ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

An Eye Tracking Scheme Employing Viola-Jones and Template Matching Algorithm

Lakshmi Sagar H.S[*1], Saneesh Cleatus.T[2], Roshini Seshadri[3], S Kala Rubini[4], Sahana Sathyanarayana[5]

[*1] Assistant Professor, 2 Associate Professor, 3,4,5 Students, Department of Electronics and Communication Engg., B.M.S. Institute of Technology, Bangalore- 560064, India

Sagar8.hs@gmail.com

Abstract

The purpose of this paper is to develop a scheme to track the movement of the eyes. Most eye tracking systems employ Infrared cameras. Considering the application of our work, we are using a regular web camera to minimize the ill effects of IR cameras. Since the implementation is in real time, OpenCV (Open Source Computer Vision) and Code::Blocks are preferred with an added advantage of being free, thus making the system economical. Analysis of the results concluded that while the accuracy of the eye tracker was adequate, more robust system can be developed by using alternate search techniques. This work allows us to develop an alternate mode of communication for those suffering from motor neuron diseases such as ALS (Amyotrophic Lateral Sclerosis). This effort would help PALS (People with ALS) to communicate through their eyes, which is the only mode of interaction available for them.

Keywords: Assistive technology, Viola-Jones algorithm, Haar-like features, Template matching algorithm, circular contours.

Introduction

Amyotrophic lateral sclerosis[1] is a debilitating, neurodegenerative disease. Its various symptoms include dysphagia, dysarthria, respiratory distress, pain, and psychological disorders. It is characterized by progressive muscular paralysis reflecting degeneration of motor neurons, corticospinal[2] tracts and the spinal cord. Most cases of ALS are readily diagnosed and the error rate of diagnosis in large ALS clinics is less than 10%. The defining feature of ALS is the death of both upper and lower motor neurons in the motor cortex of the brain, the brain stem, and the spinal cord. Sensory nerves[3] and the autonomic nervous system are generally unaffected, meaning majority of people with ALS will maintain hearing, sight, touch, smell, and taste. ALS is the most common of the five motor neuron diseases. In our paper we propose to track the eye, which is the only mode of interaction available for the PALS. Our proposed method involves tracking the eye in real time, and later subjecting it to preprocessing before performing the detection. This is followed by the layout of our work which involves using the capturing unit to obtain the required frames, and preprocessing these frames using techniques such as RGB2GRAY conversion, Histogram Equalization, and binary thresholding. The paper then elaborates on the algorithms being used for the tracking and detection process such as Viola-Jones and template matching. The method of implementation is then discussed which includes the explanation of the individual functions used for face, eye tracking and pupil detection in detail. The results of these operations are presented at the end of the paper.

Existing Methods

Assistive technology (AT) can imply a kind of service or device that is used to ameliorate the functional capabilities of disabled people. There are numerous assistive technologies available for ALS patients for each stage as the disease progresses such as arm supporter, head pointer, electrooculography[4], Eye writer[5] etc. When we consider high-tech methods like electrooculography, it performs complicated eye tracking by measuring the corneo-retinal standing potential that exists between the front and the back of the human eye. Also observing that the Eye Writer, which is a technology used when PALS are in their terminating stages, is comparatively expensive. So in our work, we attempt to provide a more cost effective, efficient technique to help them communicate better while allowing for

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

assurance that they have conveyed their messages correctly with the help of a simple feedback system.

Proposed Method

Eye tracking is a process by which the point of gaze or the movement of the eyes is measured relative to the head. One of the methods for implementing it is by capturing the eye movements in real time through a web camera at approximately 20 frames per second in our case and passing the frames to a computer interface after which the required result is displayed. The video frames obtained from the web camera are subjected to pre-processing in order to help the detection and tracking algorithms work more efficiently and precisely. Open source Computer Vision (OpenCV)[6] software is our chosen computer interface which is a library mainly meant for real time image processing that contains the functions used in our proposed algorithms. Supporting the OpenCV is Code::Blocks which is an open-source, cross-platform C/C++ IDE suitable for our work and the selected platform here is Ubuntu[7], a Linux operating system based on free software. Existing methods used for eye tracking (e.g. solely Viola-Jones algorithm [8]) have extremely long training time and high rate of false positive detections. In our work, we propose to merge Template matching algorithm [9] with Viola-Jones algorithm to overcome these inaccuracies. The face is detected first, followed by the eyes, using Viola-Jones face detector consisting of Haar cascade classifiers [10][11] which are nothing but a cascade of boosted classifiers working with haar-like features. Once detection is successful, template matching algorithm is used to track the eyes after which the region of interest is targeted and searched for circular contours representing the dark, circular nature of the pupil. The overview of our proposed technique is depicted in the block diagram as follows.

Capturing Unit

The web camera used to develop this model is a FRONTTECH Jil 2243 camera which performs real time capturing of images. The obtained images are then subjected to pre-processing. The resolution of the frames obtained is 320 x 240.

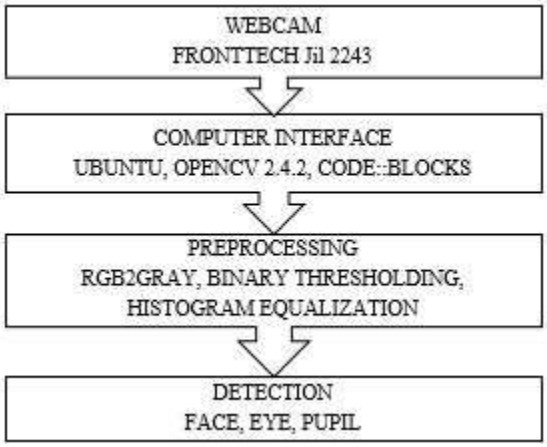
Pre-processing:

RGB to Gray

The image is converted to gray scale mainly for two reasons:

1. The color image occupies more space and it is not easy to store and process such images.
2. The color image contains redundant information which won't be of any use when face recognition is performed.

Figure 1: Block diagram including the steps employed in our proposed method.



The formula[12] used for this conversion is $Gray\ image = 0.3 \times R + 0.59 \times G + 0.11 \times B$ where R,G&B are the red, green and blue intensity values in the image. OpenCV's cvtColor() function is used for this conversion. The format of this function is: *cvtColor(src ,dest , code);* where *src* represents the input image, *dest* is the output image which is of same size as the input image, *code* represents the color space conversion code which can be BGR2GRAY, BGR2HSV and BGR2YCrCb.

Histogram Equalization

Histogram of an image is a graphical representation of intensity values i.e. it is number of pixels in an image v/s the intensity. Histogram equalization[13] converts the histogram of original image into well dispersed histogram so that image features can be highlighted. It evenly distributes the intensity values of the original image, that is, intensity values which are clustered in the middle of the entire range are spread out. Thus mapping one distribution (given histogram) to another distribution which is wide and uniform compared to the original histogram intensity distribution. It is evident that only the y axis (intensity values) has to be spread out as evenly as possible. The best solution available is that the remapping function should be cumulative distributive function. For a histogram H(i)[13] , the cumulative distributive function H′(i) will be

$$H'(i) = \sum_{0 \le j < i} H(j)$$

To use the remapping function, H'(i) should be normalized such that its maximum value is 255. Histogram equalisation is performed using the

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

OpenCV's equalizeHist() function. The format of this function is: *cvEqualizeHist(src,dest);*
where the source and the destination must be of the single channel, 8 bit images of the same size. For equalizing colour images, channels must be separated and processed one after the other.

Binary thresholding

Thresholding is a method for performing segmentation that is used for highlighting the required features in the image. It is performed by setting a threshold value and comparing each pixel intensity value with it. The output image is a binary image having two values 0 and 1. It is used to filter out noise and also, filtering out pixels with very low or very high intensity values. OpenCV's cvthreshold() function is used for this purpose. The function's format is:
cvThreshold(src, dest, threshold, maxval, type);
where src and dest are the input and output images respectively. Threshold is the value which is set so that other pixel values can be compared with it.
Maxval is used along with THRESHOLD_BINARY and THRESHOLD_BINARY_INV. Type denotes the thresholding type which can be
THRESHOLD_BINARY,
THRESHOLD_TOZERO,
THRESHOLD_TOZERO_INV etc.
Based on the relationship between the source pixel and threshold value, the destination pixel can either be set to maxval or zero.

The binary thresholding can be expressed as

$$dst(x,y) = \begin{cases} maxVal\ if\ src(x,y) > threshold \\ 0 \qquad\qquad otherwise \end{cases}$$

If the source pixel is greater than the threshold value the destination pixel is set to the maxval. If it is less than the threshold value, the destination pixel is set to zero.

Algorithms implemented:

Viola-Jones

Viola Jones algorithm, which is used for face and eye detection, is based on Haar like features[14]. They are called Haar features because of their similarity with Haar wavelets. Haar like features exhibit rectangular structure as shown below. These features are mainly used for feature detection. The sum of the pixels from the white portion is subtracted from the ones present in the grey portion. Figure (A) & (B)[15] shows the two rectangle feature which is the sum of the difference between the two regions. It is mainly for edge detection.

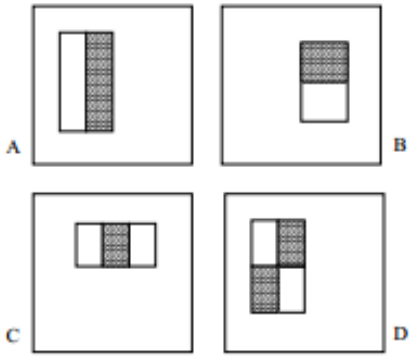


Figure 2: Haar-like features

Figure (C) shows the three rectangle feature which basically calculates the sum of the two outer rectangles and then subtracts it from the central region. Figure (D) shows the four rectangle feature, which computes the difference between the diagonal elements. Based on the position of the detection window (which acts like a bounding box for the face), the rectangles are defined. At every stage, the face candidates are scanned. The weight of each feature is generated by *AdaBoost algorithm* (a machine learning algorithm). Each Haar feature has a value generated by taking the area of each rectangle and multiplying them by their respective weights, after which it is summed. Since a stage contains several Haar features, a stage comparator compares the value of the sum obtained from each stage with a stage threshold value which is a constant obtained from *AdaBoost algorithm*. There are cascade of stages which are used for eliminating the face candidates which do not satisfy the condition. A face is detected if the candidate passes all the stages as shown in figure (3).

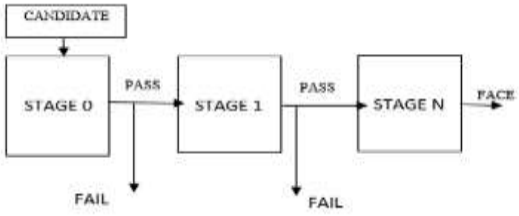


Figure 3: Cascading stages.

Based on the cascade classifier file that is loaded, the features are tracked. In our case the cascade classifiers used are
"haarcascade_frontalface_alt2.xml" and
"haarcascade_eye.xml".
The cascade classifier detects the face first and using the area of the face as reference, it detects the eye. In

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

OpenCV, there are applications which support training of a cascade classifier: *opencv_haartraining and opencv_traincascade.*

Template matching:

Template matching is a widely used method for detecting targets when provided with prearranged information regarding the target. This prearranged information provides a template image using which the most similar image pattern is searched. A parameter called similarity index is used to perform this comparison. The OpenCV function *matchTemplate()* is used to search for matches between an image patch and an input image. In template matching, all possible locations to be matched with the chosen template are stored in a resultant matrix R[16]. This matrix R stores the coefficient value for each matched location in a pixel. The size of the source image is W × H, where W and H represent the width and height, respectively. Also, w × h is the product of width and height of the template image. The matrix size is thus given by (W – w + 1) × (H – h + 1).

There are several types of template matching methods, such as the Sum of Squared Difference, Cross Correlation etc.[17] Consider *T* is the template image, *I* is the input source image & *SSD(x,y)*, *CC(x,y)*, are the numerical indices for Sum of Squared Difference, Cross Correlation respectively, in the range [0,1] at the position *(x,y)* after matching. The equations for these methods are as shown below:

$$SSD(x,y)= \frac{\sum_{x',y'}[T(x',y')-I(x+x',y+y')]^2}{\sqrt{\sum_{x',y'}T(x',y')^2\sum_{x',y'}I(x+x',y+y')^2}}$$

$$CC(x,y)= \frac{\sum_{x',y'}T(x',y')I(x+x',y+y')}{\sqrt{\sum_{x',y'}T(x',y')^2\sum_{x',y'}I(x+x',y+y')^2}}$$

Template matching has various applications such as pattern recognition, object tracking etc. The ease of implementation of template matching along with several fast algorithms in order to speed up the matching process makes its popularity in various applications evident.

Method of implementation

The sequence in which the pre-processing and detection algorithms are carried out is as indicated in figure 4.

The steps are explained in detail as follows.

- On execution of the program, a video stream is opened.
- The program consists of mainly three functions: *main(), detectEye()* and *trackEye().* In the *main()* function, we use the function *load* to load a .xml classifier file(*haarcascade_frontalface_alt2.xml* and *haarcascade_eye.xml*) to perform face and eye detection respectively.
- Two objects are created for the class

Figure 4: Flow chart for the method of implementation

CascadeClassifier (*face_cascade, eye_cascade*) with global scope, which are used to detect facial features in the video stream. The dimensions (width and height) of the bounding box are then checked. If the bounding box is found empty, the *detectEye()* function, which performs face and eye detection is called, after which the eye is tracked.

- In the *detectEye()* function, the *detectMultiScale* function is used to perform face and eye detection. If the location of the eye is successfully obtained, two parameters are returned by the function- the eye template and the bounding box.
- The *trackEye* function is called once the bounding box is fixed and three parameters- the current frame, bounding box & the eye template are given as the input. Template matching algorithm is used to compare the eye template with the frame, thereby tracking the eye.
- The frame is then subjected to pre-processing after which the frame is searched for circular contours using the OpenCV function *cv::circle*, a circle representing the pupil is drawn along the boundary of the contour, following which it is filled. Thus pupil tracking is achieved.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

Present Scope

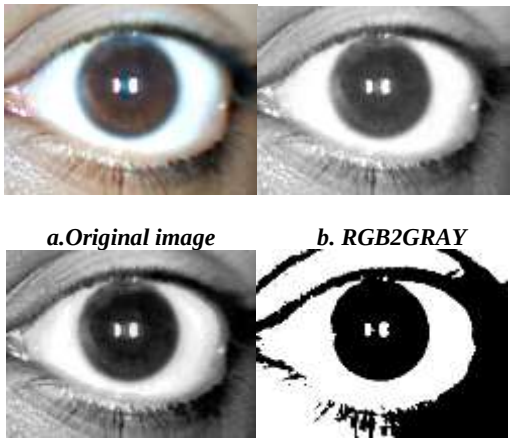
1. The eye tracking project scans the eye movement and converts it into a language for better communication.
2. It would help patients in various tasks such as communication, writing emails, drawing etc.
3. It aids people with symptoms similar to that of ALS.

Results and Conclusion

The below results were obtained when experimented under different lighting conditions and different subjects.

Pre-processing

Original 8 bit input image (figure 5.a) was subjected to gray conversion and the result is shown in figure 5.b. The gray scale image was subjected to Histogram equalisation and Binary thresholding. The respective results are shown in figure 5. c & d. The tracking algorithm was implemented for various subjects and the results of pupil tracking is shown in figure 6 & figure 7.



a.Original image ***b. RGB2GRAY***

c. Histogram equalisation ***d. Binary thresholding***

Figure 5: Snapshots indicating different pre-processing results.

Figure 6: Pupil tracking results for different subjects

Figure 7: Pupil tracking results for different eye movements

Acknowlegement

This work is funded by Karnataka State Council for Science and Technology (KSCST). We are currently working on our project ***Eye Voice: Eye controlled assistive technology for ALS patients.***

References

[1] Mitchell, J. Douglas, and Gian Domenico Borasio. "Amyotrophic lateral sclerosis." The lancet 369, no. 9578 (2007): 2031-2041.

[2] Jackson, Carlayne E., and Wilson W. Bryan. "Amyotrophic lateral sclerosis." In Seminars in neurology, vol. 18, no. 01, pp. 27-39. © 1998 by Thieme Medical Publishers, Inc., 1998.

[3] Wijesekera, Lokesh C., and P. Nigel Leigh. "Amyotrophic lateral sclerosis." Orphanet journal of rare diseases 4, no. 1 (2009): 3.

[4] Bulling, Andreas, Jamie A. Ward, Hans Gellersen, and Gerhard Troster. "Eye movement analysis for activity recognition using electrooculography." Pattern Analysis and Machine Intelligence, IEEE Transactions on 33, no. 4 (2011): 741-753.

[5] "Eye Writer" Internet: www.eyewriter.org

[6] Culjak, Ivan, et al. "A brief introduction to OpenCV." MIPRO, 2012 Proceedings of the 35th International Convention. IEEE, 2012.

[7] Al Housani, B., Bakhita Mutrib, and Hend Jaradi. "The Linux review-Ubuntu desktop edition-version 8.10." Current Trends in Information Technology (CTIT), 2009 International Conference on the. IEEE, 2009.

[8] Viola, Paul, and Michael J. Jones. "Robust real-time face detection". International journal of computer vision 57.2 (2004): 137-154.

[9] Phuong, Hoang Minh, et al. "Extraction of human facial features based on Haar feature with Adaboost and image recognition techniques." Communications and Electronics (ICCE), 2012 Fourth International Conference on. IEEE, 2012.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

[10] *Zhu, Zhaomin, et al. "Multi-view face detection and recognition using haar-like features." Proceedings of COE Workshop. 2006.*

[11] *Dabhade, Sarala A., and Mrunal S. Bewoor. "Real Time Face Detection and Recognition using Haar-Based Cascade Classifier and Principal Component Analysis." International Journal of Computer Science and Management Research 1.1 (2012).*

[12] *Chen, Haiyan, and JunHong Lu. "The image process technologies in face recognition." In Information Science and Engineering (ICISE), 2010 2nd International Conference on, pp. 4151-4154. IEEE, 2010.*

[13] *Bradski, Gary, and Adrian Kaehler. Learning OpenCV: Computer vision with the OpenCV library. O'Reilly Media, Inc., 2008.*

[14] *Padmaja, K., and T. N. Prabakar. "FPGA based real time face detection using Adaboost and histogram equalization." In Advances in Engineering, Science and Management (ICAESM), 2012 International Conference on, pp. 111-115. IEEE, 2012*

[15] *Majumder, Anima, Laxmidhar Behera, and Venkatesh K. Subramanian. "Automatic and robust detection of facial features in frontal face images." InComputer Modelling and Simulation (UKSim), 2011 UkSim 13th International Conference on, pp. 331-336. IEEE, 2011.*

[16] *Bae, Jong Sue, and Taek Lyul Song. "Image tracking algorithm using template matching and PSNF-m." International Journal of Control Automation and Systems 6, no. 3 (2008): 413.*

[17] *Chen, Jiun-Hung, Chu-Song Chen, and Yong-Sheng Chen. "Fast algorithm for robust template matching with M-estimators." Signal Processing, IEEE Transactions on 51, no. 1 (2003): 230-243.*