Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Digital Image Processing Lab
CSE 4228

Eye Direction Detection System

Submitted By

Fariha Tahsin Chowdhury	(15.02.04.043)
Ashiqur Rahman	(15.02.04.057)
Anika Salsabil	(15.02.04.062)

Submitted To

Mohammad Imrul Jubair
G. M. Shahariar

Date of Submission : 16 October, 2019

Introduction

For most of us the sense of sight is the primary source of data about surrounding environment. Therefore, it is natural to assume that information about where a gaze is focused could be helpful in determining how we communicate with the surroundings.

The aim of our project is to detect the direction of eye pupil using pupil localization concept from different eye images. It can be implemented in an application of eye controlled computer operation. The effectiveness assessment of our project is based on the collection of actual pupil location images.

Features

The prime features of our project are given below:

1. Pupil localization
2. Pupil center detection
3. Marking the pupil in the image
4. Eye gazing detection of the following directions:
    a. Left
    b. Right
    c. Up
    d. Down

Flowchart

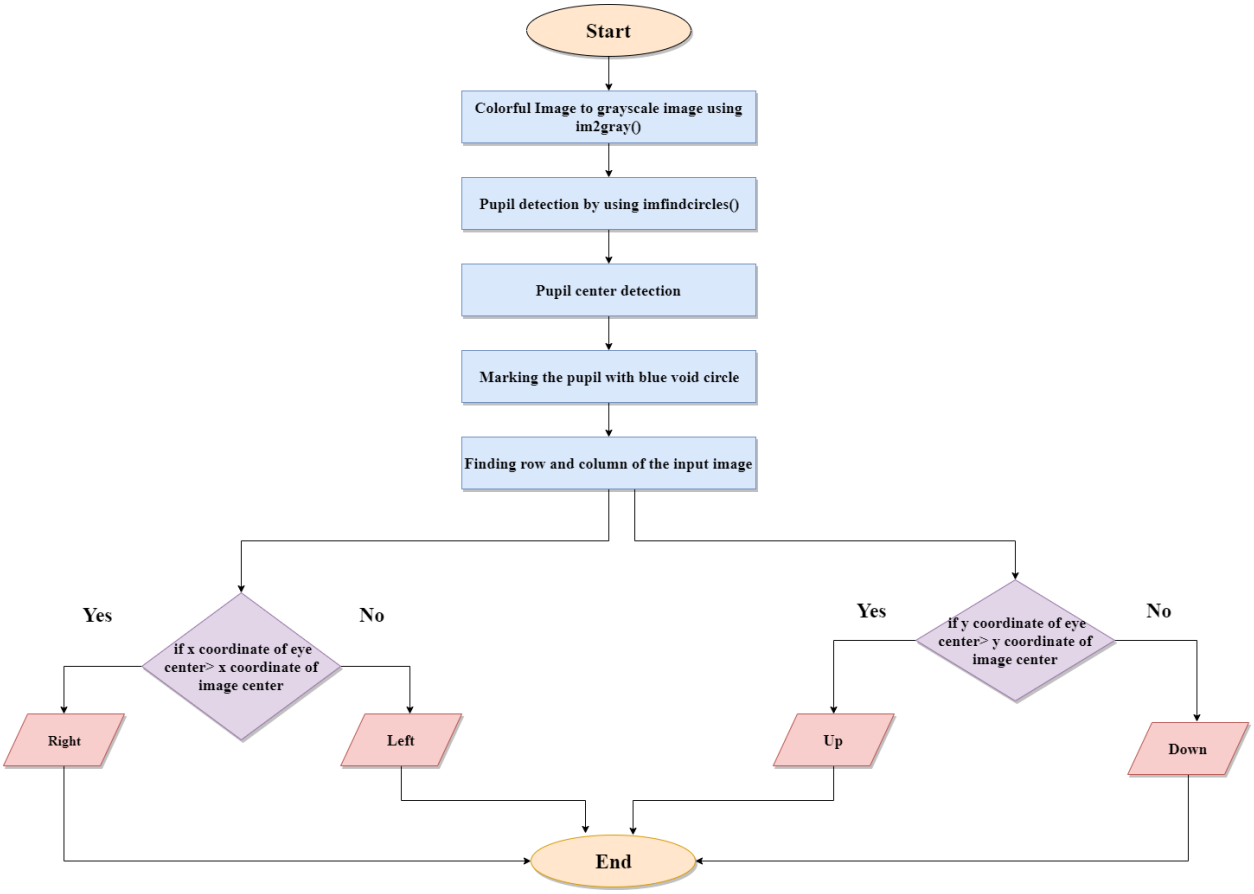


Figure: Flow Diagram of Eye Direction Detection System

Implementation

The implementation is performed by using some standard resolution images. We have implemented the following functionalities in our project. All the functionalities are given below:

1. rgb2gray( ): Converts RGB image or color-map to grayscale. This function converts the true color image **I** to the grayscale image **I** in our project. The rgb2gray function converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance.

I=imread('image/a6.jpg');

```
I=rgb2gray(I);
```

2. **imfindcircles( ):** This function finds circles using circular Hough transform in the image. We have used

```
[centers, radii, metric]=imfindcircles(Eyes,[25 60], 'ObjectPolarity',
'dark', 'Sensitivity', 0.93);
```

Here,
'Eyes' is the grayscale input image.

Range of radii for the circular objects you want to detect, specified as a 2-element vector of integers of the form [rmin rmax]. We have set the range 25 to 60.

Object polarity indicates whether the circular objects are brighter or darker than the background, specified as the comma-separated pair consisting of 'ObjectPolarity' and either of the values in the following table and 'dark' denotes the circular objects are darker than the background.

Sensitivity factor is the sensitivity for the circular Hough transform accumulator array, specified as the comma-separated pair consisting of 'Sensitivity' and a number in the range [0, 1]. If we increase the sensitivity factor, imfindcircles detects more circular objects, including weak and partially obscured circles. Higher sensitivity values also increase the risk of false detection. Here we have set sensitivity=0.93.

Centers returns the coordinates of circle centers. The estimated radii for the circle centers, returned as a column vector. The radius value at radii(j) corresponds to the circle centered at centers(j,:) and Circle strengths is the relative strengths for the circle centers, returned as a vector. The value at metric(j) corresponds to the circle with radius radii(j) centered at centers(j,:).

```
viscircles(centers, radii,'Color','b');
```

This is used to draw blue lines around the edges of the circles.

So, **imfindcircles( )** function is used to find all the dark circles in the image within the radius range and **viscircles( )** is used to draw blue edges of the circles.

3. **Pupil and Image Center Detection:** From the imfindcircles( ) method we have got the center and now we have detected the pupil and image center detection by the following bunch of code,

```
a=round(centers(1));

b=round(col/2);

a1=round(centers(2));

b1=round(row/2);
```

Here, a is the x coordinate of eye center,

b is the x coordinate of image center,

a1 is the y coordinate of the eye center

b1 is the y coordinate of the image center.

4. Conditions for Right and Left Directions: In this section we have added the condition for detecting the directions of the pupil. The code is given below:

```
if(a>b)
     subplot(2,2,3); imshow(R);

     disp(a);

    else subplot(2,2,3);

       imshow(L)

       disp(b)
```

Here,
The main condition is if the x coordinate of eye center is greater than the x coordinate of image center then is will be in the Right direction else it will detect the Left direction of the pupil.

5. Conditions for Up and Down Directions: In this section we have added the pupil detection for Up and Down. The code is attached below:

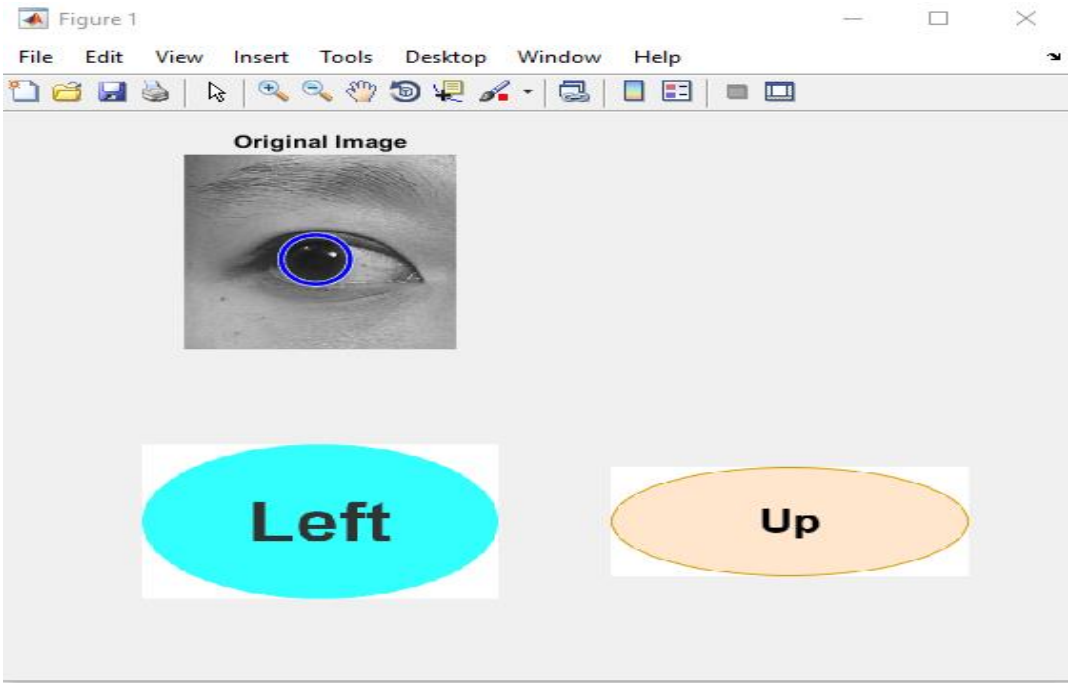
```
if(a1>b1)

     subplot(2,2,4); imshow(U);

     disp(a1);

    else subplot(2,2,4);

       imshow(D)

       disp(b1);
```

The main concept of this condition is if the y coordinate of eye center is greater than the y coordinate of image center then is will be in the Up direction else it will detect the Down direction of the pupil.

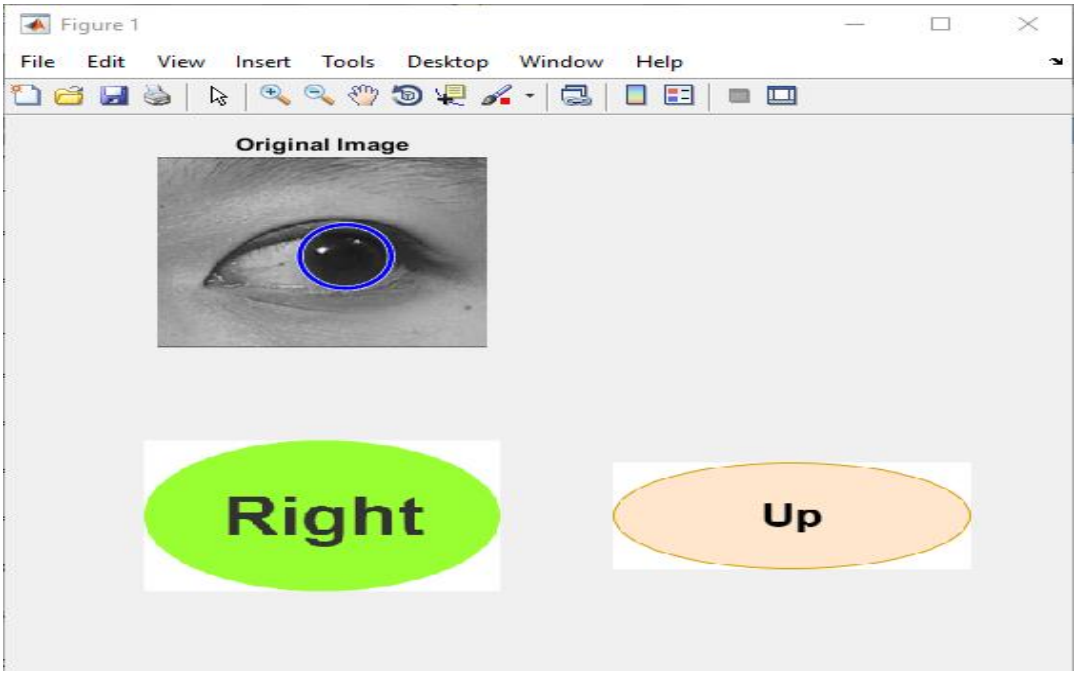
Snapshots of our Project

In this section, we have attached some screenshots of our website for a visual representation:

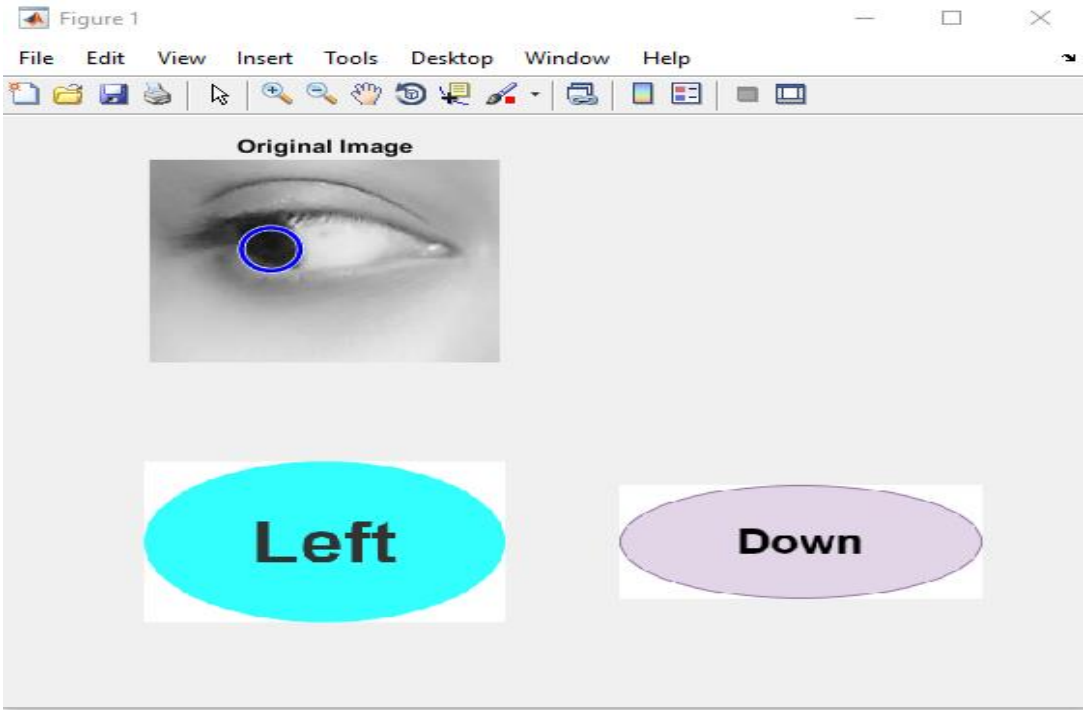
The pupil is in the Left Up direction in this image:



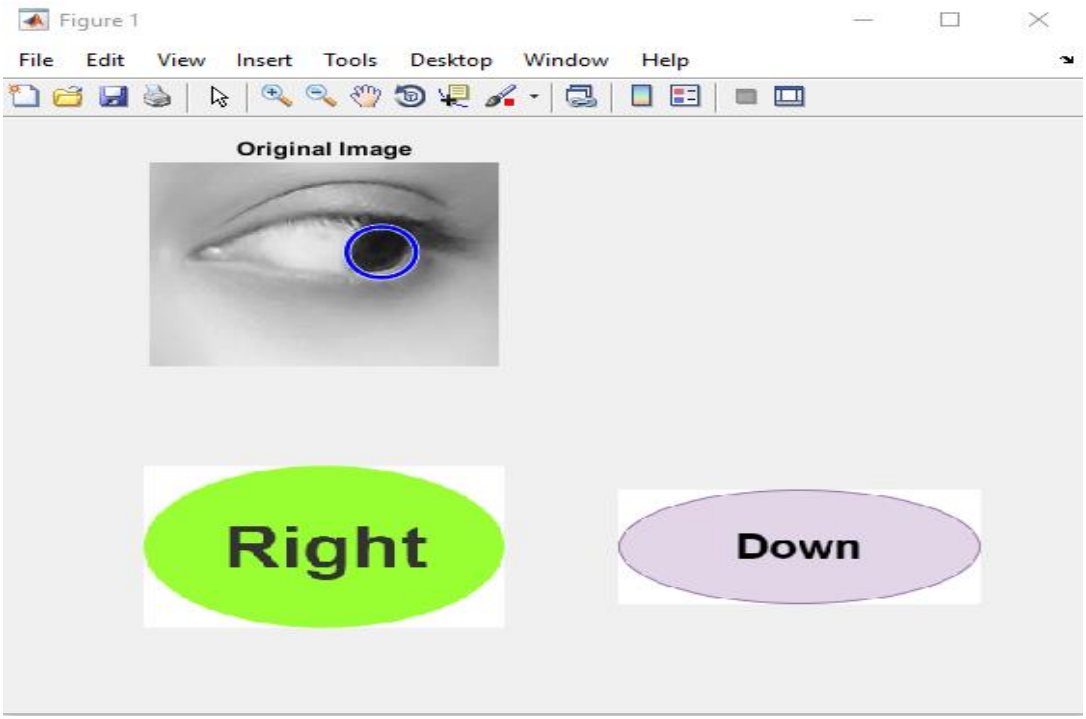
The pupil is in the Right Up direction in this image:



The pupil is in the Left Down direction in this image:



The pupil is in the Right Down direction in this image:



Contribution of the group members

Research part done by Anika Salsabil

Implementation done by Ashiqur Rahman

Report done by Fariha Tahsin Chowdhury

Applications of our Project

The pupil direction applications covers human computer interaction, brain computer interaction, assistive technology, e-learning, psychology investigation, pilot training assistance, virtual and augmented reality and so on.

Future Work

We tried our level best to make a convenient project for pupil direction detection but our future plan is to add some features like how long the person is looking in a particular direction and make it more efficient.

Conclusion

The gaze tracking system needs to become low in cost and accuracy of data capture needs to be improved in order to make them useful tools. We tried to implement the pupil direction detection project as simple as possible. So, it will be easier to implement in many eye tracking applications.