Neural Comput & Applic (2018) 29:1129–1141
https://doi.org/10.1007/s00521-016-2525-z

ORIGINAL ARTICLE

Dynamic hand gesture recognition using vision-based approach for human–computer interaction

Joyeeta Singha[1] · **Amarjit Roy**[1] · **Rabul Hussain Laskar**[1]

Received: 28 January 2016 / Accepted: 8 August 2016 / Published online: 13 August 2016

Abstract In this work, a vision-based approach is used to build a dynamic hand gesture recognition system. Various challenges such as complicated background, change in illumination and occlusion make the detection and tracking of hand difficult in any vision-based approaches. To overcome such challenges, a hand detection technique is developed by combining three-frame differencing and skin filtering. The three-frame differencing is performed for both colored and grayscale frames. The hand is then tracked using modified Kanade–Lucas–Tomasi feature tracker where the features were selected using the compact criteria. Velocity and orientation information were added to remove the redundant feature points. Finally, color cue information is used to locate the final hand region in the tracked region. During the feature extraction, 44 features were selected from the existing literatures. Using all the features could lead to overfitting, information redundancy and dimension disaster. Thus, a system with optimal features was selected using analysis of variance combined with incremental feature selection. These selected features were then fed as an input to the ANN, SVM and kNN model. These individual classifiers were combined to produce classifier fusion model. Fivefold cross-validation has been used to evaluate the performance of the proposed model. Based on the experimental results, it may be concluded that classifier fusion provides satisfactory results (92.23 %) compared to other individual classifiers. One-way analysis of variance test, Friedman's test and Kruskal–Wallis test have also been conducted to validate the statistical significance of the results.

✉ Joyeeta Singha
joyeetasingha26@gmail.com

[1] Department of ECE, NIT Silchar, Silchar, India

Keywords Human–computer interaction · Hand gesture recognition · KLT · ANOVA · IFS

1 Introduction

Nonverbal communication that includes communication through body postures, hand gestures and facial expressions makes up most of all communication among human. Hand gestures are one of the most common forms of communication to interact with human to human or human to machine. Hand gestures consist of specific linguistic content, whereas other forms of communications are general emotional state. Due to its speed, simplicity and naturalness, hand gestures have been widely used in sign languages and human–computer interaction systems [1]. Hand gesture recognition provides human to interact with computer in more natural and effective way. The hand gesture recognition system available in the literature has successful applications in computer games, sign-to-text translation systems, sign language communication [2, 3], robotics [4] and video-based surveillance.

Hand gestures may be static or dynamic. In static gesture recognition, the hand shape, size of palm, length and width of fingers need to be kept in mind [5]. Dynamic hand gestures need spatiotemporal information to track hand [6]. Hand detection and tracking is the initial step in any hand gesture recognition system. Comaniciu et al. proposed a model to track hand using color histogram [7]. They used the color histogram of the detected hand as the mean shift input to locate and track hand in the video sequences. But the drawback of the model was that it was unable to detect hand when background had similar color as object. Similarly, Chai et al. [8] and Wang et al. [9] used skin-colored information to detect hand. YCbCr color space model was

used for segmentation. Guo et al. [10] proposed a hand tracking system using skin filtering, pixel-based hierarchical feature for AdaBoosting, and codebook background cancelation. But, the background has to be known a priori. Camshift—an improved version of mean shift algorithm, was widely used for tracking objects [11]. This algorithm has found to be efficiently track hand in a simple background scene, but it cannot give the same result when the target is occluded with other skin-colored objects. Shi and Tomasi [12] selected the corner points with high intensities as the features to track target object. Though good tracking results have been observed, the feature points go on decreasing with succeeding video frames. This happens due to change in illumination or change in appearance of the hand. Asaari et al. [13] integrated adaptive Kalman filter and eigenhand to track hand under different challenging environment. But the algorithms fails when there is presence of large-scale variations and pose changes. Kolsch and Turk [14] introduced a KLT tracker-based hand tracking algorithm. This tracker fails when there is shape transformation of the hand. Nowadays, many depth-based hand detections [15] are observed, but 3D gesture interaction is not much user friendly.

The contributions of our paper are as follows. Firstly, a database has been developed using bare hand, namely '*NITS hand gesture database IV*,' for 40 class of gestures (10 numerals, 26 alphabets and 4 arithmetic operators). This database has also been made publicly available. Secondly, a new hand detection scheme has been developed by combining three-frame differencing and skin filtering. The hand was then tracked using modified KLT algorithm which included additional compact criteria, velocity and direction information along with traditional KLT algorithm. Finally color cues were used to detect the final hand region in every frame. Thirdly, ANOVA and IFS techniques were used to select the optimal features from the 44 existing features. Study on the effect of different combination of features made by IFS using individual classifiers such as ANN, SVM and kNN was made in this paper. Lastly, classifier fusion technique was developed by combining the results of individual classifiers such as ANN, SVM and kNN. Moreover, one-way analysis of variance test, Friedman's test and Kruskal–Wallis test were also conducted to validate the statistical significance of the results.

The paper is organized as follows. The architecture of the proposed gesture recognition system with details about each subsystems is presented in Sect. 2. The different experimental results obtained during hand detection, hand tracking, feature selection and classification are discussed in Sect. 3. Finally, the paper is concluded in Sect. 4.

2 Proposed system

There are five phases in our proposed system. They are hand detection, hand tracking, feature extraction, feature selection and classification. Figure 1 shows the block diagram of the proposed hand gesture recognition system. The details of each phases are provided in the following subsections.

2.1 Hand detection

The first step in any hand gesture recognition system is the segmentation of hand from the background. For this, the first three frames of the video sequence were considered. The system architecture for detection of hand is shown in Fig. 2. The algorithm for hand detection is presented as 'Algorithm 1.' This process includes three steps whose results are combined to obtain the desired hand. They are:

- Face detection followed by skin filtering
- Three-frame differencing for colored frames
- Three-frame differencing for grayscale frames

Initially, the face of the user gesticulating (if present) is detected and removed from the second frame of the video using the Viola–Jones algorithm [16, 17]. After the face is removed, skin filtering [8] is performed to obtain the skin-colored objects in the frame. On the other hand, three-frame differencing is performed with the first three frames. It is computed for both colored and grayscale frames. Morphological operations are carried out as shown in Fig. 2 in order to achieve the desired results. The results of the skin filtering and three-frame differencing are combined to obtain the desired hand from the background. But, it has been observed

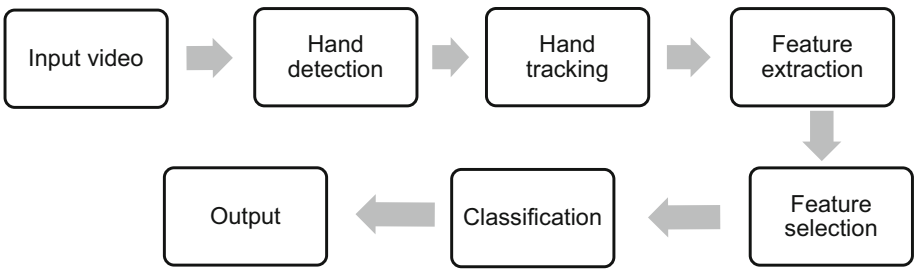


Fig. 1 Proposed system

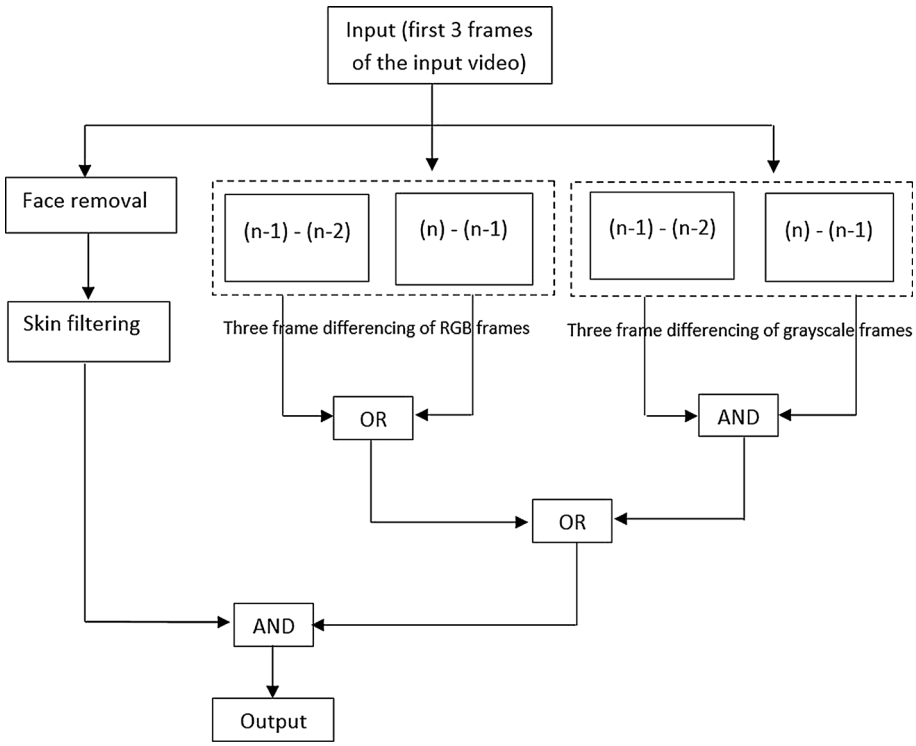


Fig. 2 System architecture of the hand detection

that along with the desired hand, there is other small skin-colored objects in the surroundings. Thus, the largest binary linked object was filtered from the other objects which are considered to be the desired hand in the background.

2.2 Hand tracking

The hand detected is tracked using three steps such as initialization of tracking region, exaction of features from

Algorithm 1 Hand segmentation

```
for i = 1 to 3 do
    A(i) = input rgb frames
    A1(i) = Convert rgb frame A(i) to grayscale frame
    if i = 2 then
      B = face detection using Viola-Jones algorithm
      B1 = remove face (B) from frame A(i)
      B2 = skin filtering of B1 image
    end if
    if i = 3 then
      C1 = skin filtering of (A(i-2) – A(i-1))
      C2 = skin filtering of (A(i-1) – A(i))
      D = OR (C1, C2)            //morphological OR operation
      E1 = gray to binary (A1(i-2) – A1(i-1))
      E2 = gray to binary (A1(i-1) – A1(i))
       F = AND (E1, E2)          //morphological AND operation
      G = OR (F, D)
      H = AND (G, B2)
      H1 = morphological opening of image H
     Final = BBLOB (H1)       //Finding the biggest binary linked object
    end if
end for
Result→'Final' provides the final hand detected
```

initialized tracking region and refining tracking region as shown in Fig. 3. The steps are presented in 'Algorithm 2.' The details of each step are discussed in the following subsections.

2.2.1 Initialization of tracking region

In traditional CamShift algorithm or KLT tracker, the initial tracking region needs to be selected manually. But to make a system robust, automatic selection of tracking region is necessary. In this proposed system, initialization of the first tracking window has been made automatic by considering the detected hand as the initial tracking window.

Algorithm 2 Hand tracking

```
1:  Find the minimum eigen features from the detected hand region
2:  Check and modify the features according to the compact criteria
3:  for i = 3 to N do
4:      features tracked using KLT feature tracker
5:      A(i) = velocities of each feature points
6:      T1= mean (A(i))
7:      if A(i) < T1 then          //if velocity of any feature < T1
8:          selected feature points are removed
9:      end if
10:     B(i) = find direction of movement of the remaining feature points
11:     B1(i) = mode (B(i))
12:     if B(i) ~= B1(i) then     //if direction of any feature is different
13:         selected feature points are removed
14:     end if
15:     bounding box over the feature points
16:     Camshift algorithm used over final selected region
17:     detect minimum eigen features and check for the compact criteria again in new tracked hand region
18: end for
```

2.2.2 Extraction of features from initialized tracking region

Selecting good features from the initial tracking window is very important. The feature points should satisfy three rules:

- It is in the tracking region.
- It should not be spread far from each other over the hand.
- It should not be concentrated on a small part of the hand.

By using KLT feature tracker, the feature points go on decreasing at next frames. This is because the points are lost in the succeeding frames of the video. A time comes when there are no more features to be tracked, and thus, tracking is lost. Moreover, if the hand is occluded with other skin-colored objects, the tracker gets confused, and thus, the some of the features are lost or it wrongly tracks the features which do not belong to the hand. To minimize such difficulties, we used compact criteria to select the optimal feature points so that the features are not too sparsely spread over the hand. This compact criterion is based on the centroid of the feature points. The traditional way of computing centroid directly by averaging the position of feature points is very sensitive to outliers. The detailed steps of compact criterion are provided below:

- The distance between each point and the remaining points and weight each point were calculated as shown in Eq. (1).

$$w_i = \frac{1}{\sum_{k \neq i} ||x_k - x_i||^2} \tag{1}$$

- According to the equation, if a feature point x_i is far from the other feature points, then it has a small weight. Alternately, if x_i is near to most of the other feature points, then it will have a larger weight. After normalizing the weights as $\sum_i w_i = 1$, the centroid is calculated as:

$$x_c = \sum_i w_i x_i \tag{2}$$

- The final step of compact criteria for a feature point x_i is expressed as $|x_k - x_i|_2 < r_o$, where r_o is a distance

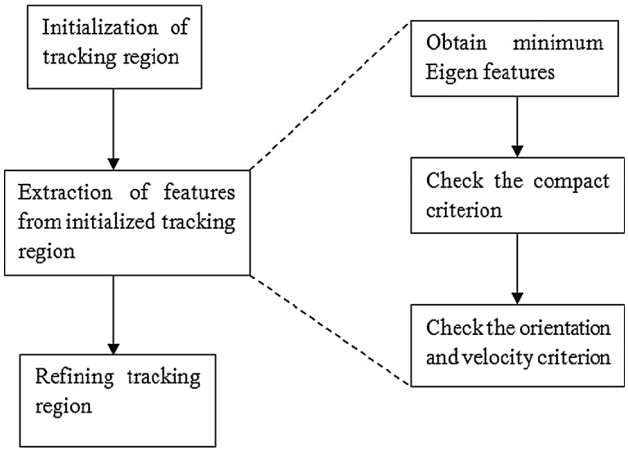


Fig. 3 System architecture of proposed hand tracking

threshold. This criterion helps to eliminate the redundant feature points and ensures that the feature points are concentrated around the centroid.

In the next step, the system was checked for the presence of any skin-colored static objects in the background which could lead to confusion with the hand. Thus, velocity information was added to remove the feature points corresponding to the static objects. The feature points having velocity less than a threshold value $((2 * V_{avg})/3)$ are rejected, where V_{avg} is the average velocity. After this, the orientation information was used to detect the direction along which the feature points are moving in the consecutive frames. Figure 4a shows an example where direction of orientation is shown with yellow lines. The red points correspond to the feature points in previous frame, and yellow points refer to feature points tracked in current frame. Firstly, the 360° is spited into 8 bins as shown in Fig. 4b. Then, all the velocity orientation of feature movements is calculated. Figure 4c shows the histogram of the direction of orientation. Finally, the bin corresponding with the most feature points is considered as the main orientation. The feature points moving in different direction compared to the direction along which majority feature points are moving are rejected for tracking in the next frame. The above steps help to remove the redundant feature points. The red bounding box in Fig. 4a shows the final tracked hand region, whereas the green bounding box shows the detected hand region without the orientation feature.

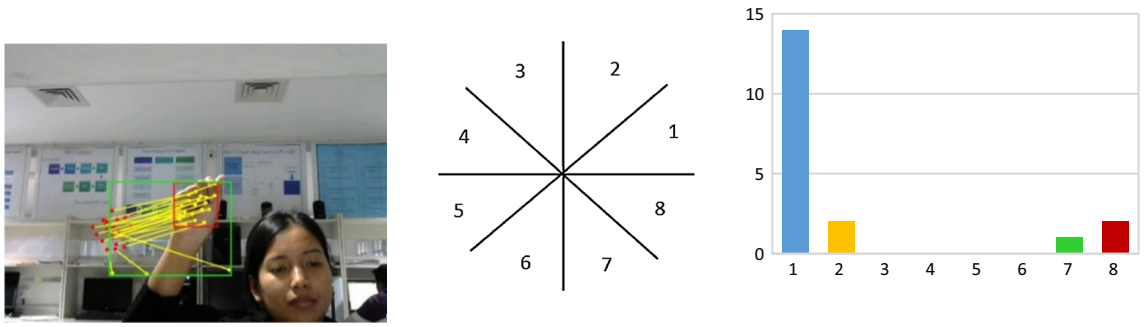
2.2.3 Refining tracking region

The bounding box calculated from former step cannot represent the hand region precisely. This is because although compact criterion selects appropriate feature points, sometimes the features are not uniform in the hand region and the existence of tracking failure makes the tracking result not reliable. Thus, CamShift algorithm can maximize the probability of skin region in the tracking window in a few iteration. This makes the tracking process more stable. Finally, the features are again generated at every 30 frames and the above process is carried out accordingly so as to avoid loss of information from feature points.

2.3 Trajectory smoothening

The trajectory of the gesture is obtained by joining the centroid points of the tracked region during tracking of hand at every video frame. This gesture trajectory is generally noisy because of the factors like movement of hand. Thus, the gesture trajectory must be smoothened before processing further steps. Douglas–Peucker algorithm [18] is applied to the gesture trajectory to smoothen the gesture trajectory. The gesture trajectory smoothened using the Douglas–Peucker algorithm showed better results compared to the smoothening process used by Bhuyan et al. [19] and Singha et al. [20, 21].

The self co-articulated strokes were detected after the smoothening of the gesture trajectory and removed from the



(a) **(b)** **(c)**

Fig. 4 **a** Tracking result, **b** 8-bin segments of orientation and **c** result histogram

gesture using the same steps performed in our previous paper [20]. These strokes were removed in this stage because these hand movements are not a part of the gesture.

2.4 Extraction of features

For matching the trajectory, 44 features were considered in this paper. We used the features from different literatures [19, 20, 22–27]. The total set of features used in our system is presented in Table 1.

2.5 Selection of optimal features

From the 44 features extracted in our system, the best set of features was selected using feature selection technique by ANOVA and IFS. This could reduce the overfitting and information redundancy in the model. The two-level feature selection technique is developed as explained in our previous paper [34] to obtain the optimal features.

2.6 Classification

2.6.1 ANN-based classification

The ANN architecture used for the proposed system has three layers: 1 input, 1 hidden and 1 output. Input layer consists of 40 neurons which represent the 40 features, and output layer has 40 neurons representing 40 gesture classes. The network was trained with different numbers of hidden units such as 50, 52, 54, 56 and 58. This helped the system to capture the network structure with highest train accuracy. The weights were adjusted by back-propagation algorithm. The optimum network achieved for our system was 40L-54N-40L. Then, the testing was performed using fivefold cross-validation process.

2.6.2 SVM-based classification

Here, the dataset used in our system was trained with different kernel functions: linear, quadratic, polynomial and

Table 1 Description on features used

Existing features	Feature description	
Rubine's feature [25]	Cosine and sine of the initial angle with respect to the x-axis	The cosine and sine of the angle between the first and last point
	The length of bounding box diagonal	The total gesture length
	The angle of the bounding box	The total traversed angle
	The distance between first and last point	Maximum speed squared
		Stroke duration
E-Rubine's feature [24]	Number of stop points	Distance from the start to end point in relation to the diagonal
	Distance from start to center point in relation to the diagonal	Total number of strokes
	Direction of the first half of the stroke	Straightness
	Direction of the second half of the stroke	Total distance between strokes
	Angle between the first and second half of the stroke	Angle between strokes
		Stroke distance in relation to each other
Location feature [19, 26, 27]	Average distance from the center of the gesture trajectory to each trajectory points	
Orientation feature [22, 26]	Motion chain code	Orientation of end hand
	Orientation of start hand	Number of significant curves in a gesture
Velocity feature [26]	Average velocity	
Acceleration feature [19]	Acceleration between consecutive trajectory points	
Velocity profile [19]	Number of maxima and number of minima in velocity profile of a gesture	
Position feature [20]	The position of the start hand and the end hand using the 6 quadrants	
Self co-articulated feature [20]	Number of self co-articulation	
	Orientation of self co-articulated strokes	
	Position of start and end of self co-articulated strokes	
Ratio feature [20]	Ratio of longest to shortest distance from the center of the gesture to trajectory points	
Distance feature [20]	Average distance from start to end of the gesture	
Ellipse fitted orientation feature [23]	Orientation is calculated for every ellipse fitted for every 6 consecutive trajectory points	
Length of major axis [23]	Length of the major axis of each ellipse fitted	
Position feature [23]	Position of the start hand and the end hand using the 3 quadrants	

radial basis function. During training, the kernel function which provides the best results was used for the testing. After the training phase, the testing was performed using fivefold cross-validation process.

2.6.3 kNN-based classification

The system was trained for different values of k such as 3, 5, 7 and 9 during the training phase. Odd values of k have been selected so as to avoid draw votes. After the training phase, the testing was performed using fivefold cross-validation process.

2.6.4 Classifier fusion

The results of individual classifiers such as ANN, SVM and kNN were combined to get the classifier fusion result. Majority voting technique was used for combining the individual classifiers. It has been observed that combining the results of the individual classifiers provides result which was more desirable than the individual ones [28]. After the classifier fusion was performed on training set, fivefold cross-validation was carried out.

3 Experimental results

For the experiments, we developed *'NITS hand gesture database IV.'* It consists of 40 gesture class (10 numerals, 26 alphabets and 4 arithmetic operators) gesticulated by 20 users. A total of 9600 gestures were used as training dataset and 2000 samples as testing dataset. Some of the samples of the total database are available in http://www.joyeetasingha26.wix.com/nits-database. The proposed system was tested on Windows 8-based Intel Core I7 processor with 4 GB RAM, and all the experiments were performed using MATLAB R2013a. The users were asked to gesticulate keeping the following conditions into account.

- The background should not consist of moving skin-colored objects at the start of the video recording.
- The hand should be already available at the start position of the gesture before the video recording starts.
- The palm of the hand should be moving at the same place for few seconds to detect the presence of hand.
- The hand is then moved smoothly and slowly to the most prominent gesture positions.
- The hand is kept in the final gesture position for few seconds to complete the gesture.

The performance of hand detection and tracking is provided in Sect. 3.1. The experimental results of optimal feature selection using ANOVA and IFS are provided in Sect. 3.2. Section 3.3 includes the results observed from the individual classifiers such as ANN, SVM and kNN. Moreover, the results of the classifier fusion have also been provided here. Three statistical tests such as 'one-way analysis of variance,' 'Friedman's test' and 'Kruskal–Wallis test' were performed to test the statistical significance of the classifier fusion. Also the stability of the algorithm to Gaussian noise is examined in Sect. 3.4. Finally, a comparison is provided in Sect. 3.5.

3.1 Performance of the hand detection and tracking

The objective analysis of the results obtained using the proposed algorithm for hand detection and tracking is provided in Figs. 5 and 6, respectively. The proposed hand detection technique is compared with the existing techniques such as skin filtering [2] and two-frame differencing [13]. Figure 5 shows the comparative analysis of the proposed hand detection algorithm with other techniques. The result obtained using the proposed hand tracking algorithm is compared with the other three state-of-the-art object tracking algorithms such as CamShift [29], KLT [14] and particle filter [30].

From Fig. 6, it can be seen that the CamShift algorithm fails if there is any overlapping between hand and other skin-colored objects in surrounding. The feature points are lost at certain interval of time when KLT algorithm is used, whereas in case of particle filter, as the number of particles is increased, the complexity of the algorithm increases resulting in delay. The proposed tracking algorithm is able to handle all the above problems during tracking discussed above.

3.2 Performance of the feature selection

The feature selection was performed as described in Sect. 2.5. The ANOVA was used to find the F value of

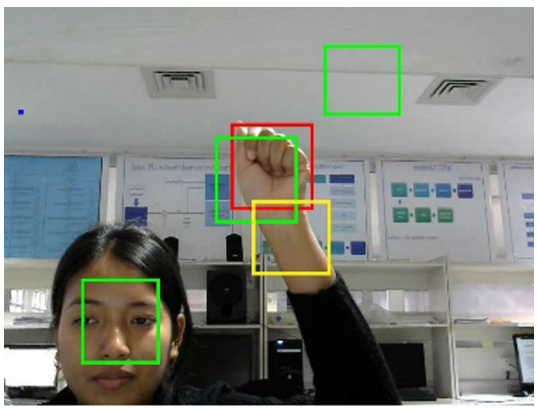


Fig. 5 Comparison of proposed hand detection algorithm with existing techniques (*green*: skin filtering, *yellow*: two-frame differencing, *red*: proposed detector) (color figure online)

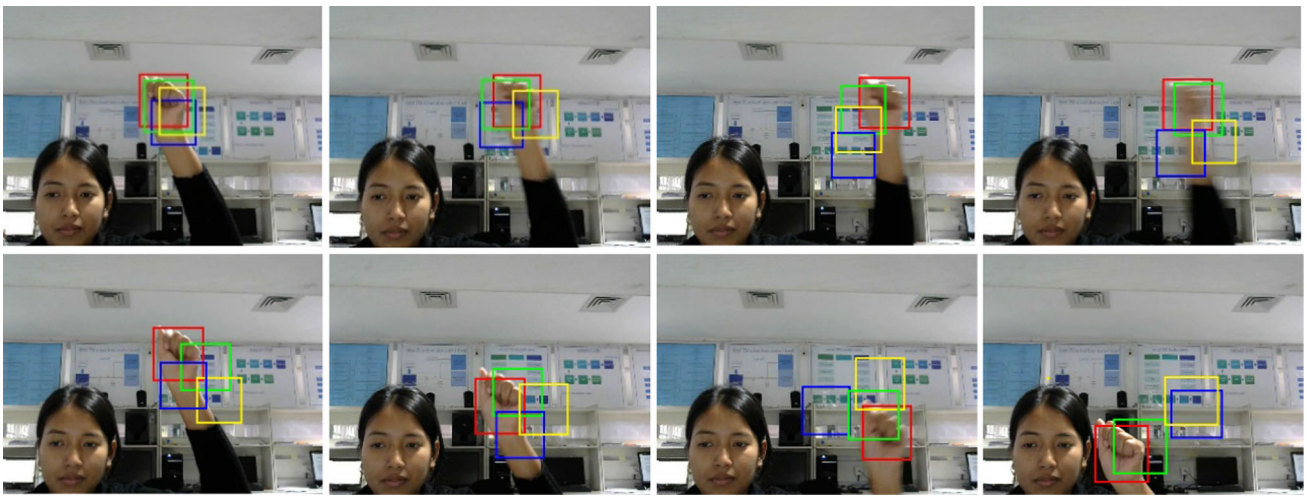


Fig. 6 Comparison of proposed tracking algorithm with existing techniques for gesture 'Five' (*green*: KLT, *blue*: CamShift, *yellow*: particle filter, *red*: proposed tracker) (color figure online)

Table 2 Total set of features used in the system

Feature set	Feature
f_1–f_{13}^*	Rubine's feature
f_{14}–f_{24}^*	E-Rubine features
f_{25}^*	Location
f_{26}–f_{29}^*	Orientation feature
f_{31}–f_{32}^*	Position
f_{33}–f_{36}^*	Self co-articulated features
f_{37}^*	Ratio feature
f_{40}^*	Length of major axis of the ellipse fitted
f_{41}^*	Position
f_{43}–f_{44}^*	Velocity profile
f_{30}	Velocity
f_{38}	Distance feature
f_{39}	Ellipse fitted orientation feature
f_{42}	Acceleration

* $p < 0.05$ indicates statistical significance

each features for checking the statistical significance of the features. A total of 40 features were found to have statistical significance which is denoted by star marked in Table 2. These 40 features are ranked in the decreasing '*F*' values as given in Table 3. Then, the performances of the combination of remaining 40 feature subsets were examined using IFS technique. The accuracy resulted in combining different feature set is shown in Fig. 7. It can be observed from the IFS curve that combination of 21, 18 and 26 features provides maximum accuracy for ANN, SVM and kNN, respectively.

3.3 Performance of the classifiers

The results of the ANN are presented in Table 4. The parameters of ANN such as hidden units and iterations were varied, and the corresponding train and test accuracies were calculated. The highest train and test accuracy were observed for network structure 40L-54N-40L (*L* and

Table 3 Features ranked in the decreasing order of '*F*' value

Rank	Feature set	Rank	Feature set	Rank	Feature set	Rank	Feature set
1	f_{34}	11	f_{31}	21	f_7	31	f_{19}
2	f_{35}	12	f_{41}	22	f_{13}	32	f_{22}
3	f_{36}	13	f_{32}	23	f_{17}	33	f_3
4	f_{33}	14	f_{27}	24	f_{14}	34	f_9
5	f_{40}	15	f_{28}	25	f_{23}	35	f_{11}
6	f_{20}	16	f_{29}	26	f_{24}	36	f_{10}
7	f_{25}	17	f_{43}	27	f_1	37	f_{16}
8	f_6	18	f_2	28	f_4	38	f_{18}
9	f_8	19	f_{21}	29	f_{12}	39	f_{44}
10	f_{26}	20	f_5	30	f_{15}	40	f_{37}

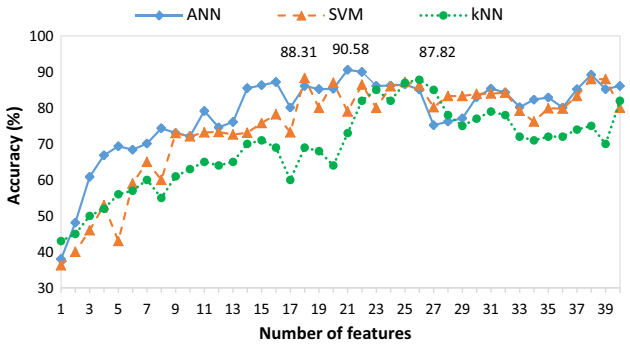


Fig. 7 IFS curve of the combination of features

Table 4 Results using ANN

Expt#	Hidden units	Network structure	Iterations	Train accuracy (%)	Test accuracy (%)
1	50	40–50–40	100	84	80
			500	86	80
2	52	40–52–40	100	86	80
			500	90	82
3	54	40–54–40	100	86	82
			500	90	86
4	56	40–56–40	100	90	84
			500	88	80
5	58	40–58–40	100	86	80
			500	90	80

N correspond to linear and nonlinear layers), iterations = 500. These network parameters were used for performing fivefold cross-validation. Cross-validation was performed to determine the validity of the proposed model. The results obtained during the cross-validation process are presented in Table 5.

The SVM classifier was tested for different kernel functions. The results of the evaluation are provided in Table 6. The highest test and train accuracies were observed for kernel function as radial basis function (rbf). The fivefold cross-validation process was performed using 'rbf' kernel function, and the results are provided in Table 7.

The system was also evaluated using kNN classifier using different values of k. The results of the evaluation are provided in Table 8. The highest train and test accuracy were obtained for $k = 5$. The fivefold cross-validation results are listed in Table 9.

Each classifiers with suitable parameters obtained from the experiments above (Tables 4, 5, 6, 7, 8 and 9) were used to generate a classifier fusion model. The fivefold cross-validation result obtained for this classifier fusion is provided in Table 10. Table 11 provides the summary of

Table 5 5-fold cross-validation result using ANN classifier

Expt#	Accuracy (%)
Subset 1	90.67
Subset 2	89.11
Subset 3	86.23
Subset 4	93.60
Subset 5	93.29
Overall accuracy	90.58

Table 6 Results using SVM

Expt#	Kernel function	Train accuracy (%)	Test accuracy (%)
1	Linear	86	82
2	Quadratic	86	80
3	Polynomial	88	84
4	Radial basis function	90	86

Table 7 Cross-validation results using SVM with 'rbf' kernel function

Expt#	Accuracy (%)
Subset 1	88.21
Subset 2	87.26
Subset 3	88.53
Subset 4	89.01
Subset 5	88.54
Overall accuracy	88.31

Table 8 Results using kNN

Expt#	k	Train accuracy (%)	Test accuracy (%)
1	3	76	70
2	5	84	80
3	7	78	72
4	9	80	74

Table 9 Cross-validation results using kNN with $k = 5$

Expt#	Accuracy (%)
Subset 1	88.92
Subset 2	85.00
Subset 3	86.20
Subset 4	90.31
Subset 5	88.67
Overall accuracy	87.82

Table 10 Cross-validation results using classifier fusion

Expt#	Accuracy (%)
Subset 1	94.27
Subset 2	92.00
Subset 3	92.30
Subset 4	91.10
Subset 5	91.48
Overall accuracy	92.23

Table 11 Comparison of success rate by different classifier using fivefold cross-validation

Classifier	Success rate (%)	Computational time (s)
ANN	90.58	0.560
SVM	88.31	0.420
kNN	87.82	0.496
Classifier fusion	92.23	1.570

Table 12 One-way analysis of variance test details

Source of variation	SS	*df*	MS	*F*	*P* value	*F*-crit
Columns	63.184	3	21.0615	5.26	0.0102	3.2389
Error	64.028	16	4.0017			
Total	127.212	19				

SS sum of squares, *df* degree of freedom, *MS* mean squared error, *F* *F*-statistic

the final cross-validation results of the different classifiers. It can be concluded that classifier fusion provides an improvement in terms of success rate of 1.65, 3.92 and 4.41 % as compared to baseline models ANN, SVM and kNN, respectively.

To test the statistical significance between the different individual classifiers, we performed one-way analysis of variance test [31], Friedman's test and Kruskal–Wallis test on the results obtained from the fivefold cross-validation process. The results of the one-way analysis of variance test are given in Table 12. The null hypothesis H_o for this experiment is that the mean accuracies of all the classifiers are same. It can be observed that $F > F_{crit}$ and $P \leq \alpha$ where $\alpha = 0.05$. We may conclude that there is a statistically significant difference between the accuracies of classifiers. Similarly, the results of Friedman's test and Kruskal–Wallis test are shown in Tables 13 and 14, respectively. In both the test, Chi - sq(our data) > Chi - sq(from table) and Prob (p) show that it is statistically significant for 0.1 significant level. Thus, we say that the mean and column effects of different classifiers are different and not all the classifiers come from the same distribution.

Table 13 Friedman's test details

Sources	SS	*df*	MS	Chi-sq	Prob > Chi-sq
Columns	16.6	3	5.533	9.96	0.0189
Error	8.4	12	0.70		
Total	25	19			

Chi-sq Chi square

Table 14 Kruskal–Wallis test details

Sources	SS	*df*	MS	Chi-sq	Prob > Chi-sq
Columns	368.6	3	122.867	10.53	0.0145
Error	296.4	16	18.525		
Total	665	19			

Post hoc analysis was performed using Tukey's HSD test for the three tests (one-way analysis of variance test, Friedman's test and Kruskal–Wallis test) to find out the most significantly different classifier. Figure 8 shows that the classifier fusion is significantly different from other classifiers.

3.4 Performance of the system with noisy data

A set of experiments were conducted using the noisy set of data. The gesture trajectories were made noisy by applying a Gaussian white noise with signal-to-noise ratio of 30 and 40. Few examples of the noisy data are shown in Fig. 9. For moderate noise level (SNR = 40), the average accuracy of the 40 gestures was 90.5 % which is comparatively similar to the results with 'no noise' (92.23 %), while for high noise (SNR = 30), the accuracy got degraded to 83.26 %. The accuracy for different sets of gestures for different SNR values is shown in Fig. 10. The low accuracy was observed due to large number of misclassifications in gesture 'One,' 'Two,' 'Seven,' '*S*,' '*Z*,' 'Divide.' This may be due to the simplicity in these gestures, while the accuracy of other gestures remains high as they have large number of variations when gesticulated by different users at different instant of time.

3.5 Comparative analysis

A comparison has been done between the performance of the system with proposed optimal feature set and the features available in the literature [19, 20, 23]. The comparison is shown graphically in Fig. 11. Figure 11a–f corresponds to the accuracy of different feature sets using different classifier models such as CRF [23, 32], HCRF [33], ANN [20, 34], SVM [35] and kNN [36], and classifier fusion for numerals, alphabets, arithmetic operators, self co-articulated, non-self co-articulated and total set of gestures, respectively. The

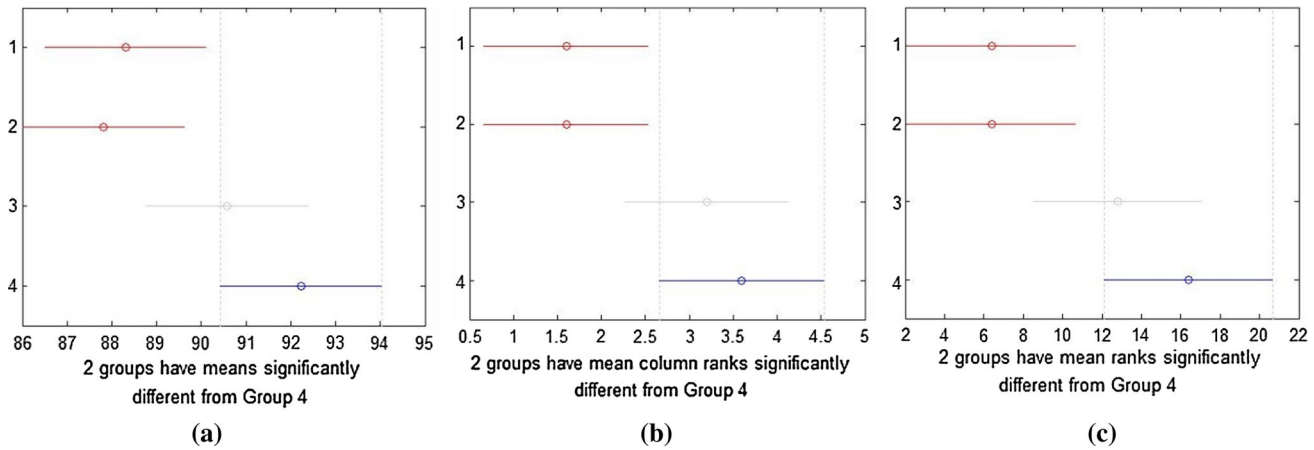


Fig. 8 Post hoc analysis using Tukey's HSD test for **a** one-way analysis of variance test, **b** Friedman's test and **c** Kruskal–Wallis test (1-SVM, 2-kNN, 3-ANN, 4-fusion)

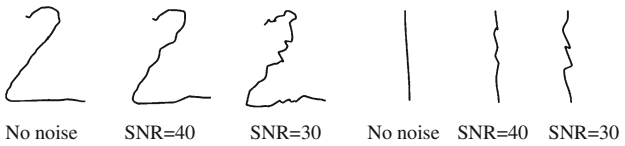


Fig. 9 Few samples of noisy data

performance of the proposed system with the optimal set of features has been observed to outperform the feature set in the other literature using classifier fusion for every set of gestures. Also it has been observed that the classifier fusion performs better than the individual classifiers for different feature sets experimented. Thus, we conclude from the comparative analysis that the combination of optimal features and classifier fusion yields the highest overall accuracy of 92.23 % which is shown in Fig. 11f.

4 Conclusion and future work

In this paper, we have developed a hand gesture recognition system where 40 set of gestures were considered. As such, database for 40 set of gestures is not available in the literature, and thus, we developed *'NITS hand gesture database IV.'* The proposed system can be used for developing a gesture-controlled hexadecimal keyboard making human–computer interaction easier. A total of 44 features were selected from the existing literature. ANOVA test was performed in order to check the statistical significance of the 44 features. The 40 significant features were then arranged in the decreasing order of the F-static value which was then fed to the IFS to select the optimal features. The total number of features or the optimal features was observed to be 21, 18 and 26 for ANN, SVM and kNN, respectively. The results of the three individual classifiers were combined to provide the classifier fusion results. After this, fivefold cross-validation was used to provide the overall accuracy of the system. The overall accuracy was observed to be 90.58, 88.31, 87.82 and 92.23 % using the ANN, SVM, kNN and classifier fusion, respectively. To test the statistical significance between the different individual classifiers, we performed one-way analysis of variance test, Friedman's test and Kruskal–Wallis test on the results obtained from the fivefold cross-validation process. It was observed from these tests that the classifier fusion is significantly different from other classifiers.

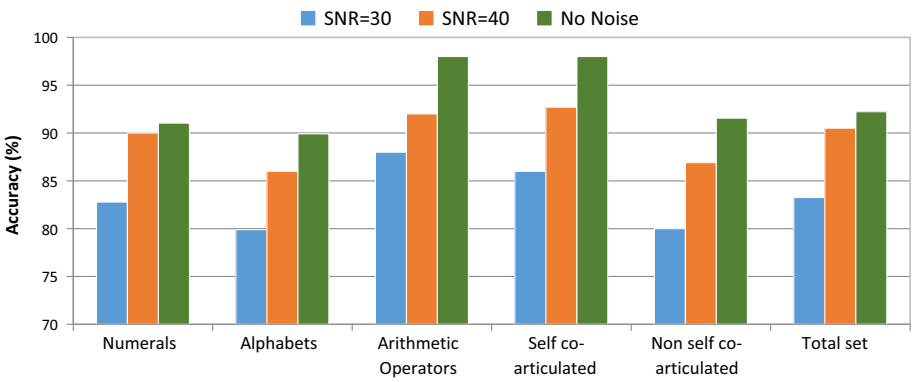


Fig. 10 Performance of the system for various gesture sets with varying SNR (30, 40, no noise)

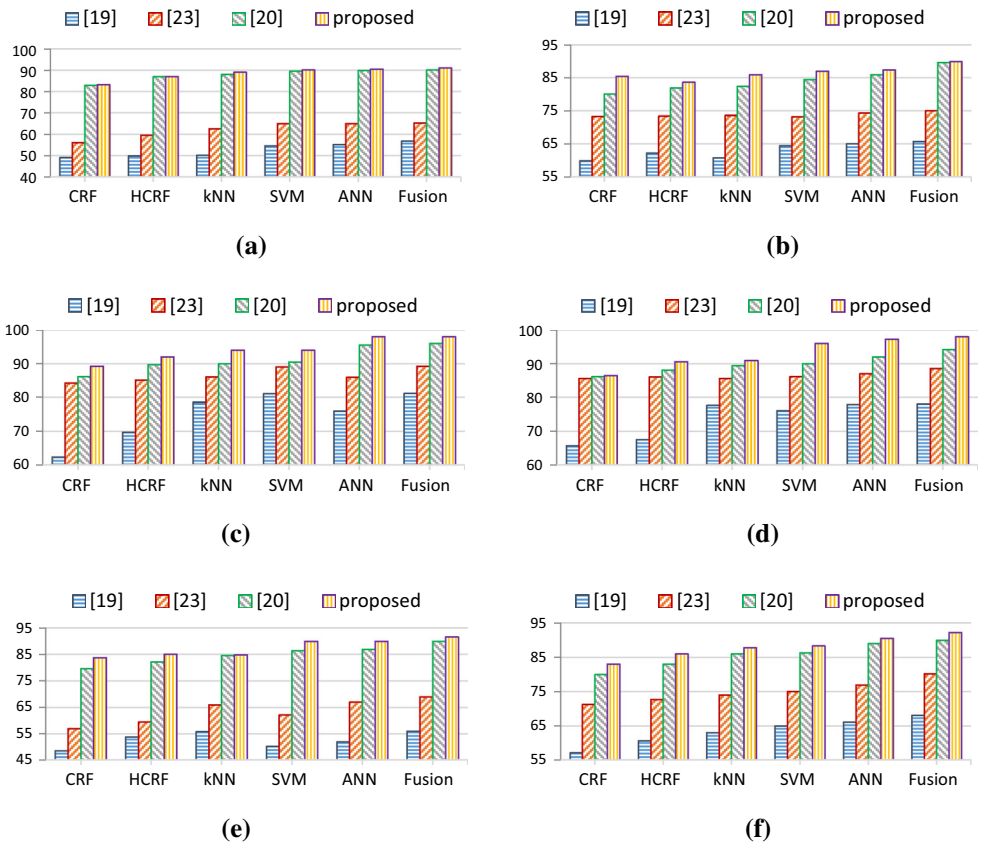


Fig. 11 Comparison of proposed features with existing features [19, 20, 23] using CRF, HCRF, ANN, SVM, kNN and classifier fusion models for gesture sets: **a** numerals, **b** alphabets, **c** arithmetic operators, **d** self co-articulated gestures, **e** non-self co-articulated gestures and **f** total set of gestures

Moreover, our system was tested for noisy set of data where Gaussian noise with SNR = 30 and SNR = 40 was added to the gesture trajectories. We can observe from the results that the low noise does not affect the system performance largely, but when the system was subjected to high noise (SNR = 30), the performance degraded from 92.23 to 83.26 %. Similarly, comparison of the proposed model was performed with the existing features in the literature. It was observed that our system provided performance better than the existing ones. However, the system was proposed for isolated gestures. This can be extended for continuous set of gestures in future. Moreover, new set of features may be added as a future work to the features used in this paper to enhance the performance of the system.

Acknowledgments The authors acknowledge the *Speech and Image Processing Lab* under Department of ECE at National Institute of Technology Silchar, India, for providing all necessary facilities to carry out the research work.

References

1. Hasan H, Abdul-Kareem S (2014) Human–computer interaction using vision-based hand gesture recognition systems: a survey. Neural Comput Appl 25(2):251–261
2. Singha J, Das K (2013) Indian sign language recognition using eigen value weighted Euclidean distance based classification technique. Int J Adv Comput Sci Appl 4(2):188–195
3. Singha J, Das K (2013) Recognition of Indian sign language in live video. Int J Comput Appl 70(19):17–22
4. Badi HS, Hussein S (2014) Hand posture and gesture recognition technology. Neural Comput Appl 25(3–4):871–878
5. Badi H, HasanHussein S, Kareem SA (2014) Feature extraction and ML techniques for static gesture recognition. Neural Comput Appl 25(3–4):733–741
6. El-Baz AH, Tolba AS (2013) An efficient algorithm for 3D hand gesture recognition using combined neural classifiers. Neural Comput Appl 22(7–8):1477–1484
7. Comaniciu D, Ramesh V, Meer P (2003) Kernel-based object tracking. IEEE Trans Pattern Anal Mach Intell 25(5):234–240
8. Chai D, Ngan KN (1999) Face segmentation using skin-color map in videophone applications. IEEE Trans Circuits Syst Video Technol 9:551–564
9. Wang H, Chang S-F (1997) A highly efficient system for automatic face region detection in MPEG video. IEEE Trans Circuits Syst Video Technol 7:615–628
10. Guo JM, Liu YF, Chang CH (2012) Improved hand tracking system. IEEE Trans Circuits Syst Video Technol 22:5
11. Bradski GR (1998) Computer vision face tracking as a component of a perceptual user interface. In: The workshop on applications of computer vision, Princeton, NJ, pp 214–219
12. Shi J, Tomasi C (1994) Good features to track. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 593–600
13. Asaari MSM, Rosdi BA, Suandi SA (2014) Adaptive Kalman filter incorporated eigenhand (AKFIE) for real-time hand tracking system. Multimed Tools Appl 70(3):1869–1898

14. Kolsch M, Turk M (2004) Fast 2D hand tracking with flocks of features and multi-cue integration. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshop, pp 158
15. Yao Y, Fu Y (2014) Contour model-based hand-gesture recognition using the Kinect sensor. Circuits Syst Video Technol IEEE Trans 24(11):1935–1944
16. Viola P, Jones M (2001) Rapid object detection using a boosted cascade of simple features. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 511–518
17. Viola P, Jones MJ (2004) Robust real-time face detection. Int J Comput Vis 57(2):137–154
18. Geetha M, Menon R, Jayan S, James R, Janardhan GVV (2011) Gesture recognition for American Sign Language with polygon approximation, IEEE international conference on technology for education, Tamil Nadu, India, 4–16 July, pp 241–245
19. Bhuyan MK, Ghosh D, Bora PK (2006) Feature extraction from 2D gesture trajectory in dynamic hand gesture recognition. In: Proceedings of the IEEE conference on cybernetics and intelligent systems, pp 1–6
20. Singha J, Laskar RH (2016) Self co-articulation detection and trajectory guided recognition for dynamic hand gestures. IET Comput Vis 10(2):143–152
21. Singha J, Laskar RH (2015) ANN-based hand gesture recognition using self co-articulated set of features. IETE J Res 61(6):597–608
22. Kao CY, Fahn CS (2011) A human-machine interaction technique: hand gesture recognition based on hidden Markov models with trajectory of hand motion. Proc Eng 15:3739–3743
23. Bhuyan MK, Kumar DA, MacDorman KF, Iwahori Y (2014) A novel set of features for continuous hand gesture recognition. J Multimodal User Interfaces 8(4):333–343
24. Signer B, Norrie MC, Kurmann U, Gesture I (2007) A Java framework for the development and deployment of stroke-based online gesture recognition algorithms, Technical report TR561, ETH Zurich
25. Rubine B (1991) Specifying gestures by example. In: Proceedings of ACM SIGGRAPH'93, 18th international conference on computer graphics and interactive techniques, USA, pp 329–337
26. Xu D, Wu X, Chen YL, Xu Y (2014) Online dynamic gesture recognition for human robot interaction. J Intell Rob Syst 77(3–4):583–596
27. Lin J, Ding Y (2013) A temporal hand gesture recognition system based on hog and motion trajectory. Opt Int J Light Electron Opt 124(24):6795–6798
28. Sharkey AJC (1999) Combining artificial neural nets: ensemble and modular multi-net systems. Springer, London
29. Nadgeri SM, Sawarkar SD, Gawande AD (2010) Hand gesture recognition using Camshift algorithm. In: Proceedings of the third ieee international conference on emerging trends in engineering and technology, Goa, pp 37–41
30. Shan C, Tan T, Wei Y (2007) Real-time hand tracking using a mean shift embedded particle filter. Pattern Recogn 40(7):1958–1970
31. Semwal VB, Mondal K, Nandi GC (2015) Robust and accurate feature selection for humanoid push recovery and classification: deep learning approach. Neural Comput Appl. doi:10.1007/s00521-015-2089-3
32. Yang HD, Sclaroff S, Lee SW (2009) Sign language spotting with a threshold model based on conditional random fields. IEEE Trans Pattern Anal Mach Intell 31(7):1264–1277
33. Quattoni A, Wang S, Morency LP, Collins M, Darrell T (2007) Hidden conditional random fields. IEEE Trans Pattern Anal Mach Intell 29(10):1848–1852
34. Bouchrika T, Zaied M, Jemai O, Amar CB (2014) Neural solutions to interact with computers by hand gesture recognition. Multimed Tools Appl 72(3):2949–2975
35. Dardas NH, Georganas ND (2011) Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. IEEE Trans Instrum Meas 60(11):3592–3607
36. Dasarathy BV (1990) Nearest neighbor (NN) norms: NN pattern classification techniques. IEEE Computer Society Press, Los Alamitos, CA