Int. J. Mach. Learn. & Cyber. (2014) 5:607–623
DOI 10.1007/s13042-013-0158-4

ORIGINAL ARTICLE

Hand gesture recognition and animation for local hand motions

M. K. Bhuyan · V. Venkata Ramaraju ·
Yuji Iwahori

Received: 21 September 2012 / Accepted: 22 February 2013 / Published online: 21 March 2013

Abstract Hand gestures are universally adopted means of communication to convey message in the form of sign language. Therefore, to communicate with a deaf and dumb person, a normal human requires to have some knowledge about the sign language and should be able to make the sign language gestures. By understanding and animating hand gestures, we can help in facilitating communication between computers and the underprivileged. In this paper, we present a method for synthesizing hand gestures with the help of a computer which may enable a normal person to convey massage to a mute person more easily without any knowledge of sign language. The proposed technique requires to train the system prior to its operation. But, gesture animation is computationally complex as it involves replication of the hand with its 27 degrees of freedom. Gesture animation also involves gesture recognition. Hence, in this paper, we have implemented a gesture animation framework after recognizing hand gestures. Computational complexity has been significantly reduced by summarizing large gesture sequence in the form of key frames. The animation process includes hand parameter calculation for every pose in a gesture sequence which is obtained using information like position of fingers, location of metacarpophalangeal joints of the fingers and the bent angles of the fingers. By using these parameters, hand pose estimation is done by imposing some constraints of the hand. Subsequently, a gesture sequence is animated using these models. For this, the hand model for the frames in between the key frames are obtained by interpolation. In our experiment, we demonstrate gesture animation with hand pose exactly same as the real gesture.

Keywords Hand gesture · Gesture animation · Hand model · Finger pose estimation

M. K. Bhuyan (✉) · V. V. Ramaraju
Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati, Guwahati 781039, India
e-mail: mkb@iitg.ernet.in

V. V. Ramaraju
e-mail: vana@iitg.ernet.in

Y. Iwahori
Department of Computer Science, Chubu University, 1200 Matsumoto-cho, Kasugai 487-8501, Japan
e-mail: iwahori@cs.chubu.ac.jp

1 Introduction

Gestures play an important role in our day to day communications. Hand gestures play the same role for deaf and dumb people as speech for normal people. They are used by the deaf and dumb people in the form of sign language. So, in order to communicate with deaf and dumb people, normal human should have some knowledge about sign language. This will require proper training of every normal person who wishes to communicate with mute person. This seems to be very much impractical and that is why, we need an automatic sign language translation system, which can translate speech into sign language and vice-versa. The idea is to build up a messenger system which can be used as a means for interaction between mute and normal person. In this system, computer should be able to recognize gestures and convert a particular sign to its corresponding message in the form of speech. This will help a dumb person to convey message to a normal person. On the other hand, the system should be able to recognize any spoken word/sentence or text and create the corresponding sign (hand gesture) through animation. This will help a normal being to communicate with a deaf person. So, the system will mainly act as an interpreter between a normal being and a mute person. That is why, automatic interpretation and animation of hand gestures has become an important research topic among researchers in virtual reality and computer animation.

Typical schematic diagram of such system is shown in Fig. 1. Such system has mainly two modules. One for translating speech/text to sign language and, the second module is for translating sign language to speech/text. Research on gestures is basically concentrated on two spheres namely gesture recognition and gesture animation. Both of them aim to facilitate communication between humans and computers. Gesture animation is simply opposite of gesture recognition. In the case of recognition, user has to perform some gestures and system has to recognize them and perform some actions correspondingly. Whereas, in the case of gesture animation, system will perform gestures and user has to recognize and understand them. In the system shown in Fig. 1, first module deals with gesture animation. Whereas, the second module deals with gesture recognition.

As a first step towards gesture animation, configuration of hand should be measurable by machine. For this step, some proposals involved the use of glove-based devices. But, glove-based gestural interface requires the user to wear a cumbersome glove that carries a load of cables connecting it to the computer. This hinders the naturalness with which the user can interact with the computer. Awkwardness in using gloves is overcome by using vision-based non-contact interaction techniques. They use color-based vision segmentation, silhouettes or edges to track the hand and fingers.

Hand gesture animation is relatively new research topic. Very few research works have been reported on gesture animation. Some of the earlier works on extraction of hand pose information and hand motion animation are given in [1–6]. They involve the complexity of having large data to be processed. For gesture synthesis, estimation of human hand posture is quite important and the research in this area is not yet advanced enough to provide a flexible and reliable solution. Mainly two research papers [7] and [8] extensively deal with this very specific research topic. In [7] and [8], a simple approach for finding hand pose from frontal view was proposed. But, all these methods could not provide a complete solution to the gesture animation process. Furthermore, these methods only deal with single-handed gestures. In view of this, we now propose a scheme in which the computational cost is highly reduced and the gestures are animated more efficiently.

In order to reduce computational complexity, large gesture sequence are summarized by only key few frames. The key frames are selected on the basis of the amount of change in hand shape—for a given key frame in the sequence the next key frame is the one in which the hand changes its shape significantly. Thus, an entire video clip is transformed into a small number of representative frames that are sufficient to represent a gesture sequence. These frames are the key frames that best represent the content of the sequence in an abstracted manner. In our method, key frames are extracted using discrete Fourier transform

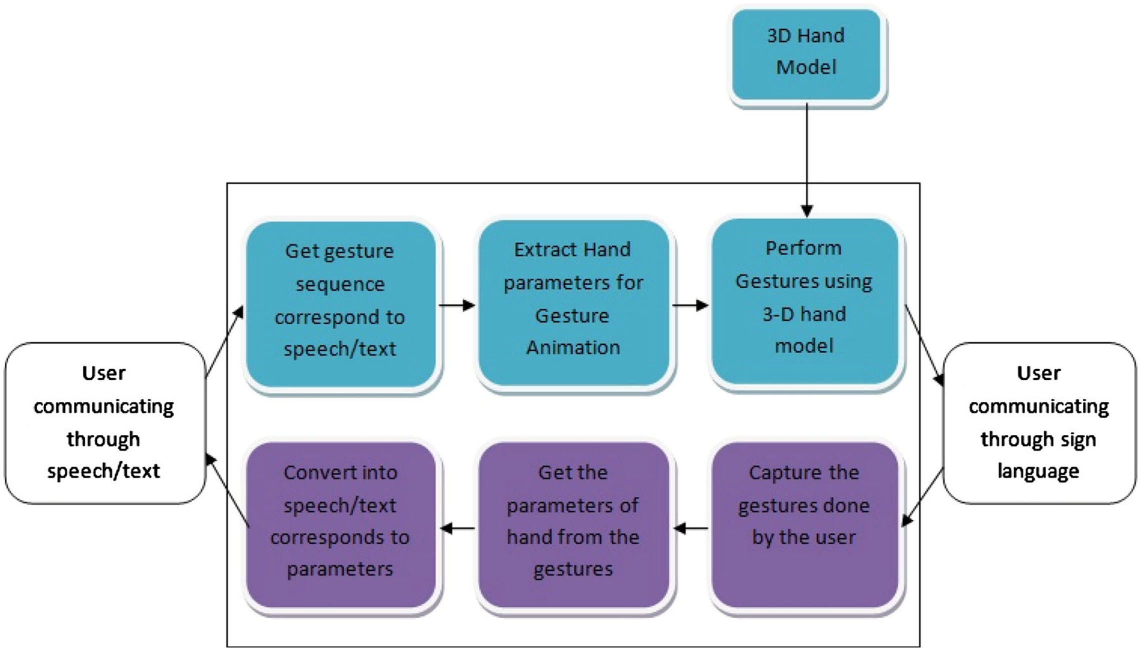

3D Hand Model
Get gesture sequence correspond to speech/text
Extract Hand parameters for Gesture Animation
Perform Gestures using 3-D hand model
User communicating through speech/text
User communicating through sign language
Convert into speech/text corresponds to parameters
Get the parameters of hand from the gestures
Capture the gestures done by the user

Fig. 1 A typical sign language communication system

(DFT) in combination with vertical projection profile (VPP). Using these key frames only, entire gesture sequence has to be reconstructed and animated by extracting some features from each key frame.

We present a method for separating hand from the arm, using "Distance Transform" followed by largest circle fit method. After separating hand from the arm, metacarpo-phalangeal (MP) joints of the hand are extracted using the corner detection technique. Finger tips are found by using contour detection. By using extracted MP joint locations and finger tips as parameters, hand pose is estimated.

Subsequently, hand pose is estimated using the hand model with 27 degrees of freedom and by imposing some constraints of the hand. Thus, the 3D hand model for a particular gesture in a key frame is obtained. Since, the estimation of hand pose is only done for key frames, we do not have any information about the pose of the hand for the frames between two consecutive key frames. For this, the hand model for the frames in between the key frames are obtained by interpolation. Finally, the gesture sequence is animated using these models and the extracted hand parameters. Our proposed method is also applied for the animation of two hands. This is successfully demonstrated in our experimental results. The proposed system was implemented using two basic modules:

- The image processing module was implemented in C with the help of OpenCV library.
- The computer graphics/animation module was implemented in C with the help of OpenGL library.

The final step is the linking and interfacing of the aforementioned modules.

The organization of the rest of the paper is as follows. In Sect. 2 we present our proposed animation method. Section 2.6 shows the approach of key frame extraction. Experimental results are shown in Sect. 3 Finally, we draw our conclusion in Sect. 4.

2 Proposed gesture animation system

The basic block diagram of the proposed hand gesture animation system is shown in Fig. 2. From the input gesture video sequence, hand region is extracted by applying skin color-based segmentation. Next, hand is separated from the arm. After extracting the key frames, hand parameters are extracted, which are subsequently used for construing the hand pose. Finally, the animation is done by exporting all the hand parameters to the animation framework which was developed in the OpenGL platform. The details of all these steps are discussed in the sections to follow.

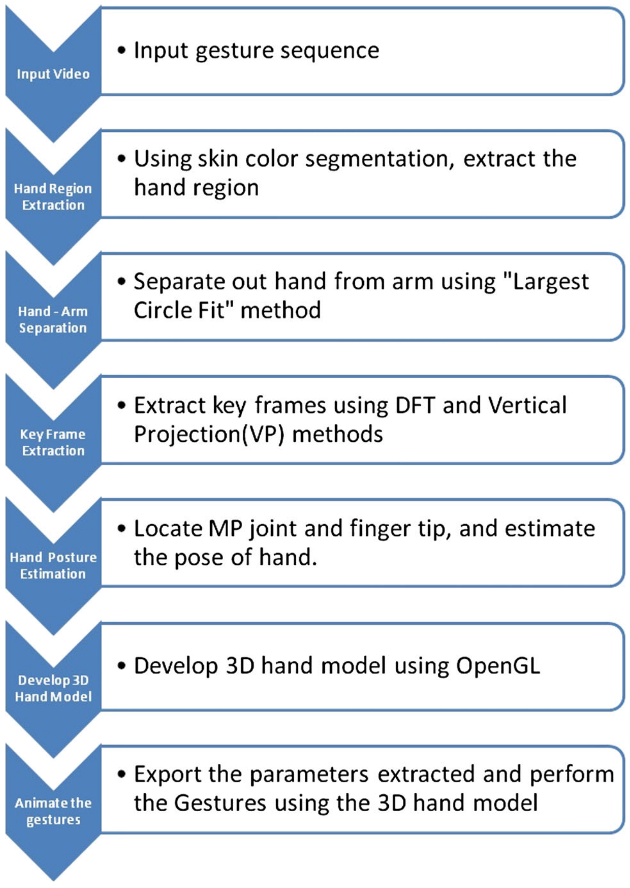

Input Video
• Input gesture sequence
Hand Region Extraction
• Using skin color segmentation, extract the hand region
Hand - Arm Separation
• Separate out hand from arm using "Largest Circle Fit" method
Key Frame Extraction
• Extract key frames using DFT and Vertical Projection(VP) methods
Hand Posture Estimation
• Locate MP joint and finger tip, and estimate the pose of hand.
Develop 3D Hand Model
• Develop 3D hand model using OpenGL
Animate the gestures
• Export the parameters extracted and perform the Gestures using the 3D hand model

Fig. 2 A schematic diagram of our proposed approach

2.1 Human hand modeling

Model-based approach for image analysis is common in computer vision applications. However, computer analysis of hand posture from actual images involves complex problems even when employing conventional and simple models. The human hand is an articulated structure with about 27 degrees of freedom and hand changes its shape in various ways by its joint movements. Hand images change by both finger movements and hand movement as a whole. However, it is also highly constrained i.e., as the hand is incapable of making arbitrary gestures. There are many examples of such constraints [3, 9, 10]. For instance, fingers cannot be bent backwards too much and the pinky finger cannot be bent without bending the ring finger. The natural movements of human hands are implicitly caused by such motion constraints. Analysis of hand constraints and inter-relationship is essential to avoid unrealistic motions during hand animation. So, it is important to implement the 3D model of human hand in accordance with its constraints in OpenGL. The inputs to this model are just the joint angels of all the fingers. Inevitably, a trade-off arises between the degree of constraints contained in a model and its resultant performance. A lack of constraints leads to a useless model, whereas too many of them require complex procedures necessitating expensive computation time. Our hand model, therefore, attempts to effectively balance these considerations.

2.2 Hand skeletal model and it's constraints

Human hand consists of 27 bones; consequently there are 27 degrees of freedom (DoF) [9,11]. Each of the four fingers has four DoF. The distal interphalangeal (DIP) joint and proximal interphalangeal (PIP) joint each has one DoF and the metacarpophalangeal (MCP) joint has two DoF due to flexion (F) and abduction (AA). Flexion refers to bending of fingers with one DoF and abduction refers to lateral movement of fingers with one DoF. The thumb has a different structure which has five degrees of freedom, one for the interphalangeal (IP) joint, and two for each of the thumb MCP joint and trapeziometacarpal (TM) joint both due to flexion and abduction. These all add up to 21 DoF. The remaining 6 degrees of freedom are from the rotational and translational motion of the palm with 3 DoF each. These six parameters can be ignored, since we will only focus on the estimation of the local finger motions rather than the global motion. In Fig. 3, a skeletal hand model is shown with DoF.

Hand constraints can be roughly divided into three types. Type I constraints are the limits of finger motions as a result of hand anatomy which is usually referred to as static constraints. Type II constraints are the limits imposed on joints during motion, which is usually referred to as dynamic constraints. Type III constraints are applied in performing natural motion, which have not yet been explored. In this paper, we have only considered Type I and Type II constraints.

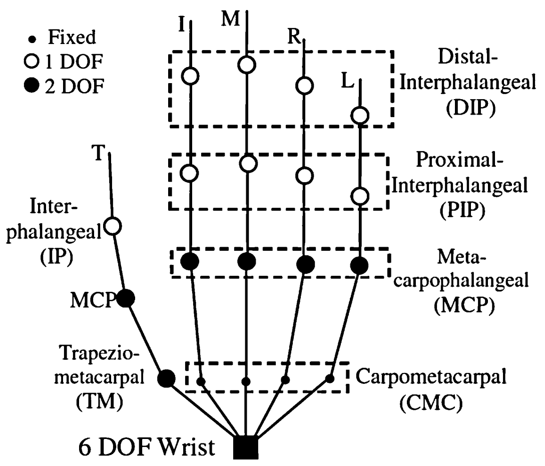


Fig. 3 A skeletal model of hand showing DoF of each joint [9, 11]

2.2.1 *Type-I constraints*

This type of constraint refers to the limits of the range of finger motions as a result of hand anatomy. We have only considered the range of motion of each finger that can be achieved without applying external forces such as bending of fingers backward using the other hand. This type of constraints are usually represented using the following inequalities:

$$\begin{aligned} 0^\circ &\le \theta_{MP-F} \le 90^\circ \\ 0^\circ &\le \theta_{PIP-F} \le 110^\circ \\ 0^\circ &\le \theta_{DIP-F} \le 90^\circ \\ -15^\circ &\le \theta_{MCP-AA} \le 15^\circ \end{aligned} \tag{1}$$

A commonly adopted constraint states that middle finger displays little abduction/adduction motions and the following approximation is made for middle finger:

$$\theta_{MCP-AA} = 0^\circ \tag{2}$$

This will reduce one DoF from the 21 DoF model. Similarly, the TM joint of thumb also displays limited abduction motion and will be approximated by 0 as well.

$$\theta_{TM-AA} = 0^\circ \tag{3}$$

2.2.2 *Type-II constraints*

This type of constraint refers to the limits imposed on joints during finger motions. These constraints are often called dynamic constraints and can be subdivided into intra-finger constraints and inter-finger constraints. The intra-finger constraints are the constraints between joints of the same finger. A commonly used constraint which is based on hand anatomy states that in order to bend the DIP joints, the PIP joints must also be bend for the index, middle, ring and little fingers. The relations can be approximated as following:

$$\theta_{DIP} = \frac{2}{3} \times \theta_{PIP} \tag{4}$$

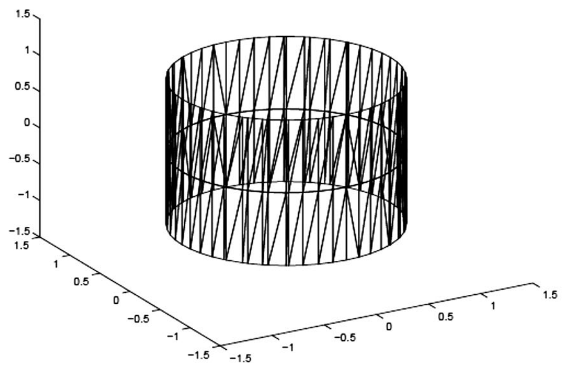
If we impose all these constraints, we can reduce the representation of our hand model to 15 DoF only.

2.3 Hand modelling

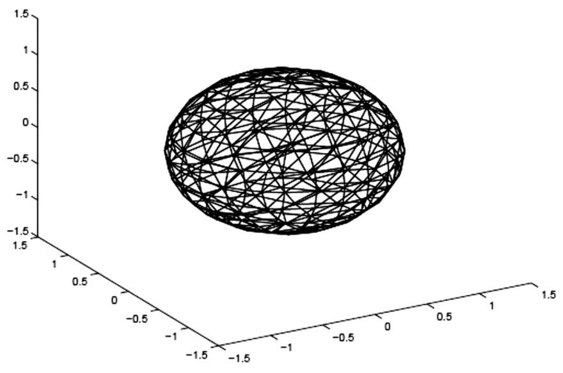
A three-dimensional hand model is constructed using truncated quadrics as building blocks, approximating the anatomy of a real human hand.

2.3.1 *Quadrics and conics*

A quadric **Q** is a second degree implicit surface in 3D space. It can be represented in homogeneous coordinates as



(a) Cylinder



(b) Sphere

Fig. 4 **a**, **b** Cylinder and sphere constructed from quadrics

a symmetric 4 × 4 matrix $\mathbf{Q}$. The surface is defined by the points X, which satisfy the equation

$$\mathbf{X}^T\mathbf{Q}\mathbf{X} = 0 \tag{5}$$

where $\mathbf{X} = [x, y, z, 1]^T$ is a homogeneous vector representing a 3D point. A quadric has 9 degrees of freedom. Accordingly, Eq. 5 may be expanded as:

$$\begin{aligned} &q_{1,1}x^2 + q_{2,2}y^2 + q_{3,3}z^2 + 2q_{1,2}xy + 2q_{1,3}xz + 2q_{2,3}yz \\ &\quad + 2q_{1,4}x + 2q_{2,4}y + 2q_{3,4}z + q_{4,4} \\ &\quad = 0 \end{aligned} \tag{6}$$

Different families of quadrics are obtained are obtained from matrices $\mathbf{Q}$ of different ranks, as described below.

2.3.2 *Ellipsoid*

Ellipsoids are represented by matrices $\mathbf{Q}$ with full rank. The normal implicit form of an ellipsoid centered at the origin and aligned with the coordinate axes is given by

$$\frac{x^2}{w^2} + \frac{y^2}{h^2} + \frac{z^2}{d^2} = 1, \tag{7}$$

and the corresponding matrix is given by

$$\begin{pmatrix} \frac{1}{w^2} & 0 & 0 & 0 \\ 0 & \frac{1}{h^2} & 0 & 0 \\ 0 & 0 & \frac{1}{d^2} & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{8}$$

Some other examples of surfaces which can be represented by matrices with full rank are paraboloids and hyperboloids. If matrix $\mathbf{Q}$ is singular, the quadric is said to be degenerate.

2.3.3 *Cones and cylinders*

Cones and cylinders are represented by matrix $\mathbf{Q}$ with rank $(\mathbf{Q}) = 3$. The equation of a cone aligned with the y-axis is given by

$$\frac{x^2}{w^2} - \frac{y^2}{h^2} + \frac{z^2}{d^2} = 1, \tag{9}$$

and the corresponding matrix is given by

$$\begin{pmatrix} \frac{1}{w^2} & 0 & 0 & 0 \\ 0 & \frac{-1}{h^2} & 0 & 0 \\ 0 & 0 & \frac{1}{d^2} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{10}$$

An elliptic cylinder, aligned with the y-axis, is described by

$$\frac{x^2}{w^2} + \frac{z^2}{d^2} = 1, \tag{11}$$

and the corresponding matrix is given by

$$\begin{pmatrix} \frac{1}{w^2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{d^2} & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{12}$$

Thus, different shapes can be obtained by using different matrices. A cylinder and a sphere can be constructed using above equations are shown in Fig. 4a, b, respectively. In this paper, we propose to build up the 3D model of the hand by combining such different shapes in OpenGL.

2.4 Hand region extraction

As a first step towards hand parameter extraction, hand region should be separated out from the background. Using skin color-based segmentation, hand region can be separated out from the complex background effectively [12]. Skin color is a unique feature, which is also rotation and scale invariant. Also, skin color is clustered in very small region in color space [13], it is easy to separate skin regions by proper thresholding. Although different people have different skin colors, some studies showed that difference largely lies in intensity rather than in color itself [14]. So, a color space should be chosen where chroma (color and hue) and intensity information is separable. In this paper, HSI and HSV color spaces are used for identifying skin regions. Mostly skin chroma of Asians is in between red and yellow. In [15], it is given that mostly their skin color falls in the range of $[105°, 150°]$. A pixel is classified as a skin pixel if the following conditions are satisfied.

$$\begin{aligned} &H(i) < 22 \\ &30 \leq I(i) \leq 100 \\ &105° \leq \theta(i) \leq 150° \end{aligned} \tag{13}$$

where $H(i)$, $I(i)$ represents Hue, intensity at ith pixel. The parameter $\theta(i)$ is given by,

$$\theta(i) = \tan^{-1}(V(i)/U(i))$$

Subsequently, morphological operations are performed to filter out unwanted small regions.

2.5 Hand arm separation

As the global motions of the hand are not at all considered for animation, the presence of the arm in the segmented hand is not important. For this purpose, only the palm region of the hand is required. So, the arm has to be separated from the segmented image. “Largest Circle Fit” method is used for separating the the arm [16], [17]. In this method, the largest circle which can be inscribed completely in the extracted hand region is found out. Ultimately palm region is the widest part of the hand. So, the largest circle will definitely be lying inside the palm. Subsequently, a tangent to that circle is drawn and the portion of the hand right to that tangent is found out. The radius and center of the largest circle can also be determined by using “Distance Transform”.

2.5.1 Distance transform

As its name indicates, it is a transform which takes a binary image, consisting of feature and non-feature pixels as an input and produces a gray level image as output such that intensity value at any pixel of the output image corresponds to the distance to the nearest feature pixel of the input image [18]. “Distance Transform” gives the skeleton of the input object. An efficient implementation of distance transform can be done using Chamfer mask. This algorithm uses small masks containing integer approximations of distances in a small neighborhood thereby converting a binary image to an approximate distance image. This is called distance transformation (DT). As shown in Fig. 5, there are two such masks, Chamfer 3–4 and Chamfer 5–7–11, that are used in the process. The horizontal, vertical and the diagonal distances in each of these masks are indicated in the figure. For example, the horizontal and vertical distances for Chamfer 3–4 are 3 each and the diagonal distance is 4. This gives a ratio of 1.333 compared to 1.414 for Euclidean (exact) distances.

In our method, we have used Chamfer mask 5–7–11 for calculating the distance transform. In Fig. 6, distance transform is shown for different hand images.

2.5.2 Steps for finding largest circle

Following steps are done to find the largest circle.

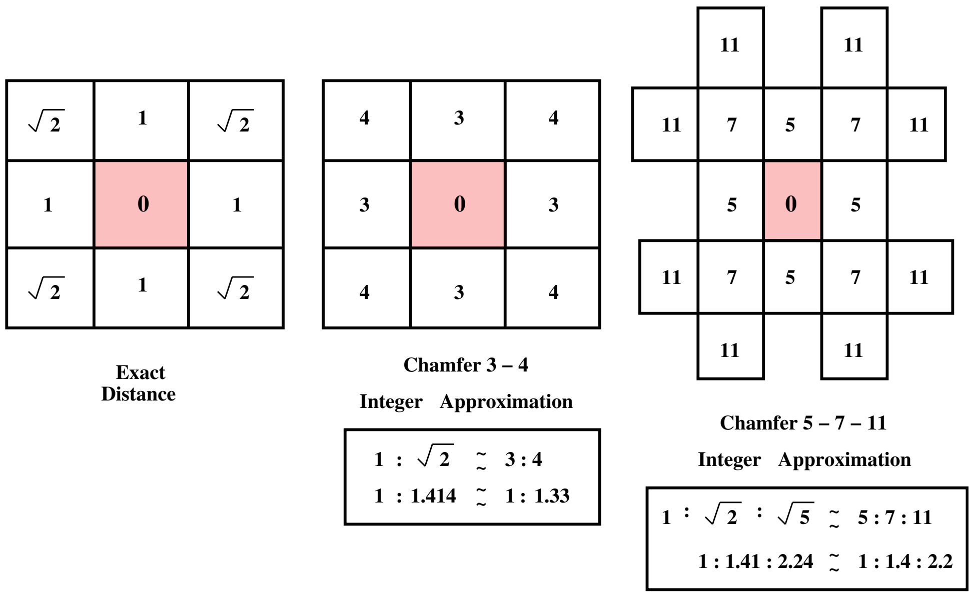
- Find the edge of the input hand image using Canny edge detector.
- Find the distance transform of the edge image.
- Normalize the distance transformed image into the range [0, 255].
- Search for the pixel having an intensity value 255. This point is the center of the largest circle to be inscribed in the hand.
- Find the edge pixel of the hand, which is at a smaller distance from the estimated center of the circle.
- Radius of the largest circle is the distance of the point obtained in the previous step to the center of the circle.

Because of normalization and geometry of hand, there may be more than one pixel having an intensity of 255. To solve this ambiguity, the center of mass of all the pixels having an intensity of 255 may be considered. This centroid may be considered as the center of largest circle inscribed in the palm region. Then, the pixels around the center of the palm will have the intensity value 255 after normalization.

2.6 Key frame extraction for gesture summarization

In the original gesture video sequence corresponding to a particular sign, we have a large number of frames. But, a real time hand gesture made by a signer is generally composed of slow and gentle movements of different parts of the hand. This implies that there is very small change in the hand pose from frame to frame resulting in large amount of temporal redundancy. Hence, an entire video sequence/clip can be compressed by a large amount by discarding frames which do not show much change in the hand pose and keeping only those frames in which the change in the hand pose is quite significant/noteable. These frames are referred to as key frames. So, the whole gesture sequence is summarized by using the key frames. Subsequently, information about hand shape parameters are extracted from these key frames only. This greatly reduces the memory requirement for storing gesture description of every sign in the dictionary. Also, this helps in reducing the total time required for training.

Fig. 5 Chamfer masks used for computing distance transformation


Exact Distance
Chamfer 3 – 4
Integer Approximation
$1 : \sqrt{2} \approx 3 : 4$
$1 : 1.414 \approx 1 : 1.33$
Chamfer 5 – 7 – 11
Integer Approximation
$1 : \sqrt{2} : \sqrt{5} \approx 5 : 7 : 11$
$1 : 1.41 : 2.24 \approx 1 : 1.4 : 2.2$

In this paper, we have proposed to use discrete Fourier transform (DFT) in combination with "Vertical Projection Profile (VPP)" for the purpose of comparing images. This in turn is used for identifying the key frames. The reason for choosing DFT is that, it offers specific advantages when comparing in frequency domain rather than directly comparing in spatial domain itself. The magnitude spectrum does not get affected by rotation and translation, unless there is a change in the information of image. So, unless there is a significant change in the pose of the hand, histogram of the magnitude spectrum of DFT will remain almost same. Magnitude spectrum of DFT of an input frame is shown in Fig. 7. Additionally, Figs. 8 and 9 show the magnitude spectrums of DFT of the translated and rotated versions of this frame respectively. Images similarity can not be judged by comparing corresponding points in the magnitude spectrum as rotation causes the magnitude spectrum to rotate by the same amount. Hence, histogram of the magnitude spectrum is considered for comparison. Distance between these histograms is a measure of similarity. To reduce dynamic range of coefficients, the logarithm of the magnitude spectrum is considered and the histogram is constructed. The histogram of the magnitude spectrum of the hand image is shown in Fig. 10. Since, we have taken logarithmic values of magnitude spectrum on x-axis, it's dynamic range is reduced. Fig. 11 also shows the histogram of the magnitude spectrum of different hand images.

Additionally, vertical projection profile (VPP) as described in [7] is also used for similarity measurement. VPP gives the number of object pixels in each column and two images are compared by using VPP.

By our convention, we start by taking the very first frame in the gesture sequence as the first key frame. We

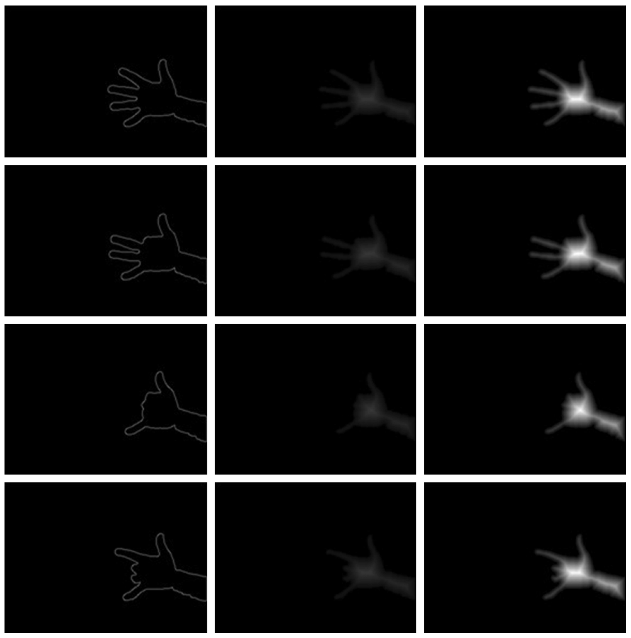


Fig. 6 Edge images of hand (*column 1*); distance transform of edge images (*Column 2*); and the normalized distance transform (*column 3*)

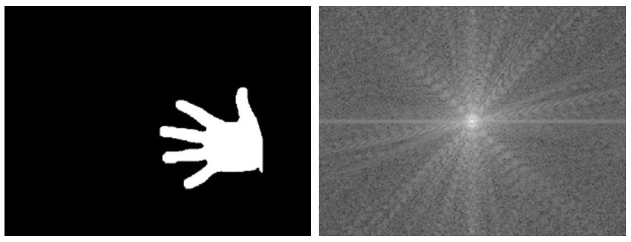


Fig. 7 Input frame (*left*); magnitude spectrum of it's DFT (*right*)

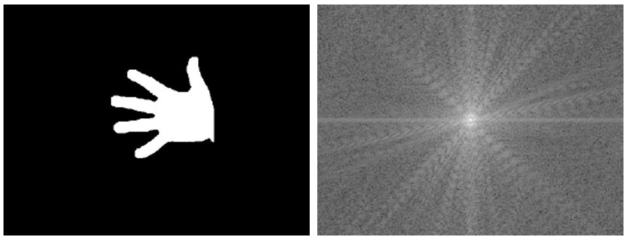


Fig. 8 Input frame translated (*left*); magnitude spectrum of it's DFT (*right*)

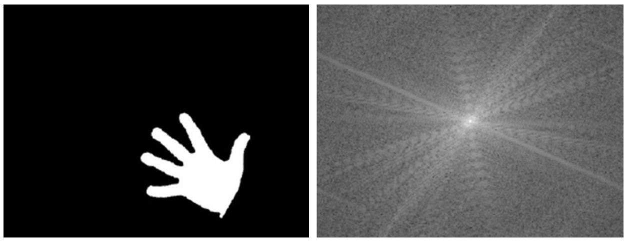


Fig. 9 Input frame rotated (*left*); magnitude spectrum of it's DFT (*right*)

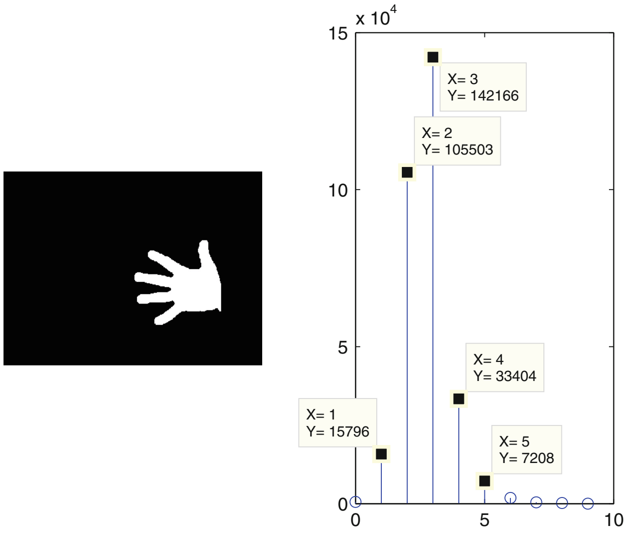


Fig. 10 Input frame (*left*); histogram of magnitude spectrum of it's Fourier transform (*right*)

then calculate the distance between histograms of the logarithm of magnitude spectrum. The distance is computed between this key frame and the next frame, i.e., the second frame, in the sequence. Similarly, the VPP of both of the frames are determined and then, the distance between these VPPs are computed. If the distance measure exceeds a predefined threshold, then this candidate frame is taken as the second key frame. Else, we continue comparing all subsequent frames with the first key frame until we obtain the second key frame. Once a new key frame is detected, the same procedure is continued to identify the next key frame in the sequence with the last identified key frame as the reference frame and its subsequent frames as the candidate frames.

Again, VPP depends on the spatial position and shape of the hand. Hence, point to point correspondence between VPPs should not be done as the VPP is circularly shifted for the variations in the spatial position of the hand. To tackle this problem, some key points are selected for comparison. These key points include center of the hand, global maximum in VPP, left most non-zero entry in VPP etc. After selecting some key points, these VPPs are aligned so that these key points coincide and subsequently, comparisons are done. The VPP of different frames of the gesture video sequence are as shown in Fig. 12.

2.7 Hand parameter calculations

2.7.1 *MP joint extraction*

Once the key frames are extracted, we now proceed with deriving hand parameters from these key frames. The hand parameters include information about different angles of different fingers, thus describing the hand pose and shape in each key frame. Using these parameters we can estimate the pose of hand.

As a next step towards hand feature extraction, MP joints of the hand should be identified. To perform this task, we use a corner detection technique. To start with, the distance transform of the palm region is found out. Subsequently, we obtain an image by thresholding, where there are some sharp points which correspond to MP joints of the fingers. So, these corner points are to be detected to find the approximate locations of the MP joints. In this paper, we propose to use Harris corner detection technique, which is invariant to translation and rotation. The corner detector is based on the local auto-correlation function of a signal, where the local auto correlation function measures the local changes of the image with patches shifted by a small amount in different directions. Since, in order a point to be a corner point, it should be an edge image. We can use Canny edge detection technique to obtain edge of the hand. Now, in order to know whether or not a particular point on the edge is a corner point, observe how the variation is present at that point in all directions. Given a shift $(\triangle x, \triangle y)$ and a point (x, y), the auto-correlation function is defined as,

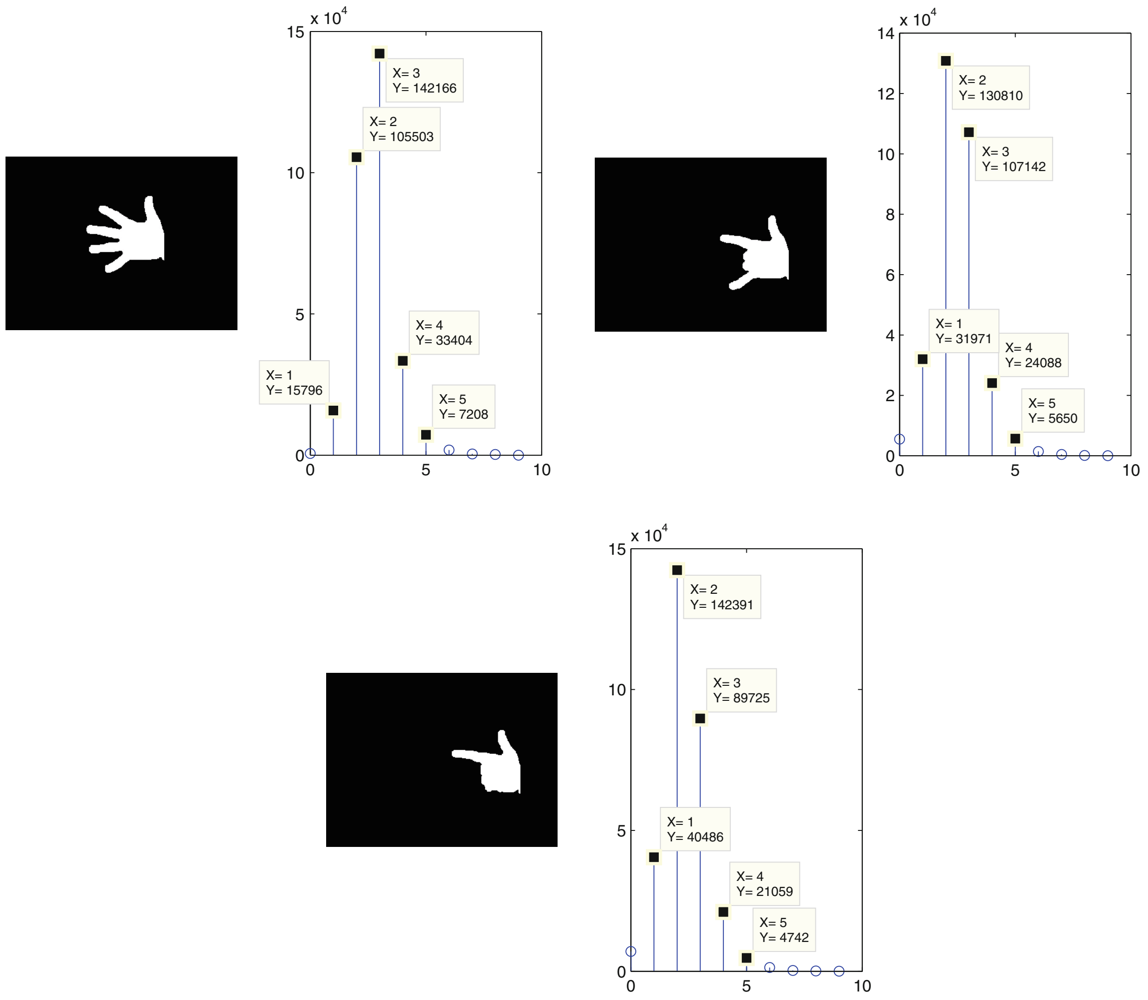


Fig. 11 Input frame (*left*); magnitude spectrum of it's Fourier transform (*right*)

$$c(x,y)=\sum_{W}[I(x_i,y_i)-I(x_i+\triangle x,y_i+\triangle y)]^2 \tag{14}$$

where, $I(..,..)$ denotes the image function and $(x_i,y_i)$are the points in the Gaussian window[1] W centered at (x, y).

The shifted image is approximated by a Taylor expansion truncated to the first order terms, as given below.

$$I(x_i+\triangle x,y_i+\triangle y)\approx I(x_i,y_i) + (I_x(x_i,y_i)I_y(x_i,y_i))\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix} \tag{15}$$

where, $I_x(..,..)$ and $I_y(..,..)$ denote the derivatives with respect to x and y, respectively. Substituting approximation Eq. (15) in (14) yields

$$\begin{aligned}c(x,y)&=\sum_{W}[I(x_i,y_i)-I(x_i+\triangle x,y_i+\triangle y)]^2\\ &=\sum_{W}\left((I(x_i,y_i)-I(x_i,y_i)-(I_x(x_i,y_i)I_y(x_i,y_i))\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}\right)^2\\ &=\sum_{W}\left(-(I_x(x_i,y_i)I_y(x_i,y_i))\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}\right)^2\\ &=(\Delta x\Delta y)\begin{pmatrix}\sum_{W}(I_x(x_i,y_i))^2 & \sum_{W}I_x(x_i,y_i)I_y(x_i,y_i)\\ \sum_{W}I_x(x_i,y_i)I_y(x_i,y_i) & \sum_{W}(I_y(x_i,y_i))^2\end{pmatrix}\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}\\ &=(\Delta x\Delta y)C(x,y)\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}\end{aligned} \tag{16}$$

where matrix $C(x, y)$ captures the intensity structure of the local neighborhood. Let λ_1, λ_2be the eigenvalues of the matrix $C(x, y)$. The eigenvalues form a rotationally invariant description. There are three cases to be considered, as follow.

[1] For simplicity, the Gaussian weighting factor $e^{-(x^2+y^2)/(2\sigma^2)}$ is omitted from the derivation.

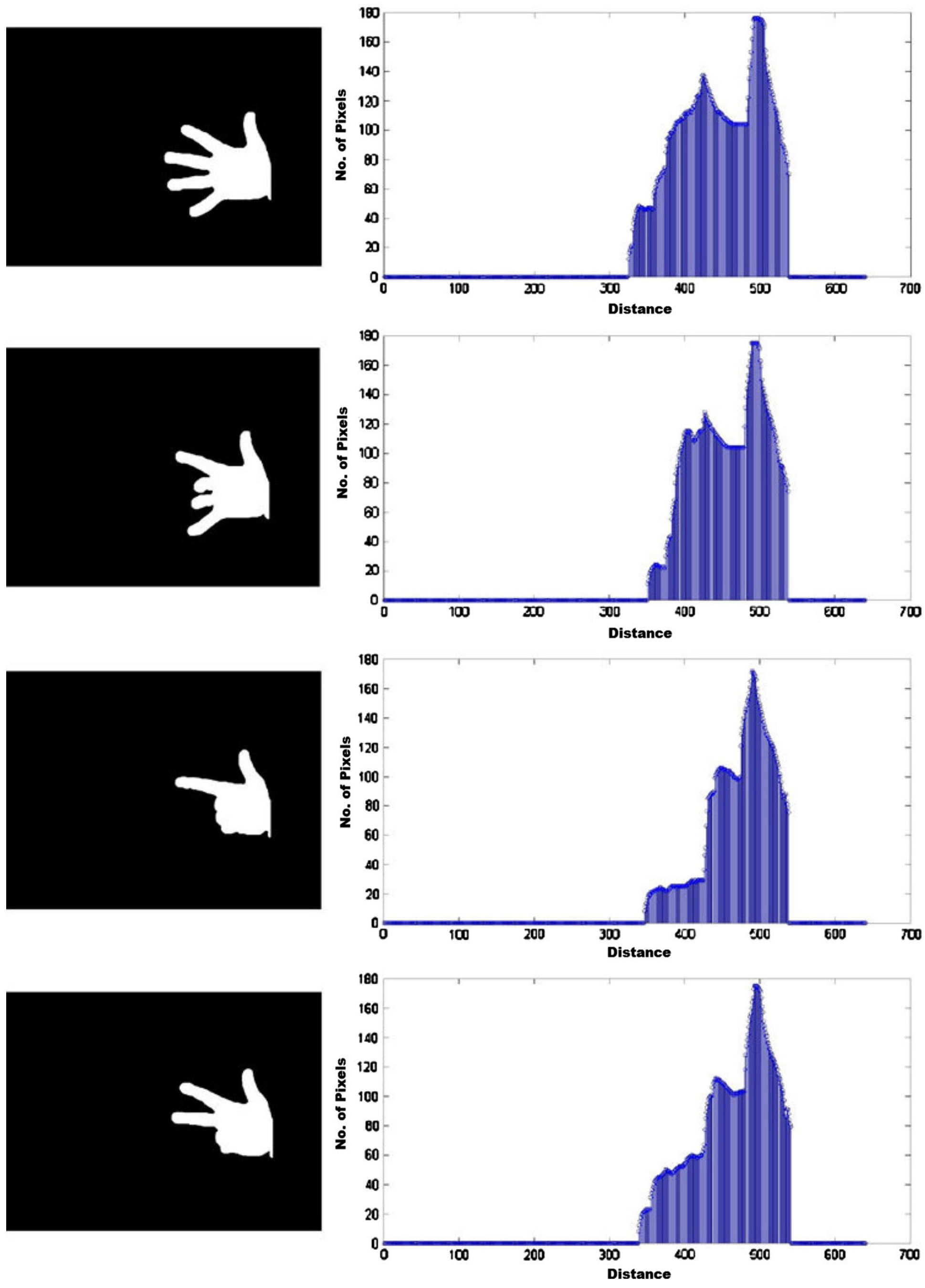


Fig. 12 *Column 1* input frames; *column 2* vertical projection profile

- If both λ_1, λ_2 are small, so that the local auto-correlation function is flat (i.e., little change in $c(x, y)$ in any direction), the windowed image region is of approximately constant intensity. This indicates a flat region.
- If one eigenvalue is high and the other is low, so that the local auto-correlation function is ridge shaped, then only local shifts in one direction (along the ridge) cause little change in $c(x, y)$ and significant change in the orthogonal direction. It is an indication of presence of an edge.
- If both eigenvalues are high, so that the local auto-correlation function is sharply packed, then shifts in any direction will result in a significant increase in $c(x, y)$. This indicates a corner.

A measure of corner response can be defined as

$$R = \mathrm{Det}(C) - k(\mathrm{trace}(C))^2 \tag{17}$$

$$\text{where,}\quad \mathrm{Det}(C) = \lambda_1, \lambda_2 \tag{18}$$

$$\mathrm{trace}(C) = \lambda_1 + \lambda_2 \tag{19}$$

and k is the empirical constant whose value is between 0.04 to 0.06. Again, R depends on the eigenvalues of C. Finally, flat, edge or corner regions can be classified based on the value of R as follow.

- R is large for a corner;
- R is negative with large magnitude for an edge; and
- $|R|$ is small for a flat region.

2.7.2 *k-curvature*

The corner detection method discussed above detects all the corner points including the MP joints in the input threshold image. So, this method gives all such points where there is a change in the contour direction. Hence, there may be some extra points in addition to MP joints. To eliminate unnecessary points, we have used a method based on "k-curvature". As MP joints correspond to a very sharp point, the curvature value is very small at the MP joints. Whereas, the curvature is very high at other corners points. So, we can eliminate false corners by thresholding the curvature value. The technique used to eliminate unnecessary corners points to detect valid MP joints is discussed below.

Let C represents a contour, and $C(i)$ is the ith point on the contour. The k-curvature at that point is defined as the angle between the vectors $[C(i), C(i-k)]$ and $[C(i), C(i+k)]$. The k-curvature can be computed quite easily, since the angle between two vectors can be found by using dot product of the vectors. There will be sharp peaks corresponding to MP joints. Hence, k-curvature values will be very low at those points. By keeping proper threshold, we can separate out the corners points belonging to the MP joints. The performance of this algorithm depends on the choice of 'k'.

2.7.3 *Finger tips detection*

Once, MP joints are located, finger tips can be found easily. To find the finger tips, the fingers from the palm are separated out [19]. Next, the contours of the separated fingers are found out. For each contour, centroid (center of mass) is estimated and the nearest MP joint from the corresponding centroid for all the contours is found out. Finally, we can find the point on the contour, which is farthest from the MP joint. All these points are the tips of the fingers. Obviously, the tip obtained from a contour and the nearest MP joint belong to a particular finger. So, the distance between these two points gives the finger pose length.

2.7.4 *Correspondence problem*

Either MP joints or finger tips individually or jointly can not give exact information about which finger they corresponds to. In order to solve this ambiguity, some radial range for each MP joint is considered. Let us consider the simple case of horizontal orientation of the hand. Angular ranges are defined to identify the fingers as shown in Fig. 13. Let us assume that, the center of the largest circle inscribed coincides with the origin of the reference co-ordinate system as shown in Fig. 13. Subsequently, the angle made by the position vector of all the MP joints with X-axis is determined. MP joint of thumb (if present) falls in the first quadrant of the co-ordinate system. Hence, we can fix the range for MP joint of thumb as $(0°, 90°)$. Similarly, the angular range for MP joints of index, middle, ring and little fingers are $(90°, 150°), (150°, 185°), (185°, 215°)$and $(215°, 270°)$respectively.

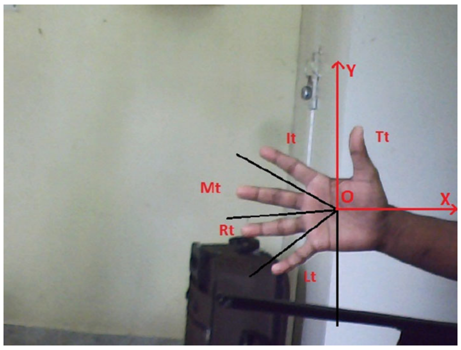


Fig. 13 Regions showing possible MP joint locations of all the fingers

2.8 Hand posture estimation

Next step of our proposed scheme is to determine finger joint angles from the information of MP joints and finger tips. The joint angles of the fingers are calculated only using the information of finger posture length i.e., the distance between MP joint and the corresponding finger tip. For this, the hand modelling constraints discussed earlier is used. Let, "L" be the original length of the finger. Similarly, "θ_1" is the joint angle corresponds to flexion of the MP joint, "θ_2" is the joint angle corresponds to PIP joint and "θ_3" is the angle corresponds to DIP joint of finger. And let "L_{obs}" be the observed length, i.e., distance between MP joint and the corresponding finger tip.

In this, the known parameters are, length of the finger (L) and observed Length (L_{obs}). Apparently, M, P, D and T represents MP joint, PIP joint, DIP joint and Tip of the finger, which are shown in Fig. 14.

It is assumed that the lengths of all the joints of a finger are of equal length and MP, PIP, DIP and finger tip (T) lie in the same plane. The finger plane is aligned with x–y plane, such that MP joint coincides with the origin of the reference coordinate system. Here, finger plane is the plane passing through M, P, D and T. Let us assume that, all phalangeal lengths are equal i.e., each phalangeal is of

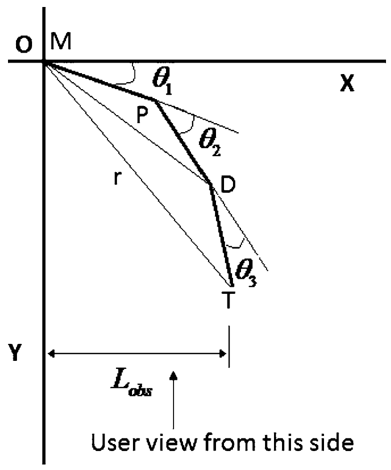


Fig. 14 MP joint, PIP joint, DIP joint and Tip of a finger

length equal to one third of the total length of the finger. The co-ordinates of *M* are (0, 0), and the coordinates of the tip, *T* are given by [11],

$$T_x = \frac{L}{3}(\cos(\theta_1) + \cos(\theta_1 + \theta_2) + \cos(\theta_1 + \theta_2 + \theta_3))$$
$$T_y = \frac{L}{3}(\sin(\theta_1) + \sin(\theta_1 + \theta_2) + \sin(\theta_1 + \theta_2 + \theta_3)) \tag{20}$$

Therefore, we can express L_{obs} as,

$$L_{obs} = \frac{L}{3}(\cos(\theta_1) + \cos(\theta_1 + \theta_2) + \cos(\theta_1 + \theta_2 + \theta_3)) \tag{21}$$

Let us consider the triangle, *MPD*. By using 'Cosine' rule and 'Sine' rule, the following relations can be obtained.

$$(MD)^2 = \left(\frac{L}{3}\right)^2 + \left(\frac{L}{3}\right)^2 - 2\left(\frac{L}{3}\right)\left(\frac{L}{3}\right)\cos(\pi - \theta_2) \tag{22}$$

$$\cos(\angle MDP) = \frac{\left(\frac{L}{3}\right)^2 + (MD)^2 - \left(\frac{L}{3}\right)^2}{2\left(\frac{L}{3}\right)(MD)} \tag{23}$$

$$\frac{\sin(\angle MDP)}{\left(\frac{L}{3}\right)} = \frac{\sin(\pi - \theta_2)}{(MD)} \tag{24}$$

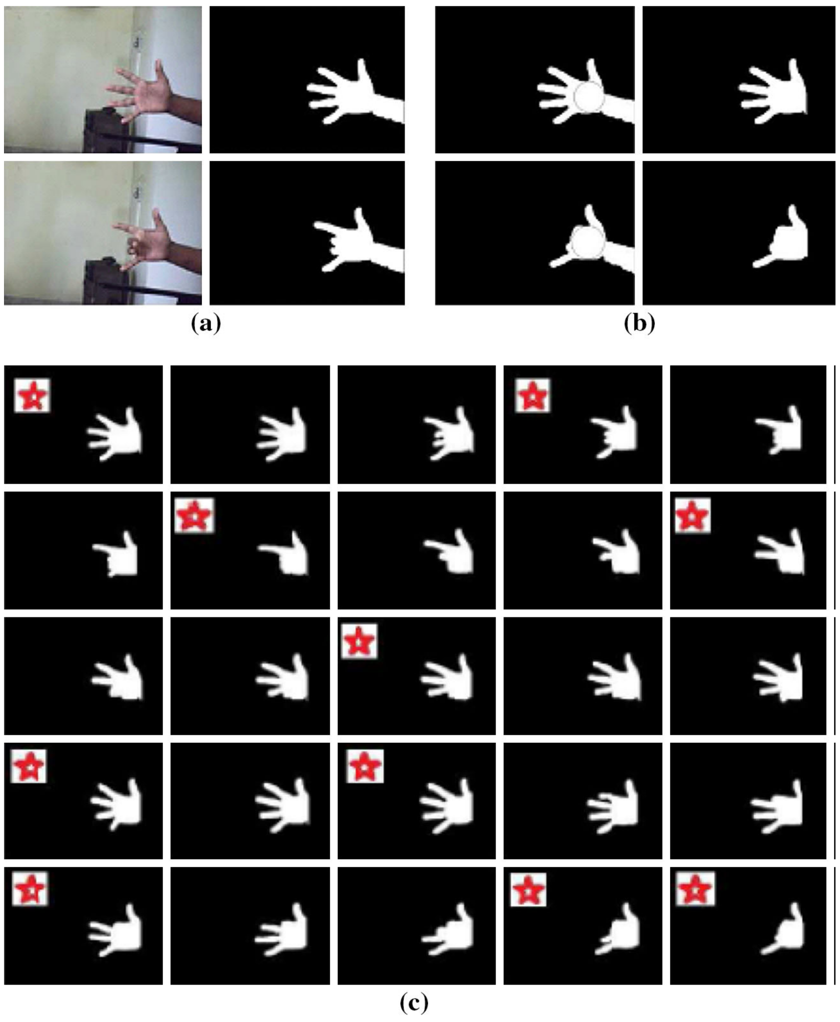


Fig. 15 **a** *Column 1* input color frame, and *column 2* skin color-based segmented region; **b** *column 1* full hand image showing largest circle inscribed, and *column 2* separated palm; **c** results of key frames extraction

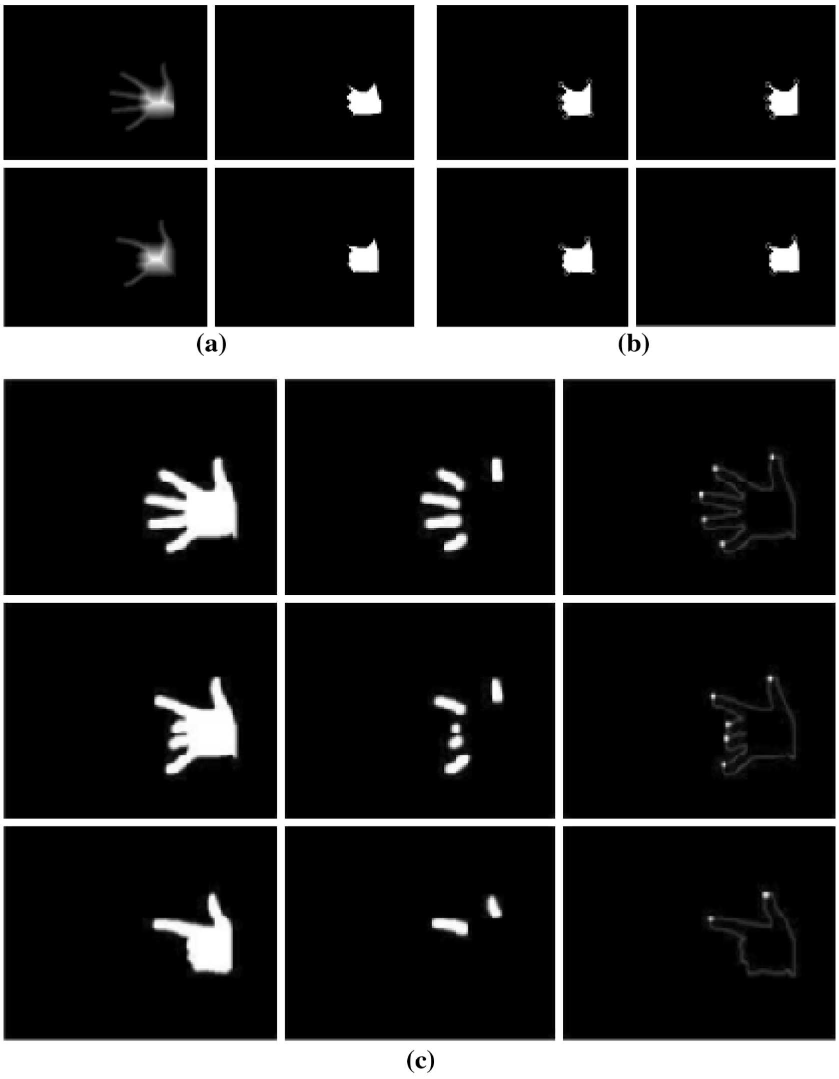


Fig. 16 **a** *Column 1* distance transformed image and *column 2* detected corners; **b** refined corners using k-curvature (MP joints detection); **c** results of fingertips detection

Again, let us consider the triangle, *MDT*. By using 'Cosine' rule for this triangle we can obtain the following relations.

$$\cos(\angle MDT) = \cos(\pi - \theta_3 - \angle MDP) \tag{25}$$

$$r^2 = (MD)^2 + \left(\frac{L}{3}\right)^2 - 2(MD)\left(\frac{L}{3}\right)\cos(\angle MDT) \tag{26}$$

We can also apply some of the constraints, given in Type-II constraints of hand model. So, θ_3 is replaced by $\frac{2}{3}\theta_2$. So, θ_3 can be removed from the above relations. Subsequently, we can solve for θ_2. After solving for θ_2, we use the Eq. (21) to solve for θ_1. For each finger, by proceeding in the similar way, we can find joint angles of all fingers.

2.9 Gesture animation scheme

After estimating the flexion angles of MP, PIP and DIP joints of all fingers, we can export these angles as parameters to 3D hand model developed in OpenGL. We can perform the animation of gestures using these parameters. Since, the extraction of the parameters has been done on the key frames only, we do not have any information about the pose of the hand, in the frames between the key frames. To give an illusion of continuous animation by presenting the different positions of hand, we should be able to estimate the pose of hand in these intermediate frames. This is called interpolation. As the name suggests, key frame interpolation attempts to generate new frames by interpolating values between two existing frames. There are many interpolation techniques available. The simplest method of interpolation is "linear interpolation". In our proposed method, we simply alter the joint angles in one key frame till they equal the joint angles of the next key frame. In this process, the human hand constraints are taken into consideration, thus ensuring that in the process of interpolation, improper gestures are not displayed.

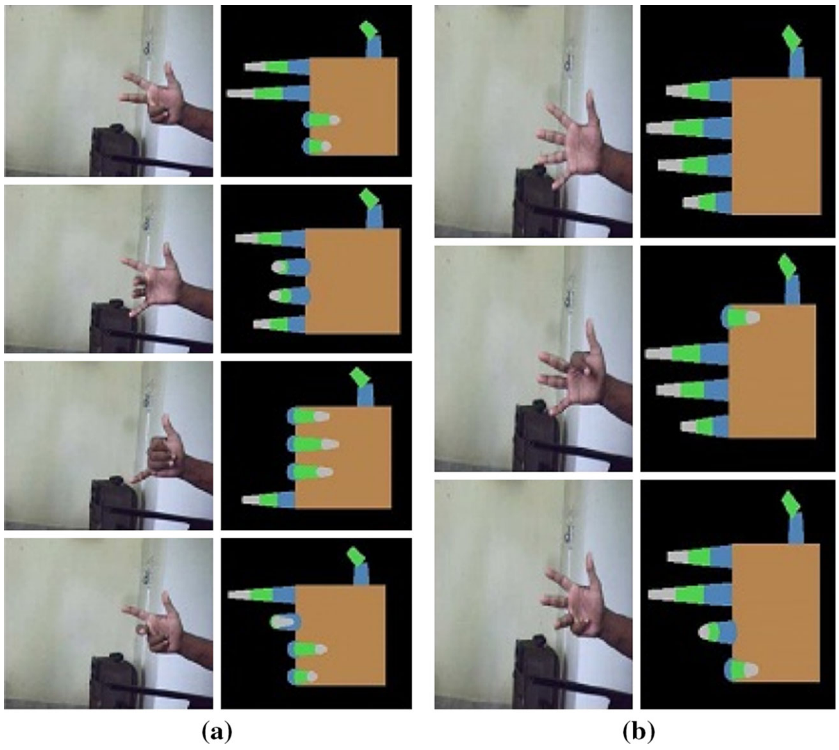


Fig. 17 **a**, **b** Reconstruction of static gestures: first column shows the original gestures; second column shows the reconstructed gestures

3 Experimental results

For our experiments, the proposed gesture recognition module was implemented using the image processing library of OpenCV 2.0 in Microsoft Visual Studio 2008 IDE in a window-based system having speed of 2.10 GHz. Whereas, gesture animation part was implemented in the OpenGL platform. The hand model was developed using cylinders and cubic in OpenGL. An web camera, with a resolution of 12 MP, 640 × 480 is used for capturing the hand movements. The performance of the proposed algorithms are reasonably satisfactory in the above mentioned setup.

Prior to gesture recognition and animation, calibration of the hand is done by keeping the hand in open position showing all the fingers. While calibrating, hand should be perpendicular to the camera axis, and the distance of the hand from the camera should not vary much.

The results of the skin color segmentation are shown in Fig. 15a, whereas Fig. 15b illustrates the results of separation of arm from the palm. Finally, Fig. 15c illustrates one example of key frame extraction from a small gesture video sequence, where key frames are marked by asterisk.

The results of MP joints and fingertip detection using corner detection technique are shown in Fig. 16a–c. MP joints are obtained by using *k*-curvature method, which is shown in Fig. 16b. Finally, Fig. 17a, b shows the animation of single-handed gestures.

For two handed gestures, face is removed after detecting the skin colored regions i.e., hand and the face. Experimental results of this step is shown in Fig. 18. Subsequently, MP joints of two hands and finger tips are detected for estimating the hand pose. Figs. 19 and 20 illustrate these steps. Finally, the results of two-handed dynamic hand gesture reconstruction and animation is shown in Fig. 21. We observe that our proposed scheme is capable of making hand pose as desired. Our scheme is capable of animating both single-handed and two-handed static and dynamic gestures. Static gestures consist of only one frame and so does not include any temporal change in the hand shape and pose.

4 Conclusion

As explained earlier, mainly two research papers [5] and [7] extensively deal with this very specific research topic. These papers did not address the problem of face detection and separation for hand parameters estimation. The problem of two-handed gesture animation was totally ignored in the earlier methods. Additionally, these methods have some constraints in estimating the hand parameters like placing of the hand in front of the camera. Some of these constraints are overcome in the proposed method. This paper presents a simple approach for finding the hand configurations from a 2D monocular image in a hand

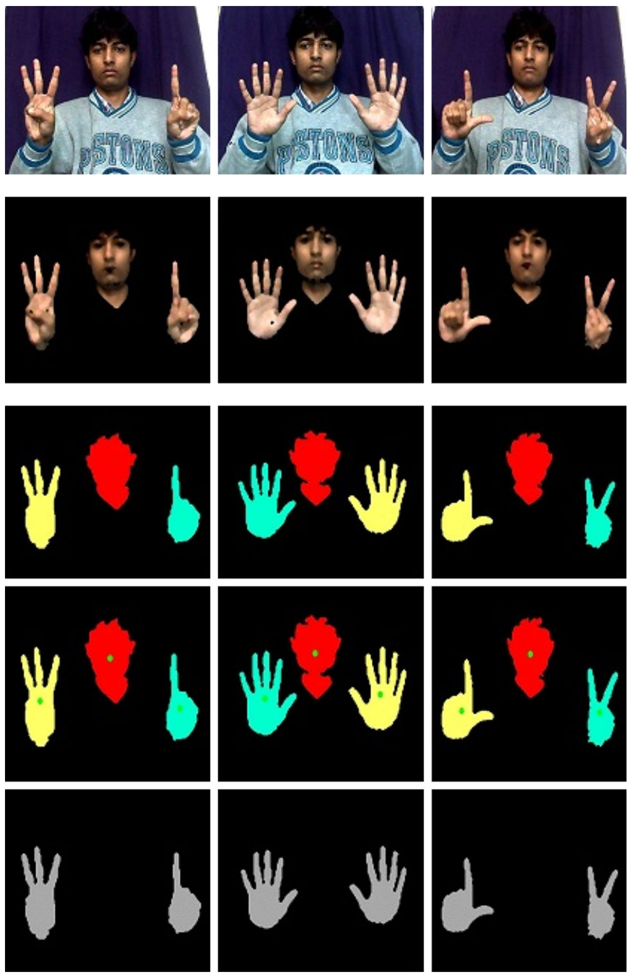


Fig. 18 Hand segmentation: *first row* shows the original images captured by the camera; *second row* shows the results of skin color-based segmentation; *third row* shows the results after component labelling; *fourth row* shows the results of centroid detection of face and hands; *fifth row* shows the separated hands

gesture sequence and subsequently extracts some hand parameters that are appropriate and sufficient to describe a particular hand pose. Computation complexity is significantly reduced by summarizing the large gesture sequence in the form of few key frames. The key video frames are selected on the basis of the amount of change in hand shape - for a given key frame in the sequence the next key frame is the one in which the hand changes its shape significantly. Thus, an entire video clip is transformed into a small number of representative frames that are sufficient to represent a gesture sequence. As discussed earlier, hand parameters are identified only from the extracted key frames that are appropriate and sufficient to describe a particular hand pose. Thus, a collection of such hand parameters, one set per key frame, constitutes the total description of the gesture in terms of some numerical values. Since only images of key hand postures are required to be analyzed to extract the control points and no

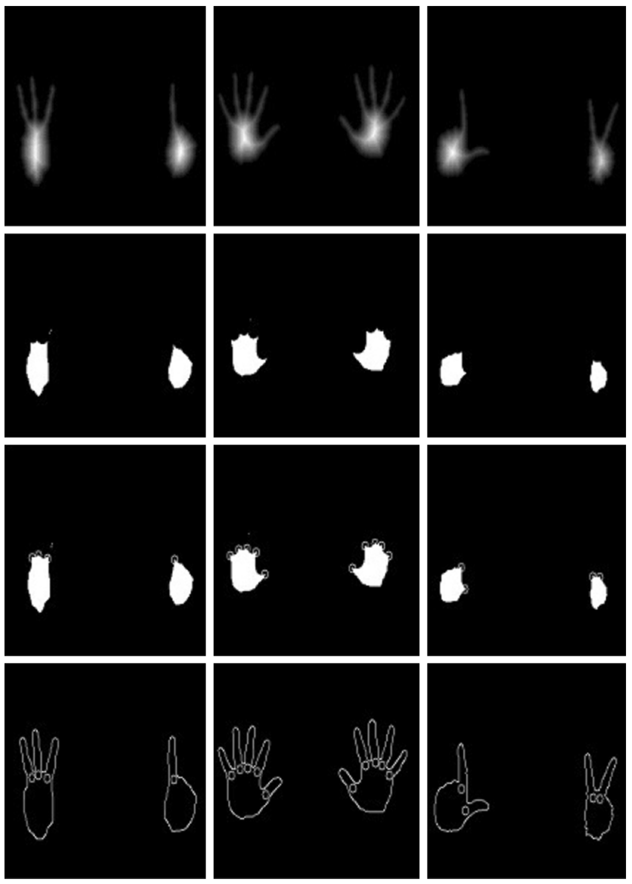


Fig. 19 Matacarpophalangial joints detection: *first row* shows distance transformed hand images; *second row* shows the results of corner detection algorithm; *third row* shows the detected MP joints; and the i shows the hand with the MP joints represented by *tiny circles*

hand motion tracking is involved, the computational complexity of the animation process is greatly reduced. Thus, the proposed method proves to be appropriate for real-time gesture synthesis in an on-line sign language conversation system.

Finally, the gesture animation is done by using the set of parameters for every key frame to crete each of the key frames and all in-between frames are interpolated i.e., all in between frames are interpolated using image metamorphosis so that the synthesized video looks like natural gesturing. One important contribution of this paper is hand pose estimation and reconstruction of two-handed dynamic hand gestures.

In this paper, we have only considered hand gestures having only local motions i.e., palm is kept fixed and gestures are created using finger movements only. However, the proposed technique can be extended with some modifications to include more natural form of gestures i.e., global motions of palm and arm. For this, along with the MP joints and finger tips of fingers, wrist and elbow

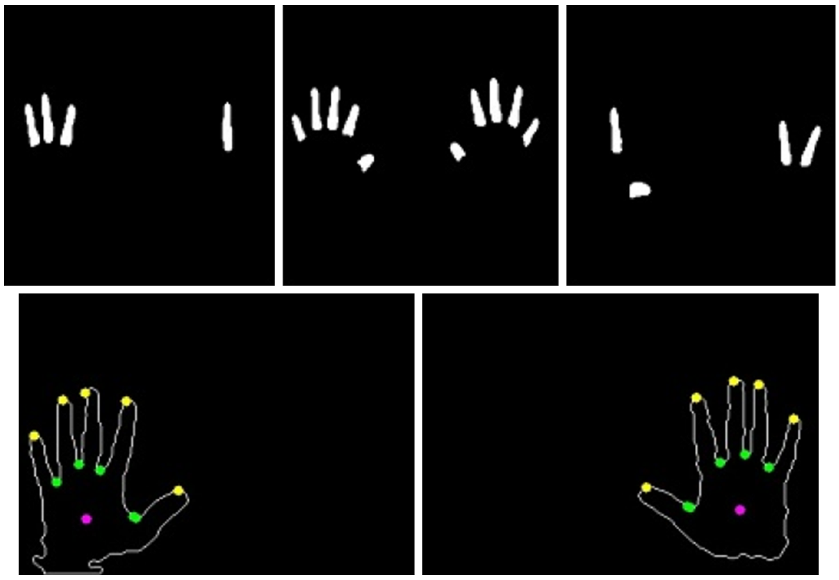


Fig. 20 Separated fingers and the detected finger tips and Matacarpophalangial joints of two hands

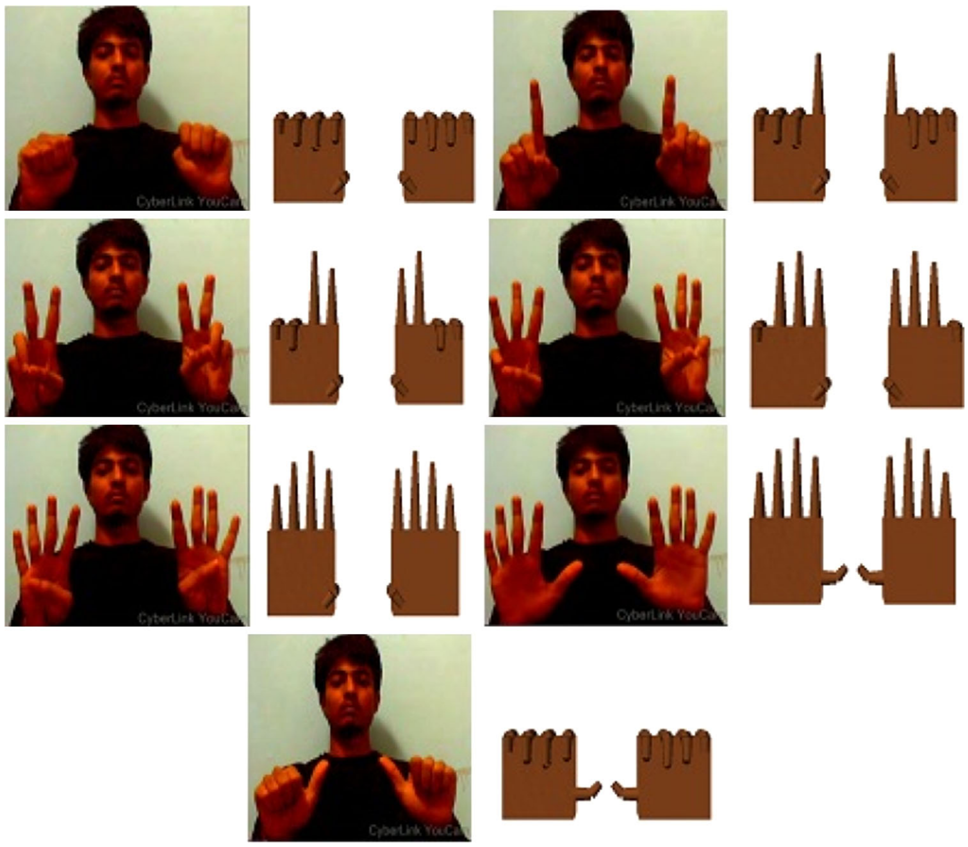


Fig. 21 Two-handed dynamic hand gesture reconstruction and animation obtained in our experiment

position of the hand should also be considered. Future works also include the segmentation of the hand in a more cluttered background and tackling the problem of interference among the fingers for gesture animation.

Acknowledgments This work was supported in part by a grant from MHRD, Government of India under National Mission for Education through ICT for the project entitled "Development of Indian Sign Language Recognition and Education System for Hearing Impaired Students of India".

References

1. Pavlovic VI, Sharma R, Huang TS (1997) Visual interpretation of hand gestures for human-computer interaction: a review. IEEE Trans Pattern Anal Mach Intell 19(7):677–695
2. Zhang S, McCullagh P, Nugent C, Zheng H, Baumgarten M (2011) Optimal model selection for posture recognition in home-based healthcare. Int J Mach Learn Cybern 2(1):1–14
3. Wu Y, Huang TS (2001) Hand modelling, analysis, and recognition for visionbased human computer interaction. IEEE Signal Process Mag 18:51–60
4. Ong SCW, Ranganath S (1997) Automatic sign language analysis: a survey and the future beyond lexical meaning. IEEE Trans Pattern Anal Mach Intell 27(6):873–891
5. Horace HSIp, Chan Sam CS, Lam Maria SW (2000) Hand gesture animation from static postures using an anatomy-based model. In: Proceedings of computer graphics international, pp 29–36
6. ElKoura G, Singh K (2003) Handrix: animating the human hand. In: Proceedings of ACM eurographics/SIGGRAPH symposium on computer animation, pp 110–119
7. Verma V, Ghosh D (2005) Hand gesture reconstruction and animation. In: Proceedings of 2nd international conference on artificial intelligence (IICAI-05), pp 537–555
8. Shankar VN, Ghosh D (2006) Dynamic hand gesture synthesis and animation using image morphing technique. In: Proceedings of international conference on visual information engineering (VIE 2006), pp 543–548
9. Lin J, Wu Y, Huang TS (2000) Modeling the constraints of human hand motion. In: Proceedings of workshop on human motion, pp 121–126
10. Lee J, Kunii TL (1995) Model-based analysis of hand posture. IEEE Comput Graph Appl 15(5):77–86
11. Guan H, Chua CS, Ho YK (2001) 3D hand pose retrieval from a single 2D image. Proc Int Conf Image Process 1:157–160
12. Tan W, Wu C, Zhao C, Chen S (2009) Hand extraction using geometric moments based on active skin color model. In: Proceedings of IEEE international conference on intelligent computing and intelligent systems, pp 468–471
13. Teng X, Wu B, Yu W, Liu C (2006) A hand gesture recognition system based on local linear embedding. In: Proceedings of IEEE conference on robotics, automation and mechatronics,vol 16, pp 1–6
14. Liu Q, Peng GZ (2010) A robust skin color based face detection algorithm. In: Proceedings of 2nd international Asia conference on informatics in control, automation and robotics (CAR), vol 2, pp 525–528
15. Porle RR, Chekima A, Wong F, Sainarayanan G (2009) Performance of histogram-based skin colour segmentation for arms detection in human motion analysis application. Int J Electr Comput Eng 4(15):950–955
16. Dong G, Yonghua Y, Ming X (2002) Vision-based hand gesture recognition for human-vehicle interaction. In: Proceedings of 7th international conference on control, automation, robotics and vision, pp 1–4
17. Amayeh G, Bebis G, Erol A, Nicolescu M (2007) A new approach to hand-based authentication. In: Proceedings of biometric technology for human identification
18. Borgefors G (1986) Distance transformations in digital images. Comput Vision Graph Image Process 34:344–371
19. Schwarz C, da Vitoria Lobo N (2005) Segment-based hand pose estimation. In: Proceedings of the 2nd Canadian conference on computer and robot vision, vol 20, pp 42–49