PERGAMON

Pattern Recognition 36 (2003) 2069–2081

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Recognition of dynamic hand gestures

Aditya Ramamoorthy[a], Namrata Vaswani[a], Santanu Chaudhury[a,*], Subhashis Banerjee[b]

[a] *Department of Electrical Engineering, IIT Delhi, New Delhi-110016, India*
[b] *Department of Computer Science Engineering, IIT Delhi, New Delhi-110016, India*

Received 6 December 2000; accepted 7 October 2002

Abstract

This paper is concerned with the problem of recognition of dynamic hand gestures. We have considered gestures which are sequences of distinct hand poses. In these gestures hand poses can undergo motion and discrete changes. However, continuous deformations of the hand shapes are not permitted. We have developed a recognition engine which can reliably recognize these gestures despite individual variations. The engine also has the ability to detect start and end of gesture sequences in an automated fashion. The recognition strategy uses a combination of static shape recognition (performed using contour discriminant analysis), Kalman filter based hand tracking and a HMM based temporal characterization scheme. The system is fairly robust to background clutter and uses skin color for static shape recognition and tracking. A real time implementation on standard hardware is developed. Experimental results establish the effectiveness of the approach.
© 2003 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Hand gesture; Hidden Markov model; Contour tracking; Real time system

1. Introduction

Gesture recognition is important for developing an attractive alternative to prevalent human–computer interaction modalities. In this paper we have focused on the problem of recognition of dynamic hand gestures. We have considered single handed gestures which are sequences of distinct hand shapes. A given hand shape can undergo motion and discrete changes. However, continuous deformations are not permitted. These gestures are distinguished on the basis of hand shapes involved and the nature of motion. We have developed a real time recognition engine which can reliably recognize these gestures despite individual variations. The engine also has the ability to detect start and end of gesture sequences in an automated fashion.

Pavlovic et al. [1] present an extensive review of the existing techniques for interpretation of hand gestures. A large variety of techniques have been used for modeling the hand. Rehg and Kanade [2] designed a system called *Digit Eyes* which modeled the hand as 3D jointed cylinders. Appearance based models exploit properties of the image of the hand. A deformable 2D template of the human hand was used in Reference [3]. An approach based on the 2D locations of fingertips and palms was used by Davis and Shah [4]. Dynamic gestures have been handled using tracking framework. Blake et al. [5] have developed a method for contour tracking using a Kalman filter and an affine contour deformation model. Michael Isard [6] has established a stochastic framework for tracking curves in visual clutter using a Bayesian random sampling algorithm (CONDENSATION—conditional density propagation-based tracking) and have shown its application in hand tracking. Bobick and Wilson [7] have developed a state-based technique for representation and recognition of gestures in which they define the gesture as a sequence of states in a measurement or configuration space. Using techniques for computing a prototype trajectory of an ensemble

* Corresponding author. Tel.: +91-11-659-1081; fax: +91-11-696-6606.
E-mail addresses: aramamoorthy@biomorphic.com (A. Ramamoorthy), namrata@cfar.umd.edu (N. Vaswani), santanuc@ee.iitd.ernet.in (S. Chaudhury), suban@cse.iitd.ernet.in (S. Banerjee).

0031-3203/03/$30.00 © 2003 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.
doi:10.1016/S0031-3203(03)00042-6

of trajectories, they have developed methods for defining configuration states along the prototype and for recognizing gestures from an unsegmented, continuous stream of sensor data. Human–computer interaction using hand gestures has been studied by a number of researchers like Starner and Pentland [8], and Kjeldsen and Kender [9]. Use of inductive learning for hand gesture recognition has been explored in Reference [10]. An HMM based approach for gesture spotting and recognition has been proposed in Reference [11]. Yoon et al. [12] have proposed a recognition scheme using combined features of location, angle and velocity.

In this work we have developed a HMM based gesture recognition system which uses both the temporal and shape characteristics of the gesture for recognition. Unlike most other schemes, our system is robust to background clutter, does not require special gloves to be worn and yet runs in real time. Use of HMMs for gesture recognition is not new but the methodology adopted for combining shape & temporal characteristics in a HMM framework is the new contribution of this work. Use of both hand shape and motion pattern is a novel feature of this work. A scheme for recognizing hand shapes has been formulated based upon MacCormick and Blake's [13] work on Contour Discriminants. We have used a Kalman filter for hand tracking to obtain motion descriptors for the HMM. A *new* technique for rejecting outliers (false contour points chosen because of poor lighting) has been developed by introducing purposive assumptions. Schemes have been devised for detecting start, end and in-between shape changes in the gesture for implementation of a real-time system. An initial version of this work was reported in Reference [14].

The organization of the rest of the paper is as follows. In Section 2 we present an overview of the recognition engine. In Section 3, we discuss our tracker framework. We discuss our gesture recognition framework in Section 5. Section 5 describes details of our real-time implementation. In Section 6, we apply the ideas in this paper to an application—remote control of a moving robot. The next section presents results of extensive experimentation with our system. We summarize the contributions of this work and identify areas for further work in the concluding section.

2. Overview of the gesture recognition scheme

In this paper we have considered single handed dynamic gestures. A gesture is composed of a sequence of epochs. Each epoch is characterized by the motion of distinct hand shapes. Our recognition engine identifies a gesture based upon the temporal sequence of hand shapes made and motion patterns followed.

The recognition process involves tracking of the gesturer's hand. The hand shape being tracked is recognized by the contour discriminant based approach. The tracker also detects discrete changes in shape. When a shape change is detected, the hand shape recognizer is invoked for identifying the new hand shape and accordingly it re-initialize the tracker. The tracker also outputs the encoded motion pattern at regular intervals. A HMM based approach uses shape and motion information for recognition of the gesture. The basic algorithmic framework for our recognition engine is the following:

1. Detect hand for boot-strapping the tracker.
2. Recognize the starting hand-shape and initialize tracker with the template of the recognized hand-shape.
3. While hand is in view
   (a) Track the hand and output encoded motion information until shape change is detected.
   (b) Recognize the new shape and initialize the tracker with template of the recognized shape.
4. Using HMM find the gesture which gives maximum probability of occurrence of observation sequence composed of shape templates and motion information.

In the next section we describe in detail the tracker.

3. Tracker framework

In our gesture representation scheme, the hand has been considered as a smooth 2D curve. The moving hand has been tracked using a simplified form of the contour tracker developed by Blake et al. [5]. The tracker, in this case, consists of a Kalman filter based estimator for a piecewise smooth image plane curve in motion:

$$r(s,t) = ((x(s,t), y(s,t)).$$

The curve representation is in terms of B-Splines. Some advantages of the B-Spline curve representation over the traditional snakes are local control, implied continuity, and compact representation. $P(t)$, the position vectors along a B-Spline curve of order k and number of control points $n+1$ is given by

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t) \quad t_{\min} \leqslant t < t_{\max}, 2 \leqslant k \leqslant n+1, \quad (1)$$

where the B_i are the position vectors of the $n+1$ defining polygon vertices and the $N_{i,k}$ are the normalized B-Spline basis functions given by the Cox–DeBoor recursion formulae.

$$N_{i,1}(t) = 1 \quad \text{if } x_i \leqslant t < x_{i+1} \quad (2)$$

$$= 0 \quad \text{otherwise}, \quad (3)$$

$$N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}}, \quad k \neq 1. \quad (4)$$

The values of x_i are elements of a knot vector satisfying the relations $x_i \leqslant x_{i+1}$ and the parameter t varies from $t_{\min}$ to $t_{\max}$ along the curve $P(t)$. The function $P(t)$ is a polynomial

of degree $k-1$ on each interval $x_i \leqslant t < x_{i+1}$. $P(t)$ and its derivatives of order $1, 2, \ldots, k-2$ are all continuous over the entire curve.

The moving hand is approximated as a planar rigid shape under the assumption that the fingers are not being flexed and the perspective effects are not significant. Since continuous change in shape has not been permitted for the gestures under consideration, for gestural epochs with fixed hand shape, hand motion can be modeled as rigid body motion. A good approximation to the curve as it changes over time can be obtained by specifying Q, a linear vector valued function of the B-Spline coordinates (X, Y). The Q parameters specify the transformation (Euclidean in our case) of (X, Y) in terms of shape. Since rotation of the hand is assumed to be negligible, the Q vector consists of X- and Y- translation and a common scale parameter. The transformation between the state Q and the control points (X, Y) is given by

$$\begin{pmatrix} X \\ Y \end{pmatrix} = WQ + \begin{pmatrix} \bar{X} \\ \bar{Y} \end{pmatrix},$$

$$Q = M\left[\begin{pmatrix} X \\ Y \end{pmatrix} - \begin{pmatrix} \bar{X} \\ \bar{Y} \end{pmatrix}\right].$$

The matrices M and W are defined in terms of the shape template $(\bar{X}, \bar{Y})$. In our case of the hand since we assume it be a planar shape and allow only X and Y translation and scaling, just three degrees of freedom are required to describe the possible shapes of the curve. The space of possible Q-vectors is expressible as a three-dimensional linear subspace of Q-vectors, and a basis for this subspace is

$$\beta = \left\{\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} \bar{X} \\ \bar{Y} \end{pmatrix}\right\}.$$

In that case the matrix M and W converting B-Spline control points (X and Y) to and from the three vector Q can be defined in terms of the template, as follows:

$$W = \begin{pmatrix} 1 & 0 & \bar{X} \\ 0 & 1 & \bar{Y} \end{pmatrix},$$

$$M = (W^{\mathrm{t}}W)^{-1}W^{\mathrm{t}}.$$

3.1. Kalman filter for tracking

The control points to be tracked in an image specify the shape template which can undergo Euclidean transformations specified by the matrix Q. A *uniform discrete time motion model* [5] is assumed. The state X for the discrete time model is described as

$$X_n = \begin{pmatrix} Q_{n-1} \\ Q_n \end{pmatrix}.$$

The state equation for uniform motion is

$$X_{n+1} - \bar{X} = A(X_n - \bar{X}) + \begin{pmatrix} 0 \\ W_n \end{pmatrix},$$

where

$$A = \begin{pmatrix} 0 & I \\ A_0 & A_1 \end{pmatrix},$$

$$\bar{X} = \begin{pmatrix} \bar{Q} \\ \bar{Q} \end{pmatrix}.$$

Here, $\bar{Q}$ denotes an estimation of the Q matrix.

For uniform motion,

$$A_0 = -I,$$

$$A_1 = 2I.$$

The Kalman Filter parameters are:

- The state update matrix A.
- The observation vector Q_n.
- The measurement noise W_n is modeled as a zero-mean uncorrelated Gaussian noise.

We discuss our measurement model in the following section.

3.2. Measurement model: Greedy algorithm

Given an estimate of the contour $r(s,t)$, the visual measurement process consists of casting rays along the normal $n(s,t)$ to the predicted curve by measuring the position of the first pixel of hand color along the normal. The following issues are considered in this context:

- Curve-fitting: Since we have used a 4th order B-Spline representation of the curve, each span of the curve is influenced by only 4 control points out of which only one has maximum influence on the span. Therefore the measurement consists of searching for that control point along its normal direction.
- Chromatic invariance: The tracker is very sensitive to light intensity variations since the measurement model is based on a color based point search. To avoid wrong matches due to non-uniform lighting, the color coordinates are subjected to chromatic invariance transformation followed by *L2 Normalization* as follows:

$$r = (R-G)^2/((R-G)^2 + (G-B)^2),$$

$$g = (G-B)^2/((R-G)^2 + (G-B)^2).$$

The transformed (r, g) coordinates are then used in the curve fitting algorithm discussed below.

3.3. The curve-fitting algorithm

The curve-fitting algorithm is used for boot-strapping the gesture recognizer and in the measurement process during tracking. This algorithm uses a rough estimate of the mean color of human skin. This estimate for the present system has been obtained from a large number of training examples.

On initiation, the user is shown a small rectangle in the center of the screen. To begin execution, the user is required to place his hand roughly in the center of that rectangle. Once this is done, the system recognizes the presence of a hand and starts the execution. The presence of the hand is detected in the following way. The system keeps a count of the number of pixels in the rectangle which fall within the calculated variance of the stored mean color. Once this count crosses a certain threshold, the following algorithm takes over.

1. Calculate the mean color of the presented hand using a small window (10×10) located near the centre. Also calculate the variance of the pixel values over that region.
2. For each given control point:-
   - Find whether the control point lies within or outside the hand region by checking its intensity value. If the intensity lies within the calculated variance of the mean, then classify it as a hand point, else as an outside point.
   - If the control point lies within the hand, then search along the direction of the outward normal to the contour till you reach a point which does not belong to the hand pixel set.
   - If the control point lies outside the hand, then search along the direction n of the inward normal to the contour until you reach a pixel belonging to the hand.

3.4. Stability of the tracker

The tracker in its raw form is quite unstable and a few wrong measurements tend to deform the contour so that it loses track. Apart from assuming an Euclidean motion model, the following three assumptions were introduced to improve tracker stability.

- *Rigid body assumption: constancy of slopes at control points*
  Due to non-uniform lighting, certain control points find incorrect matches. This leads to loop formations and contour deformation. To correct this, we use the fact that since the hand is a rigid body, the slope of the tangent to the curve at the control point remains the same as that of the initial template.

  The slope of the point $P_k = (x_k, y_k)$ with respect to the previous point P_{k-1} is

  $$\theta_k = \tan^{-1}((y_k - y_{k-1})/(x_k - x_{k-1}))$$

  and with respect to the next point P_{k+1} is

  $$\theta_{k+1} = \tan^{-1}((y_{k+1} - y_k)/(x_{k+1} - x_k)).$$

  If either of the slope values θ_k and θ_{k+1} is within $\pm 30^\circ$ of the original slope values, the point is correct i.e.

  $$P_k^c = P_k.$$

  Else it is set to the point of intersection of the normal at P_k and the line joining P_{k-1} and P_{k+1} as follows. The line joining P_{k-1} and P_{k+1} is taken as the tangent at the corrected point P_k^c. Its slope is given by

  $$\theta_{\text{tangent}} = \tan^{-1}((y_{k+1} - y_{k-1})/(x_{k+1} - x_{k-1})).$$

  The difference in angle between the tangent at the correct point P_k^c and the slope(tangent) at P_k is given by

  $$\Delta\theta = \theta_{\text{tangent}} - \theta_k.$$

  The shift along the tangent required to reach the correct point is given by

  $$\delta = \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2}\,\sin\Delta\theta.$$

  The coordinates of the correct point $P_k^c = (x_k^c, y_k^c)$ are given by

  $$x_k^c = x_k + \delta * \sin\theta_{\text{tangent}},$$

  $$y_k^c = y_k + \delta * \cos\theta_{\text{tangent}}.$$

- *Prediction on alternate frames*
  Since hand motion is jerky at times, using the observation at every frame for Kalman filter state prediction made the system unstable. So prediction was done on alternate frames.
- *Delay in start of prediction*
  To allow the hand to stabilize, the first 30 frames were not used for prediction.

An example of tracking without the proposed assumptions is shown in Fig. 1. The deformed contour is shown in the second frame which is the seventh frame in the gesture sequence. An example of improved tracking of the open hand with the assumptions is shown in Fig. 2.

3.5. Shape change detection

The centroid of the contour points is calculated and the variance of the contour points from the centroid is obtained. Different shapes will have different values of this variance at one given scale. We calculate the ratios of these variance values at this given scale and use it as a value for comparison. When there is a shape change, the ratio of the variance value of the predicted and observed contours is expected to be above the calculated value.

Fig. 1. Unstable tracking.

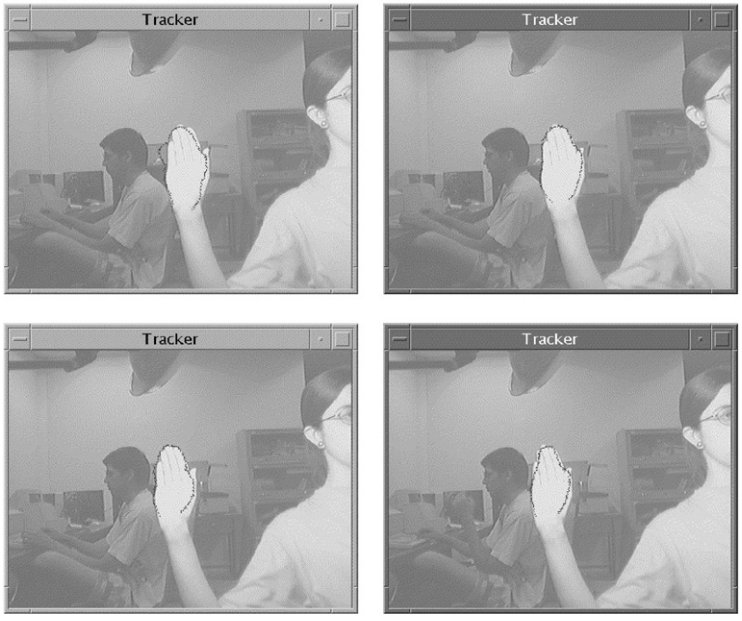

Fig. 2. Open hand tracking: moving right.

When a shape change is detected,

- The new shape is recognized using static shape recognition.
- The Kalman filter is re-initialized.
- The transformation matrix(W) between the Kalman filter state (translation and scaling) and the control points is re-calculated using the new template.
- Slopes of the new template are calculated again and stored.

4. Gesture recognition

In our work, we have considered a gesture to be characterized by the shape of the hand and the nature of motion of the hand. There are thus two components which contribute towards the recognition of a gesture.

1. Static Shape Recognition.
2. Recognition of the motion sequence.

4.1. Contour discriminant based approach for static shape recognition & tracker initialization

For recognition of hand shape, we have adapted the idea of using a probabilistic contour discriminant for object localization, proposed in Reference [13]. The approach applies to objects with complex outlines such as the human hand. It is robust to lighting changes and works well in

cluttered backgrounds. A class of hand shapes is modeled with a configuration space of specified dimension-X-translation, Y-translation and uniform scaling. Because of the measurement methodology adopted, only one-dimensional image processing is needed (search along the normal for a pixel of hand color). The recognition process is accomplished by trying to evaluate the similarity of the given hand shape to contours of the different classes.

4.1.1. The class model and prior

The hand shape classes are described by their outline, which is modeled as a B-Spline curve as is described in Reference [15]. Given an image containing one of the given hand shapes, we first sample from the space of possible configurations for a given hand shape class and adopt the following measurement method:

1. Cast normals (called measurements lines) onto the image at pre-specified points around the contour.
2. Apply a 1-D feature detector along each measurement line. The distance from a feature to the contour is called the innovation of the feature. In the general case we can have more than on feature point on a given normal line. In our case however only one feature was detected per line. The feature we were trying to detect was the presence of the first pixel of hand color.

One of the first things to be done in order for the above approach to work is, to create the space of possible configurations for each hand shape class. This is done as follows:-

1. Initially a contour corresponding to each of the static shapes is made by specifying through mouse clicks, control points on the boundary and then fitting a B-Spline curve through them. The initial template is uniformly translated in both X & Y directions and these contours are added to the prior.
2. We also generated scaled up and scaled down versions of the initial template and also added them to the prior.
3. Rotations of the initial template ($< \pm 30^\circ$) were also added to the prior.

4.1.2. Recognition

Recognition process involves following steps:

1. For recognition of a new hand initially the user has to bring his hand into the center of a 120×70 window. The mean and the variance of the hand color are estimated based on a small area of 10×10 pixels.
2. The above mentioned measurement process is now applied to prior contours of each class of hand shapes. The measurement process involves a simple search for the color of the hand along the normal to the contour at each of the control points.
3. The number of control points which do not find matches are recorded for each contour. Also the amount of distortion which each contour undergoes in trying to lock onto the hand is recorded as a sum-squared error measure from the original contour.
4. Now, the values of distortion will be low for a contour which belongs to the actual class of the new hand shape. By the above-mentioned process, a large number of contours, corresponding to the rotated, translated and scaled versions of the initial template will also have low values of distortion. On the other hand, the contours corresponding to an incorrect class shall have a large value of distortion after locking onto the given hand shape. Thus the given hand shape is declared as that class which has the lowest value of distortion among the three classes.
5. For initializing the tracker we find the closest contour match within the recognized hand shape class. This is done on the basis of the measure of distortion also calculated above. Thus the contour which has to undergo the least distortion to lock onto to the hand must be the one which matches most closely to the actual contour of the hand and thus can be used for tracker initialization (Figs. 3–5).

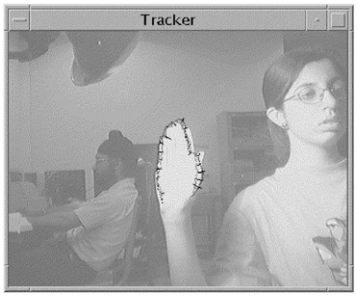

Fig. 3. Static shape: open hand.

Fig. 4. Static shape: closed hand.

4.2. Hidden Markov model based recognition

Since the gestures considered are dynamic processes, we need a mechanism to recognize gestures using their temporal characteristics. The recognition scheme must be robust and should accommodate variations in the attributes of a

Fig. 5. Static shape: deer hand.

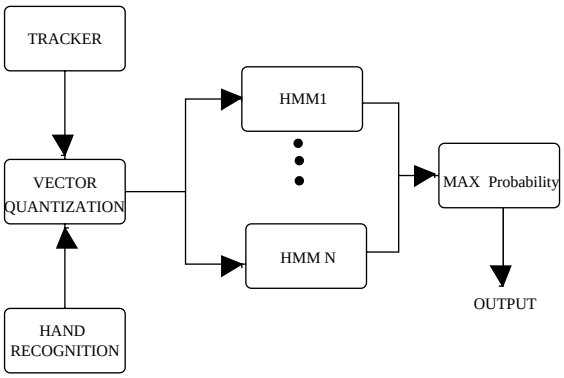

Fig. 6. Different inputs and outputs to HMMs.

gesture. We have adopted the hidden Markov model [16]-based approach.

The tracker provides temporal characteristics of the gesture. The output of the tracker is used for classifying the nature of the motion. Shape classification information is provided by the contour discriminant-based classifier. These symbolic descriptors are used as input for the HMM.

The major steps involved in recognition of gestures using HMM are:

1. Extraction of symbolic descriptors of the gesture at regular intervals from the tracker and hand shape classifier.
2. Training of HMMs by the sequence of symbolic descriptors corresponding to each of the gesture.
3. For an unknown gesture find the model which gives maximum probability of occurrence of the observation sequence generated by the gesture, as depicted in Fig. 6.

For each gesture there is a HMM. Each HMM is trained by the symbol sequences obtained from the training set of each of the gesture.

4.3. Extraction of symbolic descriptors

The state, or "Q" matrix of a Kalman filter has three elements which corresponds to three different transformations that hand can have i.e., X-translation, Y-translation and scaling. If the hand is moving to (say $+x$ direction) then only one component (corresponding to x-translation) of "Q" will be significant, all others will be nearly equal to zero. Thus at each time instant we can have information about the motion by "Q" matrix. For generation of symbols (at time $= t$) difference of elements of "Q" matrix at time t and at start of gesture i.e., at time $t = 0$ is taken. Thus we have parameters available which corresponds to the position of hand with respect to start. The parameters which are positive assigned "$+1$", negative ones correspond to "-1", and those which have value near to zero are taken as equal to zero. These temporal descriptors are generated after every 6 frames.

This symbol generation scheme can account for local variations of the motion and provide consistent set of descriptors. For example, consider a gesture in which hand first goes to right, returns back and then goes to the left. In this gesture there are mainly two parts, one is when the hand is moving to right, and other when it is moving left beyond the original position. In the first part there is positive X-displacement. As the hand moves to right, only one of the component of "Q" matrix will be positive, all others will be equal to zero most of the time. It is the same case when the hand is moving left from the original position. In this case there is negative X-displacement and the symbols generated are "-1" for X-displacement, zero for other motion parameters.

The hand shape classifier provides a unique identification code for each shape class. This, combined with motion descriptors, provides the complete symbolic description of the image sequence. The shape classifier is invoked during initialization and at each sequence segmentation point.

5. Real time implementation

For real time implementation, the gesture recognition task was divided into two threads: *Grabber* and *Tracker*, Grabber grabbed images at rate of 25 frames/s and stored the images in buffer. The size of buffer was kept at 3 (i.e., 3 images can be kept). The tracker read images from this buffer and did tracking and recognition. The grabber and tracker operated as synchronised threads. Fig. 7 shows the processing scheme used. The implementation was done on a SUN Ultra SPARC workstation.

Timing analysis of the tracking-cum-recognition process revealed that measurement module of the Kalman filter was causing the timing bottleneck. From Eq. (1) we find that the value of $N_{i,k}(t)$ is independent of the position of the control points i.e. B_i. To reduce the computation we calculated the value of $N_{i,k}(t)$ for discrete values of t and stored it in an array. Now getting a B-Spline curve became a $O(n)$ function. This optimization improved the performance of the tracker. Also note that $N_{i,k}(t)$ is nonzero only for 4 values of i (for order 4) so instead of summing it from $i = 1$ to $(n + 1)$ we are summing only for 4 values.

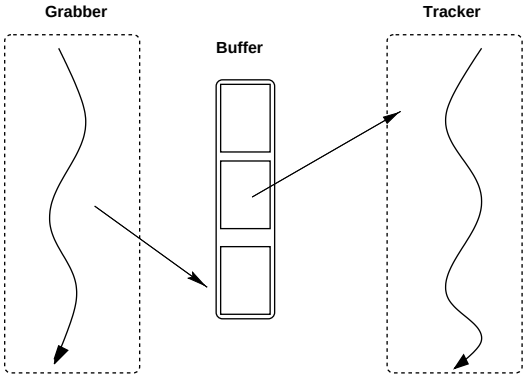

Fig. 7. Grabbing and processing of images using threads.

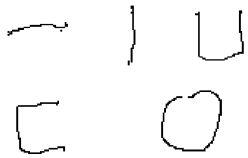

Fig. 8. Gesture motion patterns.

In the final implementation, we achieved a tracking rate of around 20 frames/s.

6. Application to remote robot control

For the purpose of experimentation we selected five different dynamic gestures. These gestures were logically associated with five different functions needed for robot motion. The gesture number was sent over the network to a Linux machine to which a robot was connected. The five different instructions given to the robot through gestures are:

1. Move forward.
2. Move forward then right.
3. Move forward then left.
4. Move backward then right.
5. Move backward then left.

Gestures corresponding to above instructions are:

1. Closed to open forward: At starting time hand is closed and it moves towards the camera then it stops and hand shape changes from closed to open (Fig. 9). Now this open hand moves back i.e. away from the camera. This complete gesture is for the robot to move forward.
2. Closed to open right: The hand is closed during start. The hand then moves towards right, stops and changes hand shape to open and then returns with the open hand (Fig. 10).
3. Closed to open left: The hand is closed during start. The hand then moves towards left, stops and changes hand shape to open and then returns with the open hand (Fig. 11).
4. Open to closed right: The hand is open during the start. The hand then moves towards right, stops and changes hand shape to closed and then returns with the closed hand (Fig. 12).
5. Open to closed left: The hand is open during the start. The hand then moves towards left, stops and changes hand shape to closed and then returns with the closed hand (Fig. 13).

In our application, camera is always grabbing images and is sending the image to "process_image" function. The "process_image" function does recognition and tracking when user is delivering a gestural instruction, otherwise it just ignores the images. So now there are two problems remaining to be solved

1. To decide when a user has started a gesture.
2. To decide completion of a gesture.

For the solution of Problem 1, we made a small rectangle in the middle of the images grabbed by the camera. In each image this rectangular area is searched for the presence of hand colored pixels. If more than 75% of the pixels are within 10% of the mean normalized hand color then it is assumed that gesture has started, otherwise tracker ignores the image. In other words, user is expected to bring his hand to the designated region for initiating a gestural action.

For the end of gesture problem our solution is whenever user wants to end the gesture he can remove his hand away from the camera so that it does not appear in the designated region. Every 10 frames of the cycle the percentage of pixels in the window lying within $\pm 10\%$ of the mean normalized hand color are checked. If this falls below 20%, the end of gesture is declared and the recognition of the gesture takes place.

7. Experimental results

In this section we present the results of recognition of gestures obtained from different individuals. We discuss results of each of the steps involved before presenting the overall results.

All the results obtained were under the following assumptions:-

1. The gesture is properly illuminated with a uniform light source.
2. Initially at the start of the gesture the static hand is placed roughly in the fovea and kept stationary for some time till the contour is initialized.
3. The user should keep his hand stationary during the time when the re-initialization takes place.

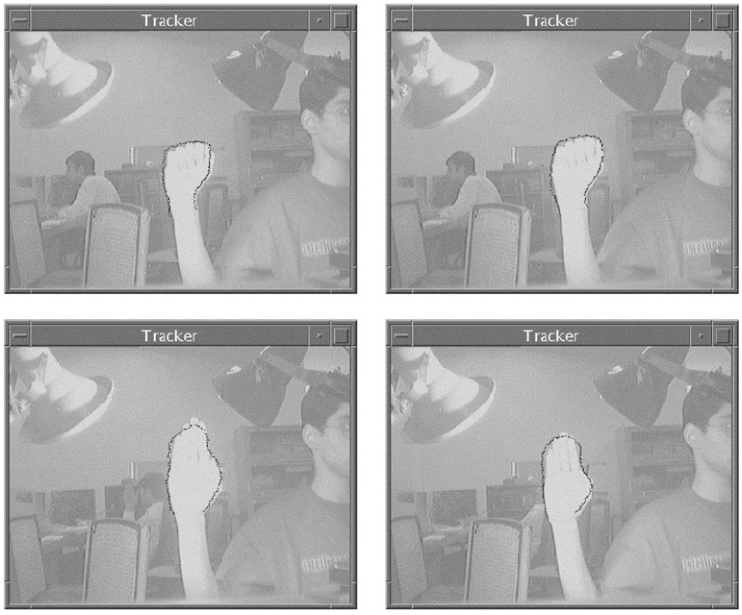

Fig. 9. Closed to open forward.

Fig. 10. Closed to open right.

4. While tracking the user should avoid sudden, jerky movements and should not rotate the hand too much.

7.1. Results of tracking

- The complete system is working at a frame rate of about 25 frames/s. The grab rate of the camera is itself 25 Hz. With this frame rate the Kalman filter works properly in most of the cases if the user keeps the above mentioned assumptions in mind.
- The re-initialization module which comes into play when the user changes hand shape also works robustly under the above assumptions.

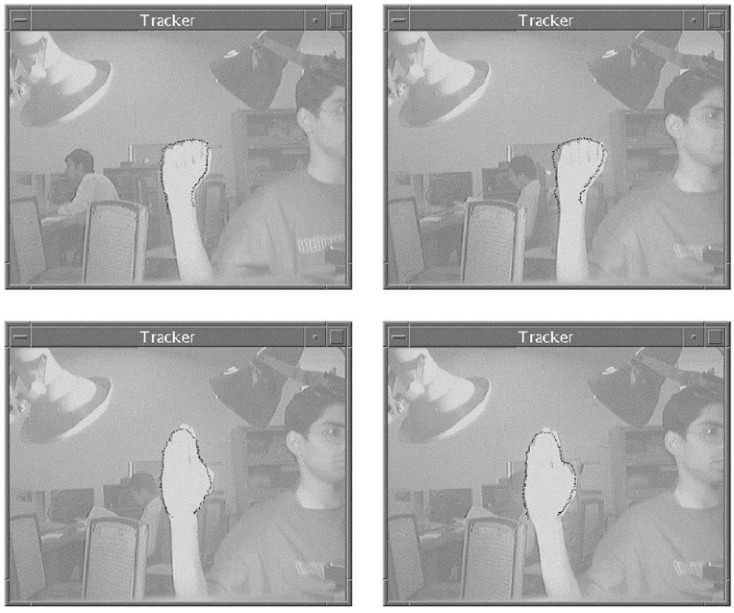

Fig. 11. Closed to open left.

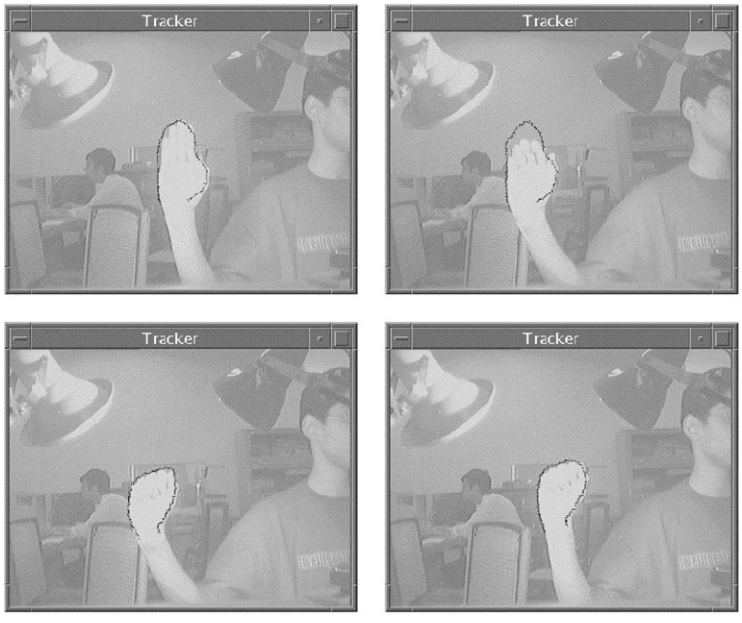

Fig. 12. Open to closed right.

The cases when the tracker loses track are as follows:-

- If the hand moves very fast:- The Kalman filter searches for a hand pixel in a certain search space (20 pixels on the right & 20 pixels on the left). If the hand moves so fast that the Kalman Filter does not find a hand pixel, then the tracker starts going out of track, because the translation & scale parameters get distorted values.
- If initialization is not good:- The tracker fails in this case because the state estimation is not proper during subsequent frames. Consequently, the error keeps on increasing resulting in loss of track.

Fig. 13. Open to closed left.

- If hand is not well illuminated:- In this case the Kalman Filter tends to get lots of false matches, which result in either repeated re-initialization or straightaway going out of track.
- Shape change detection fails:- When the shape is changed gradually, the variance error does not change suddenly and thus shape change cannot get detected.
- Re-initialization fails:- In case of open to closed hand shape change, recognition fails while moving towards the camera because a scaled up version of the closed hand is very similar to the open hand.
- Too frequent re-initialization also leads to re-initialization failure because the Kalman filter parameters have not stabilized and re-initialization takes place in a wrong window.

7.2. Results of static shape recognition

The three static hand poses which have been chosen for recognition are open hand (Fig. 3), closed hand (Fig. 4), deer hand (Fig. 5).

The results are as follows.

- The recognition of the "open hand" is correct in 95% of the cases. The "closed hand" and "deer hand" works in 80% of the cases.
- The recognition works better if only a small part of the wrist is exposed.
- Also if the scale i.e. the size of the user's hand is significantly different from the initial template's size, then too the recognition tends to fail.

The contour discriminant based approach is quite robust to translations and rotations of the hand unlike other recognition techniques like principal component analysis [29], as these were made a part of the prior model.

7.3. Results of HMM based gesture recognition

- HMM symbols (static shape, X-translation, Y-translation and Scale) were generated every six frames. Static shape can take 3 values, X-translation, Y-translation and Scale with respect to the initial template were also quantized to generate 3 symbols each. Thus total number of symbols was $3^4 = 81$.
- The HMM's were modeled as Left-Right HMM's with four states and an out degree of 3.
- HMM's corresponding to the following five gestures were trained.
  1. Closed-open-forward: Starting with closed hand, move forward and return backward with open hand.
  2. Closed-open-right: Starting with closed hand, move right and return with open hand.
  3. Closed-open-left: Starting with closed hand, move left and return with open hand.
  4. Open-closed-right: Starting with open hand, move right and return with closed hand.
  5. Open-closed-left: Starting with open hand, move left and return with closed hand.

Table 1
Table showing recognition rate for different gestures

User	*C-O-F*	*C-O-R*	*C-O-L*	*O-C-R*	*O-C-L*
1	19/20	18/20	17/20	17/20	19/20
2	18/20	17/20	18/20	17/20	17/20
3	8/10	6/10	7/10	7/10	7/10
4	9/10	7/10	6/10	7/10	6/10
5	8/10	7/10	8/10	8/10	8/10

Experimentally the best recognition results were obtained when the number of states was taken as 6 and the out degree as 3. Different users were asked to perform the gestures and the number of correctly recognized gestures is tabulated in Table 1. HMM's were trained with the first two users.

There was a tendency for the HMM module to get confused between complementary gestures eg. closed-open-left and closed-open-right. In all the cases the duration of the gestures was different and hence the recognition scheme is independent of the length of the gesture.

7.4. Comparison with other approaches

Not many vision based approaches have been reported for real-time recognition of the class of gestures considered in this work. When we compare our approach some of the recently reported approaches which work in real-time like [12,11], we find that hand shape has not been explicitly considered as a possible feature. The motion patterns in the gesture vocabulary are more complex than what we have reported here. We have found that use of hand shape makes it easier for the gesturer to remember commands. We have experimented with five gestural patterns given in [12] as shown in Fig. 8. As hand shape we have used palm. These motion patterns were generated by the movement of open hand—palm. The observables corresponding to the states in HMM were distinguished only on the basis of encoded motion patterns. After training HMM with gestures made by two users, we have experimented with inputs from five users under similar experimental conditions as reported earlier in this section. Out of a total of 100 gestures considered, our system failed to recognize three sequences. This result is better than those reported in the previous subsection. The reason for this is that contour discriminant based static shape recognition is the most difficult aspect of our recognition engine due to inter-personnel variations and non-rigid deformations of the hand shape.

8. Conclusion

We have developed a robust system for the gesture based control of a robot. The system is fully automatic and bootstraps itself. It works by the real-time tracking of the hand contour & the recognition of the gestures. It works well even in the presence of background clutter. The advantage of the system lies in the ease of its use. The user does not need to wear a glove, neither is there the need for a uniform background. The original contributions of this work are, a novel technique for combining shape and motion parameters, and system level techniques and optimizations for the achievement of real-time gesture recognition. Another important contribution is the incorporation of changing hand shapes, i.e. the flexibility given to the user to change hand shapes in between the gesture. This enables us to deal with a larger set of gestures.

Acknowledgements

We would like to acknowledge Motilal Agarwal and Shyam Sadhwani for their valuable time and ideas. Thanks are also due to Sumantra Dutta Roy for his reviews and tips on the presentation of the work.

References

[1] V.I. Pavlovic, R. Sharma, T.S. Huang, Visual interpretation of hand gestures for human–computer interaction: a review, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 677–695.
[2] J.M. Rehg, T. Kanade, Digiteyes: vision-based human hand tracking, Tech. Rep. CMU-CS-93-220, CMU, 1993.
[3] T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, Active shape models—their training and application, Comput. Vision Image Understand. 61 (1) (1995) 38–59.
[4] J. Davis, M. Shah, Recognizing hand gestures, in: Proceedings of the European Conference on Computer Vision, ECCV, 1994, pp. 331–340.
[5] A. Blake, M. Isard, D. Reynard, Learning to track the visual motion of contours, Artif. Intell. 78 (1–2) (1995) 179–212.
[6] M.A. Isard, Visual motion analysis by probabilistic propagation of conditional density, Ph.D. Thesis, Robotics Research Group, University of Oxford, September 1998.
[7] A.F. Bobick, A.D. Wilson, A state-based approach to the representation and recognition of gesture, IEEE Trans. Pattern Anal. Mach. Intell. 19 (12) (1997) 1235–1337.
[8] T.E. Starner, A. Pentland, Real-time American sign language recognition from video using hidden Markov models, Tech. Rep. 375, MIT Media Lab Vision and Modelling Group, 1995.

[9] R. Kjeldsen, J. Kender, Visual hand gesture recognition for window system control, in: IWAFGR, 1995, pp. 184–188.
[10] M. Zhao, F.K.H. Quek, Xindong Wu, Rievl: recursive induction learning in hand gesture recognition, IEEE Trans. Pattern Anal. Mach. Intell. 20 (11) (1998) 1174–1185.
[11] Hyeon-Kyu Lee, Jin H. Kim, An hmm-based threshold model approach for gesture recognition, IEEE Trans. Pattern Anal. Mach. Intell. 21 (10) (1999) 961–973.
[12] Ho-Sub Yoon, Jung Soh, Younglae J. Bae, Hyun Seung Yang, Hand gesture recognition using combined features of location, angle, velocity, Pattern Recognition 34 (2001) 1491–1501.
[13] MacCormick, Blake, A probabilistic contour discriminant for object localization, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 1998, pp. 390–395.
[14] M. Agrawal, S. Sadhwani, S. Chaudury, S. Banerjee, Recognition of dynamic hand gestures, in: S. Chaudhury, S.K. Nayar (Eds.), Computer Vision, Graphics and Image Processing: Recent Advances, Viva Books Private Limited, 1999, pp. 179–184.
[15] D.F. Rogers, J.A. Adams, Mathematical Elements of Computer Graphics, WCB/McGraw-Hill, New York, 1990.
[16] L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE 77 (2) (1989) 257–286.

About the Author—ADITYA RAMAMOORTHY did his B.Tech. in Electrical Engineering from I.I.T, Delhi, India. He is currently pursuing Ph.D. at UCLA in coding theory. His research interests are in the areas of signal processing and computer vision.

About the Author—NAMRATA VASWANI did his B.Tech. in Electrical Engineering from I.I.T, Delhi, India. She is currently pursuing Ph.D. at University of Maryland, USA in computer vision. Her research interests are in the areas of signal processing and computer vision.

About the Author—SANTANU CHAUDHURY did his B.Tech. in Electronics and Electrical Communication Eng. and Ph.D. in Computer Science and Eng. from I.I.T, Kharagpur, India. Currently, he is a professor in the department of Electrical Eng. at I.I.T, Delhi. His research interests are in the areas of Computer Vision, Artificial Intelligence and Multimedia Systems.

About the Author—SUBHASHIS BANERJEE did his B.E. from Jadavpur University, Kolkata and M.Tech. & Ph.D. from IISc, Bangalore, India. Currently, he is a professor in the department of Computer Science and Eng. at I.I.T, Delhi. His research interests are in the areas of Computer Vision, Computer Graphics, Embedded Systems.