Face Localization and its Implementation on Embedded Platform

Hemprasad Y. Patil, Sachin V. Bharambe, Ashwin G. Kothari and Kishor M. Bhurchandi
Department of Electronics Engineering
Visvesvaraya National Institute of Technology,
Nagpur, India
hemprasadpatil@gmail.com

***Abstract*—Face localization is a first step for face processing system. A lot of algorithmic work on face processing has already been reported. It has mostly been implemented on computers. For better speed, the algorithms need to be implemented on embedded platforms. The embedded algorithms need to be computationally inexpensive. This paper introduces computationally light face localization, ear and neck separation algorithms for implementation on embedded BeagleBoard-xM platform. Experimental results of the proposed algorithms are presented in the results section along with benchmarking with other contemporary algorithms.**

Keywords—Skin detection; Ear Separation; Neck Separation; Face localization; BeagleBoard-xM.

I. INTRODUCTION

To determine the image position of a single face is known as face localization. Face localization assumes that an input image contains a single face. A First step for any face processing system is face localization which is simplified problem of face detection [1], [2].

As the face recognition systems are expected to act as standalone systems which occupy less space, embedded systems are preferred for implementation of face processing systems. Suppose the face processing system is face recognition system which is to be deployed in the form of real-time embedded system. The steps in face recognition system are (a) Face detection and localization (b) Feature extraction (c) Identification [3].As steps (b) and (c) are computationally expensive, step (a) should be computationally efficient so that real-time embedded face recognition system will have less response time.

Many approaches for face localization are already been published in literature. The color processing based approaches are easy to implement [4], [5] and are computationally efficient. As neck skin region is redundant information, it has to be cropped. Reference [6] has the approach based on skin color detection but it does not separate neck and face.

A lot of algorithmic work has already been reported which is implemented on computers. Very few researchers report it as hardware realization. In order to operate embedded systems fast, the algorithm should have less computational complexity. Reference [7] Claims the hardware realization of face detection using BeagleBoard-xM platform, but it does not focus on computationally light algorithm. Operation on 1-D signals for spike detection using BeagleBoard-xM is explained in [8]. As operation on 2-D signals such as face image demands more computational power, robust algorithm is needful.

In this paper, computationally efficient face localization algorithm is proposed for implementation on embedded platforms. The same is executed on BeagleBoard-xM platform which contains asymmetric dual core architecture with an ARM Cortex-A8 and DSP processor.

The rest of paper is organized as follows. Section II describes methods and materials. Various steps for face localization such as color image segmentation for skin regions, skin segmentation based on HSV and YCbCr color spaces, face region separation from background, ear region separation and neck region separation using morphological operations are explained in section III. Section IV describes implementation of face localization algorithm on BeagleBoard-xM platform. Section V presents results and discussion. Section VI describes the future scope of this work.

II. METHODS AND MATERIALS

In this section, the general flow of face localization process is explained as shown in Fig. 1. It is difficult to detect skin regions when the image is in RGB color space. It needs to be converted into some other space for efficient detection. The RGB color image is converted in HSV as well as YCbCr color space. Further the image segmentation based on skin regions is performed by processing components from both color spaces. As ears are redundant information for face recognition systems because many other significant features are available, concerned region is removed using ear separation step. We have used canny edge detector and Morphological operations which are prerequisite steps for neck region separation. In the neck separation step, using horizontal mapping the neck region is removed from binary image. The binary image is mapped on grayscale image which is presented as face localized image.

The above proposed algorithm is implemented absolutely in spatial domain in order to ensure less computational complexity. As most of the processing is done on binary images using morphological operations, less computational cost is required. As higher stages of face processing systems require less redundant information, neck and ear regions are

978-1-4673-4529-3/12/$31.00 ©2012 IEEE

removed from skin segmented image. We implemented face localization algorithm on FEI database [9], [10].

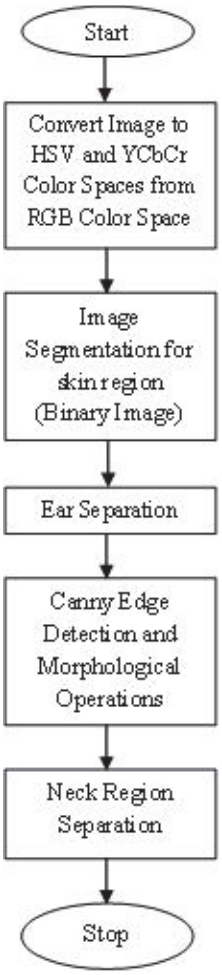

Fig. 1. Face Localization Flowchart

III. FACE LOCALIZATION

A. Color Image Segmentation for skin regions

The skin regions in color image must be separated from background. In order to achieve this, skin color space needs to be defined. As HSV and YCbCr color space has resemblance with human vision, we convert RGB color space into HSV and YCbCr color space. Using (1), RGB color space is converted to HSV color space [11].

$$\begin{cases} H = a\cos(0.5*(2*R-G-B)/\sqrt{(R-G)^2+(R-B)^2}))*57.32 \\ S = (Max(R,G,B)-Min(R,G,B))/Max(R,G,B) \\ V = Max(R,G,B)/255.0 \end{cases} \quad (1)$$

The equivalent YCbCr matrix for RGB image [12] is given by (2).

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.2568 & 0.5041 & 0.0980 \\ -0.1482 & -0.2910 & 0.4392 \\ 0.4392 & -0.3678 & -0.0714 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} \quad (2)$$

B. Skin Segmentation Based on HSV and YCbCr color space

A frontal image from FEI face database is considered for processing. The original image is as shown in fig. 2(a).

The following pseudo code is used for skin segmentation

```
If   141 ≤ Cr(i, j) ≤ 166  is TRUE
                AND
     141 ≤ Cb(i, j) ≤ 196  is TRUE
                AND
     0.001 ≤ H(i, j) ≤ 0.1  is TRUE
Then
     Denote Original.image(i, j) pixel
       as skin pixel; write 1
Else
     Original.image(i, j) pixel is
     Non – skin pixel; write 0
End
```

Where, i and j are respective row and column coordinates of original RGB image denoted by 'Original.image'. As per the given pseudo code, if pixel satisfies all conditions imposed on Cr, Cb and Hue, it is affirmed as skin pixel, else it is non skin pixel. As we wrote '1' and '0' for skin and non-skin pixels respectively, we get fig. 2(b) as binary image.

C. Face Region Separation from background

As shown in fig. 2(c), the binary image is mapped on left column of the image. The First point where 1 exists is marked as point 'A'. The last point where 1 exists is marked as point 'B'. Similarly, if the image intensities are mapped on top row, the first point at which binary 1 exists is called as 'C' and last point is marked as point 'D'. The points A and B give row wise separation and points C and D gives column wise separation. So A to B rows and C to D columns of original image are selected and image is cropped as shown in fig. 2(d).

Fig. 2 (a) Original Image (b) Color Segmented binary image (c) Marked image (d) Cropped image

D. Ear region separation

As in the case of frontal face recognition, ears cannot be treated as much significant features as most other features are

available. They should be cropped. This is done by the following statistical method stated in pseudo code.

For Col(j to j + 5) and all rows
Sum is mapped on a straight line
of Integers called as 'Sm'
End
Where, j ∈ [0 to (Number of Columns-6)]
*Area = 5 * Number of rows*
$$Percentage\ of\ face\ region = \frac{Sm}{Area} * 100$$

As the ear area is approximately 30% of the total face, if percentage is less than or equal to 30%, then that region is to be cropped.

As percentage of face region vector is computed as per pseudo code, we operated from first point up to middle and after applying 30% threshold, it is cropped. Similar operations are performed on percentage of face region vector from last point up to midpoint. The resultant cropping from left and right side is shown in fig. 3(a) and fig. 3(b) respectively. Finally both side ear regions are removed and the image is cropped as shown in fig. 3(c).

E. Neck region removal by morphological operations

As neck portion do not contribute to significant features for face recognition, it has to be removed. The chin has considerable horizontal edge which can separate neck from upper part of face.

We preferred canny edge detector because

(a) Low error rate: All close edges can be found out without forged responses, which give less error rate.

(b) Single points are well localized: The distance between true edge and edge given by detector found minimum [13].

By operating canny edge detector with a threshold of 0.01, we get edge map as shown in fig. 4(a).

As we are interested in horizontal edges, a structural element SE_1 is defined by (3),

$$SE_1 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad (3)$$

Morphological erosion operation is performed on Edge map I_e as per (4).

$$I_{MORPH1} = Erosion(I_e, SE_1) \quad (4)$$

I_{MORPH1} image is as shown in fig. 4(b). The insignificant edges are removed if the pixel neighborhood has fewer than 5 pixels. This is shown in fig. 4(c). This new image is considered as I_{MORPH2}. A new structural element SE2 is defined in (5).

$$SE_2 = [1 \quad 1 \quad 1] \quad (5)$$

As horizontal edges should grow flatly, dilation needs to be performed. As direction of this growth is determined by structuring element, we defined SE_2 with adjacent ones along with centered element as '1'. Morphological dilation operation is given by (6).

$$I_{MORPH3} = Dilation\ (I_{MORPH2}, SE_2) \quad (6)$$

I_{MORPH3} is as shown in fig. 4(d). By this operation, edges are extended in horizontal direction. Now, row-wise sum is computed for lower half region of I_{MORPH3} image starting from last row towards up direction. Largest sum indicates significant edge which separates neck and face given by chin. Fig. 4(d) is cropped from that point. The image coordinates are mapped on image 3(c) and resultant image is as shown in fig. 5.

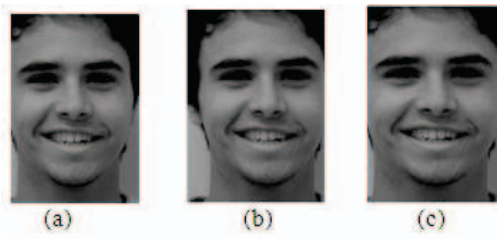

Fig. 3 (a) Left Ear cropped, (b) Right ear cropped, (c) Both Ears cropped

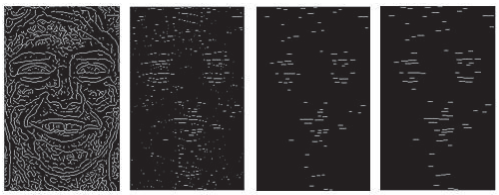

Fig. 4 (a) Edge map (b)Eroded image (c)Significant horizontal edges (d) Horizontally extended edges by dilation

Fig. 5. Neck region separated image

IV. IMPLEMENTATION ON BEAGLEBOARD PLATFORM

Texas Instruments with Digi-Key introduced the BeagleBoard-xM which is open source hardware single board computer [14]. BeagleBoard-xM has 1 GHz ARM frequency and 800 MHz DSP frequency along with 512MB POP RAM at 166 MHz. As it operates with 2W power, additional cooling arrangements are not needed. It has on board USB 2.0 port, HDMI port and Ethernet port.

Fig. 6. BeagleBoard-xM platform[14]

It is also able to process audio signal with 3.5mm audio in and audio out ports. It contains asymmetric dual core architecture with an ARM Cortex-A8 and DSP processor [14], [15]. Because of these striking features, we chose BeagleBoard-xM as embedded platform for experiment. The BeagleBoard-xM platform is as shown in fig. 6.

OpenCV [16], [17] is a library written in C and C++ language. It is free for use and under the BSD license. We implemented face localization algorithm using OpenCV library and executed it on BeagleBoard-xM platform. The experimental setup is as shown in fig. 7. Video Output of BeagleBoard-xM platform is connected to HDMI compliant display device. The Beagleboard-xM platform is initially configured for OpenCV libraries. The equivalent C++ code for pseudo codes mentioned in section II and section III is executed on Linux operating System.

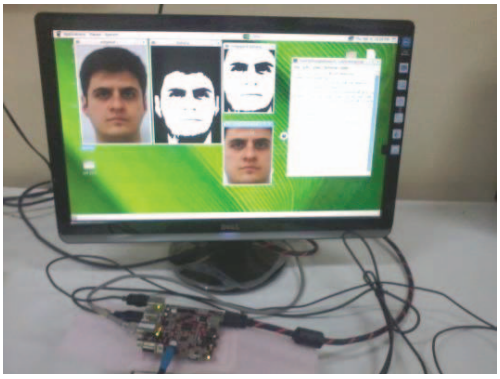

Fig. 7. Experimental Setup on BeagaleBoard-xM

V. Results and Discussion

The face localization algorithm is implemented and tested on FEI database. The results are as presented in table I.

TABLE I. Face Localization Results

No. of Images	FEI color image database	
	Successful Face localization	*% Localization Rate*
400	394	98.5 %

TABLE II. Comparison of Execution time

Software/ Operating System	Platform	Execution Time (Sec)
MATLAB/ Windows 7	Intel core i5 processor	3.2
OpenCV/ Linux	BeagleBoard-xM	0.2

The same algorithm is operated on few representative images from FEI database as shown in fig. 8. The RGB color image indicates original image and grayscale image is output after processing. The face localization algorithm is implemented and tested on MATLAB [18] on windows 7 [19] operating system with Intel core I5 [20] platform. The same is implemented and tested with OpenCV and Angstrom Linux [21] on BeagleBoard-xM platform. Comparison of execution time is presented in table II. It indicates that BeagleBoard-xM can be used for faster execution.

Reference [7] requires 950ms for face detection on BeagleBoard-xM platform, whereas our algorithm needs 200ms. Using the face localization algorithm, the face region can be successfully localized with less computational overhead. The algorithm is tested over embedded ARM platform using OpenCV library. As far as embedded platforms are concerned, they have limited processing power and memory resources. So this algorithm can be successfully used as first step to face processing systems.

Fig. 8. Original images(RGB) and their corresponding face localization (Grayscale)

VI. Future Scope

This work is implemented for still images and can be further extended for face detection in videos. BeagleBoard-xM has 1 GHz ARM frequency. If a platform of improved operating frequency is available, execution time may reduce. If the platform has its own inbuilt display sub system, the overall system size may be compact as compared to the system shown in Fig. 7.

References

[1] Ming-Hsuan Yang, David Kriegman, and Narendra Ahuja, "Detecting Faces in Images: A Survey," IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 24, no. 1, pp. 34-58, 2002

[2] R-L Hsu, M. Abdel-Mottaleb, A.K. Jain, “Face detection in color images,” IEEE Transactions on Pattern Analysis and Machine Intelligence, May 2002, Vol. 24, 696-706.

[3] Sobottka, K. Pitas, I., "Face localization and facial feature extraction based on shape and color information," Image Processing, 1996.

Proceedings., International Conference on , vol.3, no., pp.483-486 vol.3, 16-19 Sep 1996

[4] C.H. Lee, J.S. Kim, K.H. Park, “Automatic Human Face Location in a Complex Background Using Motion and Color Information,” Pattern Recognition, vol. 29, no. 11, 1996, pp. 129-140.

[5] G. Yang and T. S. Huang, "Human Face Detection in Complex Background, Pattern Recognition", vol. 27, no. 1, Jan 1994,pp. 53-63.

[6] Chai, D. Son Lam Phung, Bouzerdoum, A., "Skin color detection for face localization in human-machine communications," Signal Processing and its Applications, Sixth International, Symposium on. 2001 , vol.1, no., pp.343-346 vol.1, 2001

[7] Aby, P.K. Jose, A. Dinu, L.D. John, J,Sabarinath, G. , "Implementation and optimization of embedded Face Detection system," Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), 2011 International Conference on , vol., no., pp.250-253, 21-22 July 2011

[8] Mankodiya, K. Vogt, S. Kundu, A. Klostermann, M. Pohl, J. Ayoub, A. Gehring, H. Hofmann, U.G., "Portable electrophysiologic monitoring based on the OMAP-family processor from a beginners' prospective," Digital Signal Processing, 2009 16th International Conference on , vol., no., pp.1-8, 5-7 July 2009.

[9] C. E. Thomaz and G. A. Giraldi. “A new ranking method for Principal Components Analysis and its application to face image analysis,” Image and Vision Computing, vol. 28, no. 6, pp. 902-913, June 2010

fei.edu.br/~cet/facedatabase.html

[10] Ping Zhang, "A video-based face detection and recognition system using cascade face verification modules," Applied Imagery Pattern Recognition Workshop, 2008. AIPR '08. 37th IEEE , vol., no., pp.1-8, 15-17 Oct. 2008

[11] Soontranon, N. Aramvith, S. Chalidabhongse, T.H. , "Face and hands localization and tracking for sign language recognition," Communications and Information Technology, 2004. ISCIT 2004. IEEE International Symposium on , vol.2, no., pp. 1246- 1251 vol.2, 26-29 Oct. 2004

[12] Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing,Prentice-Hall, Inc. Upper Saddle River, NJ, USA ,2006

[13] BeagleBoard-xM Rev C System Reference Manual, Revision 1.0, April 4, 2010, http://beagleboard.org

[14] Shen Khang Teoh, Vooi Voon Yap, Chit Siang Soh, Sebastian, P. , "Implementation and optimization of human tracking system using ARM embedded platform," Intelligent and Advanced Systems (ICIAS), 2012 4th International Conference on , vol.1, no., pp.353-356, 12-14 June 2012

[15] opencv.willowgarage.com/wiki/CiteOpenCV

[16] G. Bradski and A. Kaehler, Learning OpenCV, OReilly Publications,2008

[17] www.mathworks.in/products/matlab/

[18] windows.microsoft.com/en-IN/windows7/products/home

[19] http://www.intel.com/content/www/us/en/processors/core/core-i5-processor.html

[20] www.angstrom-distribution.org