* – The trained model can be used to classify the incoming gestures. The Forward-Backward algorithm or the Viterbi algorithm can be used to classify isolated gestures. Viterbi algorithm can also be used to decode continuous gestures.

To understand the above procedure, consider an HMM-based isolated gesture recognition system. Suppose we have G meaningful gestures to be recognized. We need to build a discrete HMM for modeling each gesture. Assume that the feature vectors for each gesture can be obtained by certain signal processing techniques and a set of training data is available for each gesture. Using the vector quantization technique, we can then convert the feature vectors into finite symbols. The training and recognition processes can be summarized as follows:

1. In the training process, estimate the model parameters (A, B, π) for HMM λ^g such that the λ^g is the best representation of the training data for the gth gesture.
2. In the recognition process, convert each unknown gesture into observation symbols by feature analysis and vector quantization; follow by computing the model probabilities

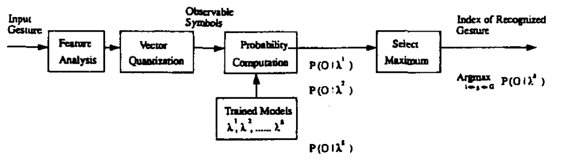

Figure 1: Illustration of the isolated gesture recognition

for all possible models, $P(O|\lambda^g)$, $1 \geq g \geq G$; follow by selecting the gesture whose probability is the highest, that is,

$$g^* = \arg \max_{1 \geq g \geq G} [P(O|\lambda^g)].$$

The procedure is illustrated in Figure 1.

2.2 Recognition

One of the advantages of the HMM-based approach is that a variety of knowledge sources can be combined into a single HMM. By representing all possible knowledge sources as HMMs, the recognition task becomes a search in an enormous HMM. In the case that no knowledge is added, we could create a recognition model by simply putting all gesture models in parallel, and adding an initial state and a final state. The initial state of recognition model has a null transition to the initial state of each gesture model; the final state of each gesture model has a null transition to the final state of the recognition model. A null transition is a transition which has a transition probability but does not emit any output symbol, and therefore does not consume any time.

2.3 Isolated vs. Continuous Gesture Recognition

The advantage of hidden Markov modeling is that it can automatically absorb a range of model boundary information for continuous gesture recognition. Other methods such as neural networks face the problems in training models for continuous gesture recognition because the gesture boundaries are not automatically detectable. Therefore tedious hand-marking is usually required.

Training HMMs for continuous gesture is similar to that for isolated gestures. To train the parameters of the HMM, the gestures are concatenated and each gesture is instantiated with its model. This large concatenated HMM can then be trained by the corresponding data. Because the entire gesture HMM is trained on the entire observation sequence for the corresponding gestures, all possible gesture boundaries are inherently considered. In other words, the parameters of each model will be reestimated based on those states which are suitable for gesture alignments regardless of the location of the gesture boundaries. Such a training method provides complete freedom to align the concatenated model against the observation, and no effort is required to find gesture boundaries.

The recognition of a continuous gesture is much more difficult than that of an isolated gesture. Because the gesture boundaries can not be accurately detected, all possible beginning and end points must be considered. This results in a tree search task. For a large vocabulary continuous gesture recognition task, an optimal full search is infeasible. Several sub-optimal search algorithms can be used instead. The Viterbi algorithm [13] is an efficient sub-optimal search algorithm that can be used for continuous gesture recognition.

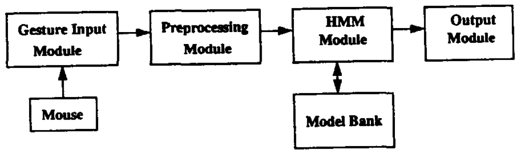

Figure 2: Block diagram of the gesture based system

3 Implementational Issues

In order to demonstrate the proposed method, we have developed a gesture-based system which is illustrated by a block diagram in Figure 2. We used a mouse as the gesture input device. The input gestures were resampled and then converted into symbols. We modeled each gesture by an HMM. The parameters of those HMMs were learned from the training data. Some important implementational issues are addressed in this section.

3.1 Input Data

A mouse is a two-dimensional single-path gesture input device. Each gesture can be represented as an array g of P time-sampled sample points:

$$g_p = (x_p, y_p, t_p) \quad 0 \leq p < P$$

We need the sampling time information t_p because our interface, X-window, does not deliver input points at regular intervals. The two-dimensional single-path gesture g_p can be projected onto x and y axes. We then obtain two independent one-dimensional signals, $x(t_p)$ and $y(t_p)$. We can re-sample $x(t_p)$ and $y(t_p)$ by linear interpolation to obtain input points with the same sampling interval. Interpolation is the same operation as "table lookup". Described in "table lookup" terms, the "table" is $[T, X]$. Linear interpolation "looks-up" the elements of t_i in T, and, based upon their location, returns values x_i interpolated within the elements of X.

3.2 Feature Extraction

To determine the extent of preprocessing a gesture, we consider factors such as the existence of a preprocessing algorithm, its necessity, its complexity, and its generality. We select the Fast Fourier Transformation (FFT) as preprocessing tool based on the following reasons. The Fourier transformation and its inverse establish a one-to-one mapping between the time domain and the frequency domain, and FFT algorithms can be implemented efficiently. Also, the Fourier transformation preserves information from the original signal, and ensures that important features are not lost as a result of FFT. Furthermore, shifting a waveform within the window changes the real and imaginary parts of the frequency domain in such a manner that the square root of the sum of the squares (the magnitude) remains constant. In fact, if we have a function given by $x(t)$ that Fourier transforms to $X(f)$, when we shift x(t) by a constant time, T, i.e., $x(t-T)$, its Fourier transformation is

$$X(f)e^{-j2\pi fT}$$

i.e., time shifting affects phase only; the magnitude remains constant throughout.

Although the Fourier transform does not explicitly show the time localization of frequency components, the time localization can be presented by suitably pre-windowing the signal in time domain. Accordingly, short time Fourier transform (STFT) of a signal $x(t)$ is defined as [20]

$$STFT_x^{\gamma}(t,f) = \int_{t'} [x(t')\gamma^*(t'-t)]e^{-j2\pi ft'}dt' \qquad (1)$$

STFT at time t is the Fourier transform of the signal $x(t')$ multiplied by a shifted analysis window $\gamma^*(t'-t)$ centered around t. (All integrals are from $-\infty$ to ∞. The superscript * denotes complex conjugation.) Because multiplication by the relatively short window $\gamma^*(t'-t)$ effectively suppresses the signal outside a neighborhood around the analysis time point $t = t'$, the STFT is simply a local spectrum of the signal $x(t')$ around analysis window t. The windows can be overlapped to prevent loss of information. Although human behavior is a nonstationary stochastic process over a long interval, it can be considered stationary over a short time interval. Thus, STFT should give a good spectral representation of the gesture during that time interval. We employ STFT to extract gesture features in our system. The Hamming window is first used with a width of 16 points in every 8 points. FFT analysis is then performed for every window. Finally, a set of 16-dimensional vectors is obtained from the amplitude of FFT coefficients.

Because we use a discrete HMM, we need to convert the feature vectors into finite symbols. Vector quantization (VQ) technique [21] is a suitable tool to map a real vector onto a discrete symbol. A vector quantizer is completely decided by a codebook, which consists of a set of the fixed prototype vectors. A description of the VQ process includes: (1) the distortion measure, and (2) the generation of the certain number of prototype vectors. In the implementation, the squared error distortion measure is used, i.e.,

$$d(\mathrm{x}, \hat{\mathrm{x}}) = ||\mathrm{x}, \hat{\mathrm{x}}|| = \sum_{i=0}^{k-1}(x_i - \hat{x}_i)^2 \qquad (2)$$

The codebook is generated by the VQ algorithm. One of the commonly used algorithms is the LBG algorithm proposed by Linde, Buzo, and Gray [22]. The LBG algorithm iteratively splits the training data into 2, 4, ..., 2^m partitions with a centroid for each partition.

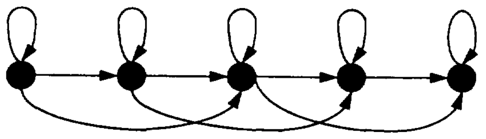
Figure 3: 5-state left-right HMM

3.3 HMM Topology

Because we consider the gesture paths as the observable symbols, the underlying state sequence associated with the model has the property that, as time increases, the state index increases (or stays the same), i.e., the states proceed from left to right. We can use an n state left-right HMM or a Bakis model to describe the task as shown in Figure 3.

The transition matrix in this case is

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & \cdots & 0 \\ 0 & a_{22} & a_{23} & a_{24} & 0 & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & a_{n-2,n-2} & a_{n-2,n-1} & a_{n-2,n} \\ \vdots & \ddots & \ddots & 0 & a_{n-1,n-1} & a_{n-1,n} \\ 0 & \cdots & \cdots & \cdots & 0 & a_{nn} \end{bmatrix}. \qquad (3)$$

Clearly this model has fewer parameters than that of the ergodic or full connected HMMs. Furthermore, the initial state probabilities satisfy:

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1. \end{cases} \qquad (4)$$

The state transition coefficients of state n are specified as

$$a_{nn} = 1,$$
$$a_{n,i} = 0, \quad i < n. \qquad (5)$$

In order to train an HMM, we need to collect a certain number of training data associated with each gesture and convert the gesture-paths into finite symbols. If we convert the continuous paths into p symbols by the certain signal processing techniques, B_i is a $n \times p$ matrix. The HMM is trained by preprocessed data to select the optimal parameters in A and B_i to best present the all training data. The resultant model represents the most likely human performance for the gesture. Based on these models, we then can evaluate the new incoming gesture and recognize the gesture according to the probability that the gesture matches with the models.

4 Experimental Results

We ran the system successfully for some case studies. We defined 9 gestures as shown in Figure 4), and employed a six

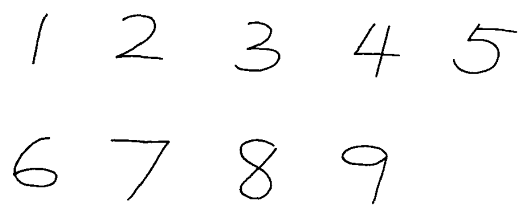

Figure 4: The defined gestures

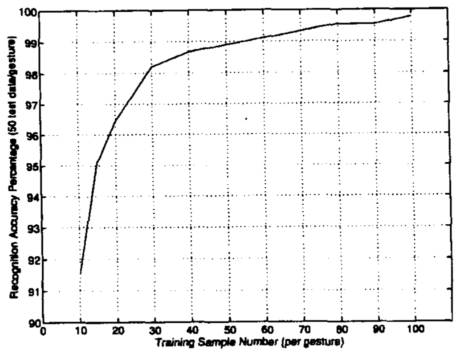

Figure 5: Recognition rate vs. training set size

state, two-dimensional Bakis model to model each gesture. For $n = 6$ we obtained the form of transition matrix A, the initial state probabilities, and the state transition coefficients of state 6 from equations (21) to (23). We employed two 256×6 observability matrices with each column representing the observation probability distribution for one state. The learning algorithms can be obtained from Equations (14)-(18) with $R = 2$.

The experiments were run on a SUN4 machine. The gestures were generated by a mouse at about 40 millisecond sampling time intervals and then resampled at 20 millisecond sampling time intervals. The FFT and VQ techniques were used for preprocessing the gestures. The Hamming window was first used with a width of 320 ms in every 160 ms. FFT analysis was then performed for every window. Finally, a set of 16-dimensional vectors was obtained from the amplitude of FFT coefficients. The LBG algorithm was used to produce VQ codebooks. First a certain number of the training vectors was used to generate a 256-vector codebook for each dimensional signal. These sets of 256 vectors were the symbols in the output probability distribution function in our discrete HMM. An input vector corresponded to the set of 16 32-bit floating point FFT coefficient amplitudes, and was mapped onto an 8-bit index which represents one of 256 prototype vectors.

To initialize the model parameters, we let output probabilities equal to $\frac{1}{256}$, where 256 is the VQ level. The transition probabilities were initialized by the uniformly distributed random number. With these initial parameters, the Forward-Backward algorithm was run recursively on the training data. The Baum-Welch algorithm was used iteratively to reestimate the parameters based on the forward and backward variables. After each iteration, the output probability distributions were smoothed using a floor between 0.0001 and 0.00001, and renormalized to meet stochastic constraints.

We first studied isolated gesture recognition. We collected 150 samples of data for each gesture, the first 100 for training and the rest of the samples for testing.For a total of 450 testing samples, the system successfully recognized 449 samples, a 99.78% accuracy rate. The effect of the training set size was studied by varying the number of training samples per gesture. Figure 5 shows the result of the recognition rate versus training set size. The testing set size was fixed at 50 per gesture (total 450) while the training set size changed from 10 to 100 samples. As expected, the recognition rate increased as the training set size increased. When the training set size was 10 samples per gesture, the recognition rate was 91.56%, and when the training set size increased to 100 samples per gesture, the rate was 99.78%.

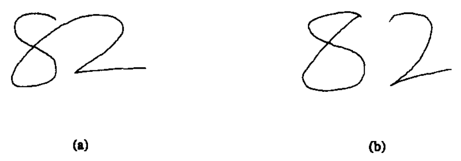

Figure 6: An example of continuous gesture: (a) connected and (b) separated

We then investigated the performance of HMM-based system for continuous gesture recognition. In this case we concentrated on the system performance to separate connected gestures. Continuous gestures may either connect or separate each other, as shown in Figure 6. We trained gesture "2" and "8" models by continuous gesture "82" for both connected and separated cases. Then we tested the trained models by inputing gesture "2", "8", connected "82", and separated "82". In all cases, the system demonstrated its capability to recognize the gesture correctly.

From the experimental results, some remarks are in order:

1. We have demonstrated that HMM is a feasible parametric model for building a gesture-based system. Although we have examined only a two-dimensional single-path gesture case in this paper, the method can be extended to multiple-path applications. For example, Data-Glove is a typical multiple-path gesture input device that allows us to encode the gestures in the joint space and represent the gestures by multi-dimensional HMM. The same method is applicable to develop a variety of gesture recognition systems that can be useful in telerobotics and human machine interfacing.

2. HMM is a doubly stochastic model and is appropriate for coping with the stochastic properties in gesture recognition. HMM can represent all the training data in the statistic sense by its parameters and its parameters can be

optimized by efficient algorithms for the accurate estimation. Therefore, the model can be updated incrementally, which is desirable for *learning*. Comparing with neural network approach, HMM converges faster because of its efficient reestimation algorithms, and is more suitable for continuous gesture recognition because no hand-marking is needed.

3. The gesture recognition of a two-dimensional single-path has much in common with on-line handwriting recognition. However, our method can potentially deal with the gestures which dimensions are other than two, are drawn from unusual symbols, specify entire commands, vary in size and orientation, and have a dynamic components.
4. Handwriting recognition systems can be broadly grouped into two classes: on-line and off-line. In on-line handwriting recognition, characters are recognized as they are drawn. In off-line handwriting recognition, characters are first drawn on paper, and then optically scanned and represented as two-dimensional rasters. Although gesture recognition is different from handwriting recognition in general, it is possible to apply the proposed method to handwriting recognition after modification. For example, the method is able to do on-line handwriting recognition by taking the FFT base on arc length instead of time t. The same idea can be applied to off-line handwriting recognition after some preprocessing.

5 Conclusion

We have proposed a method for modeling, recognizing, and learning human gestures using the hidden Markov model. HMM is a powerful parametric model and is feasible to characterize two stochastic processes – the measurable action process and immeasurable, hidden mental states. Instead of using geometric features, we convert the gestures into sequential symbols. HMMs are employed to represent the gestures, and their parameters are learned from the training data. Based on the most likely performance criterion, the gestures can be recognized by evaluating the trained HMMs.

We developed a prototype to demonstrate the feasibility of the HMM-based gesture recognition method. We defined 9 gestures and used a mouse as the gesture input device. We then applied HMM to learn and to recognize measured gestures. The experimental results showed that the proposed method can be used for learning and recognizing gestures in both isolated cases and continuous cases. The proposed method is applicable to multi-dimensional signal recognition and learning problems, and will be of significance in developing gesture interface in telerobotics and human-computer interfacing.

References

[1] T.H. Speeter, " Transformation human hand motion for telemanipulation," *Presence*, Vol. 1, No.1, pp.63-79, 1992.

[2] J. S. Lipscomb, "A trainable gesture recognizer," *Pattern Recognition*, vol. 24, No. 9, pp895-907, 1991.

[3] W. M. Newman and R. F. Sproull, *Principles of Interactive Computer Graphics*, McGraw-Hill, 1979.

[4] D. H. Rubine, "The automatic recognition of gesture," *Ph.D dissertation*, Computer Science Department, Carnegie Mellon University, December, 1991.

[5] K. S. Fu, "Syntactic recognition in character recognition," *Volume 112 of Mathematics in Science and Engineering*, Academic Press, 1974.

[6] S. S. Fels and G. E. Hinton, "Glove-talk: a neural network interface between a data-glove and a speech synthesizer," *IEEE Trans. Neural Networks*, Vol. 4, No. 1, pp.2-8, 1993.

[7] Michael L. Coleman, "Text editing on a graphic display device using hand-drawn proofreader's symbols," *Proceedings of the Second University of Illinois Conference on Computer Graphics*, University of Illinois Press, pp. 283-290, 1969.

[8] W.A.S. Buxton, R. Sniderman, W. Reeves, S. Patel, and R. Baecker, "The evolution of the SSSP score-editing tools," *Foundation of Computer Music*, Chapter 24, pp.443-466, MIT press, 1985.

[9] M.R. Minsky, "Manipulating simulated objects with real-world gestures using a force and position screen," *Computer Graphics*, Vol. 18, No. 3, pp. 195-203, 1984.

[10] S. Hirai and T. Sato, "Motion understanding for world model management of telerobot," *Proceedings of IROS'89*, pp. 124-131, 1989.

[11] K. Ikeuchi and T. Seuhiro, "Towards man assembly plan from observation: task recognition with polyhedral objects," CMU Tech. Rep. CMU-CS-91-167, Computer Science Department, Carnegie Mellon University, 1991.

[12] K.F. Lee, H.W. Hon and R. Reddy, "An overview of the SPHINX speech recognition system," *IEEE Trans. on ASSP*, Vol. 38, No. 1, pp. 35-45, 1990.

[13] X.D. Huang, Ariki and M. A. Jack, "Hidden Markov Models for Speech Recognition," *Edinburgh University Press*, 1990.

[14] B. Hannaford, P. Lee, "Hidden Markov model analysis of force/torque information in telemanipulation," *The International Journal of Robotics Research*, Vol. 10, No. 5, pp.528-539, 1991.

[15] P. K. Pook and D. Ballard, "Recognizing teleoperated manipulations," *Proceedings of 1993 IEEE Inter. Conf. on Robotics and Automation*, Atlanta, Georgia, USA, Vol. 2, pp.578-585, 1993.

[16] J. Yang, Y. Xu and C. S. Chen, "Hidden Markov model approach to skill learning and its application in telerobotics," *Proceedings of 1993 IEEE Inter. Conf. on Robotics and Automation*, Atlanta, Georgia, USA, Vol. 1, pp.396-402, 1993.

[17] L.E. Baum, T. Petrie, G. Soules, and N. Weiss, " A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chins," *Ann. Math. Stat.*, Vol. 41, No. 1, pp. 164-171, 1970.

[18] R.W. Schafer and L.R. Rabiner, "Digital representations of speech signals," *Proceedings of IEEE*, Vol. 63, No. 4, pp. 662-677.

[19] F. Hlawatsch and G.F. Boudreaux-bartels, "Linear and quadratic time-frequency signal representations," *IEEE SP Magazine*, Vol.9, No.2, 1992.

[20] R.M. Gray, "Vector quantization," *IEEE ASSP Magazine*, Vol. 1, No.2, pp. 4-29, 1984.

[21] Y. Linde, A. Buzo, and R.M. Gray, " An algorithm for vector quantizer design," *IEEE Trans. on Communication*, Vol. COM-28, pp.84-95, 1980.