Journal of Visual Languages and Computing 28 (2015) 39–55

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

Hand pose recognition from monocular images by geometrical and texture analysis ☆

M.K. Bhuyan [a,b,*], Karl F. MacDorman [a], Mithun Kumar Kar [b], Debanga Raj Neog [c], Brian C. Lovell [d], Prathik Gadde [a]

[a] School of Informatics and Computing, Indiana University Purdue University 535 West Michigan St., Indianapolis, IN 46202, USA
[b] Department of Electronics and Electrical Engineering, Indian Institute of Technology, Guwahati 781039, India
[c] Department of Computer Science, University of British Columbia, Vancouver, BC, Canada V6T 1Z4
[d] School of Information Technology Electrical Engineering, University of Queensland, Brisbane QLD 4072 Australia

ARTICLE INFO

Article history:
Received 25 July 2014
Received in revised form
20 November 2014
Accepted 3 December 2014
Available online 20 December 2014

Keywords:
Hand model
Hand pose recognition
Homogeneous texture descriptors (HTD)
Human–computer interaction (HCI)

ABSTRACT

One challenging research problem of hand pose recognition is the accurate detection of finger abduction and flexion with a single camera. The detection of flexion movements from a 2D image is difficult, because it involves estimation of finger movements along the optical axis of the camera (z direction). In this paper, a novel approach to hand pose recognition is proposed. We use the concept of object-based video abstraction for segmenting the frames into video object planes (VOPs), as used in MPEG-4, with each VOP corresponding to one semantically meaningful hand position. Subsequently, a particular hand pose is recognized by analyzing the key geometrical features and the textures of the hand. The abduction and adduction movements of the fingers are analyzed by considering a skeletal model. Probabilistic distributions of the geometric features are considered for modeling intra-class abduction and adduction variations. Additionally, gestures differing in flexion positions of the fingers are classified by texture analysis using homogeneous texture descriptors (HTD). Finally, hand poses are classified based on proximity measurement by considering the intra-class abduction and adduction and/or inter-class flexion variations. Experimental results show the efficacy of our proposed hand pose recognition system. The system achieved a 99% recognition rate for one-hand poses and a 97% recognition rate for two-hand poses.

1. Introduction

The use of the human hand as a natural interface for human–computer interaction (HCI) motivates research on hand gesture recognition. Vision-based hand gesture recognition involves the visual analysis of hand shape, position, and/or movement. Accordingly, the basic aim of gesture recognition research is to build a system that can identify and interpret human gestures automatically. Such a system can be used for manipulation, such as controlling robots or other devices without any physical contact between the human and the interface. It can also be used for communication, such as conveying information through sign language [1,2]. A sign language recognition system must be able to recognize the changing poses of the hand.

Various methods have been proposed for recognizing accurate hand poses. However, because an articulated

☆ This paper has been recommended for acceptance by Shi Kho Chang.
* Corresponding author at: Department of Electronics and Electrical Engineering, Indian Institute of Technology, Guwahati 781039, India. Tel.: +91 361 258 2523; fax: +91 361 269 0762.
E-mail addresses: mkb@iitg.ernet.in (M.K. Bhuyan), kmacdorm@indiana.edu (K.F. MacDorman), mithun@iitg.ernet.in (M.K. Kar), debanga@cs.ubc.ca (D.R. Neog), lovell@itee.uq.edu.au (B.C. Lovell), pgadde@imail.iu.edu (P. Gadde).

http://dx.doi.org/10.1016/j.jvlc.2014.12.001
1045-926X/© 2014 Elsevier Ltd. All rights reserved.

model of the hand has many degrees of freedom, detecting the finger movements has remained a challenge. Current solutions rely either on data gloves or on computer vision. Data gloves are made cumbersome by the many cables connecting them to the computer. This can render human–computer interaction unnatural. Awkwardness in using gloves is overcome by using vision-based noncontact interaction techniques.

Vision-based methods for hand gesture recognition are performed by two major approaches, namely appearance-based and model-based representations. For hand pose analysis, model-based methods are most suitable because they provide accurate estimation of hand parameters without loss of spatial information [1]. Many methods for hand pose detection take geometrical properties of the hand and model the location and movement of the fingers [3–5]. Hand poses are analyzed by applying physiological constraints on hand kinematics and dynamics. These constraints include joint-angle limits on the extension, flexion, adduction, and abduction of metacarpophalangeal (MP) joints. The constraints determine the types of movements the hand can make. Matsumoto et al. implemented this kind of approach with a skeletal hand model [6]. They used a voxel model with an estimation algorithm for recognizing different hand poses with a multi-perspective camera system.

Some methods incorporate inverse kinematics and three-dimensional (3D) reconstruction techniques to estimate the 3D hand pose from single 2D monocular images. Lee et al. proposed an articulated model with hand kinematics constrains to reconstruct a 3D image from monocular view of the hand pose [7]. They took intra- and inter-finger constrains with 20 DOF for 3D model fitting of the 2D image. They only model the static hand poses including self occlusion with physical constrains into a 3D model. Guan et al. also estimated 3D hand pose parameters from a single image only using 12 DOF with an articulated model of the hand [8]. They used eight 2D projected features from the geometrical properties of the hand to retrieve the 3D hand pose. However, they only modeled the 3D pose from different views without considering simultaneous movements of fingers with adduction/abduction angle and flexion movements of fingers. Weng et al. developed a real-time motion capturing system using a 3D hand model with a state-based particle filter to estimate the motion of individual fingers [9]. But tracking fingers is computationally expensive given their high degree of freedom. Most of the model-based gesture recognition systems used 3D articulated hand models to characterize the parameters of the hand, *viz.*, 3D positions of all joint angles, fingertip positions, and their orientation [10,11]. These methods use 3D model fitting technique through minimization of a cost function based on extracted features.

To estimate kinematic parameters and perform 3D reconstruction, these algorithms involve high computational complexity. In addition, the use of multiple cameras and the depth sensors increases the overall system complexity [12–14]. Most of the existing methods make use of hand segmentation by skin color. Skin color offers an effective and efficient way to segment out hand regions. However, this approach is degraded by variation in skin tone, lighting conditions, and dynamic scenes.

To address some of these issues, we use the concept of object-based video abstraction for segmenting the frames into video object planes (VOPs), as used in MPEG-4, where the hand is considered as a video object (VO). A binary model for the moving hand is derived and is used for tracking in subsequent frames. The Hausdorff tracker is used for this purpose [15].

A notable advantage of our proposed scheme is its robustness to background noise. The tracker can track the hand as an object very efficiently even without adopting any kind of background filtering. Moreover, unlike tracking algorithms that use Kalman filters, the VOP generation algorithm does not require extra computation for scaling and rotation. In our algorithm, the concept of "shape change" can accommodate both the scaling and the rotation of the tracked video object in successive frames of the gesture video sequence. The only computation required for the shape change is the model update in each frame of the video sequence. The model update computation using the motion vector is much simpler computationally than the other computations, *viz.*, the affine transformation required for scaling and rotation.

Subsequently, we propose a method to recognize hand poses from the 2D images by modeling abduction, adduction, and/or flexion movements of fingers. These two types of hand poses are illustrated in Fig. 1.

We earlier proposed a method for recognizing abduction and adduction movements of the fingers [16]. The hand poses having only abduction and adduction movements of the fingers are modeled by a multidimensional Gaussian distribution. In this paper, another model is proposed to recognize flexion movements of the fingers, which is integrated with our earlier proposed model to recognize different hand poses. Hand poses having only flexion movements of the fingers are modeled by homogeneous texture descriptors (HTD) [17]. Representation of flexion movements of the fingers by analyzing only the texture information of the fingers is quite simple and straightforward as compared with 3D model-based methods. Our approach has advantages over previous methods in the sense that we can model both the abduction/adduction and flexion movements of fingers together from 2D monocular images. Finally, proximity measure is used to classify the input gestures by comparing the input

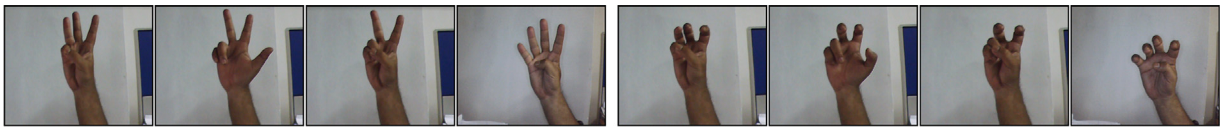
Fig. 1. First four hand poses show only the abduction and adduction movements of the fingers. Remaining four hand poses show only the flexion movements of the fingers.

features with the templates in the database. The overall block diagram of the proposed system is illustrated in Fig. 2.

The organization of the rest of the paper is as follows. Section 2 and Section 3 present our proposed hand pose recognition approach for abduction/adduction finger movements and for flexion movements, respectively. Section 4 reports experimental results. Finally, we draw our conclusion in Section 5.

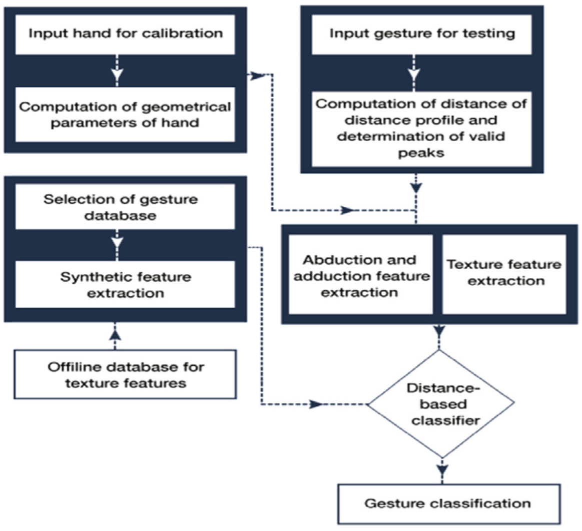
Input hand for calibration
Computation of geometrical parameters of hand
Input gesture for testing
Computation of distance of distance profile and determination of valid peaks
Selection of gesture database
Synthetic feature extraction
Abduction and adduction feature extraction
Texture feature extraction
Offline database for texture features
Distance-based classifier
Gesture classification

Fig. 2. Block diagram of the proposed scheme.

2. Proposed hand pose recognition scheme for abduction finger movements

In our proposed method, a user-specific hand model is obtained via a series of image segmentation and morphological operations. The system uses the model to determine the user's hand pose. The hand is first separated out from the forearm and some key geometric features of pre-defined gestures are obtained after hand calibration. These features are subsequently modeled as a Gaussian distribution to consider spatiotemporal variations of the finger positions during gesticulation. Based on the proximity of the incoming gestures with the pre-defined gesture patterns, the input gesture can be recognized. The proposed scheme is described in more detail in the sections to follow.

2.1. Proposed scheme for hand image segmentation and VOP generation

From the gesture video sequence, VOPs for different hand positions are obtained. During this phase Hausdorff distance is used to track the change in hand positions from one frame to the next.

Hausdorff distance measure can be used to measure the similarity between two shapes. It is defined as the maximum function between two sets of points O and I, as given below [15]

$$H(O,I)=\max\{h(O,I),h(I,O)\} \tag{1}$$

where

$$h(O,I)=\max_{o\in O}\min_{i\in I}\|o-i\|$$

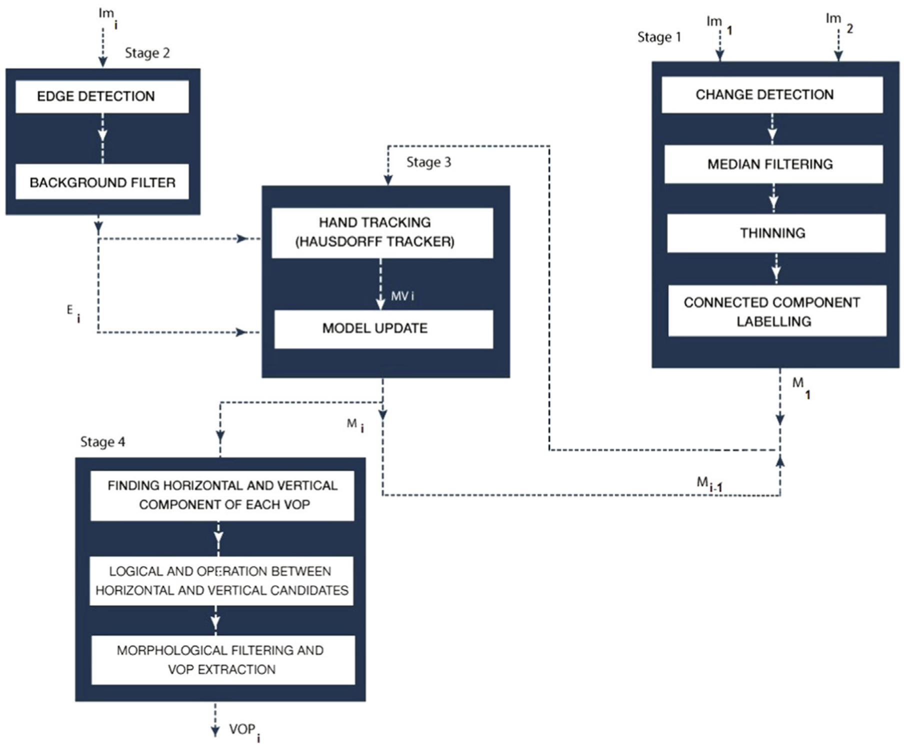
Im i
Stage 2
EDGE DETECTION
BACKGROUND FILTER
E i
Stage 3
HAND TRACKING (HAUSDORFF TRACKER)
MV i
MODEL UPDATE
M i
Stage 1
Im 1
Im 2
CHANGE DETECTION
MEDIAN FILTERING
THINNING
CONNECTED COMPONENT LABELLING
M 1
M i-1
Stage 4
FINDING HORIZONTAL AND VERTICAL COMPONENT OF EACH VOP
LOGICAL AND OPERATION BETWEEN HORIZONTAL AND VERTICAL CANDIDATES
MORPHOLOGICAL FILTERING AND VOP EXTRACTION
VOP i

Fig. 3. Block diagram for the VOP generation algorithm.

and

$$h(I,O)=\max_{i\in I}\min_{o\in O}\|i-o\|$$

Feature points are denoted by $o_1,\ldots,o_m$ for the set O and $i_1,\ldots,i_n$ for the set I. The computation of Hausdorff distance is performed by the distance transform algorithm. For this, mask chamfer 5-7-11 is used.

From the gesture video sequence, VOPs for different hand positions are obtained. The VOP generation program may be divided into four stages as depicted in Fig. 3:

- *Stage* 1: Initial hand model extraction.
- *Stage* 2: Edge detection of input video sequences.
- *Stage* 3: Object tracking and model update.
- *Stage* 4: VOP extraction.

The initial model image is generated from the first two gray-scale images, as shown in Stage 1 block of Fig. 3. This model is continuously updated in Stage 3 and is used for object tracking. The edge image for each frame is generated in Stage 2 and is also used in Stage 3. In Stage 4, VOPs are extracted from the corresponding updated model.

The core of this algorithm is an object tracker that matches a 2D binary model of the video object against subsequent frames using the Hausdorff distance. The best match found indicates the translation the object has undergone, and the model is updated in every frame to accommodate for rotation and change in shape. However, the method will be effective only if the video object changes slowly from one frame to the next, which we assume to be true in the present case.

2.1.1. Initial hand model generation

An initial model is necessary to track the video object in successive video frames of the incoming video sequence. This is accomplished by the following steps:

Change detection: Generates a change detection mask by thresholding a difference image formed from the two initial frames in a gesture sequence.

Median filtering: Removes noises in the threshold difference image thereby generating more accurate initial model.

Thinning: Reduces the width of the model by a morphological thinning process.

Connected component labeling: Eliminates short edges of the model.

An example of initial model generation using all these steps is demonstrated in Fig. 4.

2.1.2. Edge detection and background filtering

Once the initial hand model is obtained, the task is to determine in subsequent frames any change in the hand shape with respect to the model. Now, to reduce the computational cost, we propose to use edge images instead of the complete frames.

Background filtering is a desired step when VOPs are extracted from a cluttered background. Removal of background edges reduces the number of points to be considered during the process of shape change detection and thus speeds up the tracking process. However, our algorithm is capable of functioning even without background removal as long as the background does not change significantly. This is because our algorithm uses large variation in hand shape for gesture modeling. Therefore, as long as the background

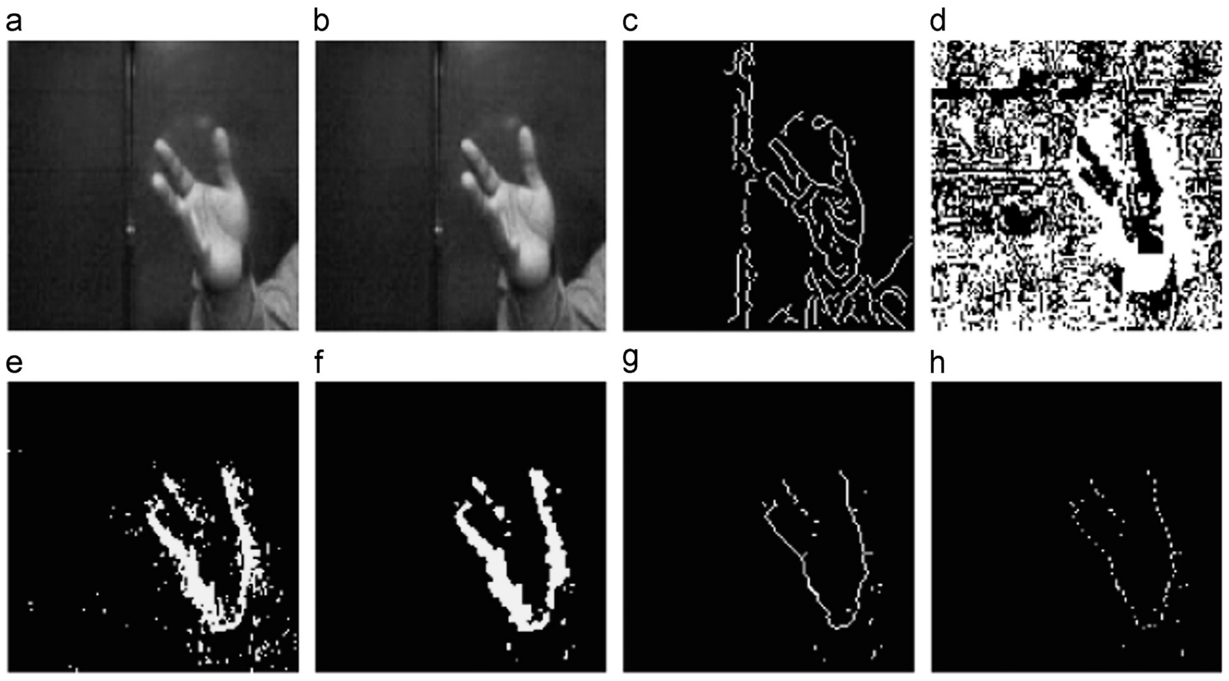

Fig. 4. Initial model generation: (a)–(b) Two initial video frames, (c) edge image of the first frame, (d) difference image, (e) threshold difference image, (f) median filtering of threshold difference image, (g) thinning, and (h) connected component labeling.

edges are somewhat stationary in a sequence, the performance of our algorithm is not degraded.

2.1.3. Hand tracking and model update

The Hausdorff object tracker finds the position where the input hand model best matches the next edge image and returns the motion vector MV_i that represents the best translation. As the tracked object moves through a video sequence, it may rotate or change in shape. To allow for this, the model must be updated every frame. The model image is updated by using the motion vector. First, the current model M_{i-1} is dilated and then shifted by motion vector MV_i. Then, the portion of edge image (E_i) that overlaps this shifted model is selected as the new model image M_i, as shown in the block diagram given in Fig. 3.

2.1.4. VOP extraction

A horizontal candidate for the object in a frame is the region in a row between the first and last edge points, while the vertical candidate is the same region in a column. After finding all the horizontal and the vertical candidates in a frame, the VOP is generated by logical AND operation and further processed by alternative use of morphological operations like closing and filling. We propose using the logical AND operation (Fig. 5e) because, unlike the OR operation (Fig. 5d), it provides information on individual fingers during hand gestures.

2.2. Detection of centroid

After determining the binary alpha plane corresponding to the palm region of the hand, moments are used to find the center of the hand. The 0th and the 1st moments are defined as

$$M_{00} = \sum_x \sum_y I(x,y), \quad M_{10} = \sum_x \sum_y xI(x,y),$$
$$M_{01} = \sum_x \sum_y yI(x,y) \tag{2}$$

Subsequently, the centroid is calculated as

$$x_c = \frac{M_{10}}{M_{00}} \quad \text{and} \quad y_c = \frac{M_{01}}{M_{00}} \tag{3}$$

In the above equations, $I(x,y)$ is the pixel value at the position (x,y) in the image. Since the background pixels are assigned **0**, the centroid of the hand in a frame is also the centroid of the total frame. Therefore, in moment calculations, we may either take the summation over all pixels in the frame or over only the hand pixels.

2.3. Hand normalization

In this step, the extracted skin region corresponding to the hand region is normalized to eliminate variations in the distance between the gesturing hand and the camera. This normalization simply nullifies the motion of the hand normal to the direction of the camera. The edge mask of the segmented hand is first extracted by using Canny edge detector. The normalization constant related to the overall hand size is then computed by the following equation:

$$N_0 = \frac{\sum_{(x_e,y_e) \in E} \sqrt{(x_e - x_c)^2 + (y_e - y_c)^2}}{M} \tag{4}$$

where E is the set of all pixels belonging to the edge mask, and M is the total number of edge pixels.

2.4. Determination of orientation of hand and hand bounding box generation

The orientation of the hand can be estimated by the position of the fingertips and the centroid of the hand (palm) in the image frame. Orientation is defined as the angle of axis of the least moment of inertia. It is obtained by minimizing $I(\theta)$ with respect to θ, *i.e.*

$$I(\theta) = \sum_{(m,n) \in R} \sum [(P_{i,y} - y_c)\cos\theta - (P_{i,x} - x_c)\sin\ \theta]^2 \tag{5}$$

That gives

$$\theta = \frac{1}{2}\tan^{-1}\left[\frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}}\right] \tag{6}$$

where

$$\mu_{p,q} = \sum_{(m,n) \in R} \sum (P_{i,x} - x_c)^p (P_{i,y} - y_c)^q$$

$(P_{i,x}, P_{i,y})$ and (x_c, y_c) are the positions of fingertips and the centroid of the palm region, respectively. Once θ is known, we use the following transformations to determine orientation:

$$\left.\begin{aligned} \alpha &= x\cos\theta + y\sin\theta \\ \beta &= -x\sin\theta + y\cos\theta \end{aligned}\right\} \tag{7}$$

For the segmented hand region, we can calculate the orientation and then construct a reference line R_{ref}, as shown in Fig. 6. The line R_{ref} is the major axis of the ellipse enclosing the palm of a hand.

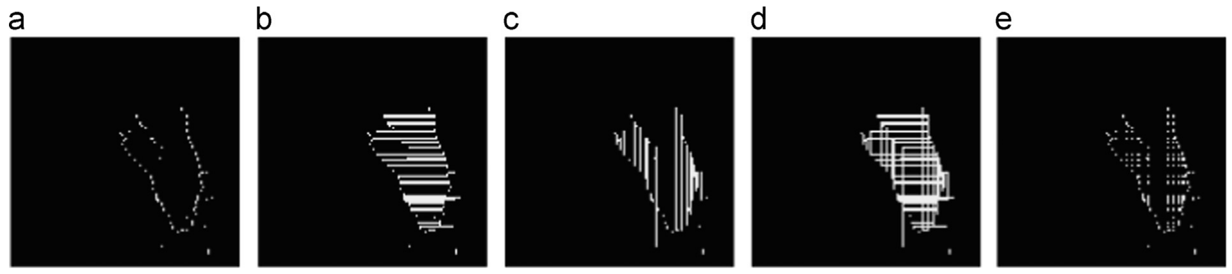

Fig. 5. Results of logical OR and AND operations on horizontal and vertical VOP candidates: (a) updated model, (b) horizontal candidate, (c) vertical candidate, (d) OR operation, in which the thumb and fingers are lost and (e) AND operation, in which they are retained.

2.5. Palm and forearm segmentation

The width of the segmented hand region is used to separate the hand from the forearm. Our proposed method determines the pair of points where the orientation vector intersects with the rectangle boundary of the segmented skin colored hand region. Hand orientation is defined as the angle of the axis of least moment of inertia. The orientation is estimated by using the boundary points and the centroid with respect to the image frame. The two intersection points of the orientation vector with the bounding rectangle are determined. The midpoints of the lines connecting the centroid and each intersection point are computed and joined. Assuming that the wrist position lies within these two points, we can find out the width of the hand region in the perpendicular direction of the line joining the midpoints. The forearm can be detected and separated by locating the wrist part, which can be recognized as two parallel lines originating at the object boundary that expand into the shape of a hand. As shown in Fig. 6, the position where the width of the hand region changes most along the perpendicular direction of the constructed line determines the boundary between the hand and the forearm.

2.6. Hand modeling

The kinematic and dynamic constraints of the hand are applied to model the hand mathematically:

- The rotation of the proximal joints of hand is responsible for the movement of a particular segment of the hand, which can be specified by the joint's rotation angles. As shown in Fig. 7, a local coordinate system can be defined on joint positions, and any joint rotation can be modeled as a sequence of rotations on the three axes of the local coordinate system.

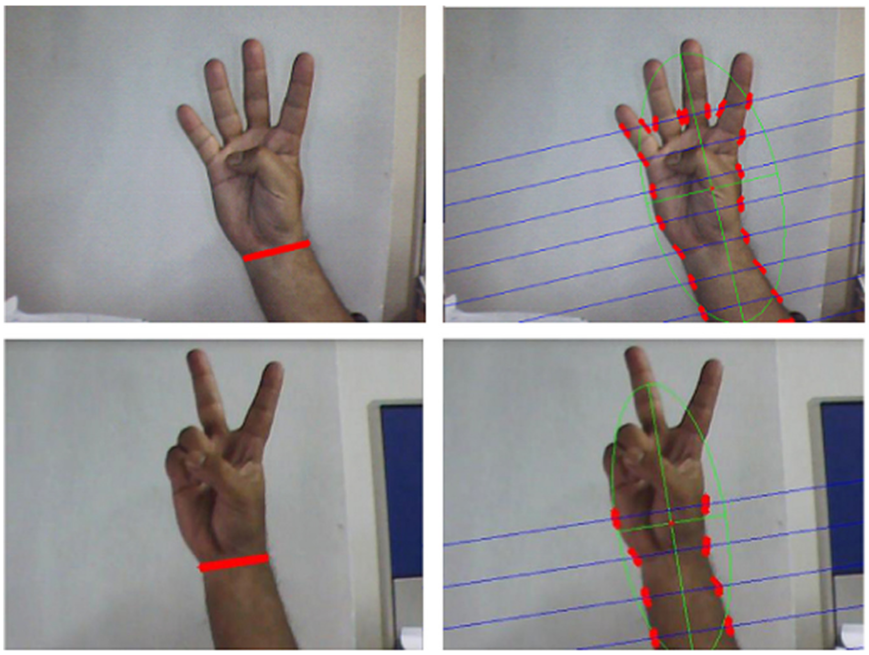

Fig. 6. Forearm segmentation.

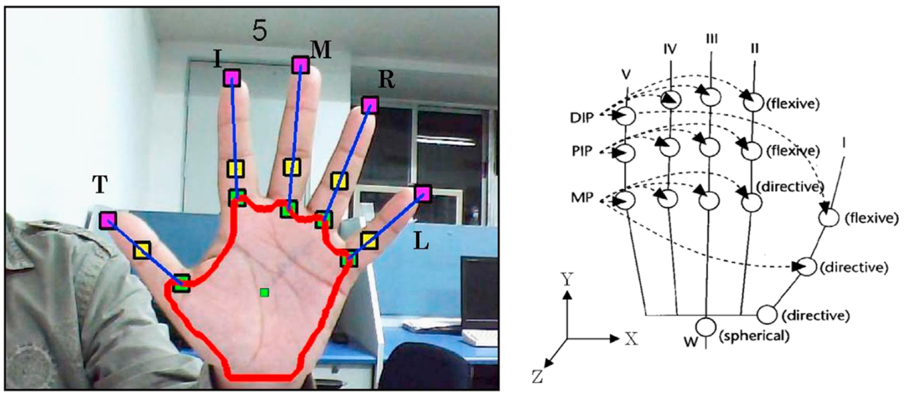

Fig. 7. (i) A normal stretched hand with fingers named as T, I, M, R, L and (ii) hand modeling.

- Only the adduction and the abduction of the metacarpo-phalangeal (MP) joints are considered for modeling. The MP joint of finger III displays limited adduction and abduction.
- The hand remains perpendicular to the optical axis of the camera so that the line joining centroid and fingertip of middle finger lies parallel to the *Y*-axis. Also, the axes of abduction and adduction motions are referenced along the *Y*-axis.
- The thumb is modeled separately by assuming that the line joining the MP joint of the thumb with the thumb fingertip is straight, and it has only abduction and adduction movement about the *Z*-axis. The total abduction and adduction angle movement for thumb is taken as 30°.
- The abduction or the adduction angle between the adjacent fingers from II to V for static constraints are taken as

$$-15^\circ \le Q^z_{MP,s} \le 15^\circ \tag{8}$$

where *Z* gives the axis of rotation of the adduction or abduction movement in the local joint centered coordinate system. Here, *s* denotes the indices of fingers from II to V.
- The flexion angle between the adjacent fingers from II to V for dynamic constraints are taken as $\theta^y_{PIP} = \frac{3}{2}\theta^y_{DIP}$ and $\theta^y_{MCP} = \frac{1}{2}\theta^y_{PIP}$ where *Y* gives the axis of rotation of the flexions/extensions in the local joint centered coordinate system [1,2].
- Only the flexions/extensions of the proximal interphalangeal (PIP) and distal interphalangeal (DIP) joints are considered for modeling. The range for PIP and DIP joints is given by the following equation:

$$45^\circ < \theta^y_{PIP} < 75^\circ \tag{9}$$

2.7. *Extraction of geometric features/calibration*

The proposed algorithm for extracting key geometric features of a hand is summarized as follows:

Step 1: Show the five fingers in front of the camera such that the plane of the hand remains perpendicular to the optical axis of camera. As shown in Fig. 8, compute the distances of the boundary points of the hand from the centroid (x_c, y_c) in a cyclic manner and store them in an array $r(t)$, where t is indices of pixels while traversing the contour. Smooth the data in $r(t)$ using B-spline interpolation and compute the local maxima. Sort the values of the maxima and store the top five elements of the sorted array as $r_1 > r_2 > r_3 > r_4 > r_5$ and their indices in $r(t)$ as $t_1 > t_2 > t_3 > t_4 > t_5$. Horizontal lines in Fig. 8 actually show the process of finding different peaks of the curve to determine the corresponding fingertips.

Step 2: Select r_A, r_B, r_C, r_D and r_E such that $r_1 < r_A < r_2, r_2 < r_B < r_3, r_3 < r_C < r_4, r_4 < r_D < r_5, r_5 < r_E < r_5 - \delta$; where δ is a safety margin, which is set as $\delta = r_5/3$. Compute six sets of points by finding six sets of solutions of the equations $r(t) = r_A$, $r(t) = r_B$, $r(t) = r_C$, $r(t) = r_D$, $r(t) = r_E$ and $r(t) = r_5 - \delta$. Let us consider the solution sets to be S_1, S_2, S_3, S_4, S_5 and S_6. The following algorithm is then implemented to determine the finger tips.

Algorithm 1. Proposed algorithm for fingertip detection.

```
for $i = 1$ to 6 do
  Consider $s_{i,j}$ as the jth element of set $S_{i,i=1:6}$
  $N_i$ is the total no of elements in $S_i$
  for $j = 1$ to $N_i$ do
    $index = \arg(\min_{p=1:5}(|s_{i,j} - t_p|)), index \in \{1, \dots, 5\}$
    Include $s_{i,j}$ in set $P_{index}$
  end
  for $k = 1$ to 5 do
    if number of elements in $P_k = 1$ and $P_k = \{p_{k,1}\}$ then
      Include spatial coordinates corresponding to $p_{k,1}$ in set $M_k$
      ELSEIF number of elements in $P_k = 2$ and $P_k = \{p_{k,1}, p_{k,2}\}$
      Include midpoint of spatial coordinates corresponding to
      $p_{k,1}, p_{k,2}$ in set $M_k$
      ELSE DO NOTHING
    end
  end
  CLEAR $P_{index,index=1:5}$
end
```

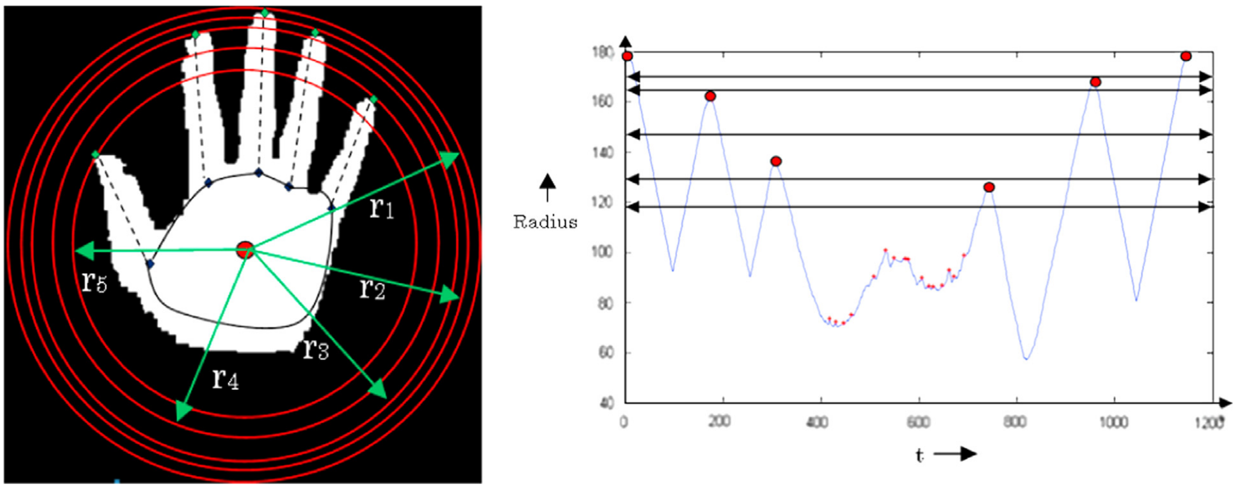

Fig. 8. Hand model showing angle of rotation of fingers. The circles drawn about the centroid with radii r_1, r_2, r_3, r_4 and $r_5(r_{\min})$.

Step 3: Construct five axes by joining elements of each of the sets $M_{k,k=1:5}$. Now for recognizing the type of fingers in the hand, if the left hand is shown for calibration, classify the peaks from left to right with respect to the reference axis R_{ref} as T (thumb or finger I), I (index or finger II), M (middle or finger III), R (ring or finger IV) and L (little or finger V). The reverse procedure is applicable for the right hand. For each of the fingers, the intersection of the axis with the hand contour and eroded hand mask will give the actual positions of fingertips P_T, P_I, P_M, P_R, P_L and positions of MP joints R_1, R_2, R_3, R_4, R_5, respectively. The extracted finger tips are shown in Fig. 9(a).

2.8. *Hand pose recognition for abduction and adduction finger movements*

In our method, the change of abduction/adduction angles are modeled as Gaussian distributions which can take care of the intra-class variability. For a normal hand gesture showing all the five fingers for calibration, we can select 10 pairs of finger combinations. Subsequently, three key features for each pair of fingers are computed based on the information of relative distance and relative angle generated by fingertips and MP joint positions. The three features proposed in our method are $d_{i,j}, \beta_{i,j}$ and $\alpha_{i,j}$, which are shown for the case of I and M finger pair in Fig. 9(b). Due to the flexion movements of fingers, the distance profile of the hand region cannot provide accurate information of the position of the tips. For this, the approximate position of the PIP joints is used to extract the features by considering the static constraint of the fingers.

Here (i,j) denotes the indices of pair of fingers, where the indices for thumb, index, middle, ring, and little fingers are 1, 2, 3, 4, and 5 respectively. Thus, spatial information of each pair of finger can be mapped to a relative 3D feature space $F_1F_2F_3$. In our proposed algorithm for finger type recognition, we consider eight hand gestures

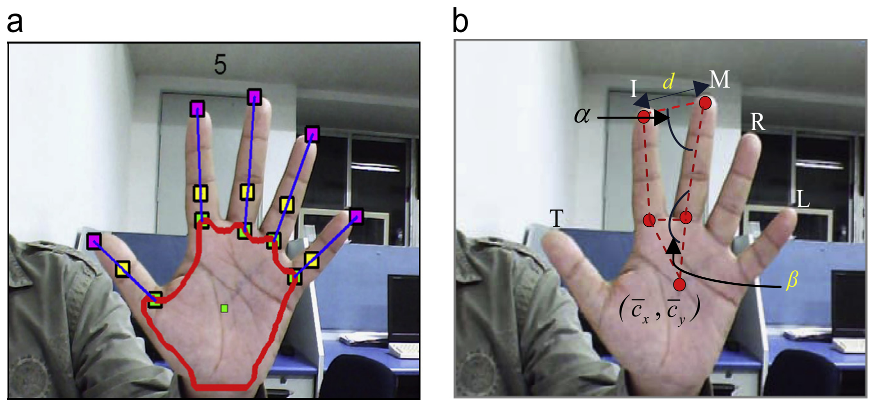

Fig. 9. (a) Results of proposed algorithm for fingertip detection and (b) features $d_{i,j}, \beta_{i,j}$ and $\alpha_{i,j}$ proposed in our method.

Table 1
Range of abduction and adduction angle for different classes of gestures.

Gestures considered	**Range of abduction and adduction angles in degrees (mean angle is shown in brackets)**				
	T	I	M	R	L
TI	θ to $\theta+30$ ($\theta+15$)	−15 to 15 (0)			
IM		−15 to 0 (−7.5)	0 to 15 (7.5)		
IL		−15 to 15 (0)			−15 to 15 (0)
TIM	θ to $\theta+30$ ($\theta+15$)	−15 to 0 (−7.5)	0 to 15 (7.5)		
IMR		−15 to 0 (−7.5)	0 (0)	0 to 15 (7.5)	
TIL	θ to $\theta+30$ ($\theta+15$)	−15 to 15 (0)			−15 to 15 (0)
TIML	θ to $\theta+30$ ($\theta+15$)	−15 to 0 (−7.5)	0 to 15 (7.5)		−15 to 15 (0)
IMRL		−15 to 0 (−7.5)	0 (0)	0 to 15 (7.5)	−15 to 15 (0)

performed by two, three, and four fingers. The valid gestures considered in our experiment are given in Table 1. If M_0 denotes the total number of finger pairs possible for a particular gesture, a particular gesture can be mapped to M_0 distinct points in $F_1F_2F_3$ space. Thus, eight selected patterns can be mapped to $F_1F_2F_3$ space into eight sets $D_{i,i=1:8}$. The gesture patterns well separated in $F_1F_2F_3$ space and also easy to gesticulate are considered in the experiment. For unique orientation of the finger axis of thumb, the gestures incorporating the thumb are analyzed separately. The reference axis for the thumb is defined as the line joining the centroid and the estimated MP joint of the finger. For other fingers, the reference axis is the line parallel to the middle finger axis and passing through the MP joint of the respective fingers. For gestures with a single finger, template matching can be performed for finger type detection.

2.8.1. *Modeling of feature distribution*

Table 1 shows approximate variations of the abduction and the adduction angle of each finger in different gesture classes. We propose two models for analyzing possible variations of the features d, β, and α in a pair of fingers. Model 1 includes all pairs of fingers including the thumb, and Model 2 includes all pairs of fingers without the thumb.

The proposed Model 1 and Model 2 are shown in Fig. 10. The notations used in the figures are summarized below:

- $d_{i,j}$: d-feature of the ith and jth fingers.
- $\beta_{1,i,j}$: β_1-feature of the ith and jth fingers.
- $\beta_{2,i,j}$: β_2-feature of the ith and jth fingers.
- L_i: length of ith finger.
- C_{ij}: distance between MP joints of ith and jth fingers.
- l_i: distance of MP joint of ith finger from the centroid.

Fingers indexed with 1–5 represent thumb, index, middle, ring and little finger. The green arrows in Fig. 10 indicate the possible range of abduction and adduction angles for a particular finger. Here θ is the offset angle of the thumb during the calibration from the reference axis of the thumb. Again, θ_1 and θ_2 are abduction and adduction angles for a particular pair of fingers in a gesture. The features are computed from the two models by using Euclidean geometry as shown in Table 2.

where

$$A_c = \sqrt{L_i^2 + C_{i,j} + 2L_iC_{i,j}\sin(\theta_2)}$$

and

$$Y = \cos\left(90^\circ + \theta_2 - \sin^{-1}\left(\frac{L_i\sin(\theta_2)}{A_c}\right)\right)$$

In our method, we only consider eight gesture classes defined as $G_{j,j=1:8}$. So, $D_{k,k=1:8}$ are the sets of points in $F_1F_2F_3$ space for the gesture classes $G_{j,j=1:8}$. For every gesture class with N fingertips, there will be $M_0 = \binom{N}{2}$ points in $F_1F_2F_3$ space. Due to spatial variations of the abduction and adduction angles, we model each of the three features as a Gaussian distribution for every points in $F_1F_2F_3$ space for a particular gesture.

Let us consider that the average values or the mean of the distributions of the three features are d_{avg}, β_{avg}, and α_{avg}. These values are computed by using the mean angles of θ_1 and θ_2, which are obtained by using Table 1 and the equations given in Table 2. The maximum possible ranges of three features are computed by putting the ranges of θ_1 and θ_2. The ranges obtained for each of the features are denoted as Δd, $\Delta\beta$ and $\Delta\alpha$, which will be subsequently used for computing the variances of the feature distributions.

Table 2
Computation of features.

Features	**Equations**
d	$d_{i,j} = \sqrt{A_c^2 + L_j^2 - 2L_jA_c \times Y}$
β	$\beta_{i,j} = 90^\circ + \theta_1 + \cos^{-1}\left(\frac{l_j^2 + C_{i,j}^2 - l_i^2}{2l_jC_{i,j}}\right)$
α	$\alpha_{i,j} = \sin^{-1}\left(\frac{A_c}{d_{i,j}}\cos\left(\theta_1 - \sin^{-1}\left(\frac{L_i\cos(\theta_2)}{A_c}\right)\right)\right)$

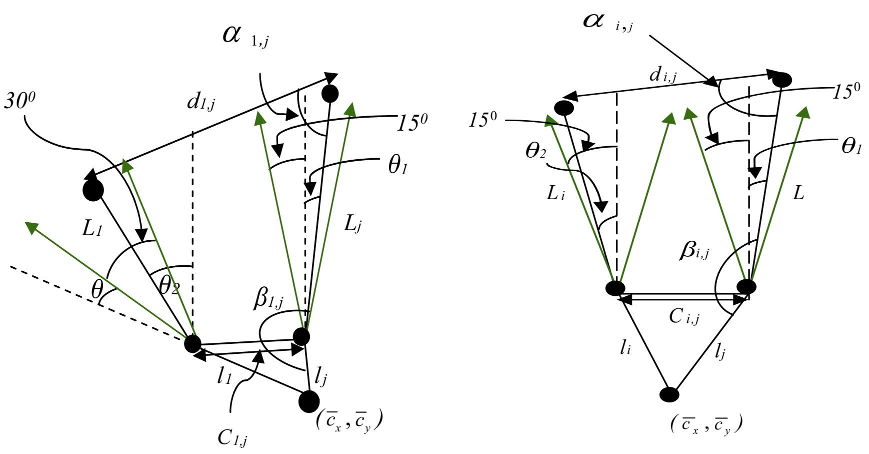

Fig. 10. Proposed Model 1 and Model 2 for modeling abduction and adduction movements of finger about the MP joint. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

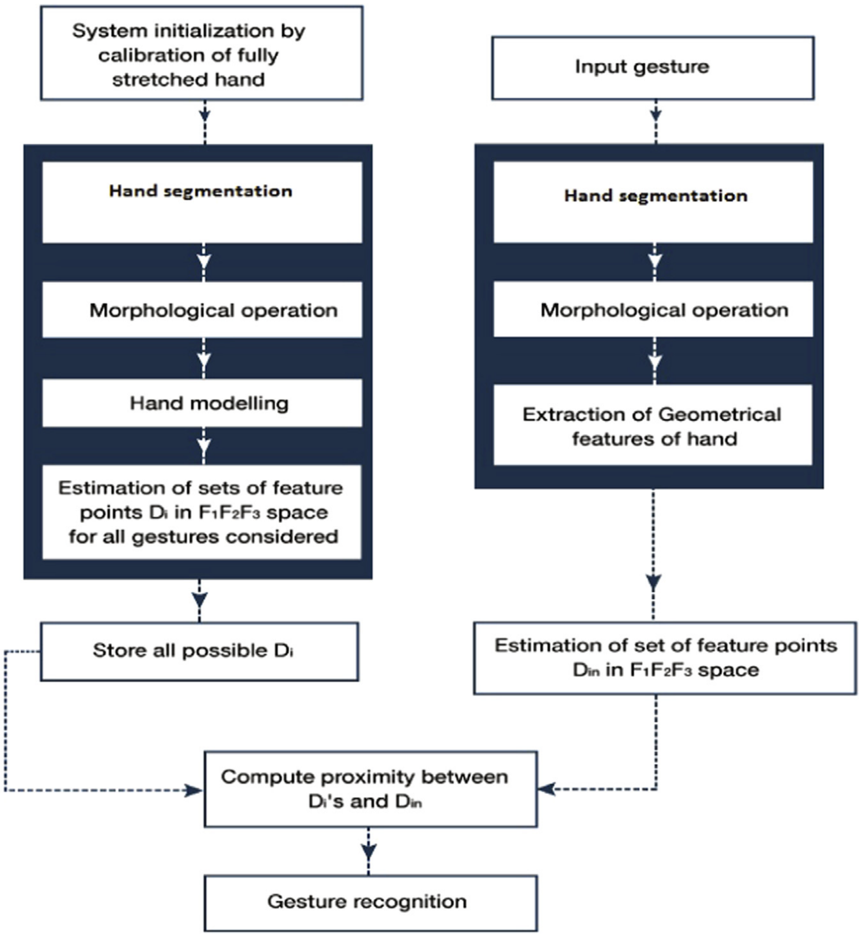
System initialization by calibration of fully stretched hand
Hand segmentation
Morphological operation
Hand modelling
Estimation of sets of feature points Dᵢ in F₁F₂F₃ space for all gestures considered
Store all possible Dᵢ
Input gesture
Hand segmentation
Morphological operation
Extraction of Geometrical features of hand
Estimation of set of feature points Dᵢₙ in F₁F₂F₃ space
Compute proximity between Dᵢ's and Dᵢₙ
Gesture recognition

Fig. 11. Block diagram of the proposed scheme.

Next, a normal distribution is defined for each of the three features from the information obtained from the geometric features and feature variations model. For each of the three features d, β and α, which are denoted as F_1, F_2, and F_3, we can define a normal distribution as

$$f(F_i) = \frac{1}{\sqrt{2\pi(\Delta F_i)^2}} e^{-(F_i - F_{i,avg})^2/2(\Delta F_i)^2} \Big|_{i \in \{1,2,3\}} \tag{10}$$

We determine all the sets $D_{k,j}$ for eight valid gesture classes during the calibration, where $D_{k,j}$ represents jth cluster of kth gesture in $F_1F_2F_3$ space. The set can be expressed as follows:

$$D_k = \{f(d_{k,j}) \times f(\beta_{1,k,j}) \times f(\beta_{2,k,j})\}_{j=1:M_0} \Big|_{k \in \{1,2,3,\ldots,8\}} \tag{11}$$

where M_0 denotes the total number of distributions in set D_k as defined before.

2.8.2. *Gesture classification*

Let $\overline{f}_{in}$ be the vector consisting of the fingertip positions and MP joint positions of the input gesture. As discussed in Section 2.7, a contour distance profile is generated by computing the distance of the hand contour from the centroid in a cyclic manner. The distance profile is interpolated using B-splines. All the small unwanted peaks in the palm boundary with value less than $r_{min} - \delta$ are excluded from further consideration. As described earlier in the fingertip detection algorithm, we perform morphological operations to find out a more accurate position of the fingertips. The input gesture pattern $\overline{f}_{in}$ is then transformed to D_{in} in the $F_1F_2F_3$ space. Next, a proximity measure is used to find the proximity of D_{in} from the pre-modeled gesture pattern distributions D_{pre}. A pre-modeled gesture pattern contains the same number of fingertips as the input gesture pattern. Mahalanobis distance is used for computing the proximity between D_{in} and all the relevant D_{pre}. The elements of D_{in} and the distributions of D_{pre} should be in the same order during the measurement of proximity between D_{in} and D_{pre}. Subsequently, the corresponding gesture pattern is recognized on the basis of minimum distance criteria. All the important steps of our proposed gesture recognition scheme for abduction finger motions are shown in Fig. 11.

2.9. *Extension of the proposed method for two-handed pose recognition*

The hands are first separated from the face and, subsequently, the binary alpha planes corresponding to

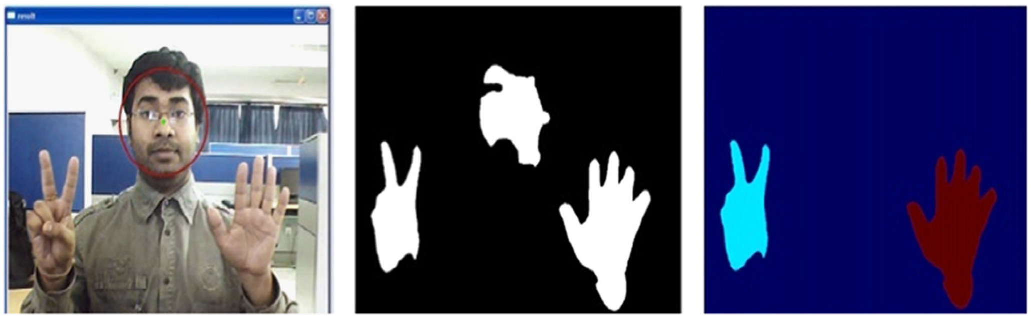
Fig. 12. Face elimination and hand localization.

two gesturing hands are extracted on the basis of spatial information of the hand. In this method, a learning-based approach and the adaboost algorithm are used for the detection of the face. In this, Haar-like features are used for recognition of the frontal face region from the background and subsequently, two hands can be labeled separately by connected component labeling [18]. Fig. 12 shows the outputs of the face elimination and hand localization process.

Two-handed gestures are treated as a composition of two independent one-handed gestures. For two-handed fingertips detection, key geometric features of each of the hands (*i.e.,* hand model) are obtained after calibrating the hands. The same calibration technique as discussed above is applied to recognize the fingertip positions of both hands. Each of the ten fingers is placed in front of the camera one by one for calibration. Both hands remain perpendicular to the camera so that the line joining the centroid and the fingertip of the middle finger lies parallel to the *Y*-axis. The positions of the fingertips are then identified based on the positions of the reference lines of the localized hand regions. If the left hand is shown for calibration, the peaks are classified from the left to right with respect to the reference axis R_{ref} as T (thumb or finger I), I (index or finger II), M (middle or finger III), R (ring or finger IV) and L (little or finger V). The reverse procedure applies for the right hand. As the two hands are already localized, the proposed algorithm is used on both hand regions separately to find the fingertips of each gesturing hand. The overall gesture is then recognized based on the information from each hand.

3. Proposed hand pose recognition scheme for flexion finger movements

As explained earlier, it is quite difficult to estimate the flexion angles from the 2D image as it involves the estimation of finger motion along the optical axis of the camera. One possible solution is the 3D modeling of the fingers to estimate all the finger movements. But model-based methods are computationally complex. So we propose a novel scheme to determine the flexion angle variations by analyzing the texture of the projected region of the fingers in the 2D image plane. So, instead of modeling the flexion angles, we use the texture features for differentiating two gestures which are different in terms of the flexion angles of the fingers. It is experimentally found that the texture features can roughly give an estimation of the finger motions along the direction of the camera. It is therefore justified to perform texture analysis in lieu of 3D modeling of the finger joints of the hand to extract the flexion and extension information inherent in the gestures. The method is simple and straightforward.

In our method, texture-based features are extracted from the texture analysis of the circular strips of the segmented hand image. The main motivation behind the proposed texture descriptors in the circular strips is that the specific patterns of texture can be extracted corresponding to the different flexion finger movements. These texture patterns indirectly give the information of flexion finger movements.

3.1. Proposed flexion features extraction

An important step towards the extraction of the proposed features is to find the textured region which can cover only the flexion finger movements. The texture analysis is performed using texture browsing descriptors (TBR).

The center of mass of the palm region is first computed as mentioned in Section 2.7. Now from the information of the calibrated hand, we estimate the relative positions of the MP joints of the fingers. The orientation of the hand is pre-computed using the approach explained in Section 2.4. For the extraction of flexion features, a textured region is chosen with a range of -120° to 120° from the reference line of the hand region. Subsequently, a circular region of radius equal to the distance between the center of mass of palm region and the highest peak of the distance profile of the hand contour is constructed. Starting from the peak value $(R_{max}/5)$, circular strips of width $5R_{max}/27$ are taken with decreasing radius for extracting the circular textured region. Three circular strips are taken to cover the total range of flexion movement of the fingers.

As shown in Fig. 13, the color region inside the circular strips are used to analyze the texture. Also, as illustrated in Fig. 14, a bounding box is constructed along the segmented portion of each circular strip to find the texture-based features. Fig. 15 shows the extracted textured region of the hand.

For extracting feature vectors from the specified textured region, a homogeneous texture descriptor (HTD) is used. The HTD characterizes the textured region using

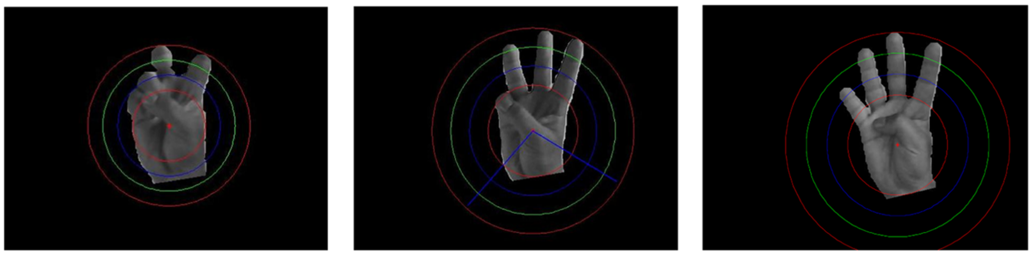

Fig. 13. Figure showing circular strips of texture. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

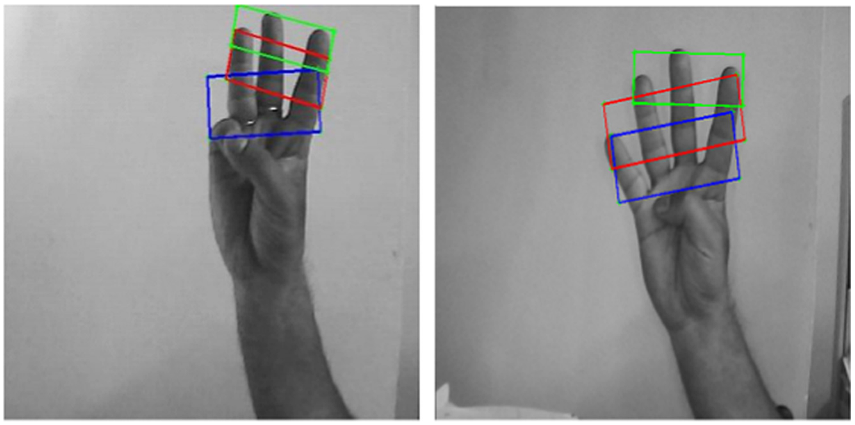

Fig. 14. Bounding box of the portion of the textures of the hand.

Fig. 15. Portions of the texture of the hand.

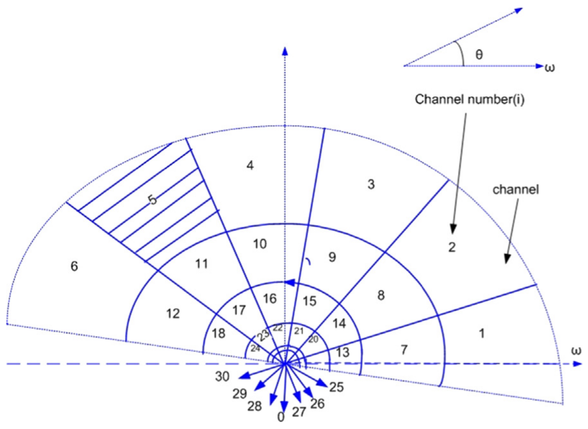

Fig. 16. Channels used for computing HTD.

the mean energy and energy deviation from the set of frequency channels [19]. The 2D frequency plane is partitioned into 30 channels as shown in Fig. 16. The mean energy and its deviation are computed in each of these 30 frequency channels in the frequency domain.

The syntax of HTD is as follows:

$$HTD = [f_{DC}, f_{SD}, e_1, e_2, \ldots, e_{30,} d_1, d_2, \ldots, d_{30}] \tag{12}$$

In this, f_{DC} and f_{SD} are the mean and the standard deviation of the image respectively, and e_i and d_i are the nonlinearly scaled and quantized mean energy and energy deviation of the corresponding ith channel, as shown in Fig. 17.

The individual channels in Fig. 17 are modeled using Gabor functions. A channel indexed by (s,r) is considered, where s is the radial index and r is the angular index. Then the (s,r)th channel is modeled in the frequency

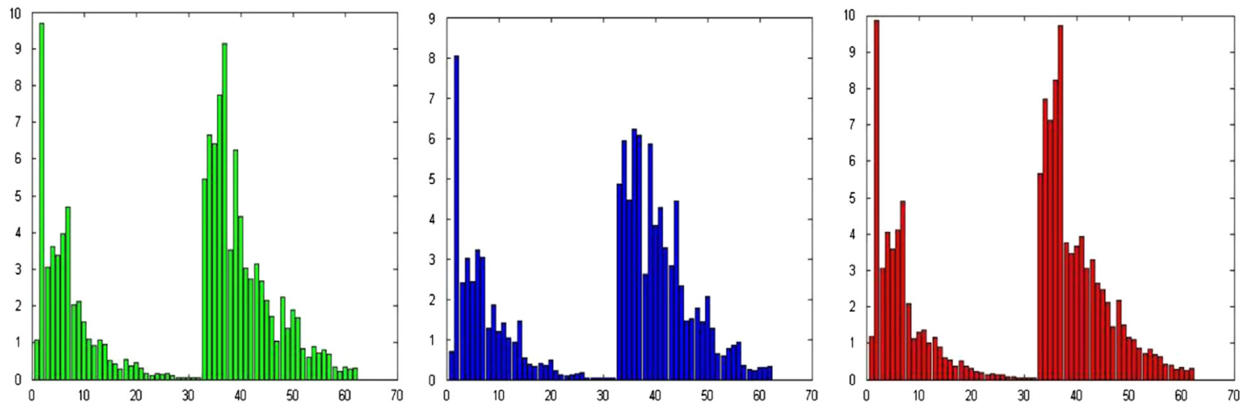

Fig. 17. Homogeneous texture descriptors of three channels.

domain as

$$G_{s,r}(\omega,\theta) = \exp\left(\frac{-(\omega-\omega_s)^2}{2\sigma_s^2}\right)\cdot\exp\left(\frac{-(\theta-\theta_r)^2}{2\tau_r^2}\right) \mathrm{ith} P(\omega,\theta) \tag{13}$$

On the basis of the frequency layout and the Gabor functions, the energy e_i of the ith feature channel is defined as the log-scaled sum of the square of the Gabor filtered Fourier transform coefficients of an image

$$e_i = \log_{10}[1+p_i] \tag{14}$$

where

$$p_i = \sum_{\omega=0^+}^{1}\sum_{\theta=(0^0)^+}^{360^0}[G_{s,r}(\omega,\theta)|\omega|P(\omega,\theta)]^2 \tag{15}$$

and $P(\omega,\theta)$ is the Fourier transform of an image represented in the polar frequency domain, that is $P(\omega,\theta) = F(\omega\cos\theta, \omega\sin\theta)$, where $F(u,v)$ is a Fourier transform in the Cartesian coordinate system. The energy deviation d_i of the future channel is defined as the log-scaled standard deviation of the square of the Gabor filtered Fourier transform coefficients of an image [20]

$$d_i = \log_{10}[1+q_i] \tag{16}$$

where

$$q_i = \sqrt{\sum_{\omega=0^+}^{1}\sum_{\theta=(0^0)^+}^{360^0}\{[G_{s,r}(\omega,\theta)|\omega|P(\omega,\theta)]^2 - p_i\}^2} \tag{17}$$

The above operations can be efficiently performed using the Radon transform. The Radon transform is defined as the integral along the line that has an angle θ counter-clockwise from the y-axis and at a distance R from the origin [17]. It can be written as

$$\begin{aligned} p_\theta(R) &= \int_{L(R,\theta)} f(x,y)\,dl \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos\theta + y\sin\theta - R)\,dx\,dy \end{aligned} \tag{18}$$

Thus, we generate three sets of HTD vectors (each containing 62 feature elements) as a representation of texture variations due to the flexion and extension movements of the fingers.

3.2. *Gesture recognition by proximity measure*

From several experiments, we observe that the hand poses due to the variations in flexion angles of the fingers can be differentiated from the analysis of the textures of the fingers. Texture patterns of the fingers change with the variations of flexion angles. For our experiments, a database of texture variations corresponding to different finger movements is collected from a number of signers. Subsequently, the texture feature HTD of the gestures under consideration are calculated and stored in the database.

To measure the similarity between an input gesture and the gestures in the database, a distance measure is proposed. This distance measure can simultaneously handle variations in abduction and flexion angles. Given an input hand image, we generate homogeneous texture descriptor features $T_{i,q}$ $(i = 1, 2, \ldots, 186)$ as mentioned in the previous section and search for their nearest neighbor in the database of mean $M_{i,d}$ $(i = 1, 2, \ldots, 186)$ for the kth $(k = 1, 2, \ldots, 6)$ hand pose. As mentioned in Section 2.8, S_A^q and S_A^d are the input adduction/abduction features of the input image and the database gesture. The exact matching score $\theta(q,d)$ between the input image, q and pose d is defined as

$$\theta(q,d) = \left.\frac{\beta}{d_M(S_A^q, S_A^d)} + \frac{(1-\beta)}{d_M(T_{i,q}, M_{i,d})}\right|_{d=1,2,\ldots,8} \tag{19}$$

where β is a normalized weighting factor which is determined experimentally as 0.6. The pose d in the database for which $\theta(q,d)$ is maximum is the recognized gesture.

4. Experimental results

To validate the proposed technique, several hand poses with important variations of the hand configurations were considered. We judiciously selected the hand poses so that they could be used as pointing gestures for HCI interfaces. The proposed system was tested in real-time on an Intel® Core I3-based personal computer. The input images are captured by a CCD camera at a resolution of 640×480 pixels. Our dataset is a challenging real-life dataset collected in cluttered backgrounds. Besides, for each gesture, the subject poses with variations in abduction/adduction and/or flexion angles. Moreover, all the experiments are performed with nearly constant uniform illumination. Our C++ implementation processes each frame in 78 ms. GPU acceleration is not used.

In our gesture recognition system, the user is constrained to the following four phases for making a gesture. These steps are required to read the hand poses from the video.

1. Insert the hand after some time within the capture range of the camera so that the background image can be read. We use a 640 × 480 image for this purpose.
2. Keep hand still (fixed) in start position until gesture motion begins.
3. Move fingers and hand smoothly and slowly to the most prominent gesture positions.
4. Complete the gesture by keeping the fingers and hand in the final gesture position for a relatively longer period of time.

Our hand gesture recognition system is robust to cluttered backgrounds, because the hand is detected using shape information; thus, the backgrounds can be easily removed. The proposed system has inherent advantages over skin color-based hand gesture recognition approaches.

For fingertips and finger types recognition of single hand, our dataset is collected from 20 subjects, and it contains eight gestures. Each subject performs 10 different poses for the same gesture. Thus in total, our dataset has 20 signers × 8 gestures/person × 10 cases/gesture = 1600 cases. Fig. 18 shows the results of finger tips detection and finger type recognition. Some additional results of our proposed finger tips detection algorithm are shown in Fig. 20 for the input hand poses shown in Fig. 19. Fig. 21 shows the results of hand pose recognition by detecting the finger tips. The recognition rate of the different hand poses, shown in Fig. 19, is about 99%.

To validate the performance of the proposed algorithm for two-handed pose recognition, we take 10 gestures showing fingertip combinations of the two gesturing hands. The proposed algorithm is evaluated by 20 signers and each signer performs three times a predefined set of 10 gestures. Therefore, a total of 600 gestures are used to evaluate the performance of the proposed system. Some of the results of two-handed fingertip detection and two-handed pose recognition are shown in Figs. 22 and 23, respectively.

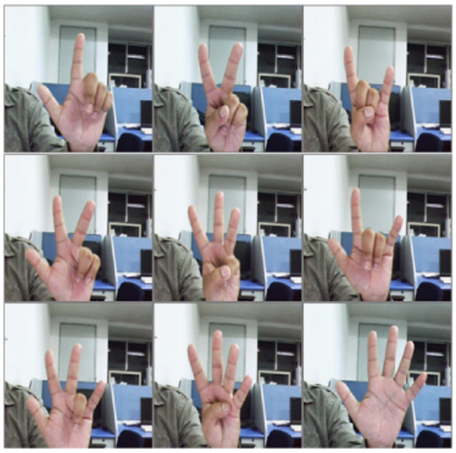
Fig. 19. Selected gestures.

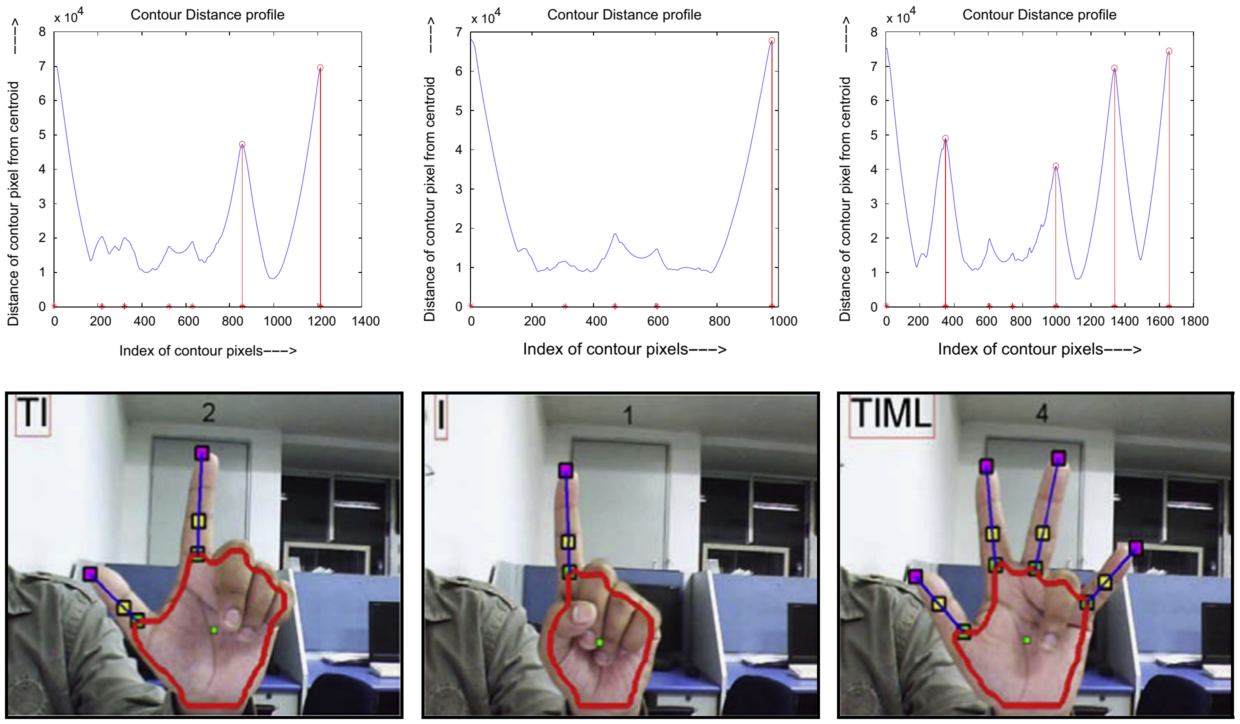

Fig. 18. Results of finger tip detection and finger type recognition.

Finally, Fig. 24 shows the gestures used to evaluate the proposed algorithm for recognizing different hand poses based on the flexion angle of the fingers. In this figure, the first row shows hand poses with straight fingers. The second row shows hand poses with the bent fingers. For our experimentation, 400 training gestures are processed under normal illumination. During the training session, eight mean HTD vectors are calculated for each of the hand gestures under consideration. Next, the proposed system is tested by 10 signers, and each signer performs a predefined set of eight gestures five times each. In this case, the recognition rate was almost 97%. These results show the efficacy of our method to handle both abduction and flexion movement of the fingers.

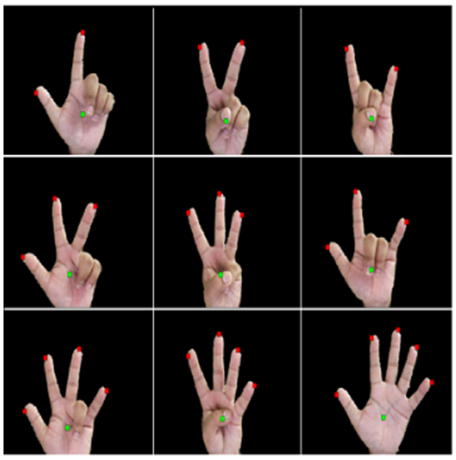
Fig. 20. Fingertips detection.

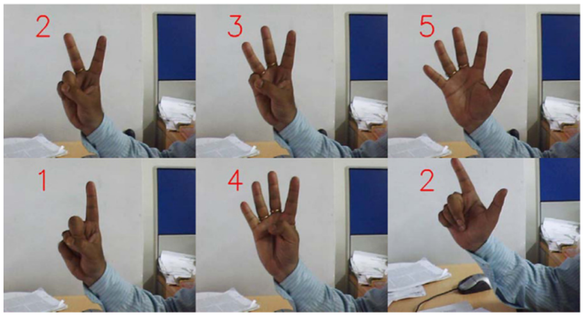

Fig. 21. Hand pose recognition by detecting the fingertips.

The real-time recognition videos are also uploaded to show the efficacy of the proposed method.

We compared our proposed method with two other existing methods as mentioned in [10,13]. For comparisons, we used our database in which 20 signers created it under normal illumination conditions. Each signer performs three times a predefined set of 10 gestures. Therefore, a total of 200 gestures are used for training and the remaining 400 gestures are used to evaluate the performance of the existing methods and the proposed system. Our proposed algorithm also offers improved recognition rates as compared to these two existing methods. The experimental results indicate the feasibility of the proposed algorithm for vision-based interfaces.

5. Conclusion

Static hand gesture recognition is a highly challenging task due to many degrees of freedom of the hand's kinematic structure. In this paper, we proposed a novel hand pose recognition method to support vision-based interfaces.

In our proposed method, MPEG-4 based video object extraction is used for finding different hand positions and shapes from the gesture video sequence. The VOP based method of segmentation does not require the rotation and scaling of the object to be segmented. The shape change is represented explicitly by a sequence of 2D models, one corresponding to each image frame. Incorporation of median filtering in the model formation stage greatly enhances tracking performance. Moreover, introduction of logical AND operation in the final VOP generation step preserves the fingers. The proposed hand segmentation algorithm overcomes the limitations of skin color-based segmentation.

Additionally, our method can recognize hand poses having both abduction/adduction and flexion movements of fingers in a single camera-based setup. In contrast to prior work on hand pose recognition by a stereo camera or by a depth sensor, the proposed method can efficiently recognize different hand poses by using only a single camera. In contrast with previous gesture recognition methods, the proposed method does not need 3D model reconstruction.

Furthermore, the incorporation of a hand calibration step in the proposed algorithm makes the system more robust against size inconsistency between the model and the user's hand. The probabilistic modeling of the

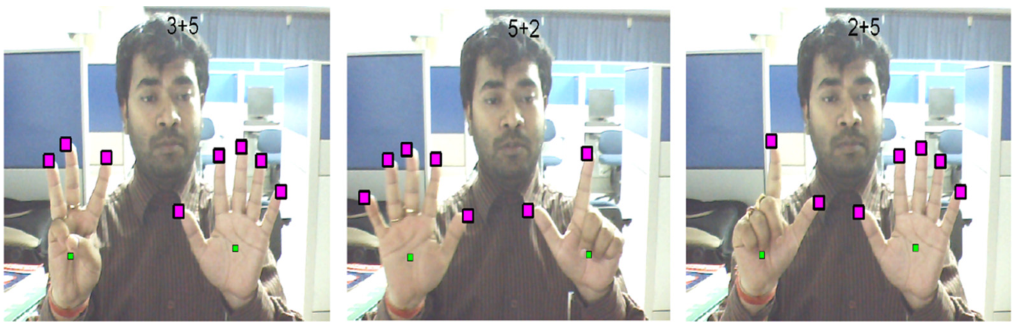

Fig. 22. Detected two-handed fingertips.

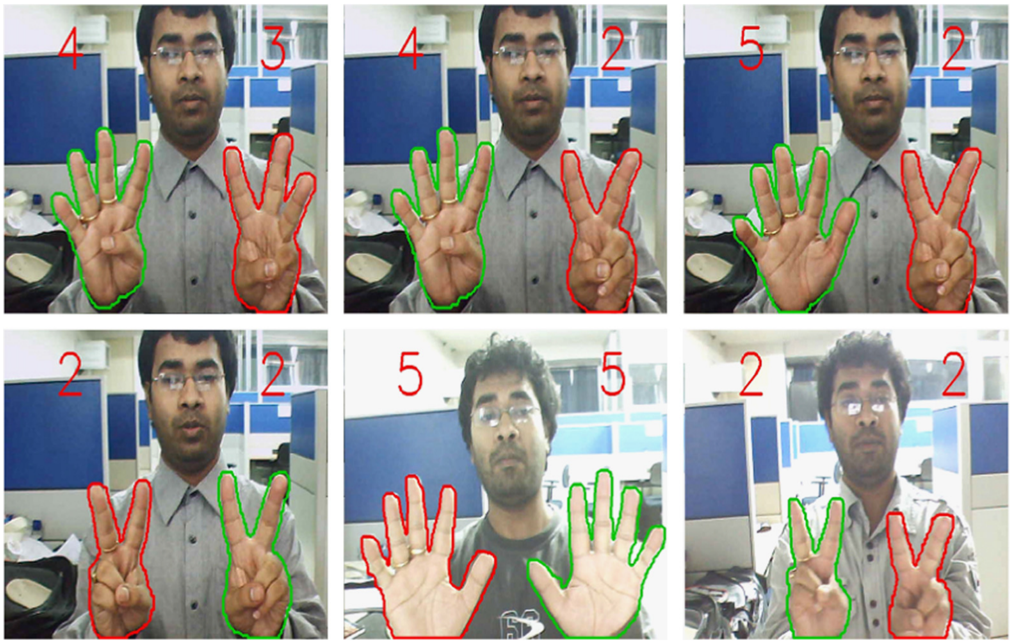

Fig. 23. Results of two-handed pose recognition.

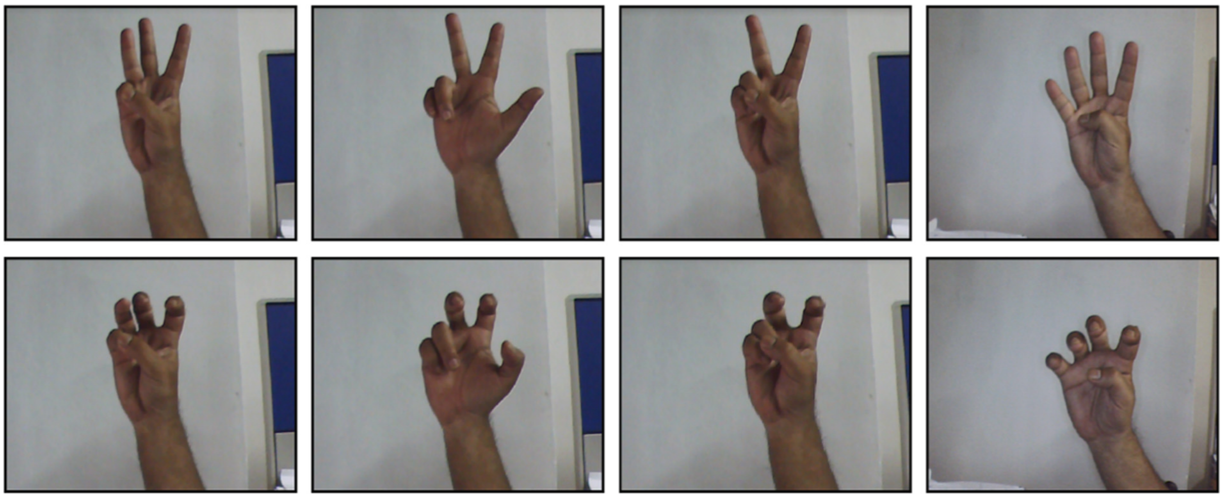

Fig. 24. Hand poses (IMR1, TIM1, IM1, IMRL1, IMR2, TIM2, IM2, and IMRL2) created by flexion and extension movements of the fingers considered for recognition.

abduction/adduction variations is performed to tackle the intra-class variation of gestures. The texture analysis using HTD has also improved the ability of the system to recognize interclass variability due to flexion movements of the fingers. Promising experimental results demonstrate the potential of our approach.

The geometrical features of the hand extracted during the calibration can also provide valuable information for applications related to finger spelling recognition for sign language recognition and hand gesture animation. The high recognition rate has shown the effectiveness of our algorithm over existing methods like silhouette matching-based pose recognition and 3D model-based hand pose recognition.

Appendix A. Supplementary data

Supplementary data associated with this paper can be found in the online version at http://dx.doi.org/10.1016/j.jvlc.2014.12.001.

References

[1] Vladimir I. Pavlovic, Rajeev Sharma, Thomas S. Huang, Visual interpretation of hand gestures for human-computer interaction, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 677–695.

[2] Ali Erol, George Bebis, Mircea Nicolescu, Richard D. Boyle, Xander twombly, Vision-based hand pose estimation: a review, Comput. Vis. Image Understand. 108 (2007) 52–73.

[3] Jintae Lee, Tosiyasu L. Kunii, Model-based analysis of hand posture, IEEE Comput. Graph. Appl. (1995) 77–86.
[4] Dung Duc Nguyen, Thien Cong Pham, Jae Wook Jeon, Finger extraction from scene with grayscale morphology and BLOB analysis, in: Proceedings of the IEEE International Conference on Robotics and Biomimetics, 2008, pp. 324–329.
[5] Sung Kwan Kang, Mi Young Nam, Phill Kyu Rhee, Color-based hand and finger detection technology for user interaction, in: Proceedings of the IEEE International Conference on Convergence and Hybrid Information Technology, 2008, pp. 229–236.
[6] Etsuko Ueda, Yoshio Matsumoto, Masakazu Imai, Tsukasa Ogasawara, A hand-pose estimation for vision-based human interfaces, IEEE Trans. Ind. Electron. 50 (4) (2003) 676–684.
[7] Sung Uk Lee, Isaac Cohen, 3D hand reconstruction from a monocular view, in: Proceedings of the IEEE International Conference on Pattern Recognition, vol. 4, 2004, pp. 310– 313.
[8] Haiying Guan, Chin-Seng Chua, Yeong-Khing Ho, 3D hand pose retrieval from a single 2D image, in: Proceedings of the IEEE International Conference on Image Processing, vol. 1, 2001, pp. 157–160.
[9] Chingyu Weng, Chuanyu Tseng, Chunglin Huang, A vision-based hand motion parameter capturing for HCI, in: Proc. IEEE Int'l Conf. Audio, Language and Image Processing, vol. 14(10), 2008, pp. 1219–1224.
[10] Martin de La Gorce, Nikos Paragios, A variational approach to monocular hand-pose estimation, Comput. Vis. Image Understand. 114 (2010) 363–372.
[11] Martin de La Gorce, David J. Fleet, Nikos Paragios, Model-based 3D hand pose estimation from monocular video, IEEE Trans. Pattern Anal. Mach. Intell. 99 (2011) 1–12.
[12] Chi Xu, Li Cheng, Efficient hand pose estimation from a single depth image, in: Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 3456–3462.
[13] Zhou Ren, Junsong Yuan, Jingjing Meng, Zhengyou Zhang, Robust part-based hand gesture recognition using kinect sensor, IEEE Trans. Multimed. 15 (5) (2013) 1110–1120.
[14] Y. Yao, Y. Fu, Contour model based hand-gesture recognition using kinect sensor, IEEE Trans. Circuits Syst. Video Technol. (99) (2014) 1–10.
[15] C. Kim, J.N. Hwang, Object-based video abstraction for video surveillance systems, IEEE Trans. Circuits Syst. Video Technol. 12 (12) (2002) 1128–1138.
[16] M.K. Bhuyan, Debanga Raj Neog, Mithun Kumar Kar, Hand pose recognition using geometric features, in: Proceedings of the National Conference on Communication (NCC 2011), 2011, pp. 1–5.
[17] Kourosh Jafari-Khouzani, Hamid Soltanian-Zadeh, Rotation-invariant multiresolution texture analysis using radon and wavelet transforms, IEEE Trans. Image Process. 14 (10) (2005) 783–795.
[18] R. Lienhart, J. Maydt, An extended set of haar-like features for rapid object detection, in: Proceedings of the IEEE International Conference on Image Processing, vol. 1, 2002, pp. 900–903.
[19] Y.M. Ro, M.C. Kim, H.K. Kang, B.S. Manjunath, J.W. Kim, MPEG-7 homogeneous texture descriptor, ETRl J. 23 (2) (2001) 41–51.
[20] George M. Haley, B.S. Manjunath, Rotation-invariant texture classification using a complete space–frequency model, IEEE Trans. Image Process. 8 (2) (1999) 255–269.