

# Evaluation of Corner Detection Algorithms for Human Emotion Modeling

Santosh Kumar Verma

Department of Computer Science & Engineering  
Jaypee Institute of Information Technology  
Sector – 128, Noida, Uttar Pradesh, India

Gaurav Parashar

Department of Computer Science & Engineering  
Integrated Academy of Management & Technology  
NH-24, Near Dasna Flyover, Ghaziabad,  
Uttar Pradesh, India

## ABSTRACT

Human emotion modelling could prove to be an important area of application for the purpose of increasing interaction between human and the computer. For modelling emotion, we have used corners as facial feature present in the image. A corner is a very important feature of an image. It represents intersection of two curves/edges, it also represents a significant change in the colour intensities in the image nearby the point itself. Extraction of corners in the image may prove to be very useful in certain areas of image processing. In this paper, various corner detection algorithms like SUSAN, Harris, Moravec and FAST corner detector algorithms are empirically evaluated with our proposed brute force approach. The comparison is based on how much time does the algorithm takes to detect the corners on the facial features of frontal human face. Furthermore, the algorithm that was found to be performing better was used in the face modelling application.

## General Terms

Computer graphics, Digital image processing, Pattern recognition.

## Keywords

Corner detection, SUSAN, Harris, Moravec, FAST, Brute force, Bezier curve, Face modelling.

## 1. INTRODUCTION

Many factors contribute in conveying emotions of an individual. Facial expressions, speech tone, physical gestures, actions are some of them. In communicating with humans, we can recognize the emotions of another person to a very precise level. If we can efficiently and effectively utilize this knowledge to find practical problems of computer science domain, we would be able to get results as accurate as that recognized by humans. And in this context to communicate with the machines, we need to understand how we can transmit information to machines to perform required tasks. In order to get our facial features being imitated by some machine, we need to first deduce what the machine might need as the input and how will it perform operations to produce the output.

Human emotion modelling could prove to be a way through which human and computer could communicate in the future. The objective of the paper is to find an algorithm that could be used to imitate the emotions of a human captured in an image and for that corners present in the image have been used. Corners are very important features that an image possesses. Corners are the pixels in the image where change in the intensity of colour in the neighbourhood and the point itself is significant. It also represents the point where two edges in the image intersect. Moreover, they are not affected even if we

apply rotation to the original image [1] i.e., they are affine invariant. This leads to the proposal of a variety of corner detection algorithms [1]-[11]. This proposal could be divided into three main groups: contour based, template based, direct corner detection.

Important areas of application of corner feature are: motion detection, video tracking, object recognition, 3D modelling, image mosaicking and many more.

We have applied these corner detection algorithms on the 2D images of facial feature viz. left eye, right eye and lips of human exhibiting four emotions (neutral, happy, sad, and shocked). The image resolutions were not more than 300 X 400 pixels in size. The images were taken from the static camera and under similar conditions, preferably in a static background.

After detecting the corners, they were used to create a modelled face in Blender which would then use those points to exhibit the similar expression to that of the image in question.

During the literature survey, we could not find a comparison between all the algorithms discussed here in a single reference. Therefore, this paper presents a comparative study of few existing corner detection algorithm along with our proposed algorithm under restricted resolution and environment.

## 2. ALGORITHMS

This section presents various corner detection algorithms that are empirically validated in this paper.

### 2.1 SUSAN Corner Detection Algorithm

SUSAN is an acronym for Smallest Univalued Segment Assimilating Nucleus [2]. SUSAN detector uses a circular mask over the pixel to be tested for corner. The pixel is generally called the nucleus. And the area under the mask is called USAN. Every pixel of the respective USAN is compared to its nucleus by a comparison function. When we get the smallest value for some USAN that tells us that the nucleus for that USAN is lying on a corner [2].

A comparison function between the nucleus and all the other pixels within the mask is given by:

$$c(r, r_0) = e^{\{-[I(r)-I(r_0)]/t\}^6} \quad (1)$$

where  $r_0$  is the position of nucleus,  $r$  is the position of any other point inside the mask,  $I(r)$  is the intensity level of point  $r$ ,  $I(r_0)$  is the intensity level of nucleus and  $t$  is the intensity threshold difference.

The size of the USAN region is calculated by:

$$n(r_0) = \sum_{r \in c(r_0)} c(r, r_0) \quad (2)$$

The initial response to a corner is given by:

$$R(r_0) = \begin{cases} g - n(r_0) & \text{if } n(r_0) < g \\ 0 & \text{else} \end{cases} \quad (3)$$

where  $g$  is the geometric threshold and is initialized as  $3n_{\max}/4$  where  $n_{\max}$  is the maximum value that  $n$  can achieve.

## 2.2 Harris Corner Detection Algorithm

Harris corner detector uses the gradient value of each pixel. If the gradient value of the pixel is significant in both the direction, then we consider that pixel as a corner else we reject that [3].

The change in intensity for the shift  $[u, v]$  can be computed using:

$$E(u, v) = \sum_{x,y} w(x, y) [I(x + u, y + v) - I(x, y)]^2 \quad (4)$$

where  $w(x, y)$  is a window function,  $I(x, y)$  is the intensity level at point  $(x, y)$ .

After Taylor expansion, the equation (4) could be re written in matrix from as:

$$E(u, v) = [u, v] M \begin{bmatrix} u \\ v \end{bmatrix} \quad (5)$$

$$\text{Where, } M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (6)$$

And  $I_x$  is gradient in  $x$ - direction and  $I_y$  is gradient in  $y$ -direction.

## 2.3 Moravec Corner Detection Algorithm

Moravec corner detector considers a portion of image centred on a pixel and computes the similarity between two such portions by taking the sum of squared differences between them. The lower the SSD value indicates more similarity. If this value is locally maximal, then a corner has been detected.

The algorithm is as follows [4]:

First, calculate the intensity variation for shift  $(u, v)$  for every pixel in the image by using:

$$E(x, y) = \sum_{a,b \in \text{window}} [I(x + u + a, y + v + b) - I(x + a, y + b)]^2 \quad (7)$$

Where the shift is in all 8 direction from the pixel.

Then check for the SSD values using

$$C(x, y) = \begin{cases} \min_{a,b} E(x, y) & \text{if } \min_{a,b} E(x, y) > \text{threshold} \\ 0 & \text{else} \end{cases} \quad (8)$$

Now find the local maxima. All these points are considered to be corners.

## 2.4 FAST Corner Detection Algorithm

FAST is an acronym for Features from Accelerated Segment Test. It uses a circle of 16 pixels (Bresenham circle of radius 3) to specify whether the centre is a corner or not. If some continuous  $N$  pixels have intensity levels different from that of the centre, then only the point was considered to be a corner otherwise not.

In the first version of the algorithm, the authors used  $N=12$ . To make the algorithm work quickly, the intensities of four points that lied along the axis were compared. If the comparison resulted in at least 3 out of 4 pixels satisfying the criteria, then only to check for the remaining points [6].

## 2.5 Brute force approach using Bezier Curves (Proposed Algorithm)

Input to this approach was a pre-processed image. Pre-processing steps included noise removal [8] and segmentation using frame differencing of background segmentation techniques [9]. This algorithm first converted the image to binary image and then checked for all the changes from black to white and vice versa. Then the largest area was found and the corners of that area were recorded. Afterwards, Bezier curve was drawn from those points and all the points that lied on the curve were recorded. All those points were considered for the evaluation. Figure 1 presents the overall architecture of the proposed algorithm. The proposed algorithm for corner detection from facial expression and emotion modelling is divided into following steps:

### 2.5.1 Skin colour Segmentation

This step has been used to get the facial part from the input image by converting the given RGB image into YCbCr color model [18, 20]. After we get the facial part from the frontal face image, we proceed to next step which is to extract facial features (like eyes and lips).

### 2.5.2 Facial feature extraction

To imitate the emotion depicted in the image, we need to extract facial features (eyes and lips) [15, 16] and for that purpose we use the concept of symmetry of human face. To extract left eye we select the pixels from top left part to the middle position. And for the extraction of right eye, we select the pixels from the middle point to extreme right part of the face. To get lips we take the lower part of the face.

### 2.5.3 Approximation curve fitting and Traversal

After obtaining the results from the previous step, we find the respective coordinates that are required for the Bezier curve drawing. We obtain the points by using BFS (Breadth First Search) algorithm to get the largest connected components on the image and thus using the coordinates  $(x, y_{\min})$ ,  $(x, y_{\max})$ ,  $(x_{\min}, y)$ ,  $(x_{\max}, y)$ . After the curve is drawn, we traverse the curve to obtain points to be stored in a file for the purpose of imitation.

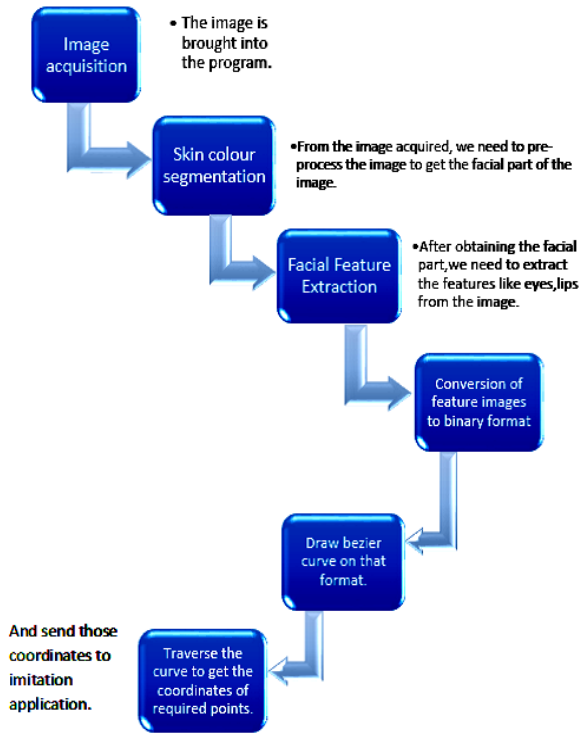


Fig 1: Overall architecture of the Proposed Algorithm

All material on each page should fit within a rectangle of 18 x 23.5 cm (7" x 9.25"), centered on the page, beginning 2.54 cm (1") from the top of the page and ending with 2.54 cm (1") from the bottom. The right and left margins should be 1.9 cm (.75"). The text should be in two 8.45 cm (3.33") columns with a .83 cm (.33") gutter.

### 3. EVALUATION OF CORNER DETECTION ALGORITHMS

There are six basic facial expressions that human being exhibit. These include emotions like happy, sad, disgust, shocked, anger and fear. There are many more expressions but those are much more complicated than these. So, four expressions (happy, neutral, sad, and shocked) were considered for the evaluation purpose. The criterion for the evaluation was the time taken by the algorithms to detect corners in the images of left eye, right eye and lips. Expressions were recorded from four individuals. The first column in the table represents the algorithm used and the values signify the time taken in milliseconds. In the tables 1,2,3,4 represents samples of emotions taken from four different individuals.

Table 1: Time taken by the algorithms for expression "Happy"

HAPPY	Sample Time (millisec)				Avg. Time (millisec.)
	Algorithm	user1	user 2	user 3	
SUSAN	1325	1949	2331	778	1595.75
Harris	1412	2185	2339	739	1668.75
Moravec	1306	1963	2904	787	1740
FAST	1345	1763	2319	778	1551.25
Brute Force	1317	1337	1615	532	<b>1200.25</b>

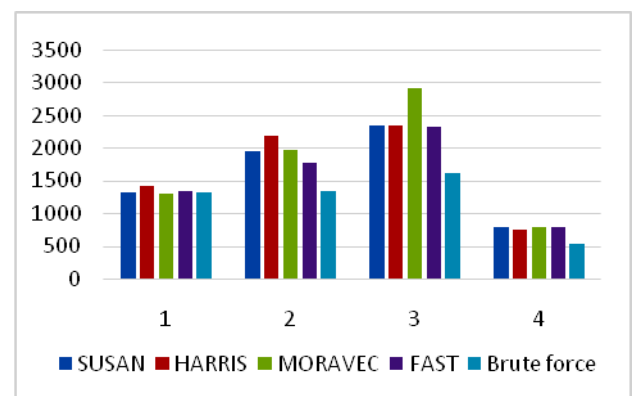


Fig 2: Graphical representation of Happy face.

Table 2: Time taken by the algorithms for expression "Neutral"

NEUTRAL	Sample Time (millisec)				Avg. Time (millisec)
	Algorithm	User1	User2	User3	
SUSAN	1406	1659	2550	1922	1884.25
Harris	1577	1864	2507	1846	1948.5
Moravec	1358	2084	2634	1877	1988.5
FAST	1681	1920	2589	1834	2006
Brute Force	1104	1213	1709	1251	<b>1319.25</b>

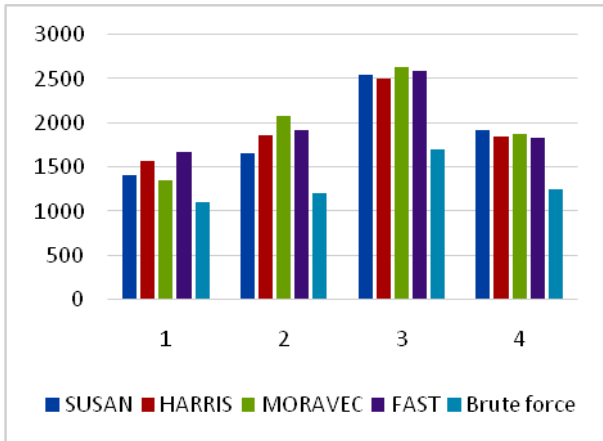


Fig 3: Graphical representation of Neutral face.

Table 3: Time taken by the algorithms for expression “Sad”

SAD	Sample Time (millisec)				Avg. Time (millisec.)
	User1	User2	User3	User4	
SUSAN	1459	1345	1623	1699	1531.5
Harris	1329	1331	1686	1520	1466.5
Moravec	1336	1318	1654	1536	1461
FAST	1298	1598	1687	1530	1528.25
Brute Force	994	1099	1137	1012	1060.5

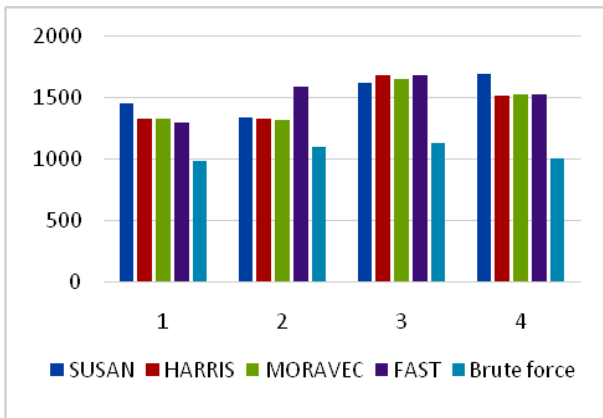


Fig 4: Graphical representation of Sad face.

Table 4: Time taken by the algorithms for expression “Shocked”

SHOCKED	Sample Time (millisec)				Avg. Time (millisec.)
	User1	User2	User3	User4	
SUSAN	1523	1655	2734	3252	2291
Harris	1300	1776	2719	3118	2228.25
Moravec	1575	1670	2822	3296	2340.75
FAST	1540	1645	2705	3301	2297.75
Brute Force	1062	1144	1794	2024	1506

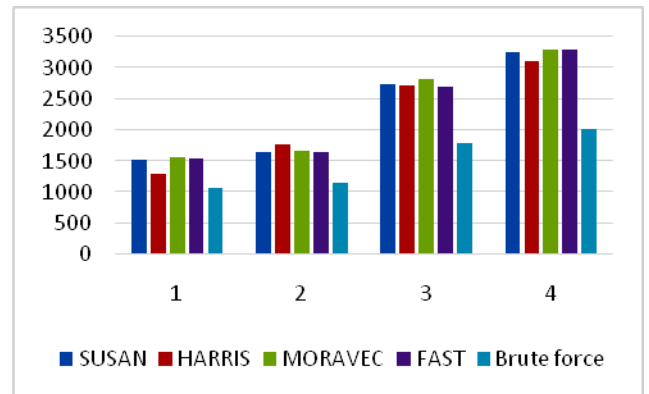


Fig 5: Graphical representation of Socked face.

#### 4. APPLICATION

After the evaluation, it was found that our approach using Bezier curve was performing better in terms of the time taken than the other algorithms for the application purpose. So, we used the approach that included Bezier curve drawing and we traced the points along the curve and then those coordinates were used to imitate a similar expression in modelled human face [13, 14].

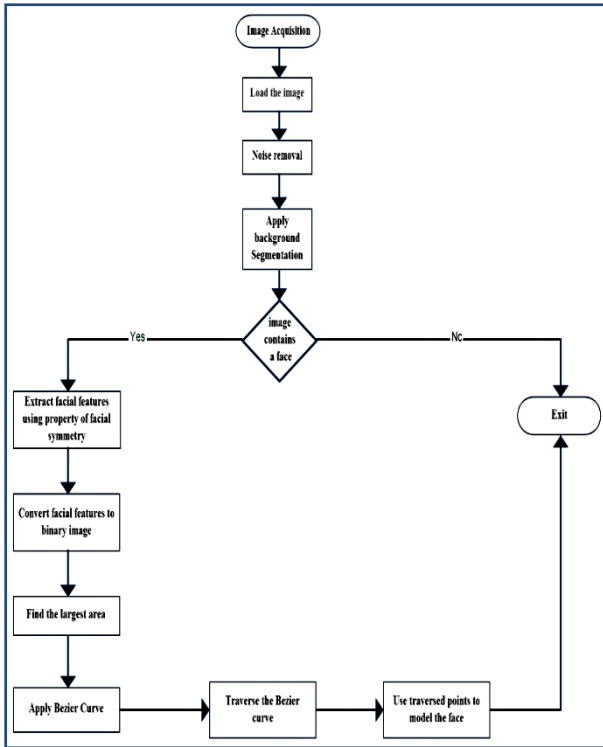

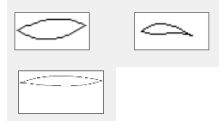
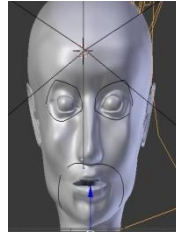

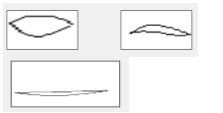
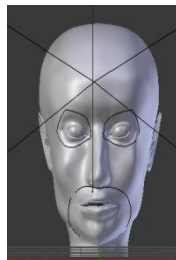
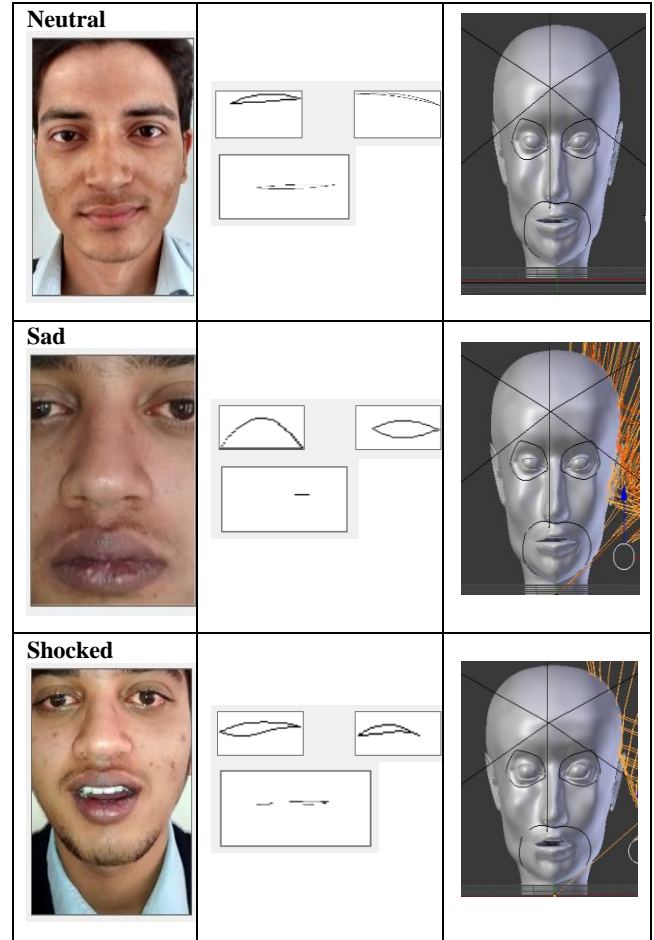


Fig 6: Flow chart for the application

Table 5: Representing actual facial expression, Bezier curve drawn on the eyes and lips of the face, modelled expression in Blender software.

Actual Expression	Bezier curve drawn on the features	Modelled expression
<b>Shocked</b> 		
<b>Happy</b> 		



## 5. EXPERIMENTAL RESULTS

Here, if we compare the response time of various algorithms for a given dataset under similar environmental criteria, we found that our proposed brute force approach is faster than the compared algorithms.

The detected human emotions were modelled on a robotic face generated via 3D mesh using Unity3D game engine.

While, we found our algorithm worthy for the situation with certain assumptions under following:

- I. Some settings need to be configured before the program removes the background from the image and retains only the facial part of the image.
- II. The success for each input cannot be guaranteed 100%.
- III. There is no security based structure to ensure that the images are kept in a secret location and the user without administrative access should not be able to access the images.
- IV. Face detection has not been applied. That is, the input is a frontal face image and if it is not the case, the system is incapable of producing some error message that the input is not a valid one.

## 6. CONCLUSION

These papers presented some important corner detection algorithms and were compared on the criterion of time consumed during the processing of human emotion detection. Then it also discusses an application of corner detection algorithms for interaction between humans and computers. It was found that our approach using Bezier curve was more efficient in terms of computation cost as compared to other algorithms for the application purpose and for the set of images considered.

The presented concept can be used to develop applications for human computer interaction as remote robot training etc. It could also be applied on mobile applications as mimic to the user and so on.

## 7. REFERENCES

- [1] S. Cordelia, M. Roger, B. Christian, "Evaluation of Interest Point Detectors," *International Journal of Computer Vision*, Vol. 37, pp. 151-172, 2000.
- [2] S. Smith, J. Brady, "SUSAN - A new approach to low level image processing," *International Journal of Computer Vision*, Vol. 23, No. 1, pp. 45-48, 1997.
- [3] C. Harris, M. Stephens, "A combined corner and edge detector," *Proceedings of the Fourth Alvey Vision Conference*, University of Sheffield Printing Unit, Manchester, pp. 147-151, 1988.
- [4] D. Nilanjan, N. Pradipti, B. Nilanjana, D. Debolina, C. Subhabrata, "A Comparative study between Moravec and Harris Corner Detection of Noisy Images Using Adaptive Wavelet Thresholding Technique," *International Journal of Engineering Research and Applications*, Vol. 2, Iss. 1, pp. 599-606, 2012.
- [5] H. Moravec, "Rover Visual Obstacle Avoidance," *Proceedings of Fifth International Joint Conference on Artificial Intelligence*, pp. 598-600, 1979.
- [6] R. Edward, D. Tom, "Machine Learning for High Speed Corner Detection," *9th European Conference on Computer Vision*, vol. 1, pp. 430-443, 2006.
- [7] T. Miroslav, H. Mark, "Fast Corner Detection," *Image and Vision Computing*, Vol. 16, pp. 75-87, 1998.
- [8] V. Santosh, G. Sanjay, "An empirical evaluation of wavelet based viz-a-viz classical state of image denoising," *IC3-2013, Sixth International Conference on IEEE*, pp. 331-336, 8-10 August 2013.
- [9] V. Santosh, K. Vinish, S. Amit, "Background Subtraction Techniques," *International Symposia ISAC-2011*.
- [10] C. Jie, Z. Li-Hui, Z. Juan, D. Li-Hua, "The Comparison and Application of Corner Detection Algorithms," *Journal of Multimedia*, Vol. 4, No. 6, pp. 435-441, 2009.
- [11] M. Farzin, M. Farahnaz, "Performance Evaluation of Corner Detectors using Consistency and Accuracy Measures," *Computer Vision and Image understanding*, Vol. 102, No.1, pp 81-94, 2006.
- [12] L. Yong- Hwan, "Detection and Recognition of Facial Emotion using Bezier Curves," *IT Convergence Practice (INPRA)*, Vol. 1, 2013.
- [13] Chavan, P. M., Jadhav M.C., Mashruwala J.B., Nehete A.K. and Panjari P.A. 2013. "Real Time Emotion Recognition through Facial Expressions for Desktop Devices", *International Journal of Emerging Science and Engineering(IJESE)*, Vol.I,7.
- [14] Gupta, N. and Kaur, N. 2013, "Design and Implementation of Emotion Recognition System by using Matlab", *International Journal of Engineering Research and Applications (IJERA)*, ISSN: 2248-9622, Vol.3,4, 2002-2006.
- [15] Gosavi, A.P and Khot, S.R. Sept 2013, "Facial Expression Recognition Using Principal Component Analysis", *International Journal of Soft Computing and Engineering (IJSCE)*, ISSN: 2231-2307, Vol. 3, 4.
- [16] Lee, Y.H, 2013, "Detection and Recognition of facial Emotion using Bezier Curves", *IT Convergence Practice (INPRA)*, Vol. 1.
- [17] [http://en.wikipedia.org/wiki/B%C3%A9zier\\_curve](http://en.wikipedia.org/wiki/B%C3%A9zier_curve)
- [18] <http://en.wikipedia.org/wiki/YCbCr>
- [19] <http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTE/S/spline/Bezier/bezier-construct.html>
- [20] <http://www.equasys.de/colorconversion.html>.