Bharatanatyam Hand Gesture Recognition Using Normalized Chain Codes and Oriented Distances

R. Amrutha
CSE Department
PESIT, Bangalore South Campus
Bangalore, India
ramrutha2504@gmail.com

Vandana M Ladwani
CSE Department
PESIT, Bangalore South Campus
Bangalore, India
vandanaladwani@gmail.com

Abstract*—We propose a system to identify Bharatanatyam hand gestures or Mudras. This involves a preprocessing stage which does a skin based segmentation to obtain the hand boundary. The feature extraction stage involves obtaining the chaincode of the entire contour of the hand followed by normalization. It also includes extracting the Euclidean distance from the centroid to the outermost boundary of the hand along 360 degrees. Extracted features from the training images are used to build four recognition models Naïve Bayes,KNN, Logistic Regression, Multiclass SVM . The system shows an accuracy of 88.47%, 87.06%, 89.83%, 92.3% using Naïve Bayes, KNN, Logistic Regression, Multiclass SVM respectively. Multiclass SVM classifier shows the best performance. This system provides an interface for newbies to learn Bharatanatyam Dance form.

Keywords*—chaincode, edge detection, floodfill, normalization, Euclidean distance,boundary distance.

I. INTRODUCTION

Hand gesture recognition is a very popular application domain where the system in meant to identify the signs that are expressive of some meaningful context or idea. Gestures is one of the most powerful ways to communicate with system which helps in building HCI systems. Gesture recognition is also helpful for physically disabled persons to communicate with the system. Dance forms a sub domain of gesture recognition. Few of the famous Indian dance forms are Bharatanatyam, Kathakali, Mohiniattam, Odissi, Kuchipudi. Bharatanatyam is a dance form which is more prevalent in the Southern regions of India among the various other dance forms present. It is a composition of tala (rhythm), raga (music), natya (dance) and bhava (expression). This dance form is an art of expression also known as Abhinayam. One of the prominent aspects of Abhinayum is physical or body movements (Anghika). Typically a Bharatanatyam dance is suggestive of a story or extract from a well known epic. The hand gestures (Mudras) is an important facet of the Anghika. Bharatanatyam highlights the use of hand gestures (Mudras) to project different emotions the dance requires. Every mudra contributes a specific meaning to the dance. Every dancer engages an extensive collection of mudras and movements to portray the expression of the dance form. Mudras are of two kinds : Asamyukta mudra (single-hand gestures) and Samyukta mudra(double hand gestures). Mudra identification along with the other modules like face expression recognition, posture recognition can help in building an online learning system for the Indian dance form and can provide a global exposure to the Indian classical Dance

The paper focuses on Asamyukta mudra identification system. Mudra recognition is more subtle compared to hand gestures and hence need an almost precise algorithm which can differentiate between the different Asamyukta mudras .In the proposed system we have considered 6 mudras named Alapadma, Araala, Chandrakala, Pataka, Tripataka and Kartarimukha. The Mudras are depicted in t Fig 1.

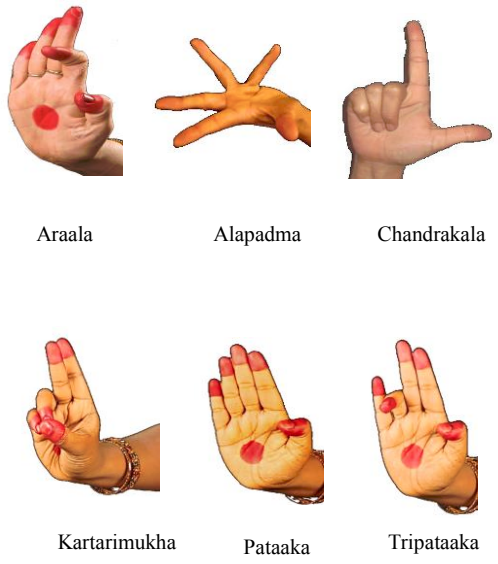

Fig. 1. Bharatanatyam Mudras

In this paper we suggest a segmentation technique which accommodates for most discrepancies that may exist while processing an image such as shadow formation, filling holes, etc. Segmentation involves skin detection in HSV color space. In the skin detection module the red coloured pixels get erased

which results in inaccurate segmentation[6]. To account this we check for the pixel values with a range of red color and add those pixels to the segmented image provided it forms a connected component. We use flood fill algorithm for filling holes that might exist. Feature extraction phase involves extracting the centroid values, chaincode of the outline and the Euclidean distance from the centroid to the outline along 0 to 360 degrees orientations. The classification phase involves a comparative study of 4 classifiers.

Thus the dataset collected has a diversity of size and individuality .we have collected data from the 10 subjects. The images in the dataset are size normalized before processing. The proposed method incorporates translation and rotation invariance. Proposed work aims at promoting and imparting training in Indian classical dance around the world. The proposed system is implemented using OpenCV 3.0.0 with python 2.7. The remaining paper is organized as follows section II deals with Literature survey, Section III deals with proposed approach followed by results and conclusion

II. LITERATURE REVIEW

Hand Gesture Recognition techniques are basically divided into Vision based and Sensor based techniques. Sensor based recognition collects the gesture data by using one or more different types of sensors. These sensors are attached to hand which records the position of the hand and then collected data is analysed for gesture recognition. An example of sensor based approach is Data glove. Sensor based approach requires expensive hardware to capture the data and it alters the natural movement of the hand.

Vision based techniques uses one or more cameras to capture the hand images. Vision based techniques use various image processing techniques to get the hand posture. Vision based techniques extract features form the region of the interest and use these features for classification using the classification model. Various features proposed in the literature for hand gesture recognition include HOG[7],Fourier Descriptors[8], Silhouette and texture contour etc.

The previous work done in this field involves coordinating the body(limb) movements to the music beat and building a computational model of these movements. Sangeeta Jadhav and Sasikumar Mukundan proposed a system that gives a specific two-charactered notation of each limb position. Hence they use a vector of these notations to define a dancer's stance at a point in time. This forms the basis of their training phase. So on the input of a beat, several combination of these vectors are chosen to form the choreography for the beats[1].

Sriparna Saha and Ramadoss Janarthanan proposed "Fuzzy L Membership Function Based Hand Gesture Recognition for Bharatanatyam Dance" which is a 3 stage system. The first stage involves extraction of boundary of the hand using a texture based segmentation and Sobel edge detection. The second stage involves finding the centre point of the outlined region of interest. It includes calculating the Euclidean distances along 8 degrees (0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°). The third stage involved matching the unknown hand gesture using Fuzzy L membership values which the membership values of the distances obtained in the previous stage. System showed an accuracy of 85.1%.[2]

Dr. Srimani and Kavitha S used a set of 3 modified images of one of the 24 grayscale images of the mudra as training .These set of 3 include the image itself, a rotated version of the image and a scaled version of that image. In the first phase of feature extraction, they have used orientation filter to obtain a feature vector. At the second phase, an outline of the double-hand gestures are obtained followed by generation of the skeletonized variant and evaluation of gradients at the corner points[3]

Sriparna Saha and Jayashree Roy proposed sensor based gesture recognition. They used Kinect sensor to obtain the joint coordinates and then used the variability of the movement to recognise if it is a single or double hand gesture. A normalization of the coordinate is done based on the hip centre coordinate of the first frame. A trajectory is then obtained which is classified as a particular known hand gesture using linear support vector machine as the classifier. They achieved an accuracy of 94.3%[4]

Sriparna Saha, Amit Konar, et al proposed "Bharatanatyam Hand gesture recognition using Polygon representation. The system approximates the extracted boundaries and compares the slopes of the straight lines to the slopes of the sides of the decagon and uses this to determine the chaincode. Classification is done by comparing the chain code of the test image with the saved templates. Accuracy of 89.3%was achieved by their approach.[5]

Mandeep kaur et al used skin color segmentation in YCbCr space. Template based matching technique using Principal Component Analysis was used for the recognition. System was tested for recognizing 4 different gestures. They reported 91.25% accuracy using their system[9]

III. PROPOSED APPROACH

Proposed Approach consists of a skin based segmentation, followed by feature extraction. The features considered are centroid, chain code , Oriented distances. Extracted Features are used to build the different classifiers and performance of different classifiers is compared.

The Various steps involved are as follows

i. Preprocessing
ii. Segmentation
iii. Feature Extraction
    a. Determining Centroid
    b. Chain code estimation
    c. Boundary Distance Measurement
iv. Training
v. Testing

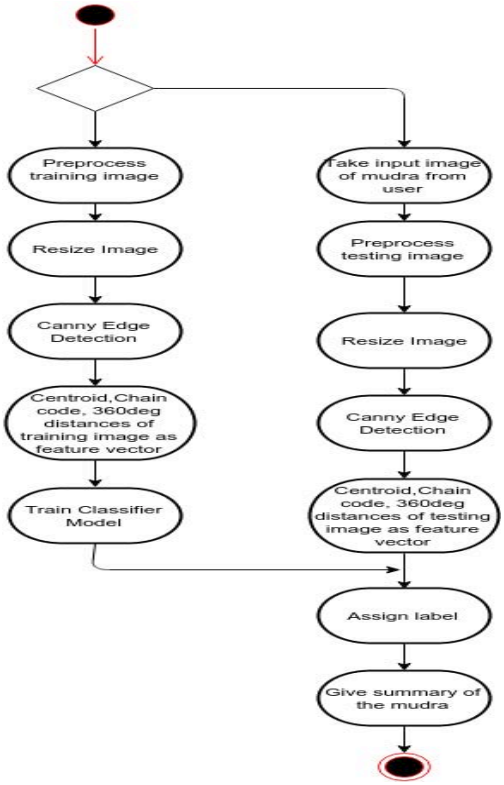

Fig. 2. Bharatanatyam Mudra Identification System

A. *Preprocessing*

Hand size varies from one image to other which may result in an error in the performance of the system. In order to overcome this error all the images are size normalized fine.

B. *Segmentation*

In this section, segmentation is done based on skin color. The isolation of the foreground from the background is done by specifying the skin tone range in HSV color space. The methodology also accounts for red color applied on the fingertips of the dancers. Each of the range boundary values are specified in numpy arrays. The pixel colors are compared with the numpy ranges and decided on whether it is the foreground or not. Sometimes, there may exist shadows or accessories(rings etc) worn by the dancers which will not fall within the ranges specified. This leads to gaps (holes) in the foreground which may contribute to the classification error. We use the flood fill algorithm to fill these gaps created. The algorithm flood fills from pixel co-ordinate (0,0) such that only the background is filled. This image is inverted and is bitwise-ORed with the original binary image to extract the hand portion. To remove any more noise that may be present, we apply morphological filters which involves opening followed by closing. Opening removes any stray white pixels that might be present and closing fills any black points on the foreground object. We account for diversity in our datasets based on how small or big the foreground object is in the original image by focussing only on the foreground. This can be done by drawing a bounding box around the foreground object and obtaining the region of interest(ROI). So that the images are comparable, we resize the image by standardizing the height of the ROI image to 200 and scaling the width according to the aspect ratio. This is done so that the features of the hand are not lost and do not appear distorted. This is referred as Image1

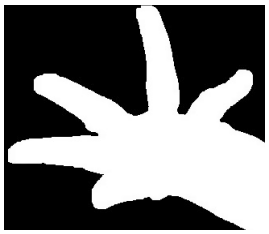

Fig 3. Image1

Canny Edge Detection algorithm is used to determine the boundary. It follows a 4 step process which involves noise reduction using a 5X5 Gaussian Filter followed by finding intensity gradient which rounds off the direction to one of (0,45,90,135) degrees. Non-maximum suppression involves removing all those points which are not a part of the edge. Hysteresis uses thresholding to check if the pixel value can be considered as an edge or not.(Can we include diagram for canny edge detector). This is referred as Image2

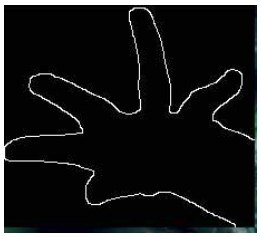

Fig 3. Image2

We observed that on applying the above algorithm, noise exists as thicker lines in some parts of the edge. To remove this and smoothen edges, we use erosion which is a morphological filter to shrink the edge, of kernel size 1X1.and size varies from one image to other which may result in an error in the performance of the system. In order to overcome this error all the images are size normalized fine.

C. *Feature Extraction*

The proposed system uses centroid, chain code and oriented boundary distances to build the feature vector.

i. Centroid

Using image1 we calculate the center of the ROI along both the axes. This is done by checking if the pixel value is white and aggregating the white pixel's x-coordinate and y-coordinate values. Also a count of the number of white pixel is maintained. Dividing the respective sums by the number of white pixels give us the mean x and y co-ordinates respectively

ii. Chain codes

Using image2 we use a connectivity graph to decide the direction of the traversal.

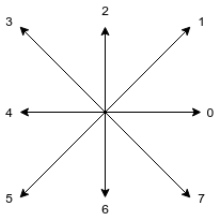

Fig. 3. Eight Connectivity Graph

To start traversal, we walk from the centroid at 0 degree to find the outermost edge pixel. This is the starting point of the traversal. We move downwards and based on the direction of the walk ,which specifies the relationship between the previous pixel and the current pixel, we map it to a direction value on the connectivity graph. This mapping is done throughout the walk around the boundary of the foreground object and recorded.

We observed that there was a difference in the number of pixels traversed in most images due to the orientation of the hand. To solve this, we standardized the size of the chaincode obtained to 80 directional values. The entire chaincode was divided into 20 blocks. Each of this block was further subdivided into 4 sub-blocks. The most frequently occurring direction value is recorded. This process is carried out for each of the 20 blocks thus finally obtaining 80 values in the end of this stage.

iii. Oriented Boundary distances

We first consider an angle matrix of the size of the image. This is obtained by performing an elementwise division of 2 matrices, each of the size of the image matrix. One matrix contains a row replication of the x-coordinate system. The other matrix contains a column replication of the y-coordinate system. After performing the division, we apply the tan inverse function on the entire matrix and convert each element in this resulting matrix to degree values.

This resultant matrix is multiplied elementwise with image2 binary matrix to obtain a matrix which has an angle values only along the edges of the foreground object. Keeping the centroid as a reference, we traverse along the edge to obtain Euclidean distances corresponding to each angle in the matrix obtained previously. In case some angles might be missing, we perform linear interpolation on those missing values. Thus for all 0 to 360 degrees we obtain the straight line distance from the centroid to the boundary of the foreground object. Thus the feature vector consists of 2 centroid values,20 standardized values for chaincode,360 values for the Euclidean Distances

$$Distance = \sqrt{(x_c - x_1)^2 + (y_c - y_1)^2}$$

D. Classification

We performed classification using 4 classifiers: Naive Bayes, KNN(for K=1 and K=5), Logistic Regression and SVM. Our dataset consists of 300 images (50 images per mudra). 80% of the images are used to train the system and remaining 20% to test the system. Thus the new incoming image is classified into one of the 6 mudras that we have used

i. Naive-Bayes

Naïve Bayes is a generative model .It builds the model with the assumption that feature vectors are iids. This classifier is trained using the maximum likelihood algorithm. For classifying the test image, features are extracted from the image and posterior probability is determined for each of the k classes and Label is predicted as the class having showing the maximum probability

The Bayes equation is given as

$$p(C_k|x) = \frac{p(C_k)p(x|C_k)}{p(x)}$$

The likelihood term can be split to account for the independency of the features. The prior term specifies the probability distribution before to the evidence taken.

ii. K-NN classifier

This classifier maps the feature vectors onto the feature space. The incoming sample is considered in term of its closeness to its 'k' neighbours for classification. The closeness factor uses distance metrics such as Euclidean or Hamming distances to approximate the class label for the unclassified sample. A good value for k reduces any noise if present and gives better approximation of the class the sample belongs to. The classifier does not use any information of the feature values until classification an hence is an instance-based learner

iii. Logistic Regression

This classifier uses a cumulative logistic curve to predict the value of the dependent variable (which is the outcome of the classification) and weigh the odds that it belongs to a particular class (categorical variable). The logarithm of ratio of the odds are plotted to give a continuous representation of the dependent variable.

iv. Multi-class SVM

This classifier uses binary SVMs where the probability of prediction to one class is compared with respect to all the other classes. Each binary SVM returns a confidence value that the sample belongs to that class. For M classes a total of {M(M-1)/2} binary classifiers are used.

$$g^j(x) = \sum_{i=1}^{m} y_i \alpha_i^j \, k(x, x_i) + b^j$$

IV. RESULTS

Once the features are determined and classification model is built using the training images the system can be tested to classify the test images. The proposed system two modes are included .Online and offline. For the online mode the test image is take at the run time and the label is predicted for the image by selecting any of the four classifiers. For the offline mode 20% of the total images(300)are used to test the system using each of the classifier system shows an accuracy of88.47%, 87.06%, 89.83%, 92.3% using Naïve Bayes,KNN, Logistic Regression, Multiclass SVM classifiers respectively. Classification error for mudra Alapadma is the maximum among all.

TABLE I. CONFUSION MATRIX FOR NAÏVE BAYES

	Class1	*Class2*	*Class3*	*Class4*	*Class5*	*Class6*
Class1	14	0	0	0	0	1
Class2	0	7	0	1	0	0
Class3	0	0	6	0	0	0
Class4	0	1	0	11	0	1
Class5	0	0	0	1	8	0
Class6	0	1	0	0	0	7

TABLE II. CONFUSION MATRIX FOR 1NN

	Class1	*Class2*	*Class3*	*Class4*	*Class5*	*Class6*
Class1	13	0	1	1	0	0
Class2	0	8	0	0	0	0
Class3	0	0	6	0	0	0
Class4	0	0	0	12	0	1
Class5	0	0	0	0	9	0
Class6	0	1	0	0	0	7

TABLE III. CONFUSION MATRIX FOR 5NN

	Class1	*Class2*	*Class3*	*Class4*	*Class5*	*Class6*
Class1	11	1	0	2	0	1
Class2	0	8	0	0	0	0
Class3	0	0	6	0	0	0
Class4	0	1	0	11	1	0
Class5	0	0	0	1	9	0
Class6	0	1	0	1	1	6

TABLE IV. CONFUSION MATRIX FOR LOGISTIC REGRESSION

	Class1	*Class2*	*Class3*	*Class4*	*Class5*	*Class6*
Class1	12	0	1	2	0	1
Class2	0	8	0	1	0	0
Class3	0	0	6	0	0	0
Class4	0	0	0	12	0	1
Class5	0	0	0	0	9	0
Class6	0	1	0	0	1	6

TABLE V. CONFUSION MATRIX FOR MULTICLASS SVM

	Class1	*Class2*	*Class3*	*Class4*	*Class5*	*Class6*
Class1	12	0	1	1	1	0
Class2	0	8	0	0	0	0
Class3	0	0	6	0	0	0
Class4	0	0	0	12	0	1
Class5	0	0	0	0	9	0
Class6	0	1	0	0	1	6

TABLE VI. RESULTS WITH DIFFERENT CLASSIFIERS

Classifier / Mudra	Naive Bayes	1-nn	5-nn	Logistic Regression	Multi Class SVM
Mudra 1	90.33	80	73.33	80	86
Mudra 2	87.5	100	100	100	100
Mudra 3	100	100	100	100	100
Mudra 4	84	92	84.6	92.3	92.3
Mudra 5	88	100	100	100	100
Mudra 6	87	75	75	75	87

V. CONCLUSION AND FUTURE SCOPE

The proposed system can be integrated with the facial expression recognition module and posture recognition module to build a complete system for giving online training of the dance form by building an interactive system which take from the video of the dance every frame and provides complete details regarding the mudra, posture and expression and can give a gist of the drama portrayed in the sequence of the frames. The system is faster as compared to the existing system as. The future work can be extending the system for recognizing other mudras and implementing the facial expression recognition and posture recognition and coupling the modules together. Also the performance of the system can be improved by using enhanced noise removal technique. Deep Learning Methods can also be explored for the same problem but it requires huge amount of data to get the better models using Deep Learning and prevent the overfitting. So the future work will be to prepare the large dataset from the dance videos and built the real time system using the deep learning methods

ACKNOWLEDGMENT

We acknowledge Nandan Kumar, Sanjana and Pallavi students of PESIT for their support in preparing the dataset.

REFERENCES

[1] Sangeeta Jadav, and Sasikumar Mukundan, “A Computational Model for Bharata Natyam Choreography,” (IJCSIS) International Journal of Computer Science and Information Security, Vol. 8, No. 7, October, 2010.

[2] Sriparna Saha, Lidia Ghosh, Amit Konar and Ramadoss Janarthanan, “Fuzzy L Membership Function Based Hand Gesture Recognition for Bharatanatyam Dance,” 5th International Conference on Computational Intelligence and Communication Networks (CICN), September,2014.

[3] Dr.Srimani.P.K, and Kavitha.S, “Recognizing Samyuktha Hand Gestures of Bharatanatyam Using Skeleton Matching and Gradient Orientation,” International Journal of Current Research, Vol. 5, Issue, 6, pp.1457-1462, June, 2013

[4] Sriparna Saha, Amit Konar and Jayashree Roy, “Single Person Hand Gesture Recognition Using Support Vector Machines,” Computational Advancement in Communication Circuits and Systems, Springer, Volume: 335, November,2014

[5] Sriparna Saha, Amit Konar, Deblina Gupta, Anasuya Ray, Abhinaba Sarkar, Pritha Chatterjee, and Ramadoss Janarthanan, “Bharatanatyam Hand Gesture Recognition Using Polygon Representation,” International Conference on Control, Instrumentation, Energy & Communication(CIEC14),2014 .

[6] S. Mitra and T. Acharya, “Gesture recognition: A survey,” Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, vol. 37, no. 3, pp. 311–324, 2007.

[7] Jing Lin, Yingchun Ding, "A temporal hand gesture recognition system based on hog and motion trajectory", Optik i2 4 pp.6795- 6798, 2013.

[8] StergiosPoularakis and Ioannis Katsavounidis, " Finger Detection And Hand Posture Recognition Based On DepthInformation", IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP). pp. 4329-4333, 2014.

[9] Mandeep Kaur Ahuja and Amardeep Singh, “Static Vision Based Hand Gesture Recognition Using Principal Component Analysis”, IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE), 2015.

[10] M Z. Ren, J. Yuan, J. Meng, and Z. Zhang, “Robust part-based hand gesture recognition using kinect sensor,” *IEEE TMM*, 2013

[11] M A. Chaudhary, J. Raheja, K. Das, and S. Raheja, “Intelligent approaches to interact with machines using hand gesture recognition in natural way: a survey,” *arXiv preprint arXiv:1303.2292*, 2013