Posture and Gesture Recognition for Human-Computer Interaction

Mahmoud Elmezain, Ayoub Al-Hamadi, Omer Rashid and Bernd Michaelis
Otto-von-Guericke-University Magdeburg
Institute for Electronics, Signal Processing and Communications (IESK)
Germany

1. Introduction

While automatic *hand posture and gesture* recognition technologies have been successfully applied to real-world applications, there are still existed several problems that need to be solved for wider applications of Human-Computer Interaction (HCI). One of such problems, which arise in real-time hand gesture recognition, is to extract (spot) meaningful gestures from the continuous sequence of hand motions. Another problem is caused by the fact that the same gesture varies in shape, trajectory and duration, even for the same person. A gesture is spatio-temporal pattern which may be static or dynamic or both. Static morphs of the hands are called postures and hand movements are called gestures. The goal of gesture interpretation is to push the advanced human-computer communication to bring the performance of HCI close to human-human interaction. Sign language recognition is an application area for HCI to communicate with computers and for sign language symbols detection. Sign language is categorized into three main groups namely finger spelling, word level sign and non manual features (Bowden et al., 2003). Finger spelling is used to convey the words letter by letter. The major communication is done through word level sign vocabulary and non-manual features include the facial expressions, mouth and body position.

The techniques for *posture* recognition with sign languages are reviewed for finger spelling to understand the research issues. The motivation behind this review is to develop a recognition system which works more robustly with high recognition rates. Practically, hand segmentation and computations of good features are important for the recognition. In the recognition of sign languages, different models are used to classify the alphabets and numbers. For example, in (Hussain, 1999), Adaptive Neuro-Fuzzy Inference Systems (ANFIS) model is used for the recognition of Arabic Sign Language. In this proposed technique, colored gloves are used to avoid the segmentation problem and it helps the system to obtain good features. Handouyahia et al. (Handouyahia et al., 1999) presents a recognition system for the International Sign Language (ISL). They have used Neural Network (NN) to train the alphabets. NN is used for the recognition purposes because it can easily learn and train from the features computed for the sign languages. Other approach

includes the Elliptic Fourier Descriptor (EFD) used by Malassiotis and Strintzis (Malassiotis & Strintzis, 2008) for 3D hand posture recognition. In their system, they have used orientation and silhouettes from the hand to recognize 3D hand postures. Similarly, Licsar and Sziranyi (Licsar & Sziranyi, 2002) used Fourier coefficients to represent hand shape in their system which enables them to analyze hand gestures for the recognition. Freeman and Roth (Freeman & Roth, 1994) used orientation histogram for the classification of gesture symbols, but huge training data is used to solve the orientation problem and to avoid the misclassification between symbols.

In the last decade, several methods of potential applications (Deyou, 2006; Elmezain et al., 2008a; Kim et al., 2007; Mitra & Acharya, 2007; Yang et al., 2007) in the advanced *Hand gesture* interfaces for HCI have been suggested but these differ from one another in their models. Some of these models are Neural Network (Deyou, 2006), Hidden Markov Models (HMMs) (Elmezain et al., 2008a; Elmezain et al., 2008b) and Dynamic Time Warping (DTW) (Takahashi et al., 1992). In 1999, Lee et al. (Lee & Kim, 1999) proposed an ergodic model based on adaptive threshold to spot the start and the end points of input patterns, and also classify the meaningful gestures by combining all states from all trained gesture models using HMMs. Kang et al. (Kang et al., 2004) developed a method to spot and recognize the meaningful movements where this method concurrently separates unintentional movements from a given image sequences. Alon et al. (Alon et al., 2005) proposed a new gesture spotting and recognition algorithm using a pruning method that allows the system to evaluate a relatively small number of hypotheses compared to Continuous Dynamic Programming (CDP). Yang et al. (Yang et al., 2007) presented a method for recognition of whole-body key gestures in Human-Robot Interaction (HRI) by HMMs and garbage model for non-gesture patterns.

Mostly, previous approaches use the backward spotting technique that first detects the end point of gesture by comparing the probability of maximal gesture models and non-gesture model. Secondly, they track back to discover the start point of the gesture through its optimal path and then the segmented gesture is sent to the recognizer for recognition. So, there is an inevitable time delay between the meaningful gesture spotting and recognition where this time delay is not well for on-line applications. Above of all, few researchers have addressed the problems on non-sign patterns (which include out-of-vocabulary signs, epentheses, and other movements that do not correspond to signs) for sign language spotting because it is difficult to model non-sign patterns (Lee & Kim, 1999).

The main contribution of this chapter is to explore two parts; the first part is related to hand posture and the second part deals with hand gesture spotting. *In posture recognition*, an approach is proposed for recognition of ASL alphabets and numbers, which is able to deal with a large number of hand shapes against complex backgrounds and lighting conditions. This approach is based on Hu-Moment, whose features are invariant of translation, rotation and scaling. Besides, geometric features are also incorporated. These feature vectors are used to train Support Vector Machine (SVM) and a recognition process that identifies the hand posture from the SVM of segmented hands. *In hand gesture*, a robust technique is proposed that executes gesture spotting and recognition simultaneously. The technique recognizes the isolated and the meaningful hand gestures in stereo color image sequences using HMMs. In addition, color and 3D depth map are used to detect hands where the hand trajectory will take place in further step using a robust stereo tracking algorithm to generate 3D dynamic features. This part covers the procedures to design a sophisticated method for

non-gesture model, which provides a confidence limit for the calculated likelihood by other gesture models. Furthermore, the confidence measures are used as an adaptive threshold for selecting the proper gesture model or spotting meaningful gestures. The proposed techniques can automatically recognize posture, isolated and meaningful hand gestures with superior performance and low computational complexity when applied on several video samples containing confusing situations such as partial occlusion and overlapping. The rest of the chapter is organized as follows. We formulate the Hidden Markov Models in Section 2 and Support Vector Machine in Section 3. Section 4 discusses the posture and gesture approach in three subsections. Experimental results are given in Section 5. We have tested image and video sequences for hand posture and gesture spotting respectively. Finally, Section 6 ends with a summary and conclusion.

2. Hidden Markov Models

Markov Model is a mathematical model of stochastic process, which generates random sequences of outcomes according to certain probabilities (Elmezain et al., 2007; Rabiner, 1989). A stochastic process is a sequence of feature extraction codewords, the outcomes being the classification of hand gesture path. In a compact mode a discrete HMMs can be symbolized with $\lambda = (A, B, \Pi)$ and is described as follows:

- The set of states $S = \{s_1, s_2, \ldots, s_N\}$ where N represents the number of states.
- The set of observation symbols $V = \{v_1, v_2, \ldots, v_M\}$ where M is the number of distinct symbols observable in each state.
- An initial probability for each state Π_i , $i=1, 2, ..., N$; such that:

$$\Pi_i = P(s_i), \quad \Pi_i \geq 0, \qquad \sum_i \Pi_i = 1 \tag{1}$$

- An N-by-N transition matrix $A = \{a_{ij}\}$ where a_{ij} is the probability of taking a transition from state i to state j at moment t:

$$a_{ij} = P(s_t = j | s_{t-1} = i), \ 1 \leq i, j \leq N, \quad a_{ij} \geq 0, \ \sum_j a_{ij} = 1 \tag{2}$$

- An N-by-M observed symbols matrix $B = \{b_{im}\}$ where b_{im} gives the probability of emitting symbol v_m in state i:

$$b_{im} = P(v_m | s_i), \ 1 \leq i \leq N, 1 \leq m \leq M, b_{im} \geq 0, \ \sum_m b_{im} = 1 \tag{3}$$

- The set of possible emission (an observation) $O = \{o_1, o_2, \ldots, o_T\}$ where T is the length of gesture path.

Based on HMMs the statistical strategy has many advantages, among them being recalled: rich mathematical framework, powerful learning and decoding methods, good sequences handling capabilities, a flexible topology for the statistical phonology and the syntax. The disadvantages lie in the poor discrimination between the models and in the unrealistic

assumptions that must be made to construct the HMMs theory, namely the independence of the successive feature frames (input vectors) and the first order Markov process (Goronzy, 2002). The algorithms developed in the statistical framework to use HMMs are rich and powerful, situation that can explain well the fact that today, hidden Markov models are the widest used in practice to implement gesture recognition and understanding systems. The main problems that can be solved with HMMs are:

1. Given the observation sequence $O = (o_1, o_2, \ldots, o_T)$, and a model $\lambda = (A, B, \Pi)$ how do we efficiently compute $P(O \mid \lambda)$, the probability of the observation sequence, given the model. This is the "***evaluation problem***". Using the forward and backward procedure provides solution.
2. Given the observation sequence $O = (o_1, o_2, \ldots, o_T)$, and the model λ, how do we choose a corresponding state sequence $S=(s_1, s_2, \ldots, s_T)$ that is optimal in some sense (i.e. best explains the observation). The ***Viterbi algorithm*** provides a solution to find the optimal path.
3. How do we adjust the model parameters $\lambda = (A, B, \Pi)$ to maximize $P(O \mid \lambda)$. This is by far the most difficult problem of HMMs. We choose $\lambda = (A, B, \Pi)$ in such a way that its likelihood, $P(O \mid \lambda)$, is locally maximized using an iterative procedure like ***Baum-Welch*** method (Rabiner, 1989).

Also, HMMs has three topologies; the first topology is Ergodic model (Fully Connected model) in which every state of the model could be reached in a finite number of steps from every other state of the model (Figure 1(a)). Other types of HMMs have been found to account for observed properties of the signal being modelled better than the standard Ergodic model. One such model is shown in Figure 8. This model is called a Left-Right Banded (LRB) because the underlying state sequence associated with model has the property that as time increases the state index increases or stays the same (i.e. no transitions are allowed to states whose indices are lower than the current state). Each state in LRB model can go back to itself or to the next state only. The last topology of HMMs is called a Left-Right (or the Bakis model) in which each state can go back to itself or to the following states. It should be clear that the imposition of the constraints of the LRB and Bakis model essentially have no effect on the re-estimation procedure. This is the case because any HMMs parameter set to zero initially, will remain at zero throughout the re-estimation procedure.

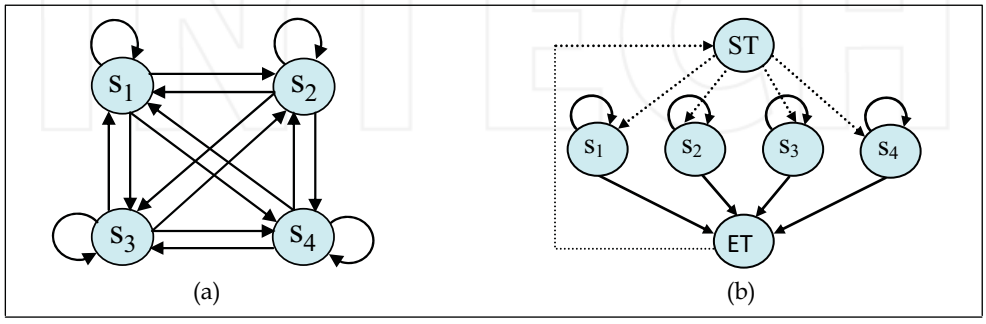


Fig. 1. (a) Ergodic topology with 4 states. (b) Simplified Ergodic with fewer transitions

3. Support Vector Machines

Support Vector Machines is a supervised learning for the optimal modelling of the data (Lin & Weng, 2004). It learns the decision function and separates the data class to the maximum width. Basically, SVM works on two-class i.e. binary classification and is also extendable for multiclass problem. In the literature, there are two types of this extension. All-together approach deals with its optimization problem. It lacks scalability and also faces optimization complexity. Second approach deals in binary fashion with multiple hyper-planes along with the combination into a single classifier. There are further two alternatives for this combination. The first one is based on one-against-all whereas other works as one-against-one. Binary classification of SVM learns on the following principle:

$$c(x) \in \{-1,1\} \tag{4}$$

The SVM's linearly learned decision function *f(x)* is described as:

$$f(x) = sign(\boldsymbol{w} \cdot \boldsymbol{x} + b) \tag{5}$$

where $\boldsymbol{w}$ is a weight vector while b is the threshold and x is the input sample.
SVM learner defines the hyper-planes for the data and maximum margin is found between these hyper planes. Because of the maximum separation of hyper-planes, it is also considered as a margin classifier. Margin of the hyper-plane is the minimum distance between hyper-plane and the support vectors and this margin is maximised. It can be formulated as following:

$$\gamma = \frac{2}{\| \boldsymbol{w} \|} \tag{6}$$

where γ is margin of the hyper-plane. Maximisation of the margin of the hyper plane is depicted in Figure 2.

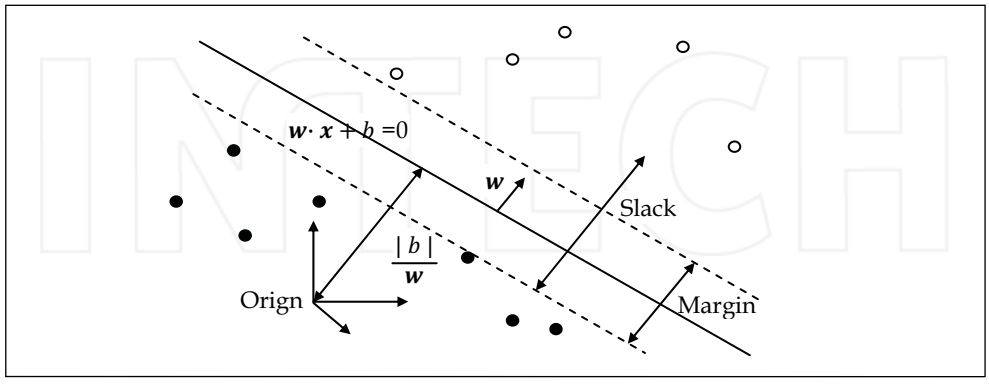


Fig. 2. Margin of the hyper plane

SVM maps input data into high dimension domain where it is utmost linearly separable as shown in Figure 3. This mapping does not affect the training time because of implicit dot product and kernel trick (Cristianini & Taylor, 2001; Suykens et al., 2005). This is also a reason that SVM is a well suited classifier where features are large in number because they are robust to the curse of dimensionality. Kernel function is the computation of the inner product $\Phi(x) \cdot \Phi(y)$ directly from the input. One of the characteristics of using the kernel is that there is no need to explicitly represent the mapped feature space. Kernel function is mathematically described as follows:

$$K(x,z) = \Phi(x) \cdot \Phi(z) \tag{7}$$

Following are some of the kernel functions which are commonly used to convert the input features into new feature space.

- Linear kernel $K(x,y) = (x \cdot y)$ (8)
- RBF Gaussian kernel $K(x,y) = e^{-\|x-y\|^2/2\sigma^2}$ (9)
- Polynomial kernel $K(x,y) = (x \cdot y + 1)^d$ (10)
- Sigmoid kernel $K(x,y) = tanh(\kappa x \cdot y + c)$ (11)

where K is a scaling factor while c is a shifting factor that controls the mapping. As discussed above, SVM outputs only the class labels for the input sample as output but not the probability information for the classes. Lin et al. describes a method to compute the class probabilities using SVM. Chang et al. developed a library ''LIBSVM'' which provides the tools for the SVM functionalities including class probability estimation (Chang & Lin, 2001).

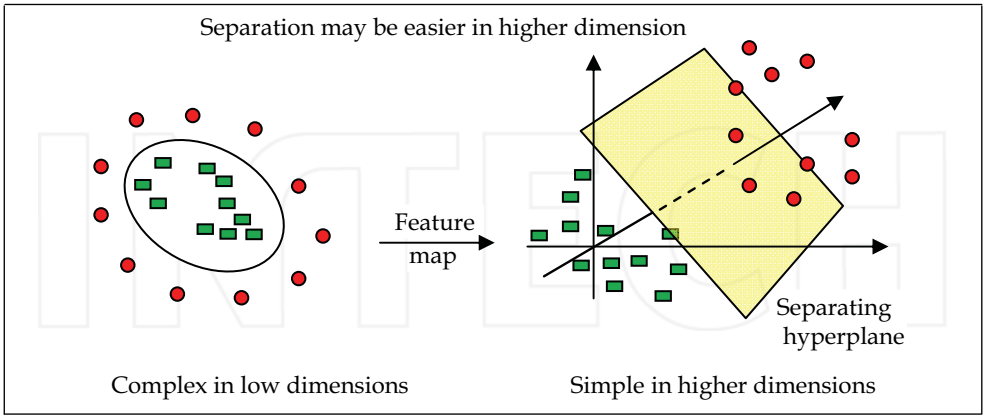


Fig. 3. Mapping from input data to a richer feature space through kernel function

SVM has been studied a lot and is being used in a large problem domain including novelty detection regression optimization along with learning and classification. It has a basic

architecture which can be modified depending upon the problem domain using margin, kernel type and duality characteristics. SVM lacks several problems which other learners do like non-linear function, problem of local minima etc. It not only distinguishes between classes but also learns to separate them optimally. In addition, the performance of SVM is declined with non-scaled data and multi-class solution is still under process (Burges, 1998).

4. Posture and Gesture Approach

In this Chapter, an approach is developed for the recognition of hand postures and gestures. Besides, improvements are done in the existing system of gesture recognition provided by IESK (Magdeburg University, Germany) whose purpose is to recognize the alphabets characters (A-Z) and numbers (0-9). The proposed approach is based on the analysis of stereo color image sequences with the support of 3d depth information. Gaussian distribution detects the skin pixels from the image sequences and depth information is used to help Gaussian distribution to build the region of interest and overcome the difficulties of overlapping regions. A framework is established to extract the posture and gesture features by the combination of various image processing techniques (Figure 4).

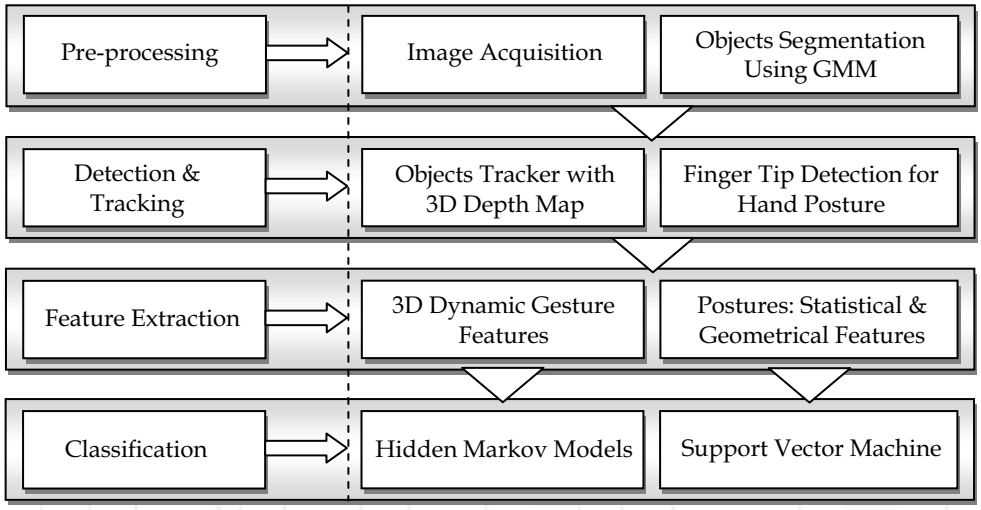


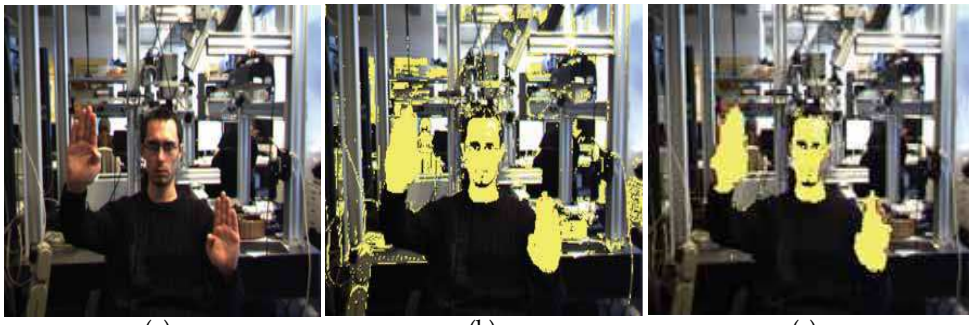
Fig. 4. Simplified structure showing the main modules for the posture and gesture approach

The computed statistical and geometrical features for the hand movements are invariant to scale, rotation and translation. These features are used for the classification of posture symbols. The classification step is divided into two steps. The first step develops the classes for some set of alphabets for hand posture. In particular, the curvature analysis determines the peaks of the hand (i.e. fingertips) which helps in the reduction of computation and to avoid the classes that are not mandatory to test for that specific posture symbol. The misclassification is also reduced due to this grouping which helps in the recognition of correct symbol. Furthermore, SVM is applied on the respective set of classes to train and test the symbols. In the second step, the hand trajectory will take place in further step using

Mean-shift algorithm and Kalman filter (Comaniciu et al., 2003) to generate 3D dynamic features for hand gesture. Furthermore, *k*-means clustering algorithm (Ding & He, 2004) is employed for the HMMs codewords. To spot meaningful gestures (i.e. Arabic numbers from 0 to 9) accurately, a non-gesture model is proposed, which provides a confidence limit for the calculated likelihood by other gesture models. The confidence measures are used as an adaptive threshold for spotting meaningful gestures.

4.1 Depth Map

Image acquisition step contains 2D image sequences and depth image sequences. For the skin segmentation of hands and face in stereo color image sequences an algorithm is used, which calculates the depth value in addition to skin color information The depth information can be gathered by passive stereo measuring based on cross correlation and the known calibration data of the cameras. Several clusters are composed of the resulting 3D-points. The clustering algorithm can be considered as kind of region growing in 3D which used two criteria; skin color and Euclidean distance (Scott, 19992; Niese et all., 2007). Furthermore, this method is more robust to the disadvantageous lighting and partial occlusion, which occur in real time environment (for instance, in case of gesture recognition). The classification of the skin pixels is improved from Figure 5 by exploiting the depth information which contains the depth value associated with 2D image pixel. In the proposed approach, the depth image is used to select the region of interest in the image and it lies in the range from minimum depth 30cm to maximum depth 200cm. However, the depth range is adaptive and can be changed. From the depth information, not only the search of object of interest is narrowed down but also the processing speed is increased. The region of interest helps to remove the computed skin pixels other than this region. Figure 6 (a)&(b) shows the normalized 2D and 3D depth image ranges up to 10m. The normalization depth images are presented for visualization in the range from 0 to 255. Figure 6 (c)&(d) shows the normalized 2D and 3D depth range of interest (i.e. range from 30cm to 200cm). It should be noted that the region of interest should include the hands and face. The improved results by using the depth information are shown in Figure 5(c).



(a) (b) (c)

Fig. 5. (a) Original 2D image (b) Skin pixel detection without using depth map (c) Yellow color shows the detection of skin pixels in the image after applying depth information

By the given 3D depth map from camera set-up system, the overlapping problem between hands and face is solved since the hand regions are closer to the camera rather than the face region (Figure 12 & 13).

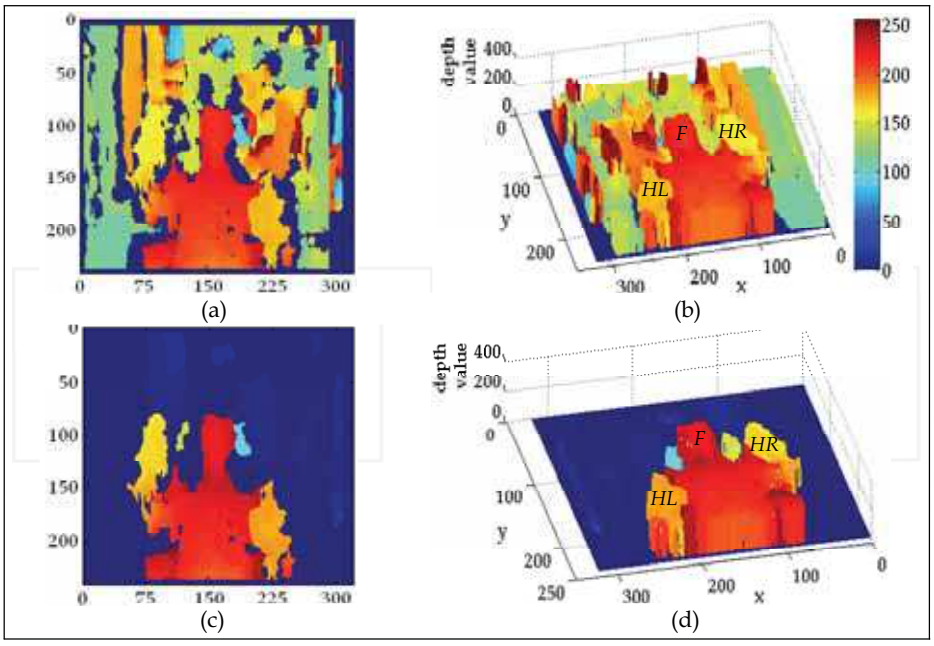


Fig. 6. (a)&(b) shows the normalized 2D and 3D depth image respectively (c)&(d) shows the normalized 2D and 3D depth image for the region of interest (30cm to 200cm). *F* referes to the face, *HL* and *HR* represent the left and right hand respectively

4.2. Feature Extraction

There is no doubt that selecting good features to recognize the hand posture and gesture path plays a significant role in any system performance. So, we will mention the features about postue and gesture in some details as follows.

4.2.1 Posture Features

In the proposed approach, the statistical and geometrical features are computed for the hand postures. These are described as under.

- **Statistical Feature Vectors**

Hu-Moments (Hu, 1962) are used in statistical feature vectors and are derived from basic moments. More specifically, moments are used to describe the properties of objects shape statistically. In image analysis, moments are considered as a binarized or grey level image with 2D density distribution functions. In this manner, an image segment is categorized with the help of moments. The properties extracted from the moments are area, mean, variance, covariance and skewness.

- **Central Moments**

If $f(x, y)$ is a digital image of *M-by-N* dimension, the central moments of order *(p+q)* is defined as:

$$\mu_{pq} = \sum_{y}^{M-1} \sum_{x}^{N-1} (x - \bar{x})^p \, (y - \bar{y})^q f(x, y), \quad \bar{x} = \frac{m_{10}}{m_{00}}, \bar{y} = \frac{m_{01}}{m_{00}} \tag{12}$$

where m_{00} gives the area of the object, m_{10} and m_{01} are used to locate center of gravity of the object, $\bar{x}$ and $\bar{y}$ gives the coordinates of the center of gravity of the object (i.e. centroid). It can be seen from the above equation that central moments are translation invariant.

- **Normalized Central Moment**

The normalized central moments are defined as:

$$\eta_{pq} = \frac{\mu_{pq}}{m_{00}^{\Upsilon}}, \qquad \Upsilon = \left(\frac{(p+q)}{2} + 1\right), \qquad p, q \, \epsilon\{2,3, \ldots, \infty\} \tag{13}$$

By normalizing the central moments, the moments are scale invariant. The normalization is different for different order moments.

- **Hu-Moments**

Hu (Hu, 1962) derived a set of seven moments which are translation, orientation and scale invariant. The equations are computed from the second and third order moments. Hu invariants are extended by Maitra (Maitra, 1979) to be invariant under image contrast. Later, Flusser and Suk (Flusser & Suk, 1993) have derived the moment invariant, that are invariant under general affine transformation. The equations of Hu-Moments are defined as:

$$\phi_1 = \eta_{20} + \eta_{02} \tag{14}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4{\eta_{11}}^2 \tag{15}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \tag{16}$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \tag{17}$$

$$\begin{aligned} \phi_5 = {} & (\eta_{30} - 3\eta_{12})\,(\eta_{30} + \eta_{12})\,[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & + (3\eta_{21} - \eta_{03})\,(\eta_{21} + \eta_{03})\,[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned} \tag{18}$$

$$\phi_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})\,(\eta_{21} + \eta_{03}) \tag{19}$$

$$\begin{aligned} \phi_7 = {} & (3\eta_{12} - \eta_{03})\,(\eta_{30} + \eta_{12})\,[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & + (3\eta_{12} - \eta_{30})\,(\eta_{21} + \eta_{03})\,[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned} \tag{20}$$

Hu-Moments are derived from a set of seven moments. These seven moments are derived from second and third order moments. However, zero and first order moments are not used in this process. The first six Hu-Moments are invariant to reflection (Davis & Bradski, 1999) and seventh moment change the sign. Statistical feature vectors contain the following set:

$$F_{stat} = (\phi_1, \phi_2, \phi_3, \phi_4, \phi_5, \phi_6, \phi_7)^T \tag{21}$$

where ϕ_1 is the first Hu-Moment. Similar is the notation for all other features in this set.

- **Geometrical Feature Vectors**

Geometrical feature set contains two features: circularity and rectangularity. These features are computed to exploit the hand shape with the standard shapes like circle and rectangle.

This feature set varies from letter to letter and is useful to recognize the alphabets and numbers. The feature set of the geometrical features is as under:

$$F_{geo} = (Cir, Rect)^T \tag{22}$$

Circularity: Circularity is the measure of the shape that how much the object's shape is closer to the circle. In the ideal case, circle gives the circularity as one. The range of circularity varies from 1 to infinity. Circularity Cir is defined as:

$$Cir = \frac{Perimeter^2}{4\pi \times Area} \tag{23}$$

where *Perimeter* is the contour of the hand and *Area* is the total number of hand pixels.

Rectangularity: Rectangularity defines the measure of the shape of the object that how much its shape is closer to the rectangle. The orientation of the object is calculated by computing the angle of all contour points using central moments. Length l and width w is calculated by the difference of largest and smallest angle in the rotation. In ideal case, the rectangularity ($Rect$) is 1 for rectangle and varies from 0.5 to infinity and is calculated as:

$$Rect = \frac{Area}{l \times w} \tag{24}$$

where area is the total pixels of the hand, l is the length and w is the width.
The statistical and geometrical feature vector set combined together to form a set of feature set. It is denoted as:

$$F_{total} = F_{stat} + F_{geo} \tag{25}$$

$$F_{total} = (\phi_1, \phi_2, \phi_3, \phi_4, \phi_5, \phi_6, \phi_7, Cir, Rect)^T \tag{26}$$

F_{total} contains all the features used for hand posture recognition.

- **Curvature Feature**

An important feature for the recognition of alphabets is the curvature feature which tells us about the peaks (i.e. fingertips) in hand. Therefore, before classifying the alphabets by SVM, four groups are made according to the numbers of fingertips detected in the hand. For ASL numbers, we classify them with a single classifier.

Normalization:
The normalization is done for features to keep them in a particular range. Geometrical features vector have the range up to infinity and these features are very different from each other, so they create a scalability problem. In order to keep them in same range and to combine them with statistical feature vector, normalization is carried out and is defined as:

$$Cir_{norm} = \frac{Cir - minCir}{maxCir - minCir}, \qquad Rect_{norm} = \frac{Rect - minRect}{maxRect - minRect} \tag{27}$$

where $minCir$ and $maxCir$ are the minimum and maximum circularity of the hand from all classes of feature vectors. Cir_{norm} The notations are the same for rectangularity. Hu-Moments are normalized by the following equation:

$$\phi_i = \frac{\phi_i - min\phi_{all}}{max\phi_{all} - min\phi_{all}} \tag{28}$$

where ϕ_i is the i^{th} Hu-Moment feature. $min\phi_{all}$ and $max\phi_{all}$ are the minimum and maximum values from the set of all classes respectively.

4.2.2 Gesture Features

There are three basic features; location, orientation and velocity. So, we will do a combination of these three basic features and using them as a main feature. A gesture path in spatio-temporal pattern that consists of hand centroid points (x_{hand}, y_{hand}) where the coordinates in the Cartesian space can be extracted from gesture frames directly. We consider two types of location features. The first location feature is *Lc* that measures the distance from the centroid to a point of the hand gesture because different location features are generated for the same gesture according to the different starting points (Eq. 29). The second location feature is *Lsc*, which is computed from the start point to the current point of hand gesture path (Eq. 31).

$$Lc_t = \sqrt{(x_{t+1} - C_x)^2 + (y_{t+1} - C_y)^2}\,, \quad t = 1, 2, \ldots, T-1 \tag{29}$$

$$(C_x, C_y) = \frac{1}{n}\left(\sum_{t=1}^{n} x_t\,, \sum_{t=1}^{n} y_t\right) \tag{30}$$

$$Lsc_t = \sqrt{(x_{t+1} - x_1)^2 + (y_{t+1} - y_1)^2} \tag{31}$$

where T represents the length of hand gesture path. (C_x, C_y) refers to the center of gravity at the point n. To verify the real-time implementation, the center of gravity is computed after each image frame.

The second basic feature is the orientation, which gives the direction along the hand when traverses in space during the gesture making process. As described above, the orientation feature is based on the calculation of the hand displacement vector at every point and is represented by the orientation according to the center of gravity (θ_{1t}), the orientation between two consecutive points (θ_{2t}) and the orientation between start and current hand gesture point (θ_{3t}).

$$\theta_{1t} = arctan\left(\frac{y_{t+1} - C_y}{x_{t+1} - C_x}\right),\ \theta_{2t} = arctan\left(\frac{y_{t+1} - y_t}{x_{t+1} - x_t}\right),\ \theta_{3t} = arctan\left(\frac{y_{t+1} - y_1}{x_{t+1} - x_1}\right) \tag{32}$$

The third basic feature is the velocity, which plays an important role during gesture recognition phase particulary at some critical situations. The velocity V is based on the fact that each gesture is made at different speeds where the velocity of the hand decreases at the corner point of a gesture path. The velocity is calculated as the Euclidean distance between

the two successive points divided by the time in terms of the number of video frames as follows:

$$V_t = \sqrt{(x_{t+1} - x_t)^2 + (y_{t+1} - y_t)^2} \tag{33}$$

Each frame contains a set of feature vectors at time t (Lc_t, Lsc_t, θ_{1t}, θ_{2t}, θ_{3t}, V_t) where the dimension of space is proportional to the size of feature vectors. In this manner, gesture is represented as an ordered sequence of feature vectors, which are projected and clustered in space dimension to obtain discrete codeword that are used as an input to HMMs. This is done using *k*-means clustering algorithm (Ding & He, 2004; Kanungo et al., 2002), which classifies the gesture pattern into *K* clusters in the feature space. This algorithm is based on the minimum distance between the center of each cluster and the feature point. We divide a set of feature vectors into a set of clusters. This allows us to model the hand trajectory in the feature space by one cluster. The calculated cluster index is used as input (i.e. observation symbol) to the HMMs. Furthermore, we usually do not know the best number of clusters in a data set. In order to specify the number of clusters *K* for each execution of the *k*-means algorithm, we considered $K = 28, 29, ..., 37$ which is based on the numbers of segmented parts in all numbers (0-9) where each straight-line segment is classified into a single cluster.
Suppose we have *n* sample of trained feature vectors $x_1, x_2, ..., x_n$ all from the same class, and we know that they fall into *K* compact clusters, $K < n$. Let m_i be the mean of the vectors in cluster *i*. If the clusters are well separated, a minimum distance classifier is used to separate them. That is, we can say that *x* is in cluster *i* if $\|x\text{-}m_i\|$ is the minimum of all the *K* distances. The following procedure for finding the *k*-means is;

- ***Build up*** randomly an initial Vector Quantization Codebook for the means $m_1, m_2, ..., m_k$
- ***Until*** there are no changes in any mean
  - ***Use*** the estimated means to classify each sample of train vectors into one of the clusters m_i
    for *i=1 to K*
      Replace m_i with the mean of all of the samples of trained vector for cluster *i*
    end *(for)*
- ***end*** *(Until)*

A general observation is that different gestures have different trajectories in the cluster space, while the same gesture show very similar trajectories.

4.3 Classification

4.3.1 Hand posture via SVM

In the classification, a symbol is assigned to one of the predefined classes and a fusion of statistical and geometrical feature vectors are used in it. A set of thirteen ASL alphabets (i.e. A, B, C, D, H, I, L, P, Q, U, V, W and Y) and seven ASL numbers (i.e. 0-6) are recognized using SVM and are shown in Figure 7(a) and Figure 7(b) respectively. Classification phase contains two parts. Curvature is analyzed in first part for ASL alphabets where as SVM classifier is used in second part for both ASL alphabets and numbers. The reason for not putting these letters with alphabets is that some letters are very similar to alphabets and it is hard to classify them. For example, 'D' and '1' are same with a small change of thumb. Therefore, unlike alphabets, ASL letters are not categorized into groups and classification is

carried out for a single group. In this way, the first part for ASL numbers is ignored and it includes only SVM classifier part.

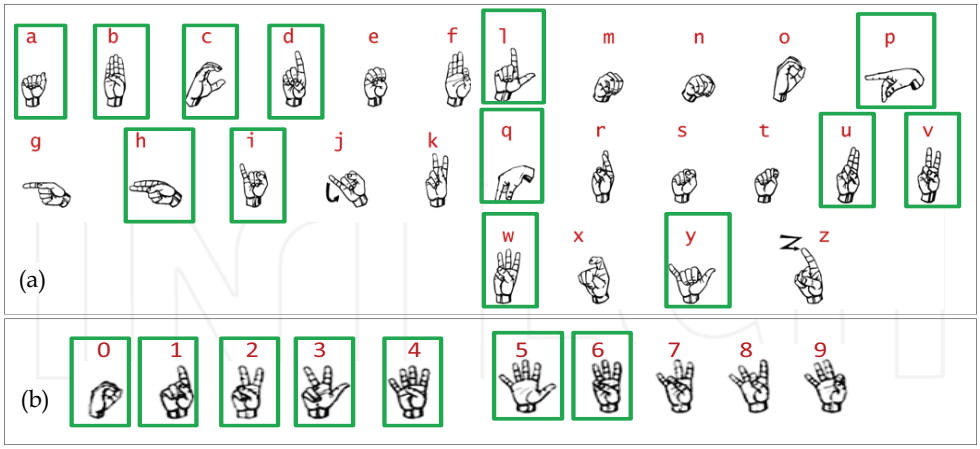


Fig. 7. (a)&(b) Set of ASL alphabets and numbers where rectangles show postures sign used

- **Curvature Analysis**

In the classification, phase, we have used the number of detected fingertips to create the groups for ASL alphabets. These groups are shown in Table 1. The analysis is done to reduce number of signs in each group and to avoid the misclassifications. In the second part, SVM classifies the posture signs based on the detected fingertips.

Group Nr.	Fingers	Posture Symbols
1	0	A, B
2	1	A, B, D, H, I, U
3	2	C, L, P, Q, V,Y
4	3	W

Table 1. The number of detected fingertips in posture alphabets

4.3.2 Hand Gesture via HMMs

To spot meaningful gestures, we construct gesture spotting network as shown in Figure 8. The gesture spotting network can be easily expanded the vocabularies by adding a new meaningful gesture HMMs model and then rebuilding a non-gesture model. Shortly, we mention how to model gesture patterns discriminately and how to model non-gesture patterns effectively. Each reference pattern for Arabic numbers (0-9) is modeled by LRB model with varying number of states ranging from 3 to 5 states based on its complexity. As, the excessive number of states can generate the over-fitting problem if the number of training samples is insufficient compared to the model parameters. It is not easy to obtain the set of non-gesture patterns because there are infinite varieties of meaningless motion. So, all other patterns rather than references pattern are modeled by a single HMM called a non-gesture model (garbage model) (Lee & Kim, 1999; Yang et al., 2007; Elmezain et al.,

2009). The non-gesture model is constructed by collecting the states of all gesture models in the system as follows:

1. Duplicate all states from all gesture models, each with an output observation probabilities. Then, we re-estimate that probabilities with gaussian distribution smoothing filter to makes the states represent any pattern.
2. Self-transition probabilities are kept as in the gesture models.
3. All outgoing transition are equally assigned as:

$$\hat{a}_{ij} = \frac{1 - a_{ij}}{N - 1}, \qquad for\ all\ j, i \neq j \tag{34}$$

where $\hat{a}_{ij}$ represents the transition probabilities of non-gesture model from state i to state j, a_{ij} is the transition probabilities of gesture models from state i to state j and N in the number of states in all gesture models.

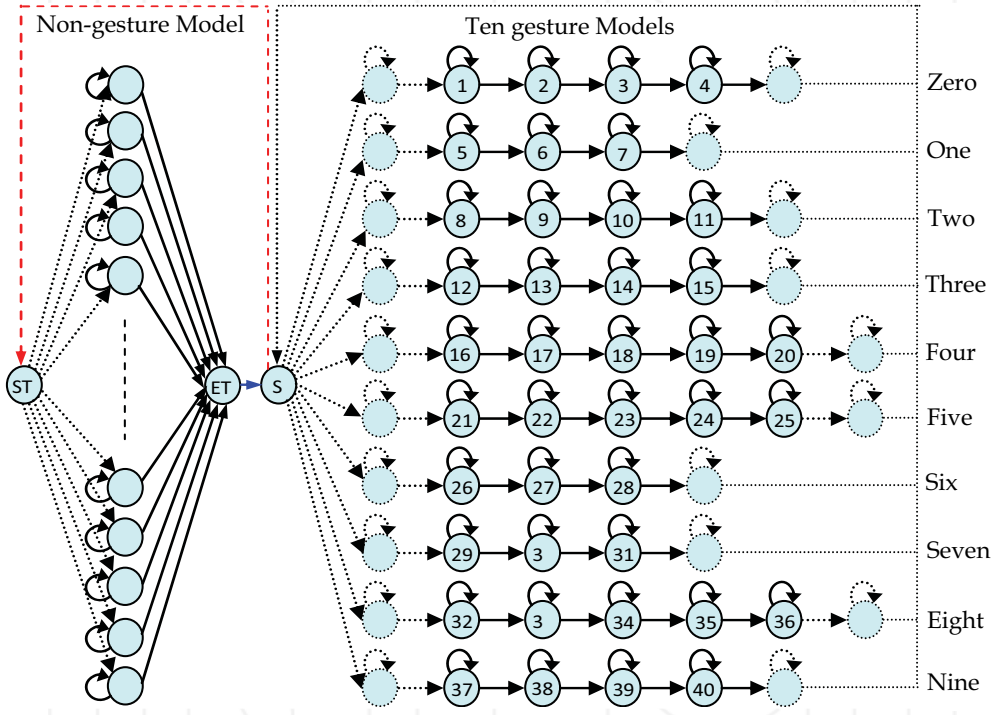


Fig. 8. Gesture spotting network which contains ten gesture models and one non-gesture model with two null states (Start: ST; End: ET)

The non-gesture model (Figure 1(b) & Figure 8) is a weak model for all trained gesture models and represents every possible pattern where its likelihood is smaller than the dedicated model for a given gesture because of the reduced forward transition probabilities. Also, the likelihood of the non-gesture model provides a confidence limit for the calculated likelihood by other gesture models. Thereby, we can use confidence measures as an

adaptive threshold for selecting the proper gesture model or gesture spotting. The number of states for non-gesture model increases as the number of gesture model increases. Moreover, there are many states in the non-gesture model with similar probability distribution, which in turn lead to a waste time and space. To alleviate this problem, a relative entropy (Cover & thomas, 1991) is used. The relative entropy is a measure of the distance between two probability distributions.

Consider two random probability distributions $P = (p_1, p_2, ..., p_M)^T$ and $Q = (q_1, q_2, ..., q_M)^T$, the symmetric relative entropy $D(P||Q)$ is defined as:

$$D(P||Q) = \frac{1}{2}\sum_{i=1}^{M}(p_i \log\frac{p_i}{q_i} + q_i \log\frac{q_i}{p_i}) \tag{35}$$

The proposed state reduction is based on Eq. 35 and works as follows:

1. Calculate the symmetric relative entropy between each probability distribution pair $p^{(l)}$ and $q^{(n)}$ of l and n states, respectively.

$$D(P^{(l)}||Q^{(n)}) = \frac{1}{2}\sum_{i=1}^{M}(p_i^{(l)} \log\frac{p_i^{(l)}}{q_i^{(n)}} + q_i^{(n)} \log\frac{q_i^{(n)}}{p_i^{(l)}}) \tag{36}$$

2. Determine the state pair *(l, n)* with the minimum symmetric relative entropy $D(P^{(l)}||Q^{(n)})$.
3. Recalculate the probability distribution output by merging these two states over the M observation discrete symbol as:

$$p_i^{(l)^*} = \frac{p_i^{(l)} + q_i^{(n)}}{2} \tag{37}$$

4. If the number of states is greater than a threshold value, then go to 1, else re-estimate probability distribution output by gaussian distribution smoothing filter to makes the states represent any pattern

The proposed gesture spotting system contains two main modules; segmentation module and recognition module. In the gesture segmentation module, we use a sliding window which calculates the observation probability of all gesture models and non-gesture model for segmented parts. The start (end) point of gesture is spotted by competitive differential observation probability value between maximal gestures (λ_g) and non-gesture (Figure 9). The maximal gesture model is the gesture whose observation probability is the largest among all ten gesture $p(O|\lambda_g)$. When this value changes from negative to positive (Eq. 38, O can possibly as gesture g), the gesture starts. Similarly, the gesture ended around the time that this value changes from positive to negative (Eq. 39, O cannot be a gesture).

$$\exists g: P(O|\lambda_g) > P(O|\lambda_{non-gesture}) \tag{38}$$

$$\forall g: P(O|\lambda_g) < P(O|\lambda_{non-gesture}) \tag{39}$$

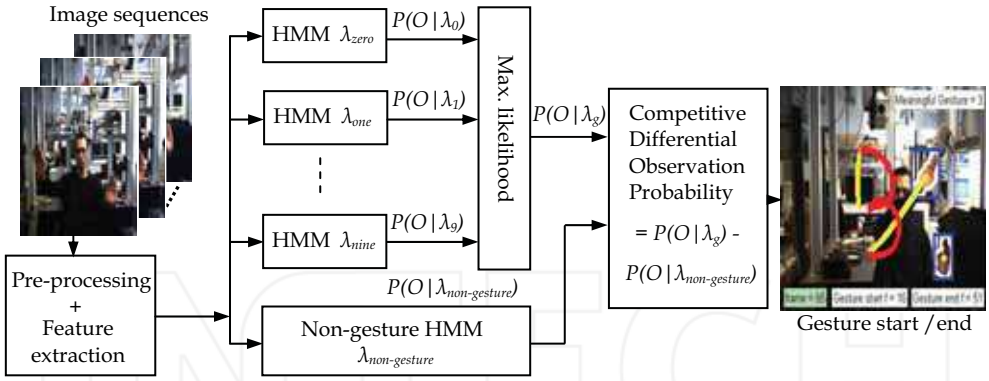


Fig. 9. Simplified structure showing the main module for hand gesture spotting via HMMs

After spotting start point in continuous image sequences, then it activates gesture recognition module, which performs the recognition task for the segmented part accumulatively until it receives the gesture end signal. At this point, the type of observed gesture is decided by Viterbi algorithm frame by frame. The following steps show how the Viterbi algorithm works on gesture model λ_g:

1. Initialization:

$$\delta_1^g(i) = \Pi_i . b_i^g(o_1), \qquad for\ 1 \le i \le N \tag{40}$$

2. Recusion (accumulative observation probability computation):

$$\delta_t^g(j) = \max_i \left[\delta_{t-1}^g(i) . a_{ij}^g\right] . b_j^g(o_t), \qquad for\ 2 \le t \le T, 1 \le j \le N \tag{41}$$

3. Termination:

$$P(O|\lambda_g) = \max_i \left[\delta_T^g(i)\right] \tag{42}$$

where a_{ij}^g is the transition probability from state i to state j, $b_j^g(o_t)$ refers to the probability of emitting o at time t in state j, and $\delta_t^g(j)$ is the maximum likelihood value in state j at time t.

5. Experiments Discussion

A method for detection and segmentation of the hands in stereo color images with complex background is used where the hand segmentation and tracking takes place using 3D depth map, color information, Gaussian Mixture Model (GMM) (Elmezain et al., 2008b; Ming-Hsuan & Narendra, 1999; Phung et al., 2002) and Mean-shift algorithm in conjunction with Kalman filter (Comaniciu et al., 2003). Firstly, segmentation of skin colored regions becomes robust if only the chrominance is used in analysis. Therefore, YC_bC_r color space is used in our approach where Y channel represents brightness and (C_b, C_r) channels refer to

chrominance. We ignore Y channel to reduce the effect of brightness variation and use only the chrominance channels, which fully represent the color information. A large database of skin and non-skin pixels is used to train the Gaussian model. In the training set, 18972 skin pixels from 36 different races persons and 88320 non-skin pixels from 84 different images are used. The GMM technique begins with modeling of skin using skin database where a variant of *k*-means clustering algorithm performs the model training to determine the initial configuration of GMM parameters.
Additionally, blob analysis is used to derive the hand boundary area, bounding box and hand centroid point (Figure 12 & 13). Secondly, after localization of the hand's target from the segmentation step, we find its color histogram with Epanechnikov kernel (Comaniciu et al., 2003). This kernel assigns smaller weights to pixels further from the center to increases the robustness of the density estimation. To find the best match of our hand target in the sequential frames, the Bhattacharyya coefficient (khalid et al., 2006) is used to measure the similarity by maximizing Bayes error that arising from the comparison of the hand target and candidate. We take in our consideration the mean depth value that is computed from the previous frame for the hand region to solve overlapping between hands and face. The mean-shift procedure is defined recursively and performs the optimization to compute the mean shift vector. After each mean-shift optimization that gives the measured location of the hand target, the uncertainty of the estimate can also be computed and then followed by the Kalman iteration, which drives the predicated position of the hand target. Thereby, the hand gesture path is obtained by taking the correspondences of detected hand between successive image frames (Figure 12). The input images were captured by Bumblebee stereo camera system that has 6 mm focal length at 15FPS with 240 $\times$ 320 pixels image resolution, Matlab and C++ implementation. Our experiments are carried out an isolated gesture recognition and meaningful gesture spotting test.

5.1 Experimental results

5.1.1 Hand Posture

For training the data, a database is built which contains 3000 samples for posture symbols taken from eight persons on a set of thirteen ASL alphabets and seven numbers. Classification results are based on 2000 test samples from five persons and sample test data used is entirely different from the training data. The computed features set are invariant to translation, orientation and scaling, therefore posture signs are tested for these properties which is an important contribution of this work. Experimental result shows the probability of posture classification for each class in the group and it is achieved for test data by the analysis of confusion matrixes. The calculated results include the test posture samples (i.e. alphabets and numbers) with rotation, scaling and under occlusion. The diagonal elements in the confusion matrixes represent the percentage probability of each class in the group. Misclassifications between the different classes are shown by the non-diagonal elements. Feature vector set for posture recognition contains the statistical feature vectors and geometrical feature vectors, so the computed confusion matrix from these features gives an inside view about how different posture symbols are similar to each other. Confusion matrixes and classification probabilities of the groups for ASL alphabets are described here:

Group 1 (No Fingertip Detected): Table 2 shows the confusion matrix of ASL alphabet 'A' and 'B'. It is to be noted that there is no misclassification between these two classes. It shows that these posture symbols are very different from each other.

Symbol	A	B
A	100.0	0.0
B	0.0	100.0

Table 2. Confusion Matrix for no fingertip detection. The alphabets in this group are completely different from one another

Group 2 (One Fingertip Detected): Table 3 shows the confusion matrix of the classes with one fingertip detected. The result of misclassification shows the tendency of a posture symbol towards its nearby posture class. Posture symbols are tested on different orientations and back and forth movements. It can be seen that alphabet 'A' results in least misclassification with the other posture symbols because alphabet 'A' is different from other postures in this group. 'H'/'U' has the maximum misclassification with the other posture alphabets. It is observed that the misclassification of 'H'/'U' with 'B' is occurred during the back and forth movement. In general, there are very few misclassifications between these posture signs because of the features which are translation, rotation and scale invariant.

Symbol	A	B	D	I	H/U
A	99.8	0.0	0.0	0.0	0.2
B	0.0	98.18	1.0	0.0	0.82
D	0.0	0.0	98.67	1.33	0.0
I	0.58	0.0	0.8	98.62	0.0
H/U	0.0	3.08	0.0	0.24	96.68

Table 3. Confusion Matrix of the alphabets for one detected fingertip

Group 3 (Two Fingertips Detected): Table 4 shows the confusion matrix of the classes with two fingertips detected. The posture symbols in this group are tested for scaling and rotations. The presented results show that the highest misclassification exists between 'P' and 'Q'. It is due to the reason that these two signs are not very different in shape and geometry. Besides, statistical features in this group are not very different from each other. Therefore, a strong correlation exists between the symbols in this group which leads to the misclassifications between them.

Symbol	C	L	P	Q	V	Y
C	98.65	0.25	0.0	0.75	0.0	0.35
L	0.38	98.5	0.0	0.76	0.0	0.36
P	0.0	0.0	98.74	1.26	0.0	0.0
Q	0.0	0.0	3.78	96.22	0.0	0.0
V	0.20	0.0	0.0	0.0	99.35	0.45
Y	0.0	0.0	0.0	0.0	0.7	99.3

Table 4. Confusion Matrix for the signs having two fingertips detected

Group 4 (Three Fingertips Detected): The posture symbol 'W' only falls in the category of three fingertips detections. Therefore, it always results in the classification of alphabet 'W'.

ASL Numbers: Table 5 shows the confusion matrix of the classes for ASL numbers and these are tested for scaling and rotations. The presented results show the least misclassification of letter '0' with the other classes because its geometrical features are entirely different from the other classes. Highest misclassification exists between letters '4' and '5' as there is a lot of similarity between these signs (i.e. thumb in letter '5' is open). Other misclassifications exists between the letters '3' and '6'.

Numbers	0	1	2	3	4	5	6
0	99.8	0.2	0.0	0.0	0.0	0.0	0.0
1	0.3	99.44	0.26	0.0	0.0	0.0	0.0
2	0.0	0.0	98.34	0.4	0.0	0.0	1.26
3	0.0	0.0	0.42	98.2	0.86	0.0	0.52
4	0.0	0.0	0.0	0.2	98.24	1.56	0.0
5	0.0	0.0	0.0	0.0	2.4	97.6	0.0
6	0.0	0.0	0.8	0.6	0.0	0.0	98.6

Table 5. Confusion Matrix of ASL numbers. The maximum and the minimum classification percentage is for the numbers '0' and '5'

Following are the classification results based on statistical and geometrical feature vectors for posture recognition as shown in Figure 10. This result shows that the SVM clearly defines the boundaries between different classes in a group. In this figure, Y-axis shows the probability of the classes and time domain (frames) are represented in the X-axis. Probabilities computed by SVM of the resultant posture alphabets are higher due to the separation between the posture classes in respective group.

Test Sequence 1 with Classification Results: In Figure 10, the major part of graph includes posture signs 'A', 'B' and some frames at the end shows the posture symbol 'D'. Posture signs 'A' and 'B' are the two signs that are categorized in two groups (i.e. no fingertip detection and one fingertip detected). These symbols in this sequence are tested for rotation and back and forth movement. During the occlusion, posture symbol 'B' is detected and recognized robustly. Figure 10 presents the test sequence with detected contour and fingertips of left hand. It can also be seen that the left hand and right hand can present different posture signs but the presented results here only show the left hand. However, it can be seen that features of posture signs does not affect much under rotation, scaling and under occlusion. Figure 11 presents the classification probabilities for test sequence in Figure 10. The classification presents good results because the probability of resultant class with respect to other classes is high. The discrimination power of SVM can be seen from this behavior and it classifies the posture signs 'A' and 'B' correctly. In the sequence, posture sign change from 'A' to 'B' in frame 90, followed by another symbol change at frame 380 from 'B' to 'D'. Posture sign 'B' is detected robustly despite of orientation, scaling and occlusion. However, misclassifications between the groups can be seen from the graph due to false fingertip detection and segmentation. For example, in the frames where no fingertip is detected, posture signs 'A' and 'B' are classified correctly but misclassifications are observed with other signs in the group with one fingertip detected.

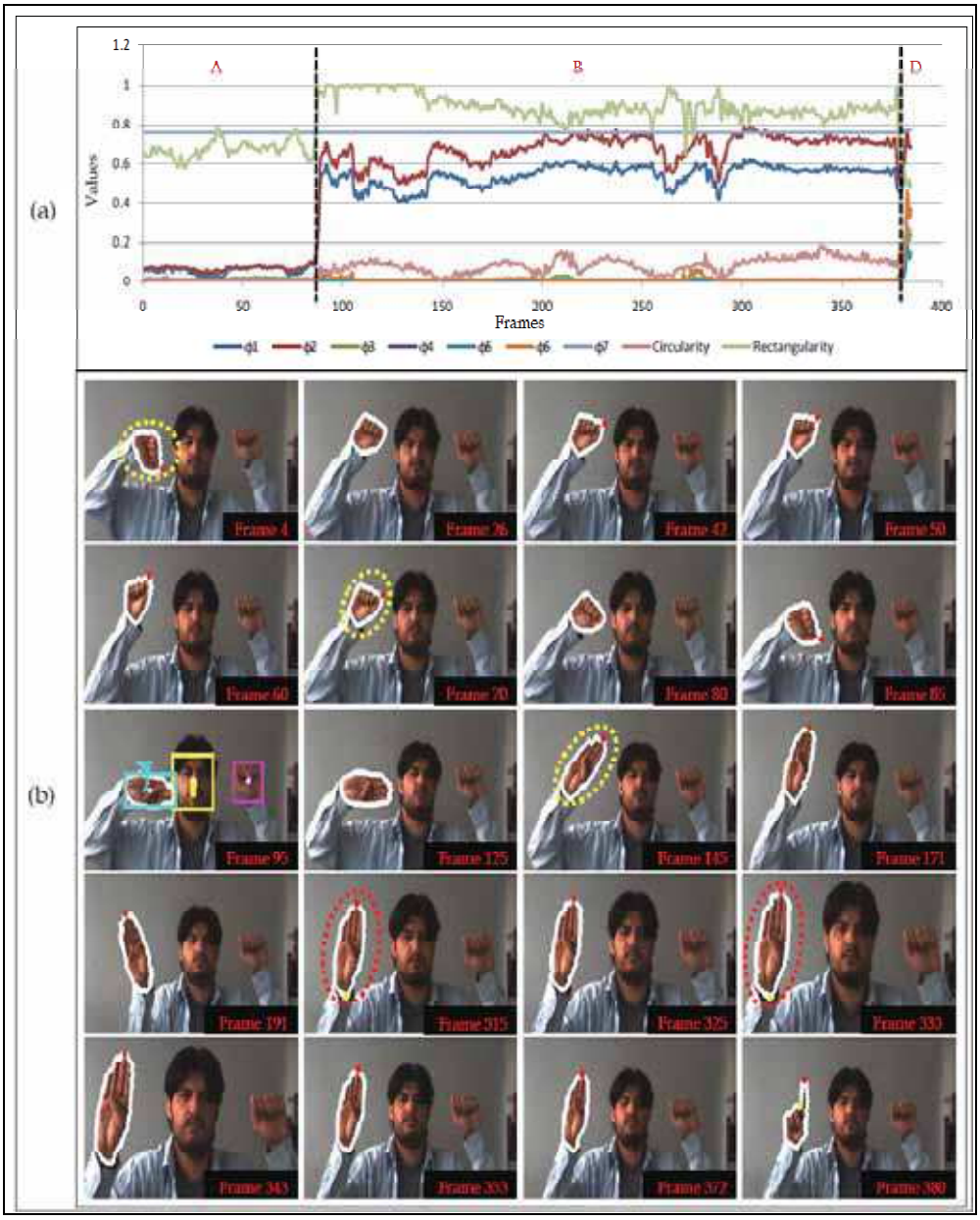


Fig. 10. (a) The graph shows the feature set of the posture signs 'A', 'B' and 'D' (b) Test Sequence "ABD-Sequence" for the postures signs 'A', 'B' and 'D' with different rotations and scaling are presented. Yellow dotted circles show rotation where as back and scaling movements are shown by red dotted circles

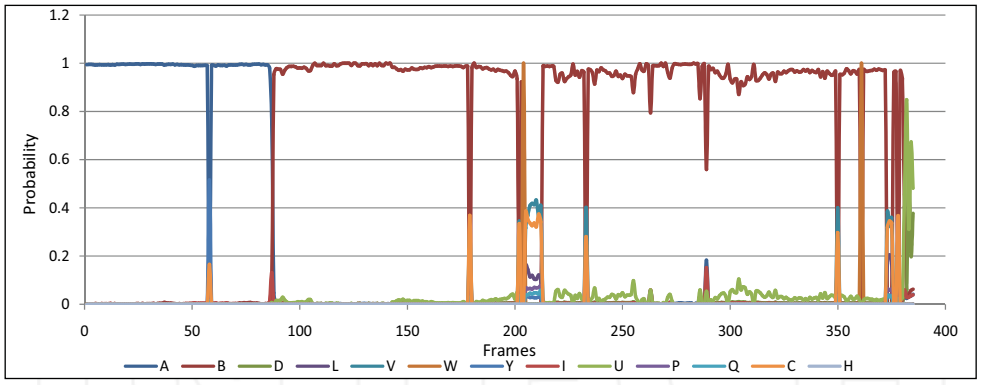


Fig. 11. Classification probability of the test sequence. Blue curve shows the highest probability in the initial frames which classifies 'A', classification for 'B' sign is shown by the brown curve and 'D' is shown in the last frames by the light green curve.

5.1.2 Hand Gesture

In our experimental results, each isolated gesture number from 0 to 9 was based on 60 video sequences, which 42 video samples for training by Baum-Welch algorithm and 18 video samples for testing (Totally, our database contains 420 video samples for training and 180 video sample for testing). The gesture recognition module match the tested gesture against database of reference gestures, to classify which class it belongs to.

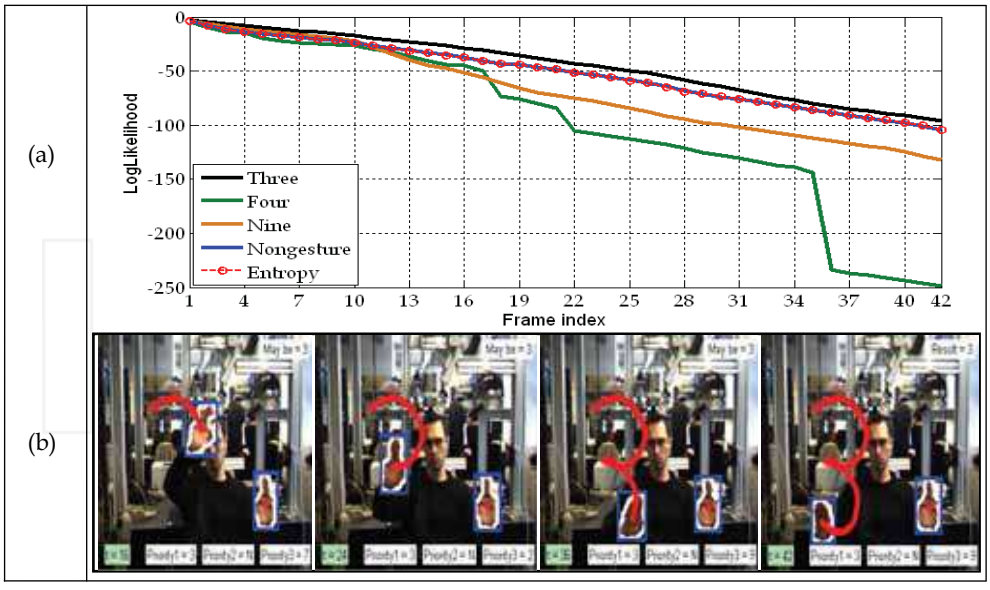


Fig. 12. (a) & (b) Isolated gesture '3' with high three priorities where the probability of non-gesture model before and after state reduction is the same

The higher priority was computed by Viterbi algorithm to recognize the numbers in real-time frame by frame over LRB topology with different number of states ranging from 3 to 5 based on its complexity. We evaluate the gesture recognition according to different clusters number from 28 to 37, based on the numbers of segmented parts in all numbers (0-9) where each straight-line segment is classified into a single cluster. Therefore, Our experiments showed that the optimal number of clusters is equal to 33 where the higher recognition is achieved. In Figure 12(a)&(b) Isolated gesture '3' with high three priorities, where the probability of non-gesture before and after state reduction is the same (the no. of states of non-gesture model before reduction is 40 and after reduction is 28). Additionally, our database also contains 280 video samples for continuous hand motion. Each video sample either contains one or more than meaningful gestures. We measured the gesture spotting accuracy according to different window size from 1 to 8 (Figure 13(a)). We noted that, the gesture spotting accuracy is improved initially as the sliding window size increase, but degrades as sliding window size increase further. Therefore, the optimal size of sliding window is 5 empirically. Also, result of one meaningful gesture spotting '6' is shown in Figure 13(a) where the start point detection at frame 15 and end point at frame 50.

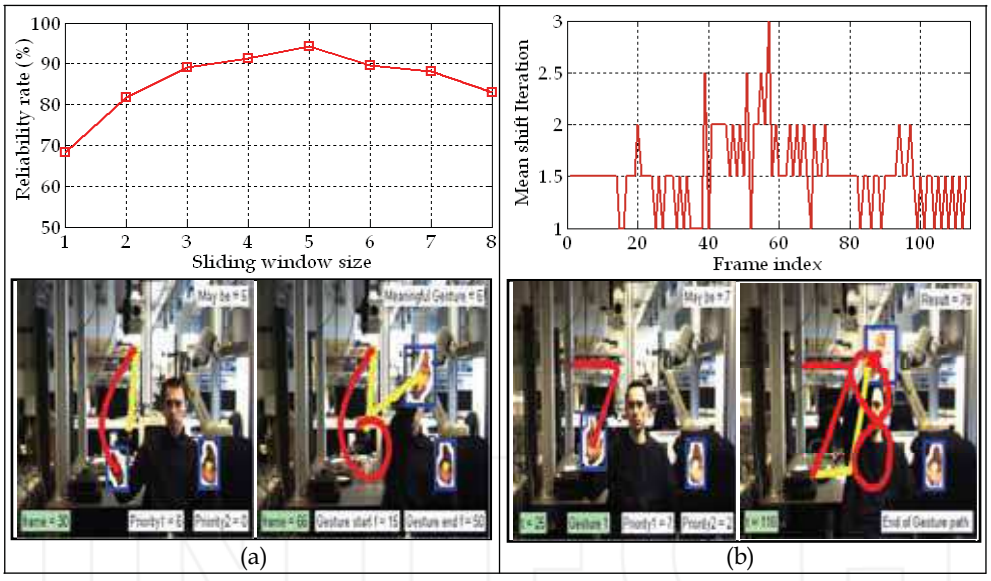


Fig. 13. (a) One meaningful gesture spotting '6' with spotting accuracy for different sliding window size (1-8). (b) Gesture spotting '78' where the mean-shift iteration is 1.52 per frame

Figure 13 (b) shows the results of continuous gesture path that contains within itself two meanningful gestures '7' and '8'. In addition, the mean-shift iteration of continuous gesture path '78' is 1.25 per frame, which in turn would be suitable for real-time implementation. In automatic gesture spotting task, there are three types of errors, namely, insertion, substitution and deletion. The insertion error occurs when the spotter detects a nonexistent gesture. A substitution error occurs when the meaningful gesture is classified falsely. The

deletion error occurs when the spotter fails to detect a meaningful gesture. Here, we note that some insertion errors cause the substitution errors or deletion errors where the insertion errors affect on the the gesture spotting ratio directly. The reliability of automatic gesture spotting approach is computed by Eq. 44 and achieved 94.35% (Table 6).

$$Reliability = \frac{\neq of\ correctly\ recognized\ gestures}{\neq of\ test\ gestures + \neq of\ inseration\ errors} \times 100\% \quad (44)$$

Gesture path	Train Data	Spotting meaningful gestures results					
		Test	Insert	Delete	Substitute	Correct	Rel. (%)
Zero	42	28	1	1	1	26	89.66
One	42	28	0	1	1	26	92.86
two	42	28	0	0	1	27	96.43
Three	42	28	0	0	0	28	100.00
Four	42	28	0	0	1	27	96.43
Five	42	28	0	1	1	26	92.86
Six	42	28	1	1	1	26	89.66
Seven	42	28	0	0	0	28	100.00
Eight	42	28	1	0	2	26	89.66
Nine	42	28	0	1	0	27	96.43
Total	420	280	3	5	8	267	94.35

Table 6. Result of spotting meaningful hand gestures for numbers from 0 to 9 using Hidden Markov Models

6. Summary and Conclusion

This chapter is sectioned into two parts; the first part is related to hand posture and the second part deals with hand gesture spotting. In the hand posture, the database contains 3000 samples for training the posture signs and 2000 samples for testing. The recognition process identifies the hand shape using SVM classifier on the manipulated features of segmented hands. The results for the hand posture recognition for thirteen ASL alphabets is 98.65% and for seven ASL numbers, the recognition rate is 98.60%. For the hand gesture, an automatic hand gesture spotting approach for Arabic numbers from 0 to 9 in stereo color image sequences using HMMs is proposed. The gesture spotting network finds the start and end points of meaningful gestures that is embedded in the input stream by the difference observation probability value of maximal gesture models and non-gesture model. On the other side, it performs the hand gesture spotting and recognition tasks simultaneously where it is suitable for real-time applications and solves the issues of time delay between the segmentation and the recognition tasks. The database for hand gesture contains 60 video sequences for each isolated gesture number (42 video sequences for training and 18 video sequences for testing) and 280 video sequences for continuous gestures. The results show that; the proposed approach can successfully recognize isolated gestures and spotting meaningful gestures that are embedded in the input video stream with 94.35% reliability. In short, the proposed approach can automatically recognize posture, isolated and meaningful hand gestures with superior performance and low computational complexity when applied on several video samples containing confusing situations such as partial occlusion.

7. Acknowledgments

This work was supported by Transregional Collaborative Research Centre SFB/TRR 62 "Companion-Technology for Cognitive Technical Systems" funded by the German Research Foundation (DFG).

8. References

Alon, J.; Athitsos, V. & Scharoff, S. (2005). Accurate and Efficient Gesture Spotting via Pruning and Subgesture Reasoning. In Lecture Notes in Computer Sciences, Springer Berlin / Heidelberg, Vol. 3766, pp. 189-198, ISBN 978-3-540-29620-1

Bowden, R.; Zisserman, A.; Kadir, T; & Brady, M. (2003). Vision Based Interpretation of Natural Sign Languages. In International Conference on Computer Vision Systems.

Burges, C. (1998). A Tutorial on Support Vector Machines for Pattern Recognition. In Proceeding of Data Mining and Knowledge Discovery, Vol. 2, No. 2, pp. 121-167

Chang, C.C. & Lin, C.J. (2001). LIBSVM: a library for support vector machines, http://www.csie.ntu.edu.tw/~cjlin/libsvm

Cristianini, N. & Taylor, J. (2001). An Introduction to Support Vector Machines and other kernel based learning methods, Cambridge University Press, ISBN-10 : 0521780195

Comaniciu, D.; Ramesh, S. & Meer, P. (2003). Kernel-Based Object Tracking. In IEEE Transaction on PAM I, Vol. 25, No. 5, pp. 564-577, ISSN 0162-8828

Cover, T.M. & Thomas, J.A. (1991). Entropy, Relative Entropy and Mutual Information. In Elements of Information Theory, pp. 12-49

Davis, J. & Bradski, G. (1999). Real-time Motion Template Gradients using Intel CVLib. In Proceeding of IEEE ICCV Workshop on Framerate Vision, pp. 1-20

Deyou, X. (2006). A Network Approach for Hand Gesture Recognition in Virtual Reality Driving Training System of SPG. In ICPR, pp. 519-522

Ding, C. & He, X. (2004). K-means Clustering via Principal Component Analysis. In International Conference on Machine Learning (ICML), pp. 225-232

Elmezain, M.; Al-Hamadi, A.; Krell, G.; El-Etriby, S. & Michaelis, B. (2007). Gesture Recognition for Alphabets from Hand Motion Trajectory Using Hidden Markov Models. In IEEE of ISSPIT, pp. 1192-1197

Elmezain, M.; Al-Hamadi, A. & Michaelis, B. (2008a). Real-Time Capable System for Hand Gesture Recognition Using Hidden Markov Models in Stereo Color Image Sequences. In Journal of WSCG'08, Vol. 16, No. 1, pp. 65-72, ISSN 1213-6972

Elmezain, M.; Al-Hamadi, A.; Appenrodt, J. & Michaelis, B. (2008b). A Hidden Markov odel-Based Continuous Gesture Recognition System for Hand Motion Trajectory. In International Conference on Pattern Recognition (ICPR), pp. 519-522

Elmezain, M.; Al-Hamadi, A. & Michaelis, B. (2009). A Novel System for Automatic Hand Gesture Spotting and Recognition in Stereo Color Image Sequences. In Journal of WSCG'09, Vol. 17, No. 1, pp. 89-96, ISSN 1213-6972

Flusser, J. & Suk, T. (1993). Pattern Recognition by Affine Moment Invariants. In Journal of Pattern Recognition, Vol. 26, No. 1, pp. 167-174

Freeman, W. & Roth, M. (1994). Orientation histograms for hand gesture recognition. In International Workshop on Automatic Face and Gesture Recognition, pp. 296-301

Goronzy, S. (2002). Robust Adaptation to Non-Native Accents in Automatic Speech Recognition. In Lecture Notes in Computer Sciences, Springer, ISBN-13: 978-540003250

Handouyahia, M.; Ziou, D. & Wang, S. (1999). Sign Language Recognition Using Moment-Based Size Functions. In International Conference of Vision Interface, pp. 210-216

Hu, M. (1962). Visual Pattern Recognition by Moment Invariants. In IRE Transaction on Information Theory, Vol. 8, No. 2, pp. 179-187, ISSN 0096-1000

Hussain, M. (1999). Automatic Recognition of Sign Language Gestures. Master Thesis, Jordan University of Science and Technology

Kang, H. ; Lee, C. & Jung, K. (2004). Recognition-based Gesture Spotting in Video Games. In Journal of Pattern Recognition Letters, Vol. 25, No. 15, pp. 1701-1714, ISSN 0167-8655

Kanungo, T. ; Mount, D. M. ; Netanyahu, N. ; Piatko, C. ; Silverman, R. & Wu, A.Y. (2002). An Efficient k-means Clustering Algorithm: Analysis and Implementation. In IEEE Transaction on PAMI, Vol. 24, No. 3, pp. 881-892, ISSN 0162-8828

Khalid, S. ; Ilyas, U. ; Sarfaraz, S. & Ajaz, A. (2006). Bhattacharyya Coefficient in Correlation of Gary-Scale Objects. In Journal Multimedia, Vol. 1, No. 1, pp. 56-61, ISSN 1796-2048

Kim, D.; Song, J. & Kim, D. (2007). Simultaneous Gesture Segmentation and Recognition Based on Forward Spotting Accumlative HMMs. In Journal of the Pattern Recognition Society, Vol. 40, No. 11, pp. 3012-3026, ISSN 0031-3203

Lee, H. & Kim, J. (1999). An HMM-Based Threshold Model Approach for Gesture Recognition. IEEE Transaction on PAMI, Vol. 21, No. 10, pp. 961-973, ISSN 0162-8828

Licsar, A. & Sziranyi, T. (2002). Supervised Training Based Hand Gesture Recognition System. In International Conference on Pattern Recognition, pp. 999-1002

Lin, C.J. & Weng, R. (2004). Simple Probabilistic Predictions for Support Vector Regression. In Technical report, Department of Computer Science, National Taiwan University

Maitra, S. (1979). Moment Invariants. In Proceeding of the IEEE, Vol. 67, pp. 697-699

Malassiotis, S. & Strintzis, M. (2008). Real-time Hand Posture Recognition using Range Data. In Image and Vision Computing, Vol. 26, No. 7 pp. 1027-1037, ISSN 0262-8856

Mitra, S. & Acharya, T. (2007). Gesture Recognition: A Survey. In IEEE Transaction on Systems, MAN, and Cybernetics, Vol. 37, No. 3, pp. 311-324, ISSN 1094-6977

Ming-Hsuan, Y. & Narendra, A. (1999). Gaussian Mixture Modeling of Human Skin Color and Its Applications in Image and Video Databases. In SPIE/EI&T Storage and Retrieval for Image and Video Databases, pp. 458-466

Niese, R.; Al-Hamadi, A. & Michaelis, B. (2007). A Novel Method for 3D Face Detection and Normalization. In Journal Multimedia, Vol. 2, No. 5, pp. 1-12, ISSN 1796-2048

Phung, S.L.; Bouzerdoum, A. & Chai, D. (2002). A Novel Skin Color Model in YCbCr Color Space and its Application to Human Face Detection. In IEEE International Conference on Image Processing (ICIP), pp. 289-292, ISSN 1522-4880

Rabiner, L. R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In Proc. of the IEEE, Vol. 77, No. 2, pp. 257-286, ISSN 0018-9219

Scott, D., W. (1992). Multivariate Density Estimation. In Wiley, 1992.

Suykens, J. ; Gestel, T.; Brabenter, J.; Moor, B. & Vandewalle, J. (2005). Least Squares Support Vector Machines, World Scientific, ISBN 9812381511.

Takahashi, K.; Sexi, S. & Oka, R. (1992). Spotting Recognition of Human Gestures From Motion Images. In Technical Report IE92-134, pp. 9-16

Yang, H.; Park, A. & Lee, S. (2007). Spotting and Recognition for Human-Robot Interaction. In IEEE Transaction on Robotics, Vol. 23, No. 2, pp. 256-270, ISSN 1552-3098

Advanced Technologies

Edited by Kankesu Jayanthakumaran

ISBN 978-953-307-009-4
Hard cover, 698 pages
Publisher InTech
Published online 01, October, 2009
Published in print edition October, 2009

This book, edited by the Intech committee, combines several hotly debated topics in science, engineering, medicine, information technology, environment, economics and management, and provides a scholarly contribution to its further development. In view of the topical importance of, and the great emphasis placed by the emerging needs of the changing world, it was decided to have this special book publication comprise thirty six chapters which focus on multi-disciplinary and inter-disciplinary topics. The inter-disciplinary works were limited in their capacity so a more coherent and constructive alternative was needed. Our expectation is that this book will help fill this gap because it has crossed the disciplinary divide to incorporate contributions from scientists and other specialists. The Intech committee hopes that its book chapters, journal articles, and other activities will help increase knowledge across disciplines and around the world. To that end the committee invites readers to contribute ideas on how best this objective could be accomplished.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Mahmoud Elmezain, Ayoub Al-Hamadi, Omer Rashid and Bernd Michaelis (2009). Posture and Gesture Recognition for Human-Computer Interaction, Advanced Technologies, Kankesu Jayanthakumaran (Ed.), ISBN: 978-953-307-009-4, InTech, Available from: http://www.intechopen.com/books/advanced-technologies/posture-and-gesture-recognition-for-human-computer-interaction

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821