THE INTERNATIONAL JOURNAL OF SCIENCE & TECHNOLEDGE

Trace Graph Generation of Eye and Head by Tracking Eye Pupil Movements and Head Movements

Thakur Navneetha Singh
Master of Technology, Computer Science & Engineering
Vasavi College of Engineering, Ibrahimbagh, Hyderabad, India

Abstract:
There are number of approaches to track eye and head movements for variety of applications. But most of the techniques and approaches used to track eye and head movements rely on Intrusive and Expensive techniques in which various devices are connected such to track head movements person is made to wear trail frame and to track eye movements some glass like devices are made used.
This paper provide the simple approach which is non-expensive and non intrusive to track eye and head movements. This system requires simple integrated web camera. This project is done by Digital Image processing and Computer Vision based techniques and algorithms in a practical approach. The main objective of this project is to design algorithm for eye and head movement tracking device. Firstly that device is made learnt about what does eye looks like and where it is located on face by using some eye and head movement tracking algorithms. Once the eye and head movements are tracked it will store the eye and head movements data in a file and simultaneously plot the trace graph of eye movements made and head movements made respectively.
.

Keywords: *Eye Tracking , Head Tracking, Eye Pupil Movements, Head Movements, Cascade Classifiers, Detect Eye, Detect Head , Motion Estimation, Motion Detection*

1. Introduction

In the field of Computer Science there are many subfields exist where eye and head movements tracking has become very important.
Eye and Head Tracking devices could be classified into two types :

- Remote and non intrusive and
- Head mounted or intrusive method.

In intrusive methods where the person is made to wear the trail frame etc the main problem is it is expensive and also it will cause un-comfort to the person. And though there are remote and non intrusive methods the problem is it cause the lack of accuracy due to inexpensive system.
The main objective of this project is to provide the simple non expensive and non intrusive approach to track eye pupil movements and head movements with accuracy.
This project will be developed in C++ and will involve computer vision technologies such as OpenCV and the use of an analytical approach called Haar-like features. The user interface is developed using the QT Framework.

1.1. Objective

The main objective of this project is to provide the simple non expensive and non intrusive approach to track eye pupil movements and head movements with accuracy.

1.2. Modules

This project is divided into two modules .

- Eye movements Tracking and Trace Generation of Eye Movements.
- Head movements Tracking and Trace Generation of Head Movement

1.3. Required Specifications

1.3.1. Hardware Requirements are

- Integrated Web Camera.
- PC.

1.3.2. Software Requirement are
- Operating System : Linux
- Simulation Tools : OpenCV
- Coding Language : CPP
- IDE : Qt Creator

2. Experimental Results

2.1. Face and Eye Detection in a live video Stream
In order to detect face and eyes using opencv there are predefined classifiers using which we can detect the face and eyes in a image or live video stream using the following steps:
- Load the cascades.
- Read the video stream
- Apply the classifier to the frame
- Detect faces in frame.
- In each face, detect eyes.
- Display the results in a window.

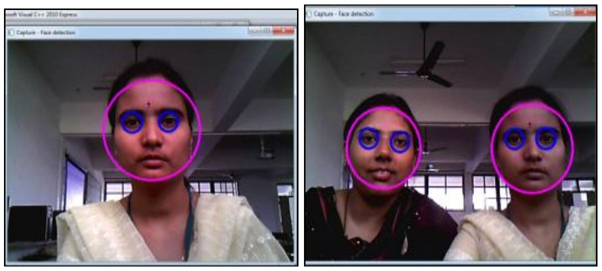


Figure 1: Face and Eye Tracking in live video

2.2. Detect Iris in the given image using Circular Hough Transform
- Read in the image as downloaded, crop out the starting image, convert to grayscale.
- Find edges. Note that Canny method incorporates Gaussian smoothing and gradient calculation, so no need to do these separately.
- Detect most prominent circle.

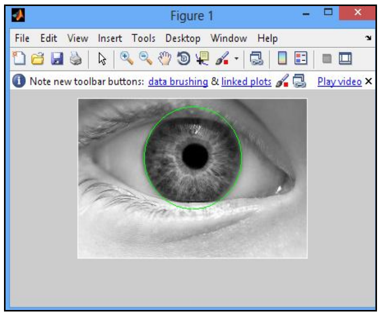


Figure 1
File Edit View Insert Tools Desktop Window Help
Note new toolbar buttons: data brushing & linked plots
Play video
Figure 2

2.3. Detect Iris and pupil with their center
- Read the image
- Convert it to gray
- Reduce the noise so we avoid false circle detection.
- Apply the Hough Transform to find the circles
  - HoughCircles(src_gray, circles, CV_HOUGH_GRADIENT, 1, src_gray.rows/20, 50,50, 0, 0);
  - HoughCircles(src_gray, circles, CV_HOUGH_GRADIENT, 1,src_gray.rows/20, 50,60, 0, 0);
- Draw the circles detected with circle center and circle outline.

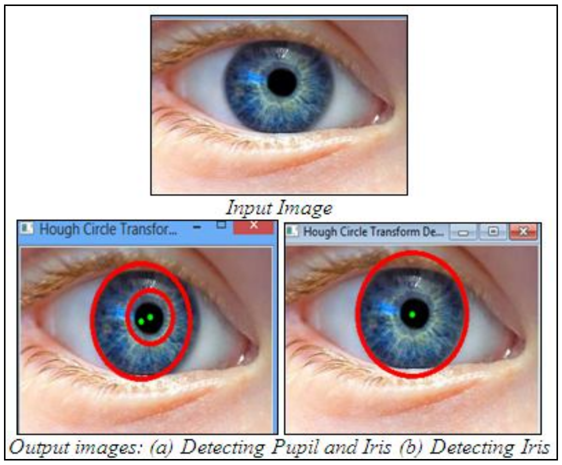

Input Image
Hough Circle Transfor...
Hough Circle Transform De...
Output images: (a) Detecting Pupil and Iris (b) Detecting Iris

Figure 3

2.4 . Detect Pupil With In The Eye Window In Live Video

- Read the frame
- Convert it to gray
- Reduce the noise so we avoid false circle detection.
- Apply the Hough Transform to find the circles cvHoughCircles(gray,storage,CV_HOUGH_GRADIENT,1,gray>height, 35, 25);
- Draw the circles detected.

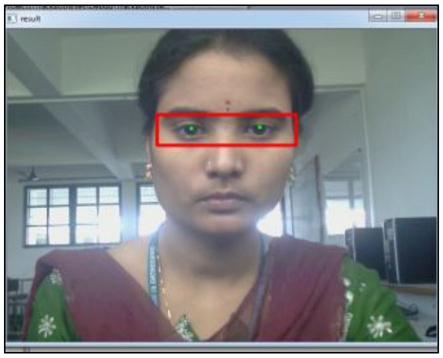


Figure 4: Tracking of Pupil in Eye window

2.5. Pupil Detection and pupil movement Trace graph generation

In order to detect pupil first we need to locate the face in captured frame then locate the possible position of the eye window in the detected face, to do this we use Haarcascade_frontalface classifier and Haarcascade_eye classifier. Once as we obtain the region of interest i.e eyes position the we use houghcircle method to detect the circle of specified radius in eye window region and detected circles and generate the trace graph of both left and right eye movements made thereafter.

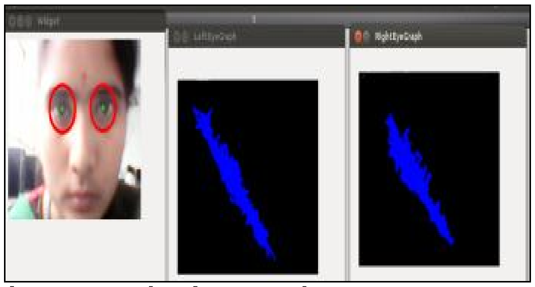
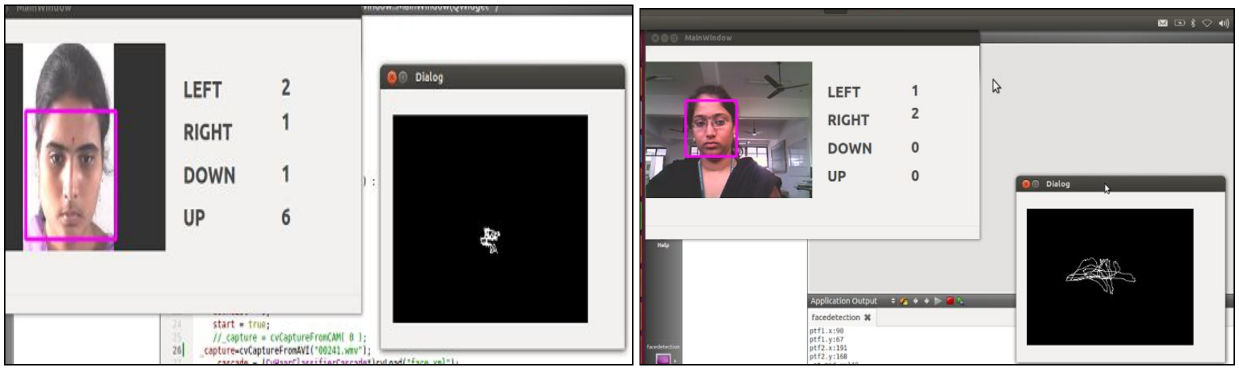


Figure 5: Tracking Eye pupil and Eye Pupil movements Trace Graph generation

In order to generate the graph we have taken a structure which use two points at a time i.e first point in first frame and second point from second frame if there is a movement then draw a line segment using cvLine() function and the processes is continued till the program is exited and if there is no eye movements then no line will be drawn. This is done for both the left and right eye and graph is as shown in the above figure.

2.6. Head Detection and head movement Trace graph generation
In the given input video detect the Face or Head using Haarcascade_frontalface classifier and indicate the detected face with a rectangle bounding box.



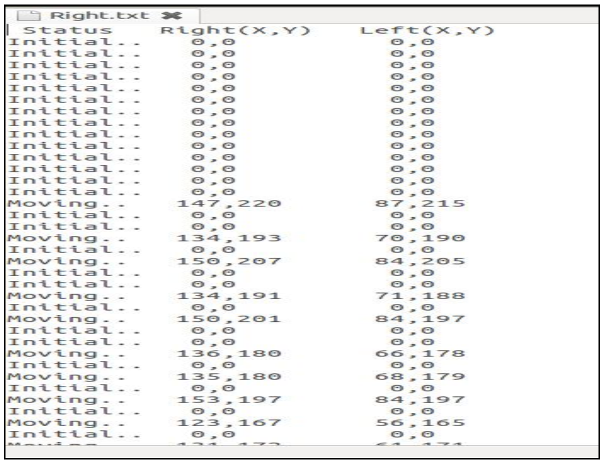
```
start = true;
//_capture = cvCaptureFromCAM( 0 );
_capture=cvCaptureFromAVI("00241.wmv");
```

```
ptf1.x:90
ptf1.y:67
ptf2.x:191
ptf2.y:168
```

Figure 6: Tracking Head and Head movements Trace Graph generation

In order to generate the trace graph of Head Movements we have taken a structure which use two points at a time i.e. first point in first frame and second point from second frame if there is a movement then draw a line segment using cvLine() function and the processes is continued till the program is exited and if there is no Head movements then no line will be drawn

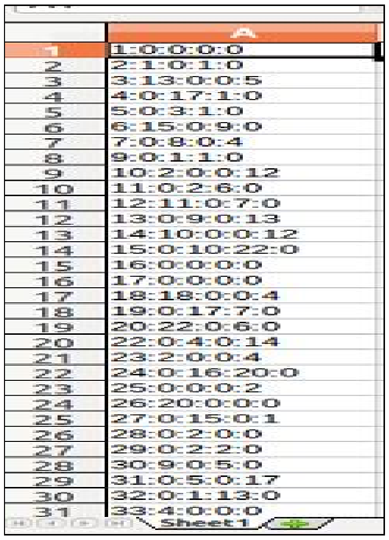
2.7. Storing the Coordinates of the Pupil Position into a File
As soon as the webcam capture the first frame and when the pupil position is detected for the first time then that position is considered as the coordinate position and thereafter we compare the other frames with initial frame if there is any movement then new coordinate point is stored in the file indicating the motion, if there is no movement then coordinated points (0,0) are stored for both Left and Right eye separately.



```
Right.txt
 Status       Right(X,Y)      Left(X,Y)
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Initial..        0,0              0,0
Moving..      147,220          87,215
Initial..        0,0              0,0
Initial..        0,0              0,0
Moving..      134,193          70,190
Initial..        0,0              0,0
Moving..      150,207          84,205
Initial..        0,0              0,0
Initial..        0,0              0,0
Moving..      134,191          71,188
Initial..        0,0              0,0
Moving..      150,201          84,197
Initial..        0,0              0,0
Initial..        0,0              0,0
Moving..      136,180          66,178
Initial..        0,0              0,0
Moving..      135,180          68,179
Initial..        0,0              0,0
Moving..      153,197          84,197
Initial..        0,0              0,0
Moving..      123,167          56,165
Initial..        0,0              0,0
```

Figure 7: Eye Movement Information File

This file contains three fields first status ,left and right eye. Status field contains two values Initial if no movement is found and moving if movement is detected.
And if Status field contains moving then left and right field contains the coordinate position of the pupil of left and right eye.
In Status field contains Initial the it means there is no movement and left and right eye contains coordinates as (0,0).

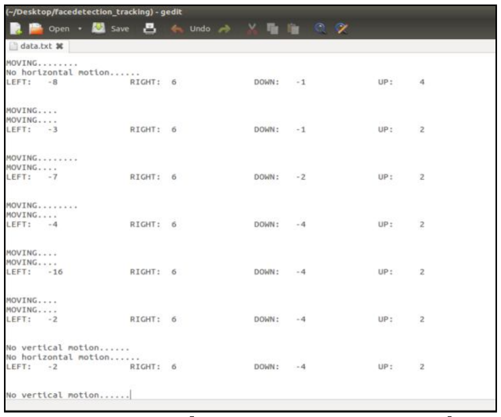


	A
1	1:0:0:0:0
2	2:1:0:1:0
3	3:13:0:0:5
4	4:0:17:1:0
5	5:0:3:1:0
6	6:15:0:9:0
7	7:0:8:0:4
8	9:0:1:1:0
9	10:2:0:0:12
10	11:0:2:6:0
11	12:11:0:7:0
12	13:0:9:0:13
13	14:10:0:0:12
14	15:0:10:22:0
15	16:0:0:0:0
16	17:0:0:0:0
17	18:18:0:0:4
18	19:0:17:7:0
19	20:22:0:6:0
20	22:0:4:0:14
21	23:2:0:0:4
22	24:0:16:20:0
23	25:0:0:0:2
24	26:20:0:0:0
25	27:0:15:0:1
26	28:0:2:0:0
27	29:0:2:2:0
28	30:9:0:5:0
29	31:0:5:0:17
30	32:0:1:13:0
31	33:4:0:0:0

Sheet1

Figure 8: Eye Movement Direction File

The above is the file that contains the following fields frame number: Left: Right: Top and Down. The field are separated by colon ":" . In Frames When ever the movement is detected then it is found that in which direction the movement is and how much distance, in that particular field value is displayed and the other fields contains 0 indicating that there is no motion in this direction. Example say if Eye has moved top left then top and left field contains some non zero value i.e the distance moved and right down contains 0.



```
MOVING........
No horizontal motion......
LEFT:   -8              RIGHT:  6               DOWN:   -1              UP:     4

MOVING....
MOVING....
LEFT:   -3              RIGHT:  6               DOWN:   -1              UP:     2

MOVING........
MOVING....
LEFT:   -7              RIGHT:  6               DOWN:   -2              UP:     2

MOVING........
MOVING....
LEFT:   -4              RIGHT:  6               DOWN:   -4              UP:     2

MOVING....
MOVING....
LEFT:   -16             RIGHT:  6               DOWN:   -4              UP:     2

MOVING....
MOVING....
LEFT:   -2              RIGHT:  6               DOWN:   -4              UP:     2

No vertical motion......
No horizontal motion......
LEFT:   -2              RIGHT:  6               DOWN:   -4              UP:     2

No vertical motion......
```

Figure 9: Head Movements Direction File

Similarly for head also Head movement direction file is generated and used to generate trace graph of head movements.

2.8. Trace Graph Generation

<u>2.8.1. Pupil Movement Trace Graph</u>

By using the Coordinates of the pupil position that are stored in a file we would like to generate the trace graph of a pupil movement as shown in a sample figure below for both the left and right eye pupil movements separately.

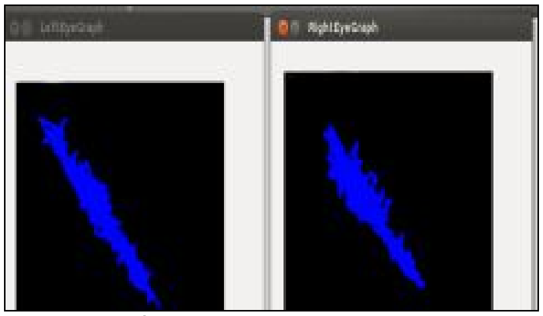


Figure 10: Left and Right Eye movements Region Map

2.8.2. Head Movements Trace Graph Generation

By using the Coordinates of the Head position that are stored in a file we would like to generate the trace graph of a pupil movement as shown in a figure below.

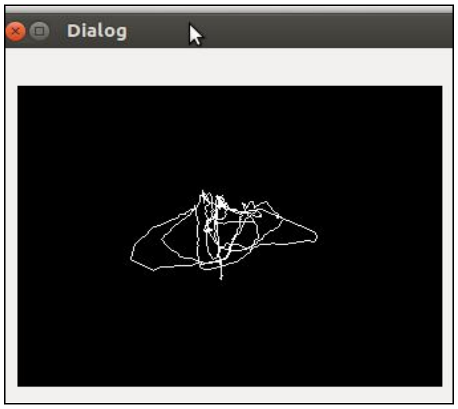


Figure 11: Head movements Region Map

3. Conclusion

The main objective of this project is to generate the Trace graph of Eye Movement based on the Eye Pupil movements made the person while observing the screen and similarly generate the trace graph of head movements made while observing the screen.

4. Future Work

This Study can be used in the design of real time Eye and Head Movement Tracking Device which can be used in the Eye sight testing centers to prescribe the patient with the appropriate shape of the lens which best fits the patient according to his or her eye and head movements made to look at particular object.

5. References

1. http://civanim.blogspot.in/2010/04/install-opencv-21-for-microsoft-visual.html
2. http://opencvfacedetect.blogspot.in/2010/10/face-detectionfollowed-by eyesnose.html
3. http://opencv-srf.blogspot.in/2011/09/capturing-images-videos.html
4. http://stackoverflow.com/questions/10506393/how-to-detect-pupil-in-matlab.
5. http://www.mathworks.in/matlabcentral/fileexchange/36137-motion-detection-in-live-video-stream
6. http://docs.opencv.org/doc/tutorials/objdetect/cascade_classifier/cascade_classifier.html
7. http://matlabsproj.blogspot.in/search/label/Iris%20Detection%20matlab
8. http://www.cvmt.dk/education/teaching/e07/MED3/IP/Simon_Pedersen_CircularHoughTransform.pdf
9. https://opencv-code.com/tutorials/eye-detection-and-tracking/
10. https://github.com/bsdnoobz/opencv-code/blob/master/eye-tracking.cpp
11. https://sites.google.com/site/learningopencv1/getting-started/example_code_7
12. https://sites.google.com/site/learningopencv1/eye-dimensions
13. http://research.microsoft.com/~viola/Pubs/Detect/violaJones_CVPR2001.pdf**.**
14. http://docs.opencv.org/doc/user_guide/ug_traincascade.html
15. http://note.sonots.com/SciSoftware/haartraining.html
16. http://www.opencv.org.cn/opencvdoc/html/modules/imgproc/doc/feature_detection.html
17. http://en.wikipedia.org/wiki/Hough_transform
18. http://docs.opencv.org/doc/tutorials/imgproc/imgtrans/hough_circle/hough_circle.html