Correspondence

A Fuzzy Rule-Based Approach to Spatio-Temporal Hand Gesture Recognition

Mu-Chun Su

***Abstract*—Gesture-based applications widely range from replacing the traditional mouse as a position device to virtual reality and communication with the deaf. In this paper, we present a fuzzy rule-based approach to spatio-temporal hand gesture recognition. This approach employs a powerful method based on hyperrectangular composite neural networks (HRCNNs) for selecting templates. Templates for each hand shape are represented in the form of crisp IF-THEN rules that are extracted from the values of synaptic weights of the corresponding trained HRCNNs. Each crisp IF-THEN rule is then fuzzified by employing a special membership function in order to represent the degree to which a pattern is similar to the corresponding antecedent part. When an unknown gesture is to be classified, each sample of the unknown gesture is tested by each fuzzy rule. The accumulated similarity associated with all samples of the input is computed for each hand gesture in the vocabulary, and the unknown gesture is classified as the gesture yielding the highest accumulative similarity. Based on the method we can implement a small-sized dynamic hand gesture recognition system. Two databases which consisted of 90 spatio-temporal hand gestures are utilized for verifying its performance. An encouraging experimental result confirms the effectiveness of the proposed method.**

***Index Terms*—Gesture recognition, neuro-fuzzy system, spatio-temporal pattern recognition.**

I. Introduction

The wish to provide a more natural means of interacting with computers has led to considerable interest in recognizing hand gestures. A variety of gesture-based applications have been created so far. Minsky built a gestural interface to the LOGO programming language [1]. Buxton's group produced a musical score editor that uses gestures for entering notes [2]. Ko and Yang built a finger mouse that enables a user to specify commands and additional parameters by drawing single intuitive gestures with his or her finger [3]. Recently, many researchers have devoted themselves to developing verbal communication aids for the deaf [4]–[10]. Due to congenital malfunctions, diseases, head injuries, or viral infections, deaf or nonvocal individuals are unable to communicate with hearing persons through speech. Deaf or nonvocal persons use sign language or hand gestures to express themselves, however, most hearing people do not have special sign language expertise. Therefore, conversations between deaf and hearing people are done in writing or with an interpreter. However, conversing through the use of these two methods is troublesome and sometimes causes misunderstandings. This phenomenon motivated the development of speaking aids with which deaf or nonvocal persons can make themselves easily understood by hearing persons. Additional surveys on techniques and applications of gesture recognition are given by Watson [11].

Manuscript received May 27, 1998; revised November 30, 1998 and August 1, 1999. This work was supported by the National Science Council of Taiwan, R.O.C., under Grant NSC-88-2213-E032-014.

The author is with the Department of Computer Science and Information Engineering, National Central University, Chung-Li 320, Taiwan, R.O.C.

Publisher Item Identifier S 1094-6977(00)04799-4.

Hand gestures can be described in terms of the following four major attributes:

1) hand configuration (i.e., posture);
2) palm orientation;
3) hand position;
4) hand movement.

In our opinion, hand gestures can be classified into two categories: 1) static hand gestures which rely only on the information about the flexure angles of the fingers and 2) dynamic hand gestures which rely not only on the fingers' flex angles but also on the hand trajectories and orientations. Dynamic hand gestures can be further divided into two subclasses. The first subclass consists of hand gestures involving hand movements, and the second subclass consists of hand gestures involving only the fingers' movements without changing the position of the hands. That is, it requires at least two different hand shapes connected sequentially to form a particular hand gesture. Fig. 1 gives an example to illustrate the difference between these different types of hand gestures. Therefore samples of these hand gestures are spatio-temporal patterns. Note that samples of static hand gestures can be viewed as a special case of spatio-temporal patterns.

There are several different approaches to recognizing spatio-temporal hand gestures. The simplest way is first to turn the temporal sequence into a spatial pattern and then to employ the template matching technique. The dynamic time warping (DTW) algorithm provides the effect of a nonlinear normalization process in order to make the similarity measure operate successfully [12]. The DTW algorithm operates by stretching the template pattern and measuring the amount of stretching required. The less stretching needed, the more similar the patterns are. Although the pattern matching technique provides good recognition performance for a variety of practical applications, it has a number of deficiencies. For example, the selection of appropriate templates for each class is a difficult task. In addition, another problem associated with the DTW algorithm is that it usually requires substantial computation to reach an optimal DTW path. This is the most vulnerable drawback of the conventional pattern matching technique incorporated with the DTW algorithm. The other approach is to train recurrent neural networks [8]. One may find that it usually takes a lot of time to train a recurrent neural network. Another approach is to employ hidden Markov models (HMMs) to recognize hand gestures. The property of HMMs to compensate for time and amplitude variances has been proven for speech and character recognition [13]. This property makes HMMs appear to be an ideal approach for hand gesture recognition [14]–[17]. The price paid for the efficiency in this case is that we have to collect a great amount of data and a lot of time is required to estimate corresponding parameters in HMMs.

In this paper we propose a fuzzy rule-based approach to spatio-temporal hand gesture recognition. We are then able to implement a hand gesture to speech system for the deaf. This paper is organized into five sections. In Section II, we briefly introduce the architecture of a two-layer HRCNN and the method of extracting rules from the values of the network's parameters. The proposed method of recognizing spatio-temporal hand gestures is presented in Section III. In Section IV, we give the results obtained by applying the method to databases consisting of 90 sign words from the Taiwanese sign language (TSL) signed by four persons. In Section V we give a few concluding remarks.

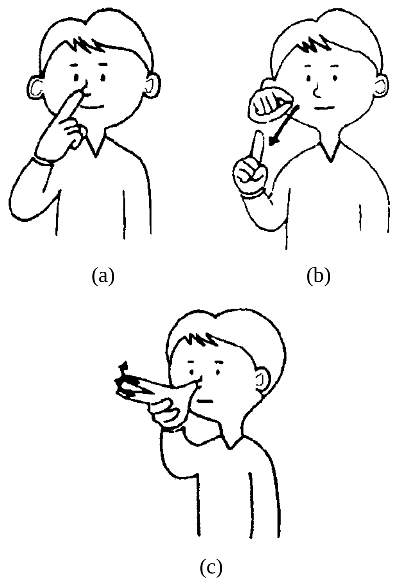


Fig. 1. Examples of sign words in TSL: (a) static sign word "I"; (b) dynamic sign word "one month"; and (c) dynamic sign word "mouse".

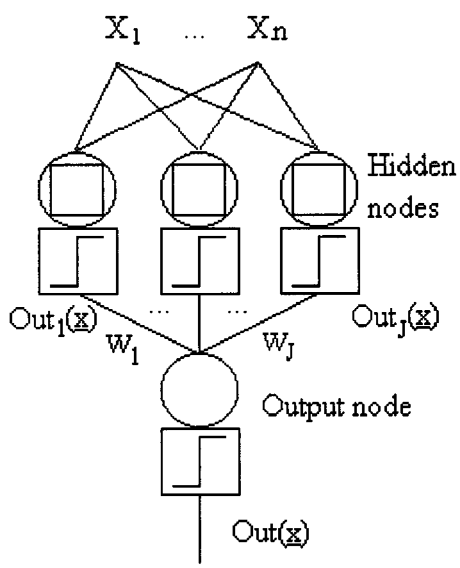


Fig. 2. A symbolic representation for a two-layer HRCNN.

II. Review of Hyperrectangular Composite Neural Networks

A symbolic representation of a two-layer HRCNN is illustrated in Fig. 2. The mathematical description of a two-layer HRCNN is given as follows:

$$Out(\underline{x}) = f\left(\sum_{j=1}^{J} Out_j(\underline{x}) - \eta\right) \tag{1}$$

$$Out_j(\underline{x}) = f(net_j(\underline{x})) \tag{2}$$

$$net_j(\underline{x}) = \sum_{i=1}^{n} f((M_{ji} - x_i)(x_i - m_{ji})) - n \tag{3}$$

and

$$f(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{4}$$

where M_{ji} and $m_{ji} \in R$ are adjustable synaptic weights of the jth hidden node, $\underline{x} = (x_1, \cdots, x_n)^T$ is an input pattern, η is a small positive real number less than 1, $Out_j(\underline{x})$ is the output function of the jth hidden node, and Out($\underline{x}$): $R^n \rightarrow \{0, 1\}$ is the output signal of a two-layer HRCNN with J hidden nodes. Note that $Out_j(\underline{x})$ is 1 if and only if $\underline{x}$ is in the jth hyperrectangle which is defined by $[m_{j1}, M_{j1}] \times \cdots \times [m_{jn}, M_{jn}]$ and $Out(x)$ is 1 if and only if $\underline{x}$ is in at least one of the J hyperrectangles. In our previous works [18]–[20], we have shown that the values of the synaptic weights of a trained HRCNN can be interpreted as a set of crisp IF-THEN rules. The IF-THEN classification rules extracted from a trained HRCNN with J hidden nodes can be represented as

$$\begin{aligned} &IF \quad (\underline{x} \in [m_{11}, M_{11}] \times \cdots \times [m_{1n}, M_{1n}]) \\ &THEN \quad out(\underline{x}) = 1; \\ &\qquad \vdots \\ &IF \quad (\underline{x} \in [m_{J1}, M_{J1}] \times \cdots \times [m_{Jn}, M_{Jn}]) \\ &THEN \quad out(\underline{x}) = 1; \\ &ELSE \quad out(\underline{x}) = 0 \end{aligned} \tag{5}$$

where the rule antecedents define a set of n-dimensional hyperrectangles.

The supervised decision-directed learning (SDDL) algorithm generates a two-layer HRCNN in a sequential manner by adding hidden nodes as needed. As long as there are no identical data over different classes, we can obtain a 100% recognition rate for the training data. First of all, training patterns are divided into two classes: 1) a "positive class" from which we want to extract the "concept" and 2) a "negative class which provides the counterexamples with respect to the concept." A "seed" pattern is used as the base of the "initial concept" (a hyperrectangle with arbitrarily small size). The seed pattern is arbitrarily chosen from the positive class. Then we try to generalize (expand) the initial concept (hyperrectangle) to include next positive pattern. Note that this initial hyperrectangle should not contain any negative pattern. The following step is to fetch the next positive pattern and to generalize the initial concept to include the new positive pattern. This process involves growing the original hyperrectangle to include the new positive pattern. After the process of generalization, again we use negative patterns to prevent overgeneralization. Fig. 3 illustrates the growing and shrinking procedure. From the figure we see clearly that the present rectangle (i.e., at time t) includes three positive examples represented by the symbol $*$. Then we expanded the rectangle to include another positive example. As a result, the expanded rectangle (i.e., at time $t+1$) includes a negative example represented by the symbol X; therefore, we have to shrink the expanded rectangle back so as to exclude the negative example. It should be emphasized that the shrunken rectangle (i.e., at time $t+2$) should include the original rectangle (i.e., at time t) in order to guarantee that the originally recognized positive examples (in this case, there are three positive examples) will not be forgotten after the procedure to prevent overgeneralization. This process is repeated for all the remaining positive patterns. If there is any unrecognized positive pattern, another initial hyperrectangle (hidden node) is generated and the process of learning is repeated again and again until all positive patterns are recognized. The flowchart of the SDDL algorithm is given in Fig. 4. A more detailed description of the training procedure is given in [18]–[20]. Here, we give a pseudocode description of two important procedures.

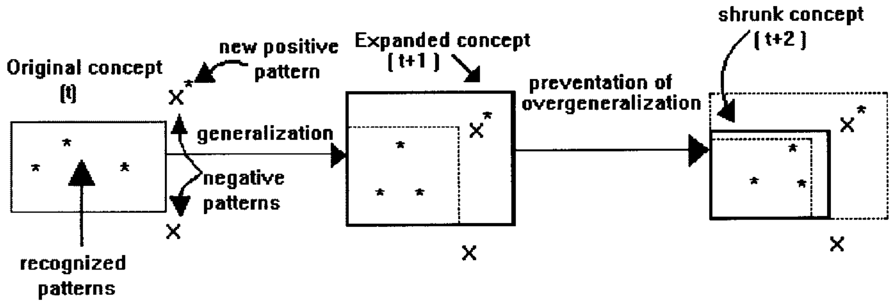


Fig. 3. An example of the growing and shrinking procedure involved in the SDDL algorithm.

- Procedure of generalization

begin ($\underline{x}$ is a positive pattern)
for i from 1 to dimeensions of input
begin
if $x_i \geq M_{ji}(t)$
then $M_{ji}(t+1) = x_i + \varepsilon$;
else if $x_i \leq m_{ji}(t)$
then $m_{ji}(t+1) = x_i - \varepsilon$;
else $M_{ji}(t+1) = M_{ji}(t)$ and $mji(t+1) = m_{ji}(T)$;
end;
end;
end.

- Procedure of prevention-of-overgeneralization

begin ($\underline{x}$ is a counterexample)
for i from 1 to dimensions-of-input
begin
if $x_i > M_{ji}(t)$
then $M_{ji}(t+2) = x_i - \delta$ (δ *should be chosen to ensure* $x_i - \delta \geq M_{ji}(t)$);
else if $x_i < m_{ji}(t)$
then $m_{ji}(t+2) = x_i + \delta$ (δ *should be chosen to ensure* $x_i + \delta \leq m_{ji}(t)$);
else $M_{ji}(t+2) = M_{ji}(t+1)$ and $m_{ji}(t+2) = m_{ji}(t+1)$;
end;
end;
end.

The value of the ε can be equal to or greater than zero. As for specification of the value of the parameter δ, one simple method is to make δ be equal to $\frac{1}{2}(x_i - M_{ji}(t))$ if $x_i > M_{ji}(t)$ or $\frac{1}{2}(m_{ji}(t) - x_i)$ if $x_i < m_{ji}(t)$.

III. Proposed Recognition Method

Our method of recognizing spatio-temporal hand gestures involves the following four steps.

Step 1) Sampling: Approaches to the problem of hand gesture recognition can be divided into two main categories: methods that rely on computer vision and methods that use data gloves. Computer vision based techniques are noninvasive, but it is difficult to implement successful systems using this approach because they cannot deal with complex hand gestures at this time. Although the obvious problem with the second approach is the need for wearing a pair of data gloves which limit the user's freedom of movement, we decided to adopt this approach because it can deal with more complex problems. The ten fingers joint angles are measured by a pair of sensing gloves which convert hand gestures into computer readable data. The three most popular models of sensing gloves are VPL Data-Glove, Virtex Cyber-Glove, and Mattel Power-Glove [21]. They all have sensors that measure some or all of the finger joint angles. Each has its own advantages and disadvantages (e.g., precision, stability, and cost). In our experiments, we used a pair of low-cost EMI-Gloves developed by ourselves as the interface. The EMI-Glove uses low-priced electro-mechanical strain gauges to sense the flexure information of the finger joint angles. There are a total of ten strain gauges on each hand in order to measure ten joints: the metacarpophalangeal joints of the five fingers, the interphalangeal joint of the thumb and the proximal interphalangeal joints of the other four fingers. Each sample is then a 20×1 column vector, therefore, samples of a hand gesture are a sequence of 20-dimensional vectors.

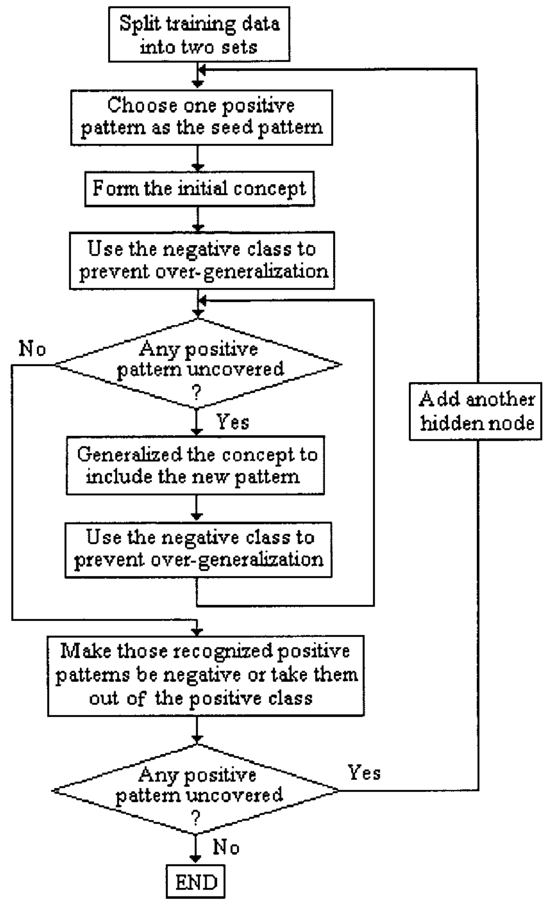
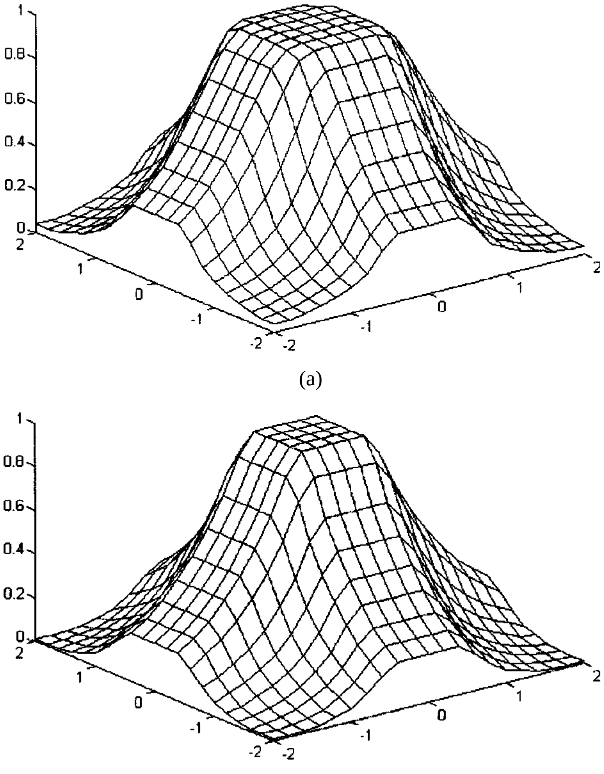


Fig. 4. Flowchart of the SDDL algorithm.

Step 2) Generation of Templates: Generally, in TSL every sign word consists of one or two basic hand shapes. Suppose there are N sign



(a) (b)

Fig. 5. An example of $s_k(l, k)$, with (a) $s = 5.0$ and (b) $s = 10.0$.

words in the vocabulary and these N sign words consist of N_s $(N_s \leq N)$ basic hand shapes. We then train N_s HRCNN's to generate templates for these N_s basic hand shapes. For convenience, we denote the number of hidden nodes corresponding to each basic hand shape as $H_k,\ k = 1, 2, \cdots, N_s$. Therefore, the kth basic hand shape has H_k templates which are represented as H_k IF-THEN rules.

Step 3) Pattern Recognition: When an unknown sign word is to be classified, every sample vector of the sign word is compared with each template of each basic hand shape in the vocabulary and a local measure of similarity between the test sample vector and each template is computed. To be precise, let $\underline{sx}_l$ be the sample vector of the unknown sign word with total L sample vectors. The local similarity between the sample vector $\underline{x}_l$ and the hth hyperrectangle of the kth basic hand shape is denoted as $S_k(l, h)$

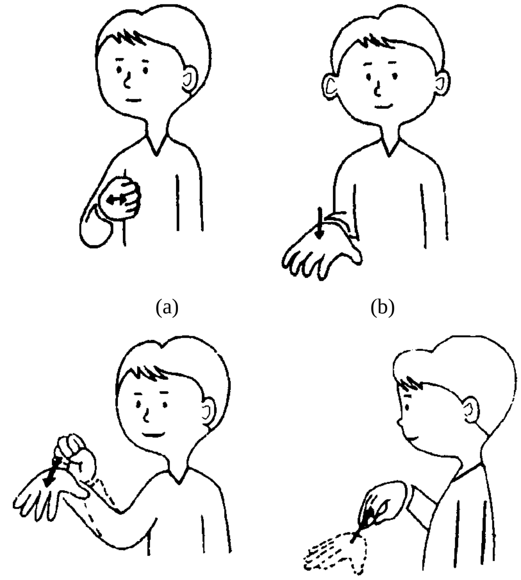
$$S_k(l, h) = \exp[-s^2(Per_h^{(k)}(\underline{x}_l) - Per_h^{(k)})^2] \tag{6}$$

where

$$Per_h^{(k)} = \sum_{i=1}^{20} (M_{hi}^{(k)} - m_{hi}^{(k)}) \tag{7}$$

$$Per_h^{(k)}(\underline{x}_l) = \sum_{i=1}^{20} \max(M_{hi}^{(k)} - m_{hi}^{(k)},\ M_{hi}^{(k)} - x_{li}, x_{li} - m_{hi}^{(k)}) \tag{8}$$

and s is a sensitivity parameter which regulates how fast the similarity value decreases as the distance between $\underline{x}_l$ and the hyperrectangle defined by $[M_{h,1}^{(k)}, m_{h,l}^{(k)}] \times \cdots \times [M_{h,20}^{(k)}, m_{h,20}^{(k)}]$. Actually, we may regard $S_k(l, h)$ as a membership function representing the grade of membership of $\underline{x}_l$ in the fuzzy set defining a hyperrectangle on the 20-dimensional input space. Fig. 5 illustrates an example of $S_k(l, h)$ for the two dimensional case. The reason why we fuzzify these crisp rules is as follows. Since there are many factors (e.g., sensor noise) would disturb the measurements of the finger's joint angles it is probable that many patterns would not fall inside these crisp hyperrectangles. Therefore we incorporate fuzzy sets introduced by Zadeh [22] into our recognition method.



(a) (b) (c) (d)

Fig. 6. A tie will happen when the following four sign words are simultaneously included in the vocabulary: (a) "familiar," (b) "place," (c) "throw away," and (d) "take."

Suppose the pth sign word in the vocabulary consists of the k_1th basic hand shape and the k_2th basic hand shape. The accumulative similarity, S_p, measures the degree to which the unknown sign word is similar to the pth sign word. The value of S_p is computed as

$$S_p = \sum_{l=1}^{L} \max\left\{\max_{1 \leq h_1 \leq H_{k_1}} S_{k_1}(l, h_1),\ \max_{1 \leq h_2 \leq H_{k_2}} S_{k_2}(l, h_2)\right\}. \tag{9}$$

Step 4) Decision Logic: Finally the unknown sign word is classified as the p^*th sign word in the vocabulary if the following condition holds:

$$S_{p^*} > S_p \text{ for } p \neq p^* \text{ and } p = 1, \cdots, N. \tag{10}$$

However, a tie may exist if one of the following two conditions happen:

1) two different sign words consist of the same two basic hand shapes but in a different order;
2) a sign word in the vocabulary is contained within another word.

Fig. 6 gives an example to illustrate such a tie. From the figure, we find the sign word "throw away" consists of the two basic hand shapes which represent the sign words "familiar" and "place," respectively. Reversing the order, the sign word "place" followed by the sign word "familiar" consists of the sign word "take." In order to break such a tie, we apply the concept of transition states to further process the patterns. To be precise, let the numbers 1 and 2 represent the first basic hand shape ("familiar") and the second basic hand shape ("place"), respectively. Suppose a user signs one the four sign words shown in Fig. 6. According to our algorithm, we will label every sample vector of the unknown hand gesture as either 1 or 2 based on the local similarity, therefore, the sample vectors of the unknown sign word may be represented by one of the following four labeling sequences:

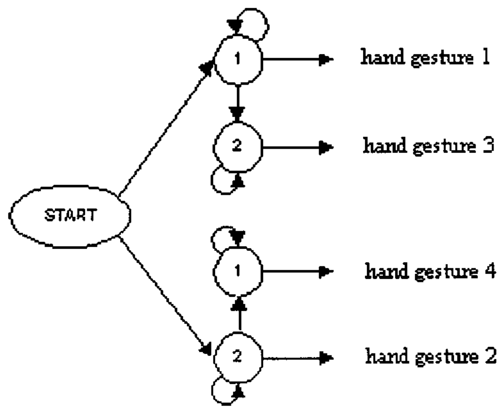


Fig. 7. An example of a two-state finite state network used to solve the tie condition.

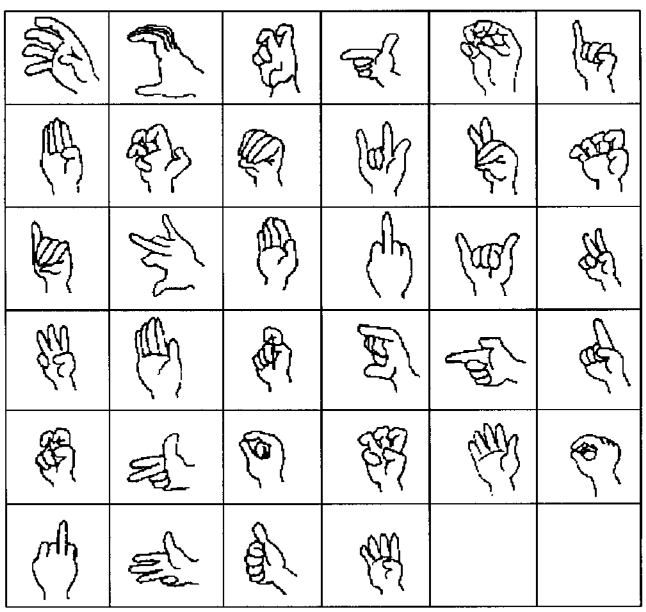


Fig. 8. Thirty—four basic hand shapes used in our vocabulary.

1) 111...1111 (the first sign word "familiar");
2) 222...2222 (the second sign word "place");
3) 111...2222 (the third sign word "throw away");
4) 222...1111 (the fourth sign word "take").

Obviously, if he or she signed the first sign word then $S_1 = S_3$. On the other hand, if the second word is signed then S_2 may be equal to S_4. To break such a tie, we feed the labeling sequence to the finite state network (FSN) shown in Fig. 7 and then the output of the FSN will provide us with a correct answer.

Apparently, we can directly make a recognition decision by computing (6)–(10) without involving any other computation to normalize out speaking (or signing) rate fluctuation, therefore our method reduces the substantial computation required by the DTW algorithm.

IV. Experimental Results

To evaluate the performance of the proposed method of recognizing spatio-temporal patterns, two databases consisting of 90 sign words from TSL were used. These 90 sign words consisted of the 34 basic hand shapes shown in Fig. 8. Four persons volunteered to sign the 90 sign words. Each sign word was repeated four times by each user. The first two persons contributed to the first database and the remaining two persons contributed to the second database. Two repetitions of the first two users were for training and the remaining two repetitions for testing. On average there are 125 sample vectors for every sign word. Note that for each sign word we just select the first (last) 10 sample vectors of all sample vectors to represent the first (second) basic hand shape. The reason is that we hope to keep the number of training data as small as possible. The experimental results showed that the correct recognition rates were 100% and 94.1% for the training set and testing set, respectively. We then used the rules extracted from the first database to test the second database. The correct recognition rate was 91.2%. Table I tabulates the recognition results obtained by the proposed method. The reason why we can not achieve 100% correct recognition rates is obvious. First, human variability causes the same sign word to be conducted differently each time. Even for the same user, different instances of the same sign word may not be identical. That is, both inter-person and intra-person time-scale variances exist in the hand gesture recognition problem. Second, some hand shapes are indeed similar, therefore, they are not easily classified because it is probable that one of the hand shapes formed by a user is recognized as another hand shape formed by another user. Indeed, our experimental results showed almost all missclassified errors are due to this kind of recognition error. Fig. 9 gives some examples of the hand shapes which are prone to causing recognition errors. Based on these discussions, our experimental results seem very encouraging.

TABLE I
THE RECOGNITION RESULTS

Database	The first database		The second database
Recognition Rate	training set	testing set	91.2%
	100%	94.1%	

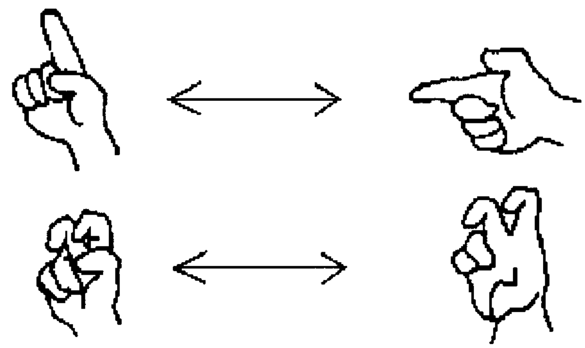


Fig. 9. Examples of hand shapes which are prone to causing recognition errors in our recognition system.

V. Conclusions

A method of recognizing spatio-temporal hand gestures was proposed in this paper. Based on the method we were able to implement a small-sized dynamic hand gesture recognition system. We used two databases which consisted of 90 sign words to evaluate the effectiveness of the method. By training a HRCNN, we efficiently generated templates for each basic hand shape. Then an unknown sign word was classified to the corresponding sign word in the vocabulary by computing accumulative similarities. In this manner, we obviated substantial computation for time alignment.

ACKNOWLEDGMENT

The author would like to thank H. Huang, C. H. Lin, C. L. Huang, and C. D. Lin for designing the EMI-gloves and writing the software.

REFERENCES

[1] M. R. Minsky, "Manipulating simulated objects with real-world gesture using a force and position sensitive screen," *ACM Comput. Graph.*, vol. 18, no. 3, pp. 195–203, 1984.

[2] W. Buxton, R. Sniderman, W. Reeves, S. Patel, and R. Baecker, "The evaluation of the SSSP score-editing tools," in *Foundations of Computer Music*, L. Roads and J. Strawn, Eds. Cambridge, MA: MIT Press, 1985, pp. 387–392.

[3] K. Ko and H. S. Yang, "Finger mouse and gesture recognition system as a new human computer interface," *Comput. Graph.*, vol. 21, no. 5, pp. 555–561, 1997.

[4] J. Kramer and L. Leifer, "The talking glove: An expressive and receptive verbal communication aid for the deaf, deaf-blind, and nonvocal," in *Proc. 3rd Annu. Conf. Computer Technology, Special Education, Rehabilitation*, Northridge, CA, Oct. 1987, pp. 335–340.

[5] ——, "The talking glove: A speaking aid for nonvocal deaf and deaf-blind individuals," in *Proc. RESNA 12th Annu. Conf.*, New Orleans, LA, 1989, pp. 471–472.

[6] S. S. Fels and G. E. Hinton, "Glvoe-Talk: A neural network interface between a Data-Glove and a speech synthesizer," *IEEE Trans. Neural Networks*, vol. 1, no. 1, pp. 2–8, 1993.

[7] K. Kamata, T. Yoshida, M. Watanabe, and Y. Usui, "An approach to Japanese sign language translation system," in *IEEE Int. Conf. Systems, Man, Cybernetics*, 1990, pp. 1089–1090.

[8] K. Murakami and H. Taguchi, "Gesture recognition using recurrent neural networks," in *Proc. CHI'91*, 1991, pp. 237–242.

[9] K. Morimoto, T. Izuchi, E. Fujishige, S. Watanabe, T. Morichi, and T. Kurokawa, "Analysis of spatio-temporal structure of sign language for its machine translation," in *8th Symp. Human Interface*, 1992, pp. 621–626.

[10] M. C. Su, "A speaking aid using neural networks for the deaf," *Biomed. Eng. Applicat., Basis, Commun.*, vol. 8, no. 4, pp. 33–39, 1996.

[11] R. Watson, "A survey of gesture recognition techniques," Trinity College, Dublin, Ireland, Tech. Rep. TCD-CS-93-11, 1993.

[12] L. R. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[13] R. Rebiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–285, 1989.

[14] H. Liang and M. Ouhyoung, "A sign language recognition using hidden Markov model and context sensitive search," in *ACM Symp. Virtual Reality Software Technology*, 1996, pp. 59–66.

[15] K. Grobel and M. Assan, "Isolated sign language recognition using hidden Markov models," in *IEEE Int. Conf. Systems, Man, Cybernetics*, 1997, pp. 162–167.

[16] Y. Nam and K. Y. Wohn, "Recognition of space-time hand-gestures using hidden Markov model," in *ACM Symp. Virtual Reality Software Technology*, 1996, pp. 51–58.

[17] C. Vogler and D. Metaxas, "Adapting hidden Markov models for ASL recognition by using three-dimensional computer vision methods," in *IEEE Int. Conf. Systems, Man, Cybernetics*, 1997, pp. 156–161.

[18] M. C. Su, "A neural network approach to knowledge acquisition," Ph.D. Dissertation, Univ. Maryland, College Park, Aug. 1993.

[19] M. C. Su, "Use of neural networks as medical diagnosis expert systems," *Comput. Biol. Med.*, vol. 24, no. 6, pp. 419–429, 1994.

[20] M. C. Su, C. T. Hsieh, and C. C. Chin, "A neuro-fuzzy approach to speech recognition without time alignment," *Fuzzy Sets Syst.*, vol. 98, no. 1, pp. 33–41, 1998.

[21] R. Skalwsky, *The Science of Virtual Reality and Virtual Environments*. Reading, MA: Addison-Wesley, 1993.

[22] L. A. Zadeh, "Fuzzy sets," *Inform. Contr.*, vol. 8, pp. 338–353, 1965.

A Fuzzy Expert System for Fault Detection in Statistical Process Control of Industrial Processes

Shendy M. El-Shal and Alan S. Morris

***Abstract*—Little work has previously been reported on the use of fuzzy logic within statistical process control when this is used for fault detection as part of quality control systems in industrial manufacturing processes. Therefore, this paper investigates the potential use of fuzzy logic to enhance the performance of statistical process control (SPC). The cumulative sum of the deviation in the monitored parameter is combined with the deviation in an attempt to discriminate between false alarms and real faults and, consequently, to improve the quality of the solution. Combinations of control rules are utilized and trained to cope with different inputs such that rejection of false alarms is achieved and quick detection of real faults is obtained. The design and implementation of this fuzzy expert system (FES) are presented, and a comparative rule-based study is performed.**

***Index Terms*—Expert system, false alarm, fault detection, fuzzy logic, statistical process control.**

I. INTRODUCTION

Statistical process control (SPC) is defined as the application of statistical methodology in process monitoring, and it is used as a tool in the control and improvement of manufacturing processes in a wide range of industries. Some examples of its application are the monitoring of paper thickness in a paper mill, oil viscosity in oil production and component value in the manufacture of electronic components such as resistors. One frequent source of confusion with the term SPC arises from the fact that it only describes a process monitoring function and does not encompass "control" in the usual engineering sense. Traditionally, in engineering jargon, the word "control" has been associated with engineering process control (EPC) methods. These have been utilized successfully in industry for decades, to make compensating adjustments in some manipulatable variables to counteract the effects of disturbances in an input stream on some process output. At present, SPC provides only a process monitoring function that detects when the monitored variables have deviated beyond expected values. In response, open-loop control action is applied in which a plant operator looks for the reason for variation in the monitored parameter values and takes corrective control action as appropriate. However, it has been noted [1] that SPC could be integrated with EPC to provide an automated response to process faults if the reliability level of SPC was at a high enough level.

SPC traditionally looks for deviations in measurements that are more than 3σ (three times the standard deviation) away from the mean expected value when the manufacturing process is in statistical control. There is only a 0.3% chance that a measurement is more than 3σ away from the mean value due only to random effects. However, 0.3% means three measurements out of each thousand. If the process is stopped every time that a measurement with a deviation greater than 3σ is detected, then it will stop three times unnecessarily for each thousand samples. These three unnecessary stoppages are called false alarms.

Manuscript received March 29, 1999; revised March 8, 2000.

S. M. El-Shal is with the Department of Electrical and Electronic Measurements, National Institute of Standards, 12211 Giza, Egypt (e-mail: smelshal@hotmail.com).

A. S. Morris is with the Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, S1 3JD (e-mail: a.morris@sheffield.ac.uk).

Publisher Item Identifier S 1094-6977(00)04800-8.