Global Scientific JOURNALS

GSJ: Volume 6, Issue 3, March 2018, Online: ISSN 2320-9186
www.globalscientificjournal.com

EYE MOVEMENT DETECTION USING MATLAB

Aditi Ghosh, Divya Chandan, Tanisha Ghosh, Prof. Rajini G.K.
VIT (Vellore Institute of Technology)

ABSTRACT:

Life is crucial hence safety precaution should always be taken before any kind of accident occurs. These days road accident is one of the major causes of life insecurity. Even a single moment carelessness can cause lifetime regret. According to many studies more than half of the road accidents occur because of carelessness and inactiveness of the driver. Drowsiness is one the major factor that causes inactiveness of the driver. It leads to increase in number of road accidents every year. If drowsiness is detected early enough then it could save many road accidents. Drowsiness can be detected by monitoring the eye movement of the driver. Here we are developing a prototype of the same. The system uses a small camera (webcam in this case) that points directly towards the driver's face and monitors the driver's eyes in order to monitor the eye movement. Firstly the system detects the face and then eyes, and then it determines if the eyes are open or closed. The system uses information obtained from the binary version of image to find the edges of the face, which narrows the area of where the eyes may exist. Once the eyes are located, measuring the distances between the intensity changes in the eye area determine whether the eyes are open or closed. If the eyes are found closed for 5 or more consecutive frames, then the system concludes the inactiveness of the driver. The system detects the direction in which driver moves his eyes and we get to know it, so that we can keep a check on him.

Keywords: Eye blinking, Face detection, Haarcascade Face samples, Viola Jones algorithm

I. INTRODUCTION

Driver fatigue results in over fifty percent of the road accidents each year. It's high time and we need technology to avoid such cases. In the last decade, many countries have begun to pay attention to the automobile driver safety problem. Researchers have been working on the detection of automobile driver's drowsiness using various techniques, such as physiological detection and Road monitoring techniques. However, all the research till date in this approach need electrode contacts on the automobile drivers' head, face, or chest making it non-implementable in real world scenarios. The major challenges of the proposed technique include (a) Developing a real time system (b) Face detection (c)Eye detection (d) eyeball movement detection. Our main focus is to design a real time system which will accurately monitor the eye movement of the driver. If Symptoms of the driver fatigue is detected early enough, accidents can be avoided. Fatigue detection involves observation of eye movements and the blink patterns in sequence of images extracted from a live video.

II. PROPOSED WORK

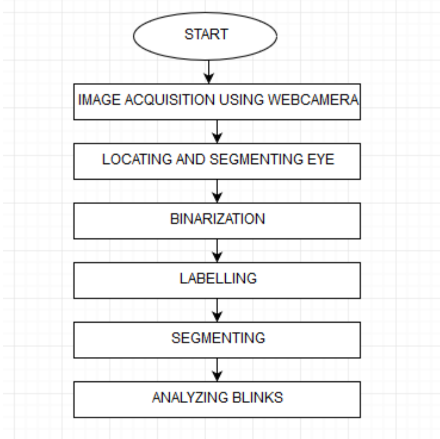
Video acquisition: It involves obtaining the live video feed of the Automobile driver. It is achieved, by using camera (webcam in this case) and then dividing into frames. This module takes live video as its input and then convert it into a series of frames or images, which are then processed.

Face detection: The face detection function takes one frame at a time from t frames provided by the frame grabber. In every frame it tries to detect the face of the driver and is achieved by making use of a set of pre-defined Haarcascade samples.

Eyes detection: Once the face has been detected via the face detection function, the eyes detection function tries to detect the driver's eyes and is done by Voila Jones algorithm.

Name: Aditi Ghosh, Divya Chandan, Tanisha Ghosh pursuing degree in B.Tech third year in branch Electronics and Instrumentation Engineering, VIT (Vellore Institute of Technology)
Guided by: Prof. Rajini G. K.

III. FLOW CHART



For detecting the face, we can avoid processing the image at the corners which will reduce a significant amount of processing required. As the region of interest is the face, first the face is detected. Once it is done, next step involves detecting the eyes. For detecting the eyes, instead of processing the entire face region, we take a region of interest within the face which helps in achieving the primary goal .Next we use Viola Jones algorithm for eye detection, and process only the region of interest. The next step is to determine whether the eyes are in open/closed state, once the eyes have been detected. It is achieved by extracting and examining the pixel values from the eye region. If eyes are in the left direction, it indicates output as left. If eyes are in the right direction, it indicates output as right. if the driver is looking straight, it indicates output as straight. If there is any distortion or if the eyes are closed, it must give the output as no face.

IV. DESCRIPTION

A. Our project is based on the Viola-Jones algorithm. This module takes live video as its input and then convert it into a series of frames or images, which are then processed. From this, only the eyes are sectioned out. Position, width and height of the region are given as inputs to the rectangle( ) function to detect the eye region of the image. Position, width and height are obtained by using the Vision class in MATLAB. Built in object detector function CascadeObjectDetector is used to detect the eyes. The Eye Detect object is given as input to the step function along with the image and the values returned correspond to the X-Coordinate, Y Coordinate, Width and Height of the eye region. The image is now cropped using the imcrop( ) function with one input as n*4 matrix and the other being the image itself. Using the rgb2gray( ) function, the RGB image obtained is converted to its equivalent grayscale form. This is followed by converting the obtained gray scale image to its black and white form using im2bw( ). The black and white image thus obtained is dilated to get only the eyes. The dilation function is used to enhance the foreground features. IM2=imdilate(IM,SE) dilates grayscale, binary, or packed binary image IM, returning the dilated image, IM2. SE is structuring element object or array of structuring element objects, returned by STREL function. It gradually enlarges the boundaries of regions of foreground pixels (white pixels).

Our next step is processing this on a live feed. The accuracy of this method is based on the sensitivity of the camera. It has a direct relationship with the accuracy. The greater the accuracy needed, better the quality of webcam has to be used. First we need to install the webcam and then configure it to obtain the necessary video feed. The associated webcams are identified using the imaqhwinfo( ) function. Next step is to configure the webcam and assign the video properties. This involves setting the Frames per Trigger. The live feed was then obtained using the start (video object) function. To initialize an object FaceDetect, the vision CascadeObjectDetector statement was used. The next step was to crop the image such that only the face is retained for further eye detection. This is achieved by visualizing the live video feed as individual frames and then processing each frame distinctly. The vision CascadeObjectDetector is used to initialize an object EyeDetect. The video was converted to individual frames using the snapshot( ) function which returns a matrix corresponding to an RGB image. Next step involved was similar to identifying the eye region in a static image but instead of the image being stored in the computer memory, it is stored virtually in a MATLAB script. Since the snapshot( ) function works by contacting the webcam each time it is called, processing time is increased. The EyeDetect object is given as input to the step function along with the image & the values returned correspond to the X-Coordinate, Y-Coordinate, Width and Height of the eye region. The image is cropped using imcrop( ) with one input as the n cross 4 matrix and other being the image itself. The RGB image hence obtained is converted to its equivalent grayscale form using the rgb2gray( ) function. It is then converted to black and white form using the im2bw( ) function. The black and white image thus obtained is then dilated to get only the eyes. If eyes are in the left direction, it indicates output as left. If eyes are in the right direction, it indicates output as right. if the driver is looking straight, it indicates output as straight. If there is any distortion or if the eyes are closed, it must give the output as no face.

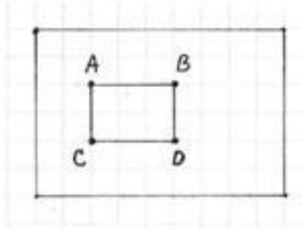
B. Viola Jones algorithm

The Viola-Jones algorithm is a widely used mechanism for object detection. The main property of this algorithm is that training is slow, but detection is fast. This algorithm uses Haar basis feature filters, so it does not use multiplications.

The efficiency of the Viola-Jones algorithm can be significantly increased by first generating the integral image.

$$II(y,x) = \sum_{p=0}^{y} \sum_{q=0}^{x} Y(p,q)$$

The integral image allows integrals for the Haar extractors to be calculated by adding only four numbers. For example, the image integral of area ABCD (Fig.1) is calculated as $II(y_A,x_A) - II(y_B,x_B) - II(y_C,x_C) + II(y_D,x_D)$.



Detection happens inside a detection window. A minimum and maximum window size is chosen, and for each size a sliding step size is chosen. Then the detection window is moved across the image as follows:

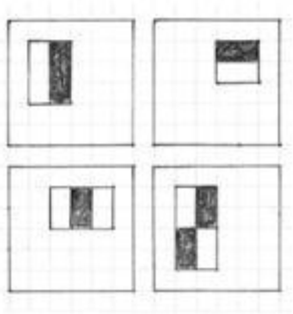
1. Set the minimum window size, and sliding step corresponding to that size.

2. For the chosen window size, slide the window vertically and horizontally with the same step. At each step, a set of *N* face recognition filters is applied. If one filter gives a positive answer, the face is detected in the current widow.

3. If the size of the window is the maximum size stop the procedure. Otherwise increase the size of the window and corresponding sliding step to the next chosen size and go to the step 2.

Each face recognition filter (from the set of *N* filters) contains a set of cascade-connected classifiers. Each classifier looks at a rectangular subset of the detection window and determines if it looks like a face. If it does, the next classifier is applied. If all classifiers give a positive answer, the filter gives a positive answer and the face is recognized. Otherwise the next filter in the set of *N* filters is run.

Each classifier is composed of Haar feature extractors (weak classifiers). Each Haar feature is the weighted sum of 2-D integrals of small rectangular areas attached to each other. The weights may take values ±1. Fig.2 shows examples of Haar features relative to the enclosing detection window. Gray areas have a positive weight and white areas have a negative weight. Haar feature extractors are scaled with respect to the detection window size.



The classifier decision is defined as:

$$C_m = \begin{cases} 1, & \sum_{i=0}^{I_m - 1} F_{m,i} > \theta_m \\ 0, & \text{otherwise} \end{cases}$$

$$F_{m,i} = \begin{cases} \alpha_{m,i}, & \text{if } f_{m,i} > t_{m,i} \\ \beta_{m,i}, & \text{otherwise} \end{cases}$$

$f_{m,i}$ is the weighted sum of the 2-D integrals. is the decision threshold for the i-th feature extractor. $\alpha_{m,i}$ and $\beta_{m,i}$ are constant values associated with the i-th feature extractor. θ_m is the decision threshold for the m-th classifier.

a.

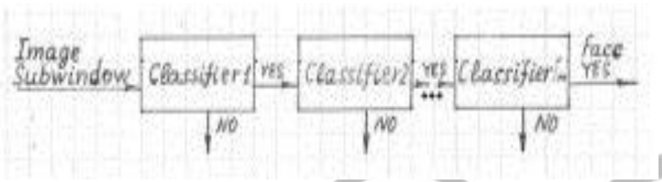
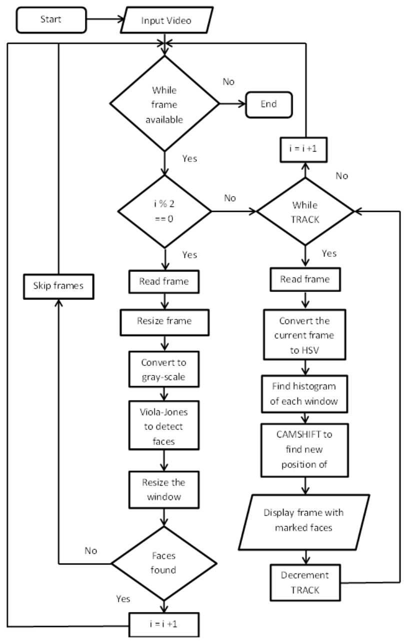


Fig: Object detection Viola Jones algorithm

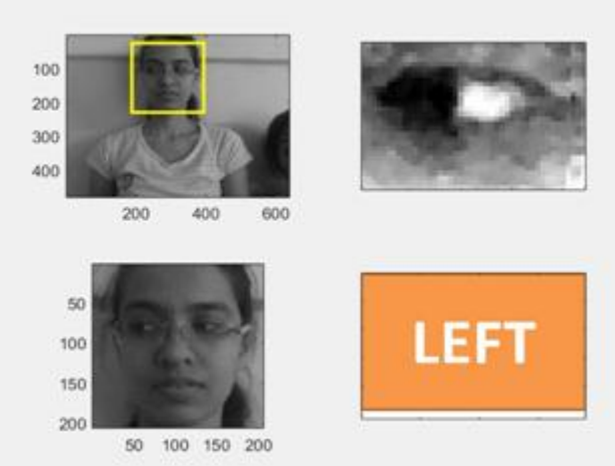
The cascade architecture is very efficient because the classifiers with the fewest features are placed at the beginning of the cascade, minimizing the total required computation. The most popular algorithm for features training is AdaBoost.



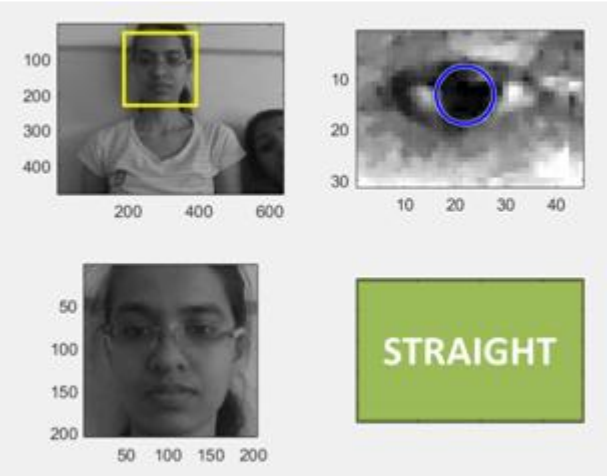
CODE

```
clear all
clf('reset');
cam=webcam();   %create webcam object
right=imread('RIGHT.jpg');
left=imread('LEFT.jpg');
noface=imread('no_face.jpg');
straight=imread('STRAIGHT.jpg');

detector = vision.CascadeObjectDetector(); % Create a detector for face using Viola-Jones
detector1 = vision.CascadeObjectDetector('EyePairSmall'); %create detector for eyepair

while true % Infinite loop to continuously detect the face

vid=snapshot(cam);  %get a snapshot of webcam
vid = rgb2gray(vid);    %convert to grayscale
img = flip(vid, 2); % Flips the image horizontally

bbox = step(detector, img); % Creating bounding box using detector

if ~ isempty(bbox)  %if face exists
biggest_box=1;
fori=1:rank(bbox) %find the biggest face
ifbbox(i,3)>bbox(biggest_box,3)
biggest_box=i;
end
end
faceImage = imcrop(img,bbox(biggest_box,:)); % extract the face from the image
bboxeyes = step(detector1, faceImage); % locations of the eyepair using detector

subplot(2,2,1),subimage(img); hold on; % Displays full image
fori=1:size(bbox,1)    %draw all the regions that contain face
rectangle('position', bbox(i, :), 'lineWidth', 2, 'edgeColor', 'y');
end

subplot(2,2,3),subimage(faceImage);     %display face image

if ~ isempty(bboxeyes)  %check it eyepair is available

biggest_box_eyes=1;
fori=1:rank(bboxeyes) %find the biggest eyepair
ifbboxeyes(i,3)>bboxeyes(biggest_box_eyes,3)
biggest_box_eyes=i;
end
end


bboxeyeshalf=[bboxeyes(biggest_box_eyes,1),bboxeyes(biggest_box_eyes,2),bboxeyes(biggest_box_eyes,3)/3,bboxeyes(biggest_box_eyes,4)];   %resize the eyepair width in half
eyesImage = imcrop(faceImage,bboxeyeshalf(1,:)); %extract the half eyepair from the face image
eyesImage = imadjust(eyesImage);    %adjust contrast

        r = bboxeyeshalf(1,4)/4;
        [centers, radii, metric] = imfindcircles(eyesImage, [floor(r-r/4) floor(r+r/2)], 'ObjectPolarity','dark', 'Sensitivity', 0.93); % Hough Transform
        [M,I] = sort(radii, 'descend');

eyesPositions = centers;

subplot(2,2,2),subimage(eyesImage); hold on;

viscircles(centers, radii,'EdgeColor','b');

if ~isempty(centers)
pupil_x=centers(1);
disL=abs(0-pupil_x);      %distance from left edge to center point
disR=abs(bboxeyes(1,3)/3-pupil_x);%distance from right edge to center point
subplot(2,2,4);
ifdisL>disR+16
subimage(right);
else if disR>disL
subimage(left);
else
subimage(straight);
end
end

end
end
else
subplot(2,2,4);
subimage(noface);
end
set(gca,'XtickLabel',[],'YtickLabel',[]);

hold off;
end
```

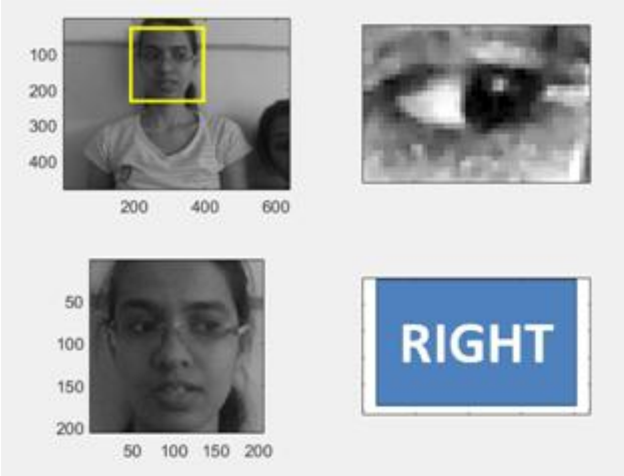
OUTCOMES



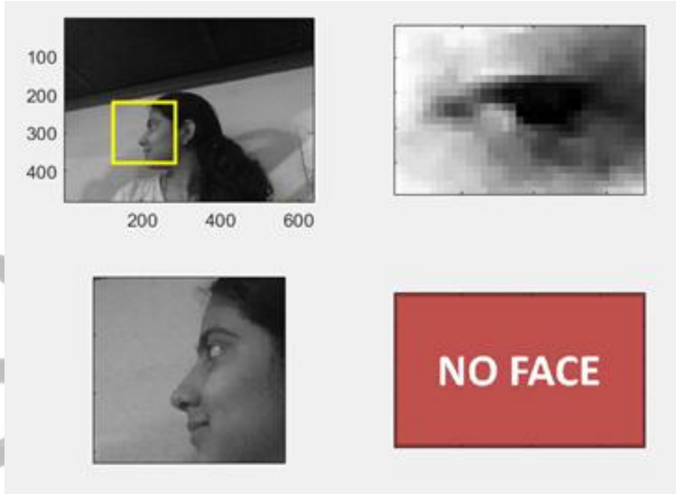
If eyes are in the left direction, it indicates output as left.



If the driver is looking straight, it indicates output as straight



If eyes are in the right direction, it indicates output as right.



If there is any distortion or if the eyes are closed, it must give the output as no face.

References

[1] Ruian Liu,et.a;, "Design of face detection and tracking system," Image and Signal Processing (CISP), 2010 3rd International Congress on , vol.4, no., pp.1840,1844, 16-18 Oct. 2010.

[2] Paul Viola, Michael J. Jones, *Robust Real-Time Face Detection*, International Journal of Computer Vision 57(2), 2004.

[3] Neeta Parmar Instructor: Peter Hiscocks, "Drowsy Driver Detection System" Department of Electrical and Computer Engineering", presented at Ryerson University © 2002**.**

[4] P. Viola and M. J. Jones, Robust real-time face detection, International Journal of Computer Vision, 57 (2004), pp. 137{154.

[5] Learning Voila Jones Algirithm by Gary Bradski and Adrian Kaehler.