Gesture-Based Interaction and Communication: Automated Classification of Hand Gesture Contours

Lalit Gupta and Suwei Ma

***Abstract*—The accurate classification of hand gestures is crucial in the development of novel hand gesture based systems designed for human computer interaction (HCI) and for human alternative and augmentative communication (HAAC). A complete vision-based system consisting of hand gesture acquisition, segmentation, filtering, representation, and classification is developed to robustly classify hand gestures. The algorithms in the subsystems are formulated or selected to optimally classify hand gestures. The gray scale image of a hand gesture is segmented using a histogram thresholding algorithm. A morphological filtering approach is designed to effectively remove background and object noise in the segmented image. The contour of a gesture is represented by a localized contour sequence whose samples are the perpendicular distances between the contour pixels and the chord connecting the end-points of a window centered on the contour pixels. Gesture similarity is determined by measuring the similarity between the localized contour sequences of the gestures. Linear alignment and nonlinear alignment are developed to measure the similarity between the localized contour sequences. Experiments and evaluations on a subset of American Sign Language (ASL) hand gestures show that, by using nonlinear alignment, no gestures are misclassified by the system. Additionally, it is also estimated that real-time gesture classification is possible through the use of a high-speed PC, high-speed digital signal processing chips, and code optimization.**

***Index Terms*—Contours, hand gestures, morphological filtering alignment, segmentation.**

I. Introduction

This paper describes the design and implementation of a vision-based hand gesture classification (VHGC) system which can be used for novel human-computer-interaction (HCI) applications and for human alternative and augmentative communication (HAAC) applications. The main approaches for analyzing and classifying hand gestures for HCI and HAAC applications include glove-based techniques and vision-based techniques. The glove-based techniques use sensors to measure the positions of the fingers and the position of the hand in real-time. However, gloves tend to be quite expensive and the weight of the glove as well as the cables of the associated measuring equipment hinders free movement of the hand. The vision-based techniques are usually glove-free and can be divided into the three-dimensional (3-D) and the two-dimensional (2-D) approaches. In the 3-D approach, gesture classification is based upon the parameters of a 3-D model of the human hand. Gesture classification is based upon the parameters of an image of the gesture in the 2-D approach. Because 3-D hand models are quite complex, the classification of gestures from parameters derived from 3-D models is computationally extensive making real-time classification difficult. The 2-D models are relatively less complex than the 3-D models. However, 2-D models do not carry the finger movement and finger position information required for the classification of complex dynamic gestures. Therefore, the 2-D approach is restricted to the less complex problem of classifying well-defined static gestures. References [1]–[4] contain notable work in addressing issues and developing methodologies to solve problems related to vision based hand gesture classification. Additionally, [5] contains an excellent review of the advantages, limitations, applications, and vision based approaches for hand gesture classification. A thorough review of glove-based methods is contained in [6]. The proceedings of the International Workshop on Automatic Face and Gesture Recognition, Zurich, Switzerland, 1995, also contains valuable information on vision- and glove-based analysis and classification methods.

Manuscript received January 31, 2000; revised January 26, 2001. This paper was recommended by Associate Editor R. A. Hess.

The authors are with the Department of Electrical Engineering, Southern Illinois University, Carbondale, IL 62901 USA (e-mail: gupta@engr.siu.edu).

Publisher Item Identifier S 1094-6977(01)03531-3.

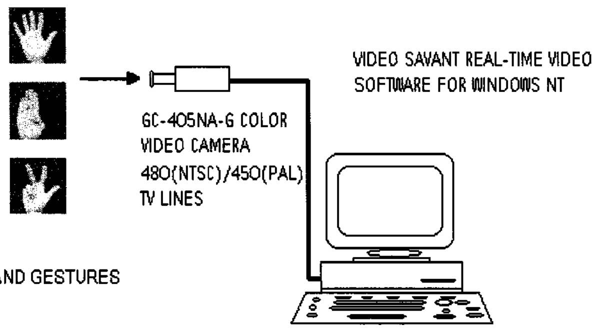

Fig. 1. Vision-based hand gesture classification system.

The VHGC system described in this paper is designed to be capable of classifying static hand gestures for real-time HCI and HAAC applications. Although the use of static hand gestures may appear restrictive, the potential applications for the VHGC system are, in fact, quite numerous. This is because a single gesture may represent a single command or a sequence of commands, a single word or a sequence of words to form a phrase, and single alphanumeric character or a collection of alphanumeric characters. Additionally, just two gestures representing dichotomous options such as "yes" and "no" can be used to build a rather complex hierarchical sequence of control and communication commands. As a result, general examples of the potential applications of the VHGC system include:

2) the remote control of devices and household appliances by disabled individuals;
3) a means for communications for vocally impaired individuals;
4) the execution of commands by a computer system.

II. System Description and Operational Flowchart

The VHGC system is shown in Fig. 1. The relatively simple system consists of a single video camera to acquire an image of a gesture and an HP 586-200 MHz personal computer for processing and classifying the gesture. The Video Savant™ real-time video software for Windows NT is used for the real-time acquisition, sampling, quantizing, and storing of the gestures. Static hand gestures are represented by their contours (discrete boundaries) and the similarity between gestures is determined by measuring the similarity between contour representations. An operational flowchart of the processing steps, which will be described in the subsequent sections, is shown in Fig. 2. The algorithms for the segmentation, filtering, representation, and classification steps were developed using the C++ programming language.

In operation, hand gestures are formed between the front of the video camera and a uniform background in a laboratory with florescent lights in the ceiling. No additional illumination sources are used. Individuals are instructed to form gestures in front of the camera with no restrictions on the distance between the hand and the camera nor any strict restrictions in the orientation of the hand in the plane parallel to the camera. The individuals, however, are instructed to keep the hands approximately parallel to the camera lens in order to maintain the gesture shape.

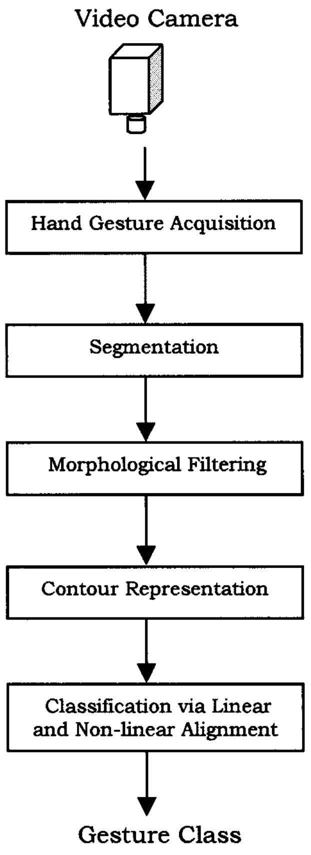

Fig. 2. Operational flowchart of the VHGC system.

For system development and off-line testing, ten gestures from the American Sign Language (ASL) were selected. These gestures were selected because they are typical of the hand gestures that can be used for HCI and HAAC applications. Five individuals, not trained in ASL, were instructed to form, twice, each of the ten gestures and a database of (5 × 10 × 2) = 100 gestures was generated. The spatial resolution of the uniformly sampled gesture image was selected as (l28 × l28) and the amplitudes were quantized into 256 gray levels.

Examples of one gesture from each of the ten gesture classes are shown in Fig. 3. Hereafter, gesture $k, k = 1, 2, \cdots, K$, from class m, $m = 1, 2, \cdots, M$, will be denoted by $g_{m,k}(x, y)$, where $(K = 10)$ is the number of gestures in each class and $(M = l0)$ is the number of gesture classes. Fig. 4 shows examples of five gestures, each made by a different individual, from class 1 in the data base. Observe the within-class temporal variations which occur because very few restrictions were imposed on forming the gestures. The temporal variations include changes in the position, orientation, dimension, and the shape of the gestures in the images.

III. Gesture Segmentation

The goal of gesture segmentation is to extract the hand gesture as accurately as possible from the background of the image. That is, the segmented hand gesture should not have parts of the background due to under segmentation nor should it have parts of the hand deleted due to over segmentation. In general, the selection of an appropriate segmentation algorithm depends largely on the type of images and the application areas. Because the laboratory environment (lighting conditions) of the VHGC system is fixed, an autonomous segmentation algorithm which gives good segmentation results in the laboratory environment

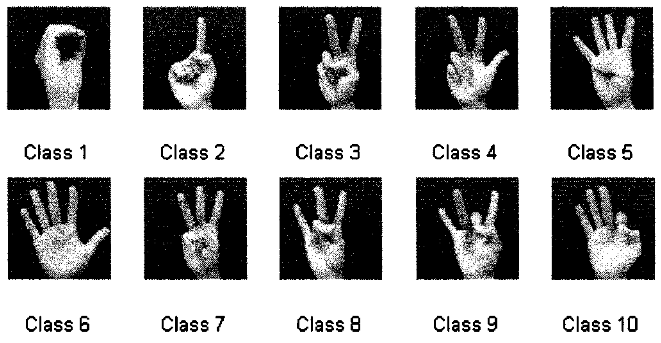

Fig. 3. Examples of one gesture from each gesture class.

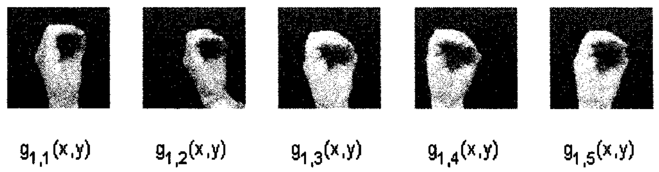

Fig. 4. Examples of five gestures belonging to class 1.

and is also computational simple can be selected. The Otsu segmentation algorithm [7] was tested and found to give good segmentation results for the hand gestures and was, therefore, selected. Briefly, the Otsu segmentation algorithm is a gray level thresholding algorithm based on discriminant analysis. The algorithm treats the segmentation of a gray scale image into a binary image as a classification problem in which the two classes (in this case, hand and background) are generated from the set of pixels within the gray scale image. Using a threshold T for an image with L gray levels, the image is segmented in two classes $\Omega_0 = \{1, 2, \cdots, T\}$ and $\Omega_1 = \{T+1, T+2, \cdots, L\}$. The optimum threshold T^* is determined as that value of T which maximizes the ratio of the between-class variance σ_B^2 to the total variance σ_T^2. If the number of pixels at gray level i is denoted by n_i and the total number of pixels is N, then, for a given T, the between class variance and the total variance are defined and computed as follows:

$$\sigma_B^2 = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2$$
$$\sigma_T^2 = \sum_{i=1}^{L} (i - \mu_T)^2 P_i$$

where

$$\omega_0 = \sum_{i=1}^{T} P_i, \quad \omega_1 = \sum_{i=T+1}^{L} P_i$$
$$\mu_0 = \sum_{i=1}^{T} (iP_i)/\omega_0, \quad \mu_1 = \sum_{i=T+1}^{L} (iP_i)/\omega_1$$
$$\mu_T = \sum_{i=1}^{L} (iP_i)$$
$$P_i = n_i/N, \qquad \left(P_i \geq 0 \text{ and } \sum_{i=1}^{L} P_i = 1\right).$$

Typically, the entire histogram is scanned to find the optimum T. However, it was found that for the hand gesture images in the laboratory environment, the optimum T tended to be close to the estimated total mean μ_T. To decrease computation time, the histogram was, therefore, scanned in a small window of size 11 gray levels centered on the integer value of μ_T. The images $s_{1,5}(x, y)$ and $s_{3,5}(x, y)$ resulting from

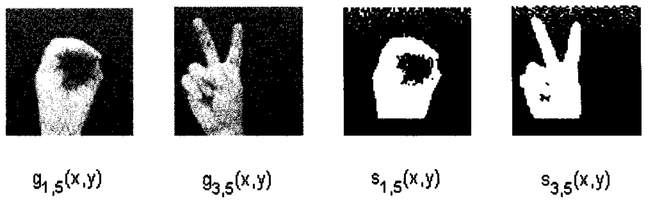

Fig. 5. Examples of segmentation.

segmenting images $g_{1,5}(x, y)$ and $g_{3,5}(x, y)$ using the segmentation algorithm are shown in Fig. 5. The hand pixels are assigned "1" and the background pixels are assigned "0."

The Otsu algorithm was also tested for small variations in the lighting conditions of the laboratory and it was found that although the threshold changed, the segmentation result was not adversely affected. That is, there were only minor changes in the segmented gesture mainly because of the uniform background. Algorithms designed to segment diverse images [8] can be employed to accurately segment gestures in varying backgrounds, however, such algorithms tend to be computationally extensive.

IV. Morphological Filtering

A close examination of the segmented gesture images revealed that the segmentation was seldom perfect. That is, as evident in Fig. 5, the background may have ls (background noise) and the gesture may have 0s (object noise). The background noise and object noise can cause problems in extracting the contour of the gesture, especially when they are close to the contour. It is, therefore, desirable to decrease the background noise and object noise prior to the extraction of the gesture contour. A morphological filtering [9] approach using a sequence of dilation and erosion operations was developed to obtain a smooth, closed, and complete contour of a gesture. In general, the dilation and erosion operations on a binary image A and with a structuring element B are defined as follows.

1) Dilation: If A and B are sets in the 2-D integer space $Z^2, x = (x_1, x_2)$ and ϕ is the empty set, then, the dilation of A by B is

$$A \oplus B = \{x | (\hat{B})x \ \cap \ A \neq \phi\}$$

where, $\hat{B}$ is the reflection of B. Dilation consists of obtaining the reflection of B about its origin and then shifting this reflection by x. The dilation of A by B is the set of all x displacements such that $\hat{B}$ and A overlap by at least one nonzero element. Set B is commonly referred to as the structuring element.

2) Erosion: The erosion of A by B is

$$A \otimes B = \{x | (B)x \subseteq A\}.$$

That is, the erosion of A by B is the set of all points x such that B, translated by x, is contained in A. Note that dilation expands an image and erosion shrinks it.

3) Opening: The opening of set A by structuring element B is

$$A \circ B = (A \otimes B) \oplus B.$$

That is, the opening of A by B is simply the erosion of A by B followed by a dilation of the result by B. Opening generally smoothes the contour of an image, breaks narrow isthmuses, and eliminates thin protrusions.

4) Closing: The closing of set A by structuring element B is

$$A \cdot B = (A \oplus B) \otimes B.$$

That is, the closing of A by B is simply the dilation of A by B followed by the erosion of the result by B. Closing also tends to smooth sections of contours but, as opposed to opening, it generally fuses narrow breaks and long thin gulfs, eliminates small holes, and fills gaps in the contour.

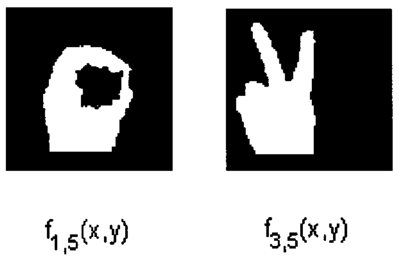

Fig. 6. Examples of morphological filtering for noise removal.

Fig. 6 shows the results of applying an opening operation followed by a closing operation on the noisy segmented images $s_{1,5}(x, y)$ and $s_{3,5}(x, y)$ shown in Fig. 5 to obtain gestures with smooth and complete contours. For each case, the set A is the noisy segmented image and a 3 × 3 structuring element B of 0s is used in these examples. The opening operation which is erosion followed by dilation is used to remove the background noise and the closing operation which is dilation followed by erosion is used to remove object noise. It is seen that through morphological filtering, the resulting images $f_{1,5}(x, y)$ and $f_{3,5}(x, y)$ of the segmented gestures $s_{1,5}(x, y)$ and $s_{3,5}(x, y)$ are noise-free and the contour is an undistorted outline of the gesture.

V. Contour Representation

A careful examination of the filtered gestures reveals that what distinguishes one gesture form another gesture is the shape of the contour. Therefore, the contour can be used as a basis for the discrimination of the hand gestures. The localized contour sequence (LCS), which has been proven to be a very effective representation of contours [10], is selected to represent the gesture contours. A contour tracking algorithm is used to track the contour of a gesture in the clockwise direction and the contour pixels are numbered sequentially starting from the arbitrarily selected contour pixel. If $h_i = (x_i, y_i)$, $i = 1, 2, \cdots, N$, is the ith contour pixel in the sequence of N ordered contour pixels of a gesture, the ith sample $h(i)$ of the LCS of the gesture is obtained by computing the perpendicular Euclidean distance between h_i and the chord connecting the end-points $h_{[i-(w-1)/2]}$ and $h_{[i+(w-1)/2]}$ of a window of size w boundary pixels (w odd) centered on h_i. That is

$$\begin{aligned} h(i) &= |u_i / v_i|, \quad \text{where} \\ u_i &= x_i[y_{i-(w-1)/2} - y_{i+(w-1)/2}] \\ &\quad + y_i[x_{i+(w-1)/2} - x_{i-(w-1)/2}] \\ &\quad + [y_{i+(w-1)/2}][x_{i-(w-1)/2}] \\ &\quad - [y_{i-(w-1)/2}][x_{i+(w-1)/2}], \quad \text{and} \\ v_i &= [(y_{i-(w-1)/2 - yi+(w-1)/2})^2 \\ &\quad + (x_{i-(w-1)/2} - x_{i+(w-1)/2})^2]^{1/2}. \end{aligned}$$

The computation of $h(i)$ is illustrated in Fig. 7. A gesture $f_{m,k}(x, y)$ with $N_{m,k}$ contour pixels results in an $N_{m,k}$ point sequence represented by $h_{m,k}(i)$, $i = 1, 2, \cdots, N_{m,k}$. The LCS has the following properties that make it attractive for representing hand gesture contours.

a) The LCS is not restricted by shape complexity and is, therefore, suitable for gestures which typically have convex and concave contours.

b) The LCS can also be used to robustly represent partial contours [10]. Therefore, the representation of the visible part of the gesture will not be affected when a part of the gesture is obscured because the hand is not parallel to the camera.

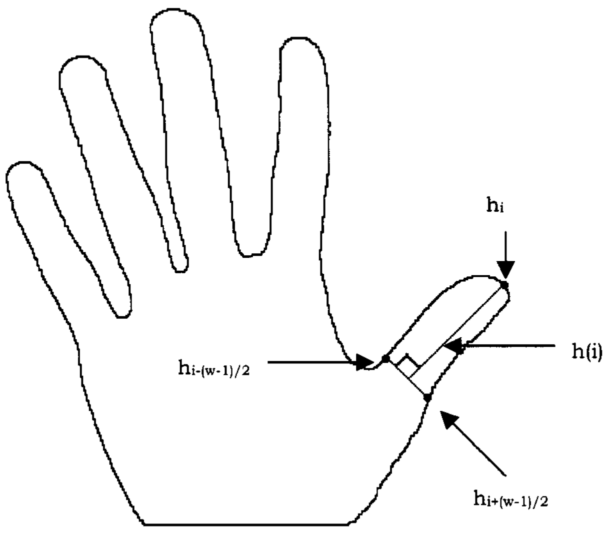

Fig. 7. Computation of the samples of the LCS.

c) Because the representation does not involve derivative computations such as slopes or curvature, the representation is quite robust with respect to contour noise (random variations in the contour).

d) Increasing w tends to increase the amplitudes of the samples of the localized contour sequence. An increase in the amplitudes has the effect of increasing the signal-to-noise ratio for a fixed contour noise level, therefore, the robustness with respect to contour noise can be increased by increasing w.

Because the gestures have closed boundaries, $h_{m,k}(i)$ may be regarded as a circular sequence. Fig. 8 shows the LCSs $h_{1,5}(i)$ and $h_{3,5}(i)$ of the filtered gesture $f_{1,5}(x,y)$ and $f_{3,5}(x,y)$ shown in Fig. 6 using $w = 99$. To aid visual analysis, the discrete LCSs are displayed as continuous signals in the figures.

VI. Linear Alignment of LCSs

Recall that during operation, no restrictions are placed on the position, distance, and orientation of the gesture in front of the camera. The shape of the gesture does not change when the position, distance, and orientation of the gesture change. A change in the position of a gesture results in a translation of the gesture in the image plane. The LCS is clearly invariant to gesture translation in the image plane. The start-point is determined by locating the first contour pixel using a left-to-right and top-to-bottom scan of the image. Therefore, a change in the orientation of a gesture results in a circular shift in the samples of the LCS. The dimension of the contour is scaled when the distance of the gesture from the camera changes; therefore, the duration and the amplitude of the LCS are also scaled. However, the shape of the LCS does not change because the shape of the contour does not change. The scaling of the amplitude of the LCS can be easily normalized by dividing the samples of the LCS by the standard deviation of the LCS. The scaling of the duration can also be easily normalized by uniformly expanding or compressing the LCS to have a fixed duration $\hat{N}$ using a linear transformation. Invariance to shifts in the start point can be incorporated into the classification stage by determining the position of best match using a circular shifting operation.

In the first method developed to compute the similarity between LCSs, which will be referred to as the linear alignment method, the LCSs are amplitude normalized using the standard deviation and are duration normalized using a linear transformation. That is, if the amplitude and duration normalized LCSs of the reference gesture of class m

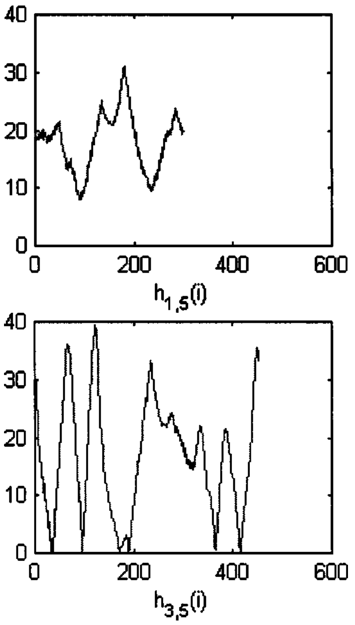

Fig. 8. LCSs of the contours of the gestures in Fig. 6.

and a test gesture are represented by $\hat{h}_m(i)$ and $\hat{t}(i)$, $i = 1, 2, \cdots, \hat{N}$, respectively, then, the dissimilarity between the two LCSs can be found by first determining

$$D_m(j) = \sum_{i=1}^{\hat{N}} |\hat{h}_m(i) - \hat{t}((i+j))|$$
$$j = 0, 1, \cdots, (\hat{N} - 1)$$

where $\hat{t}((i+j))$ denotes a circular shift of j samples in $\hat{t}(i)$.

The best match between $\hat{h}_m(i)$ and $\hat{t}(i)$, is then given by

$$D_m = \min_j [D_m(j)].$$

The test gesture is hypothesized to belong to each gesture class to compute $D_m, m = 1, 2, \cdots, M$, and the test gesture is assigned to class m^* given by the minimum distance rule

$$m^* = \arg\ \min [D_m].$$

VII. Nonlinear Alignment of LCSs

It should be noted that besides the variations due to changes in the orientation and distance of the gestures, the contours of gestures from the same class also experience additional within-class variations because a) a gesture formed by different individuals will not be exactly the same and b) a gesture formed by an individual may not be exactly the same each time the gesture is formed. The difference in the gestures formed by an individual or individuals will result in variations in the shape of contour. These variations will be regarded as nonlinear distortions in the contour because some segments of the contour may be compressed and some segments may be expanded when the positions of the fingers vary in forming the gesture. Consequently, segments of the LCS experience compression and expansion resulting in nonlinear distortion in the LCS. Clearly, uniformly expanding or compressing the LCS will not compensate for the nonlinear distortions in the segments of the LCS.

The second method developed to measure the similarity between the LCSs takes the nonlinear distortions into account. In order to accommodate nonlinear distortions, nonlinear alignment techniques [11]–[13] have been developed to measure the similarity between sequences. In nonlinear alignment, the goal is to optimally align the samples of the two sequences so that the dissimilarity between the two sequences is minimized.

In the nonlinear alignment formulation to follow, it will be assumed that the LCSs are amplitude and duration normalized as well as start-point aligned using circular shifting. Although duration normalization is not necessary in the formulation, it is incorporated so that exactly the same sequences are used in the comparison between the performances of linear and nonlinear alignment classification methods. If the amplitude and duration normalized LCSs of a gesture from class m and a test gesture are now represented as $\hat{h}_m(q)$ and $\hat{t}(r), q, r = 1, 2, \cdots, \hat{N}$, respectively, then, the samples of two sequences are optimally aligned by determining an alignment function W of the form

$$W = w(1), w(2), \cdots, w(Z)$$

where $w(z) = [i(z), j(z)]$.

W provides a mapping between the axes q and r via an intermediate axis z of length Z such that the overall dissimilarity between the two samples is minimized. For each $w(z)$, a cost $d[w(z)]$ is assigned to reflect the discrepancy between the aligned samples. Assuming that the absolute difference is used for the cost function, the alignment function is determined such that the overall cost

$$\sum_{z=1}^{Z} d[w(z)] = \sum_{z=1}^{Z} |\hat{h}_m[i(z)] - \hat{t}[j(z)]|$$

is minimized subject to the following constraints.

1) Monotonicity: The alignment function must be monotonic to preserve the natural ordering of the samples in the sequences. That is

$$i(z) \geq i(z-1)$$
$$j(z) \geq j(z-1).$$

2) End-Point Alignment: The end-points (first and last samples) of the sequences must be aligned. That is

$$i(1) = j(1) = 1$$
$$i(Z) = j(Z) = \hat{N}.$$

3) Continuity: The alignment function must not skip any samples in the two sequences; therefore

$$i(z) - i(z-1) \leq 1$$
$$j(z) - j(z-1) \leq 1.$$

The solution to the above optimization problem is given by solving the recursive equation

$$D[w(z)] = d[w(z)] + \min_{w(z-1)} \{D[w(z-1)]\}$$

with initial conditions

$$D[w(1)] = d[w(1)].$$

The overall dissimilarity between the two sequences after alignment is given by

$$D_m = (1/Z)D[w(Z)].$$

From the monotonicity and continuity constraints imposed on the alignment function, if $w(z) = (i, j)$ then $w(z-1)$ consists of $[i-1, j]$, $[i, j-1]$, and $[i-1, j-1]$ and the recursive equation becomes

$$D[i, j] = d[i, j] + \min \{D[i-1, j], D[i, j-1], D[i-1, j-1]\}.$$

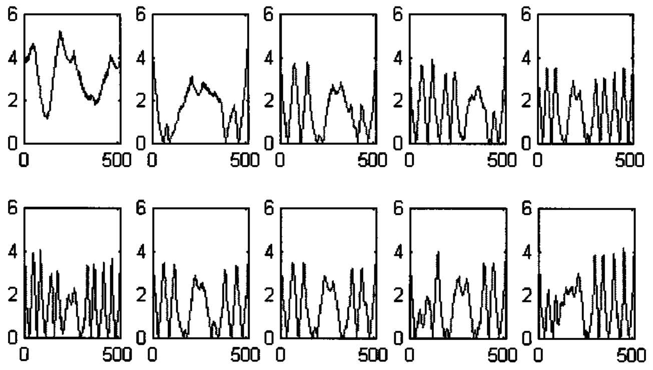

Fig. 9. Amplitude and duration normalized LCSs of the gestures in Fig. 3.

$D[i, j]$ is the dissimilarity remaining between the sequences $\hat{h}_m(q)$ and $\hat{t}(r)$ after alignment and is referred to as the discrepancy measure. If $D[i, j]$ is stored in an $\hat{N}x \times \hat{N}$ array, $D[i, j]$ is computed only for $[i, j]$ values in a narrow band along the diagonal of the $\hat{N} \times \hat{N}$ array because the optimal alignment path for sequences with similar samples tends to fluctuate in the neighborhood of the diagonal. Restricting the computations in the band along the diagonal serves two important purposes. The band restricts the region of search for the optimal alignment path in a meaningful manner. That is unreasonable alignment between very dissimilar sequences (e.g., LCSs from different gesture classes) is prevented. The band also reduces the number of computations required in determining the discrepancy measure.

In order to classify a test gesture represented by the LCS $\hat{t}(r)$, the discrepancy $D_m, m = 1, 2, \cdots, M$, between $\hat{t}(r)$ and each reference gesture $\hat{h}_m(q), m = 1, 2, \cdots, M$ is computed. The test gesture is assigned to the class m^* given by

$$m^* = \arg \min [D_m].$$

VIII. Classification Results

The amplitude and duration normalized LCSs of the 100 gesture images in the database were computed. That is, each gesture was represented by its corresponding normalized LCS. The LCSs were normalized to have a duration equal to 516 which was the average duration of the 100 LCSs. Fig. 9 shows the amplitude and duration normalized LCSs of the ten gestures in Fig. 3. In order to robustly evaluate the performance, a random sampling approach was used to generate multiple classification trials. For each trial, a reference LCS for each class was randomly selected from the ten LCSs of the class. The remaining nine LCSs formed the test set for each class. Therefore, the total number of LCSs tested in each trial was (9 × 10) = 90. The probability of classification error for trial number j was estimated as

$$p_j = (\text{total number of LCSs misclassified in trial } j/90).$$

The overall probability of misclassification was estimated as

$$p_e = (1/J) \sum_{j=1}^{J} p_j$$

where J is the total number of trials.

Tables I and II show the results for 50 classification trials. Column 1 of Table I shows that out of the (50 × 9) = 450 tests conducted using LCSs from class 1, no misclassifications were obtained. Column 4 shows that out of the 450 tests conducted using the LCSs from class 4, 90 were misclassified into class 7 and 20 were misclassified into class 10. The other elements in the tables are interpreted in a similar fashion. The overall probability of misclassification is shown in the last row

TABLE I
CLASSIFICATION RESULTS AND PROBABILITY OF MISCLASSIFICATION USING NONLINEAR ALIGNMENT

Reference Gestures	Test Gestures									
	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9	Class 10
Class 1	**450**	0	0	0	0	0	0	0	0	0
Class 2	0	**450**	0	0	0	0	0	0	0	0
Class 3	0	0	**450**	0	0	0	0	0	0	0
Class 4	0	0	0	**340**	0	0	0	0	0	**2**
Class 5	0	0	0	0	**438**	***23***	0	0	0	***18***
Class 6	0	0	0	0	**12**	**427**	0	0	0	0
Class 7	0	0	0	***90***	0	0	**450**	***23***	0	***47***
Class 8	0	0	0	0	0	0	0	**427**	0	0
Class 9	0	0	0	0	0	0	0	0	**450**	0
Class 10	0	0	0	***20***	0	0	0	0	0	**383**
Probability of misclassification $p_e = 235/4500 = 0.052$										

TABLE II
CLASSIFICATION RESULTS AND PROBABILITY OF MISCLASSIFICATION USING NONLINEAR ALIGNMENT

Reference Gestures	Test Gestures									
	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9	Class 10
Class 1	**450**	0	0	0	0	0	0	0	0	0
Class 2	0	**450**	0	0	0	0	0	0	0	0
Class 3	0	0	**450**	0	0	0	0	0	0	0
Class 4	0	0	0	**450**	0	0	0	0	0	0
Class 5	0	0	0	0	**450**	0	0	0	0	0
Class 6	0	0	0	0	0	**450**	0	0	0	0
Class 7	0	0	0	0	0	0	**450**	0	0	0
Class 8	0	0	0	0	0	0	0	**450**	0	0
Class 9	0	0	0	0	0	0	0	0	**450**	0
Class 10	0	0	0	0	0	0	0	0	0	**450**
Probability of misclassification $p_e = 0/4500 = 0$										

of the tables. The results show that the performance of the nonlinear alignment classification method is superior to that of the linear alignment method.

There are two factors that need to be considered in the development of gesture based classification systems for HCI and HAAC applications. The first, which is the goal of this paper, is to demonstrate that the gestures can be classified accurately by the system. This has been demonstrated clearly by the results shown in Table II. The next is to demonstrate that the gestures can be classified in real-time. The segmentation, morphological filtering, LCS computation, and the alignment classification algorithms were coded in the C++ programming language. No attempt was made to optimize the code. Additionally, the gesture images are read from the hard drive during operation because the video capturing software is designed to store the images directly onto the hard drive. Clearly, the system is not optimized with respect to the classification speed. It took 20 s to classify each gesture using linear alignment. It took 30 s to classify each gesture when nonlinear alignment was used in a band of nine samples centered along the diagonal. Both methods incorporated start-point normalization by determining the best match using ten circular shifts centered on the initially detected start point. The computations involved in the segmentation, filtering, LCS, start-point normalization, and the nonlinear alignment algorithms are typical of the computations that can be performed quite rapidly using available high-speed processors and high-speed digital signal processing chips. Further reductions in the classification time are also possible through the development of a buffer-based image capturing system and code optimization. It is, therefore, estimated that significant reductions in the classification time are possible if the VHGC system is developed using high-speed processors, buffer-based acquisition, and code optimization, thus, making real-time gesture classification possible.

IX. CONCLUSION

The goal of the paper was to develop a complete system capable of robustly classifying hand gestures for HCI and HAAC applications. From a visual analysis of hand gestures, it was determined that essential shape information for discriminating gestures was in the boundary of the gestures. Therefore, a contour and vision based classification approach was formulated. The relatively simple system consisted of a video camera, video capturing software, and a personal computer. For flexible operation, no constraint other than holding the gesture approximately parallel to the camera lens was imposed. The processing steps to classify a gesture included gesture acquisition, segmentation, morphological filtering, contour representation, and alignment based classification. Rather than forming an arbitrary set of gestures, the database for off-line evaluation consisted of the gestures for numbers 0 through 9 of the ASL. These gestures were selected because they are typical of the hand gestures that can be used for HCI and HAAC applications. The ten-class database consisted of ten example gestures for each class. The Otsu algorithm was selected to autonomously segment the gesture images and a morphological filtering approach was developed to remove background and object noise. The contour of a gesture was represented by the LCS and a linear alignment and a nonlinear alignment method were formulated to determine the similarity between two LCSs. The classification results showed that no misclassifications were obtained using nonlinear alignment even though the within-class variations were high because the gestures were formed by individuals not trained in ASL and with few constraints. The performance of the nonlinear alignment method was superior to that of the linear alignment method because, unlike linear alignment which simply uniformly expands or compresses the duration of a sequence, nonlinear alignment optimally aligns the samples of two sequences to minimize the dissimilarity between the sequences. Nonlinear alignment is clearly the better choice for classifying hand gestures because of the inherent nonlinear distortions that can be expected in the contours of the gestures.

The main goal was to show that robust classification of hand gestures is possible using the system developed in this paper. No attempt was made to optimize the code nor was any attempt made to use the fastest available processors. For the system to be successfully applied to the potentially numerous real-time HCI and HAAC applications, the hand gestures should not only be classified accurately but must also be classified rapidly. It is estimated that if the system is developed using a high-speed PC with high-speed digital signal processing chips, images are read directly from a buffer, and the code for the processing steps is optimized, gestures can be classified fast enough to make real-time applications possible. Additionally, through the further reduction in classification time, the VHGC could be extended to classify dynamic gestures represented by sequences of static gestures.

REFERENCES

[1] J. Lee and T. L. Kunii, "Model-based analysis of hand posture," *IEEE Comput. Graph. Appl.*, pp. 77–86, Sept. 1995.

[2] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 696–710, July 1997.

[3] T. Staner, J. Weaver, and A. Pentland, "Real-time American sign language recognition using desk and wearable computer based video," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 1371–1375, Dec. 1998.

[4] Y. Cui and J. Weng, "A learning-based prediction-and-verification segmentation scheme for hand sign image sequence," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 798–804, Aug. 1999.

[5] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human computer interaction: A review," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 677–694, July 1997.

[6] D. J. Sturman and D. Zeltzer, "A survey of glove-based input," *IEEE Comput. Graph. Appl.*, vol. 14, pp. 30–39, Jan. 1994.

[7] N. Otsu, "A threshold selection method from gray-level histogram," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, pp. 62–66, Jan. 1979.
[8] L. Gupta and T. Sortrakul, "A Gaussian mixture based image segmentation algorithm," *Pattern Recognit.*, vol. 31, no. 3, pp. 315–325, 1998.
[9] E. R. Dougherty, *An Introduction to Morphological Image Processing*. Bellingham, WA: SPIE, 1992.
[10] L. Gupta, T. Sortrakul, A. Charles, and P. Kisatsky, "Robust automatic target recognition using a localized boundary representation," *Pattern Recognit.*, vol. 28-10, pp. 1587–1598, 1995.
[11] H. Sakoe and S. Chiba, "Dynamic programming optimization for spoken word recognition," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-26, pp. 43–49, Feb. 1978.
[12] L. Gupta, D. L. Molfese, R. Tammana, and P. G. Simos, "Non-linear alignment and averaging for estimating the evoked potential," *IEEE Trans. Biomed. Eng.*, vol. 43, pp. 348–356, Apr. 1996.
[13] L. Gupta and K. Malakapalli, "Robust partial shape classification using invariant breakpoints and dynamic alignment," *Pattern Recognit.*, vol. 23-10, pp. 1103–1111, 1990.

Nonparametric Genetic Clustering: Comparison of Validity Indices

Sanghamitra Bandyopadhyay and Ujjwal Maulik

***Abstract*—Variable string length genetic algorithm (GA) is used for developing a novel nonparametric clustering technique when the number of clusters is not fixed *a priori*. Chromosomes in the same population may now have different lengths since they encode different number of clusters. The crossover operator is redefined to tackle the concept of variable string length. Cluster validity index is used as a measure of the fitness of a chromosome. The performance of several cluster validity indices, namely, Davies–Bouldin (DB) index, Dunn's index, two of its generalized versions and a recently developed index, in appropriately partitioning a data set, are compared.**

***Index Terms*—Clustering, cluster validity, Davies–Bouldin (DB) index, generalized Dunn's index, genetic algorithms (GAs), pattern recognition.**

I. Introduction

Genetic algorithms (GAs) [1], [2] are randomized search and optimization techniques guided by the principles of evolution and natural genetics, and have a large amount of implicit parallelism. They provide near optimal solutions of an objective or fitness function in complex, large, and multimodal landscapes. In GAs the parameters of the search space are encoded in the form of strings (or, *chromosomes*). A *fitness* function is associated with each string that represents the degree of *goodness* of the solution encoded in it. Biologically inspired operators like *selection*, *crossover*, and *mutation* are used over a number of generations for generating potentially better strings.

Clustering [3], [4] is a popular unsupervised pattern classification technique which partitions the input space into K regions based on some similarity/dissimilarity metric where the value of K may or may not be known *a priori*. The aim of any clustering technique is to evolve a partition matrix $U(X)$ of the given data set X (consisting of, say, n patterns, $X = \{x_1, x_2, \ldots, x_n\}$) such that

$$\sum_{j=1}^{n} u_{kj} \geq 1 \quad \text{for } k = 1, \ldots, K$$
$$\sum_{k=1}^{K} u_{kj} = 1 \quad \text{for } j = 1, \ldots, n \quad \text{and}$$
$$\sum_{k=1}^{K} \sum_{j=1}^{n} u_{kj} = n.$$

The partition matrix $U(X)$ of size $K \times n$ may be represented as $U = [u_{kj}]$, $k = 1, \ldots, K$ and $j = 1, \ldots, n$, where u_{kj} is the membership of pattern x_j to cluster C_k. In crisp partitioning $u_{kj} = 1$ if $x_j \in C_k$; otherwise $u_{kj} = 0$. Cluster validity is a measure associated with different partitions that indicates their relative goodness.

In most real-life situations the number of clusters in a data set is not known beforehand. The real challenge is to be able to automatically evolve a proper value of the number of clusters and provide the appropriate clustering under this circumstance. Some attempts in this regard can be found in [4] and [5]. The ISODATA algorithm [4] uses a combination of splitting, merging and deleting clusters to adjust the number of cluster centers. Each of these operations depends on several user supplied parameters, which are often very difficult to estimate *a priori*. Recently, Ravi and Gowda [5] used a distributed GA based on the ISODATA technique for clustering symbolic objects. However, this method also suffers from the same limitations as present in the ISODATA clustering technique.

Our aim in this paper is to develop a nonparametric clustering technique which will not assume any particular underlying distribution of the data set, while it will be able to evolve a proper value of number of clusters as well as provide the appropriate clustering automatically. Variable string length genetic algorithm (VGA) [6], with real encoding of the cluster centers in the chromosome [7], is used as the underlying search tool for this purpose. Several cluster validity indices viz., Davies–Bouldin (DB) index [8], Dunn's index [9], two of its generalized versions [10], and a newly developed validity index are utilized for computing the fitness of the chromosomes. The results provide a comparison of these indices in terms of their utility in determining the appropriate clustering of the data. Several artificial and real-life data sets with different characteristics are used for performing the experiments.

II. Clustering Using Variable String Length Genetic Algorithms

In this section, we describe the use of VGAs for automatically clustering a data set. This involves determination of the number of clusters as well as the appropriate clustering of the data. The technique, described below, is subsequently referred to as the *VGA-clustering* (VGA-based clustering).

String Representation and Population Initialization: In *VGA-clustering*, the chromosomes are made up of real numbers (representing the coordinates of the centers). If chromosome i encodes the centers of K_i clusters in N dimensional space $K_i \geq 2$, then its length l_i is taken to be $N * K_i$.

Each string i in the population initially encodes the centers of a number K_i of clusters, where K_i is given by $K_i = \text{rand}(\,) \bmod K^*$. Here, $\text{rand}(\,)$ is a function returning an integer, and K^* is a soft estimate of the upper bound of the number of clusters. Note that K^* is used only for the generation of the initial population. The actual number of

Manuscript received June 23, 2000; revised December 22, 2000. This paper was recommended by Associate Editor V. Kolmanovskii.

S. Bandyopadhyay is with the Machine Intelligence Unit, Indian Statistical Institute, Calcutta, India (e-mail: sanghami@isical.ac.in).

U. Maulik is with the Department of Computer Science & Technology, Kalyani Government Engineering College, Kalyani, India (e-mail: ujjwal_maulik@kucse.wb.nic.in).

Publisher Item Identifier S 1094-6977(01)03532-5.