Bharatnatyam Hand Gesture Recognition using Contour Detection

Joshi Akanksha Ajay
PICT, Sahakarnagar-2,
Pune-411009, India

Mhasawade Vishwali Vinay
PICT, Sahakarnagar-2,
Pune-411009, India

ABSTRACT
For detecting the hand gestures or *mudras* used in Bharatnatyam a system has been accomplished. The input consists of an image of a hand gesture out of the 28 *asamyuktamudras.* The input image is processed and then compared with the various hand gesture images, and the system determines which hand gesture the input image resembles to. It then outputs the name of the input image. This system provides a way to determine the asamyukta mudra that an image comprises of.

Keywords
color space , threshold , Binarization , contour detection

1. INTRODUCTION
Bharatnatyam is a form of classical dance that originated in the temples of Tamil Nadu. Lord Shiva in his Natraja form is considered the God of this dance [1]. In Bharata Natyam, approximately fifty-five root mudras (hand/finger gestures) are used to clearly communicate specific ideas, events, actions, or creatures. Out of these 28 require only one hand, and are classified as `Asamyukta Hasta', along with twenty-three other primary mudras which require both hands and are classified as 'Samyukta Hasta'. The proposed system uses the 'Asamyukta Hastas' as its input. In this a method to process these images and develop a self-learning system that learns the nature of these images is proposed. The 28 mudras are as shown in Figure 1:

Figure 1

The system consists of four stages.First stage includes training the system to learn to interpret all the 28 'Asamyukta mudras' by studying their contours and remembering their names. Second stage involves capturing a hand gesture and converting the captured hand gesture image into a binary image. Third stage involves finding the contours of the binary image. Fourth stage compares the contour values with the trained image's contour values and determines to which trained image it resembles. The stages can be represented in Figure 2:

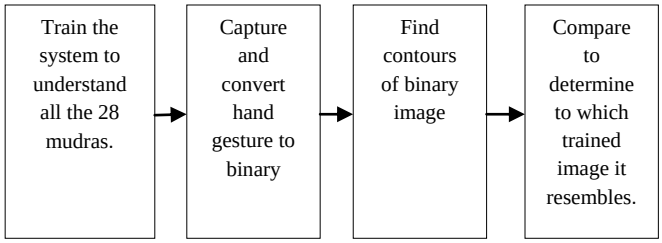


Figure 2

2. PROPOSED SYSEM

2.1 Training the system
The first stage of the training consists of capturing the image of the hand gesture. The extracted image is in RGB form. It uses the RGB colour space [2]. This image has three channels: red, green and blue [3]. RGB channels roughly follow the colour receptors in the human eye. The RGB image is converted into a gray scale image. A gray scale image is composed of different shades of grey colour. A true colour image can be converted to a gray scale image by preserving the luminance (brightness) of the image. The RGB image is a three dimensional image whereas the gray scale image is a two dimensional image. [4] The gray scale image is obtained from the RGB image by using the following equation:-

$$Y \Leftarrow 0.299R + 0.587G + 0.114B$$

Where Y is the gray scale weighted average and R, G, B are integers between 0 and 255. [9]

The RGB colours are not weighted equally. Since pure green is lighter than pure blue and pure red, it has the highest weight [0.587]. Pure blue is the darkest of the three, so it has the least weight [0.114].

The gray scale image is then converted into a binary image. A binary image is a digital image that has only two possible values. In a gray scale image, each pixel in the image is assigned a value corresponding to its intensity. Thus, in a gray scale image each pixel has an intensity value from 0-255, where 0 = black, 255 = white and in between are the shades of gray. A binary image is produced from the gray scale image by taking each pixel's gray-scale value and reducing the 0-255 range down to simple on/off (0 or1) value where 0 = black and 1= white

The system uses*Otsu's* Binarization [6] for obtaining the required binary image. Otsu's Binarization works for bimodal images(In simple words, bimodal image is an image whose histogram has two peaks). So in simple words, this algorithmautomatically calculates a threshold [7] value from image histogram for a bimodal image.

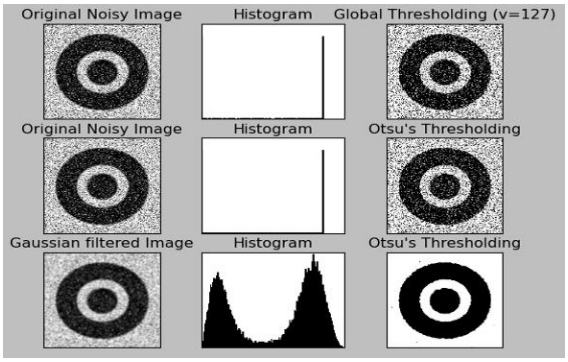


Figure 3

Otsu's algorithm tries to find a threshold value (t) .

1. Compute histogram and probabilities of each level
2. Set up initial $\omega_i(0)$ and $\mu_i(0)$
3. Step through all possible thresholds $t = 1 \ldots$ maximum intensity
    1. Update ω_i and μ_i
    2. Compute $\sigma_b^2(t)$
4. Desired threshold corresponds to the maximum $\sigma_b^2(t)$
5. You can compute two maxima (and two corresponding thresholds). $\sigma_{b1}^2(t)$ is the greater max and $\sigma_{b2}^2(t)$ is the greater or equal maximum
6. Desired threshold = $\dfrac{\text{threshold}_1 + \text{threshold}_2}{2}$

[8]
It actually finds a value of *t* which lies in between two peaks such that variances to both classes are minimum. After obtaining the binary image the system determines the contours of the binarized image. Contours can be defined as a curve joining all the continuous points along the boundary of an image having same colour or intensity value.[5] After finding the contours, the (x,y) co-ordinates of the contributing pixels of the contour were stored in an array. Some methods are applied to remove the redundant points and compress the contour, thereby saving memory. In the last stage of the training, the contour of the image is mapped to the name of the input hasta.

The system is trained with all the 28 hastas by providing the input images of all the 28 hastas.

2.2 Capturing the Image And Converting To Gray Scale

Once the system has been trained with all the 28 'Asamyukta hastas', it is tested. A hand gesture is captured and the captured image is given as an input to the system to convert into corresponding gray scale image. The gray scaled image is then converted to the corresponding binary image using Otsu's Binarization.

2.3 Finding contours of the binary image

Contours are the points having the same intensity level. As the binary image consists of only two intensity values (0 and 1), a contour is formed by joining the points having the same intensity. This gives the contour of the binary image.

2.4 Determining the Resemblance between the Input Image and the Trained Images

After the training, the system has learnt the contours of the 28 'Asamyukta Hastas' and has mapped each Hasta image contour to its name. After the contour of the input image has been obtained, the system finds the resemblance between the contour obtained and the contours of all the trained images. It computes the difference between the input contour and the trained contours. The lesser the difference, greater the resemblance. The system outputs the name of the input hasta on the basis of this resemblance.

3. EXPERIMENTAL EVALUATION

3.1 Training images

The system is trained for the hasta images as follows:

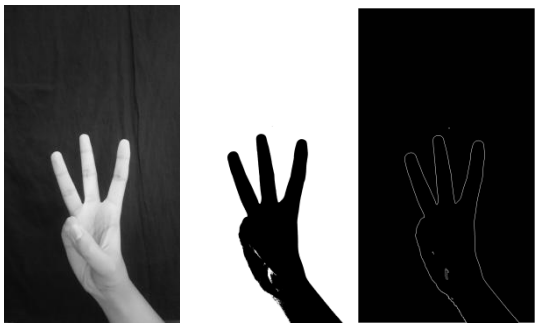


Figure 4. Trishula Hasta

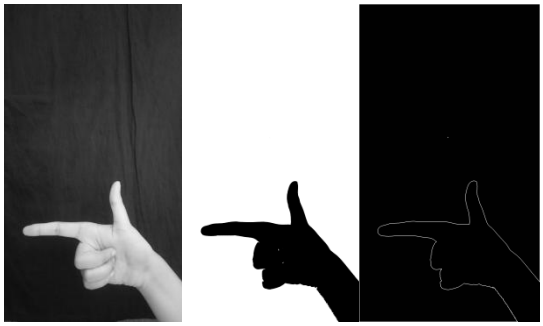


Figure 5. Chandrakala Hasta

3.2 Testing images consisting of Alta

The system is tested for images consisting of hastas of a hand painted with *Alta* (red markings on the finger tips and palm).

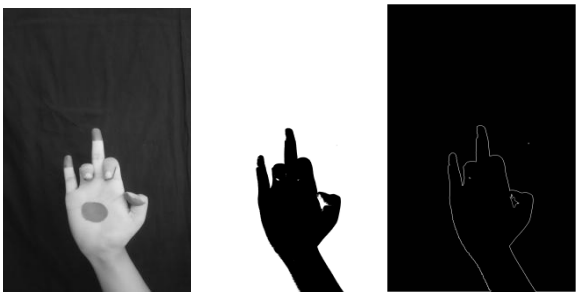


Figure 6. Tripataka Hasta

Figure 7. Padmakosha Hasta

3.3 Testing images consisting of a hand other than the training hand used

The system is tested for another hand

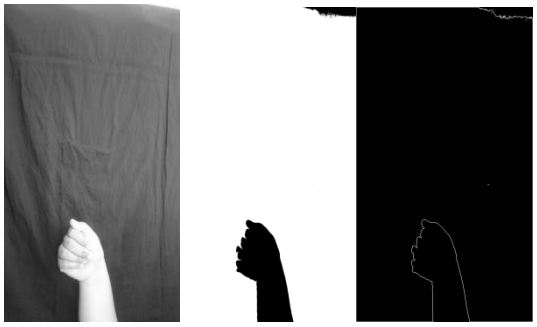


Figure 8. Mushthi Hasta

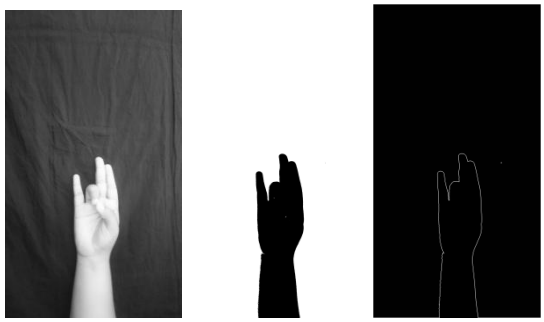
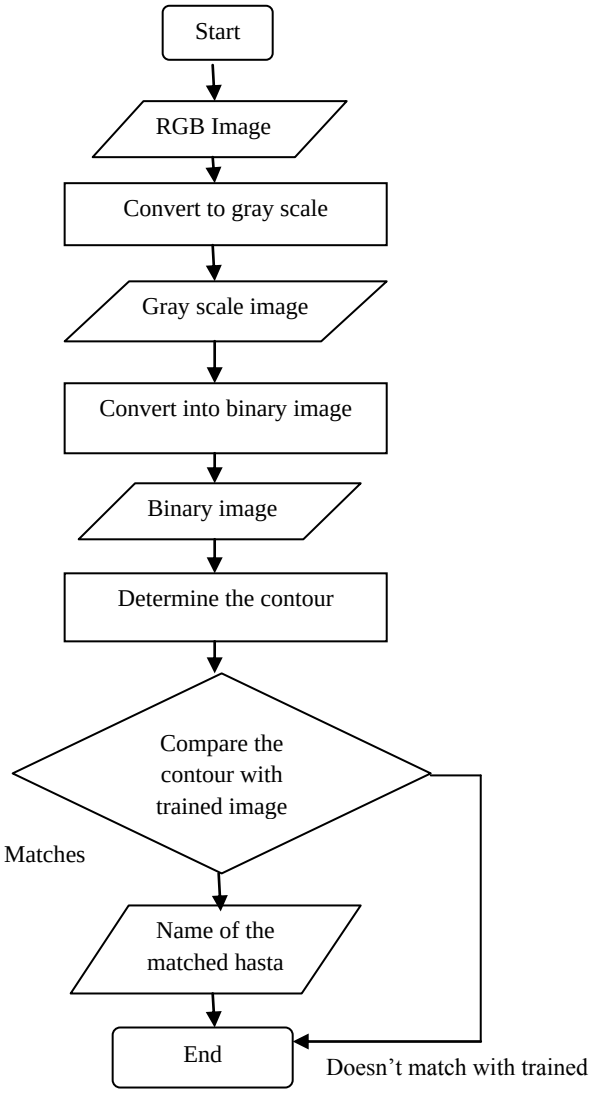


Figure 9. Mayurakhya Hasta

4. CONTROL FLOW DIAGRAM OF THE SYSTEM



5. GRAPHS

5.1 Tested images same as trained images

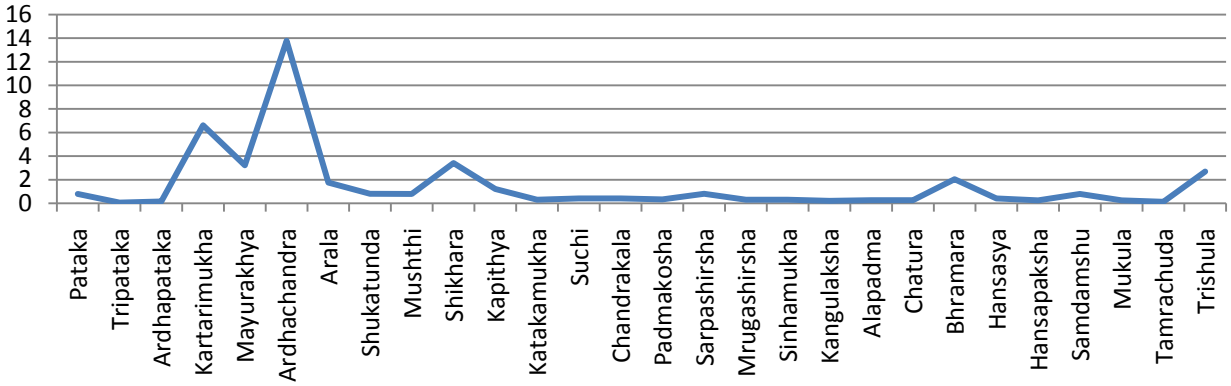


Figure 10

A trained image of 'Tripataka hasta' is tested against all the trained images. The value of the difference between the input image contour value and the trained image contour values is plotted. The minimum in the graph corresponds to the closest resemblance to the hasta which the input image consists.

5.2 Tested image consists of Alta

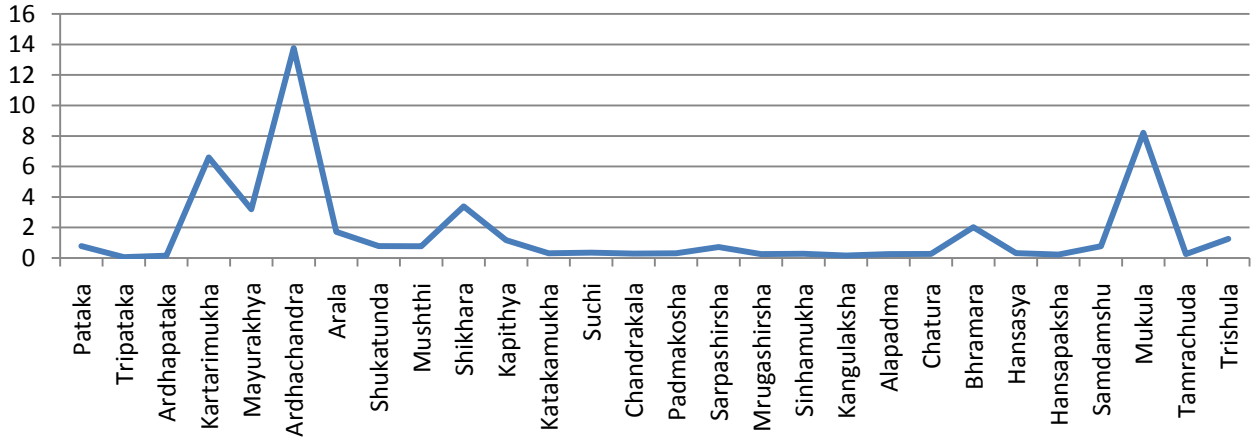


Figure 11

An Alta image of 'Tripataka hasta' is tested against all the trained images. The value of the difference between the input image contour value and the trained image contour values is plotted. The minimum in the graph corresponds to the closest resemblance to the hasta which the input image consists. The minimum here corresponds to the 'Tripataka hasta'.

5.3 Tested image is of different hand

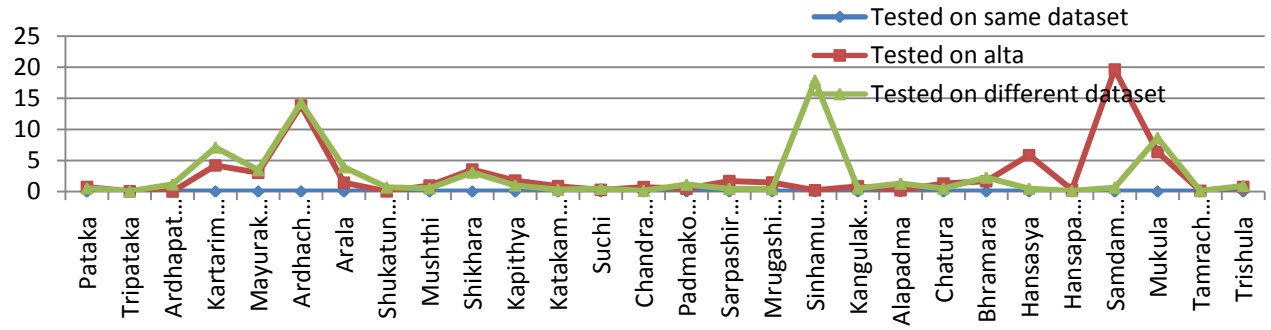


Figure 12

Every hasta is compared to the corresponding trained hasta. In the case where the trained and tested images are the same the difference between the contours values of the trained and tested are zero. When the tested images are of Alta and a different hand the difference varies as the contours won't exactly be similar and hence there is a difference between the contour values of a trained hasta and tested hasta.

6. CONCLUSION

The proposed system was implemented with the help of opencv library. An intelligent system was designed for *asamyukta hastas* hand gesture recognition of Bharatnatyam. It was able to handle different static hand poses of Bharatnatyam and process them to find contours. The implemented system gives 89 percent correct results for the mapping of same hand gestures to the trained hasta images. Due to the effect of differences in contours, the results for mapping of the Alta images and the gesture images of a different hand produces around 80 percent correct results. However, there are certain shortcomings. In the proposed system, a simple background has been considered. Noisy images have been avoided. Runtime images have not been captured. Moreover only 420 images were considered .

In future dynamic runtime images could be analyzed. Also various classification algorithms can be used to classify the images and then recognize them.

7. REFERENCES

[1] 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.

[2] C. Mythili and V.Kavitha, "Color Image Segmentation using ERKFCM" International Journal of Computer Applications (0975 – 8887) Volume 41– No.20, March 2012.

[3] Michal Podpora and Grzegorz Paweł Korbas, Aleksandra Kawala-Janik, "YUV vs. RGB – Choosing a Color Space for Human-Machine Interaction," Position papers of the 2014 Federated Conference on Computer Science and Information Systems pp. 29–34 DOI: 10.15439/2014F206 ACSIS, Vol. 3.

[4] Roy, S and Mitra, A, "Color Scale & grayscale image representation using multivector"Computer, Communication, Control and Information Technology (C3IT), 2015, 978-1-4799-4446-0.

[5] I. Sobel, "Neighborhood coding of binary images for fast contour following and general binary array processing," Computer Graphics and Image Processing,vol. 8, no. 1, pp. 127–135, 1978. .

[6] Sergey Milyaev, Olga Barinova , Tatiana Novikova , Pushmeet Kohli and Victor Lempitsky, "Image binarization for end-to-end text understanding in natural images,mbnlk_icdar2013.pdf.

[7] H. Devi and T. Acharya, "Thresholding: A Pixel-Level Image Processing Methodology Preprocessing Technique for an OCR System for the Brahmi Script," Ancient Asia, Journal of the Society of South Asian Archaeology, Published on 01 Dec 2006.

[8] Image Binarization using Otsu Method by XuLiang.

[9] Image Processing for Computer Graphics By Jonas Gomes, Luiz Velh