Driver›s Drowsiness Fatigue Detection System Based on Eye Gaze Estimation Scheme

نظام الكشف عن إرهاق النعاس للسائق بناءً على مخطط تقدير نظرات العين

Assist.Lecturer Noor Hayder Abdul Ameer1

110048@uotechnology.edu.iq

Assist.Lecturer Nada Najeel Kamal2

110136@uotechnology.edu.iq

University of Technology- Iraq, Department of Computer Sciences

Abstract

The accidents of traffic produce a state of disquiet to the individuals over the world, affect the main life component which is the human, and exhaust the material resources. Therefore, the proposals and solutions should be found to decrease the traffic accidents or at least determine the reasons and decrease the passive-effects and identify the core issues that lead to the traffic accidents occurrence, such as driver's drowsiness and fatigue, road problems, vehicle breakdown, and etc. These issues need to develop effective safety systems capable of decreasing accidents by warning the driver under the various bad driving situations. Eye gaze estimation systems are one of the fundamental technologies for the upcoming driving assistance systems. This paper proposes a low-cost, passive and remote eye gaze estimation scheme to monitor the eye gaze direction of the driver in real-time, in addition to a perfect audible sound for alerting the driver. The proposed eye gaze estimation scheme shows a greater reliability, and accuracy.

Keywords: *Driver's drowsiness-fatigue detection, face detection, eye region detection, eye gaze estimation, sound alerting.*

1. Introduction

Drowsiness is a certain state between awake and being sleepiness. In this state, the driver's vigilance and attention to the road are decreased. The rising of fatal traffic accidents owing to a drivers' drowsiness and fatigue is a dangerous issue for society. In order to avoid these accidents, vehicles-driver intelligent systems are required which are capable of detecting driver's drowsiness-fatigue and sending an alert.

Recently, various schemes for developing effective driving assistance systems have been presented. Most of these schemes are based on measuring the physiological situations such as brain waves, heart rate, and etc. for detecting maximum accuracy. But these schemes are not comfortable (intrusive) since they require to wear sensing elements to a driver's body [1].

Driver's drowsiness-fatigue can also be noticed with ease by specific visual attitudes, and changes in their facial features such as driver yawning, eye closure, the slow movement of the eyelid, eye gaze, and head movement. [2]. Thus based on these visual behaviors, other schemes to deliver good results are appeared which are less intrusive such as the head or eye gaze movement monitoring schemes based on wearable or unwearable devices [1].

In this paper, a driver's drowsiness-fatigue detection system based on eye gaze estimation scheme is proposed.

ISSN: 2075 - 2954 (Print)

The proposed scheme firstly detects the driver's face from a real-time video sequence, then determines the eye region and finally estimates the eye gaze direction. The major concentrate is to utilize this eye gaze for detecting the driver's drowsiness-fatigue whilst driving to reduce the vehicle accidents happening daily. This scheme is implemented on the Arduino board for creating a comfortable mobile system. The driver's eyes has been closed the direction of driver's eye gaze is out of the road for a time more than two seconds, then the driver's drowsiness-fatigue is detected and the driver is warned by an audible sound. The paper remainder has been constructed , in the second section, the related works are presented, the proposed scheme is presented in the third section. Section four focuses on experimental results. Finally, the conclusions and future work are presented in section five.

2. Related Works

In order to improve the acceptance and comfort of the driver's drowsiness-fatigue detection system, considerable researchers have exhausted a lot of effort in studying the driver's driving features in particular situations. The number of publications in the field of driver drowsiness and fatigue detection system at "Institute of Electrical and Electronics Engineers (IEEE) Explore website", and "Science Direct website" for the years 2009-2018.

Some researchers employ data obtained from physiological sensors like electrocardiogram (ECG) [3], and electroencephalogram (EEG) data [4, 5, 6]. The ECG signals point out to the heartbeats dynamics which are beneficial for studying the variability of heart rate during the state of drowsiness and fatigue. In a similar way, EEG signals are capable of measuring the neuronal changes happening owing to drowsiness and fatigue and can quantitatively be analyzing these signals for determining beneficial information for estimating the brain states [7]. Systems based quantitative features provided high rates of accuracy. Nevertheless, they are intrusiveness.

On the other hand, nonintrusive systems for bio-signals are much less precise. lots of researches work on extraction the facial features. H. Singh et al. [8] proposed a system for detecting eye movement to detect driver's fatigue early enough to avoid an accident. According to Xuesong W. and Chuan X. [9]. The 23 nonintrusive metrics besides eye features and driving behavior metrics were evaluated depending on a high fidelity driving simulation in a controlled environment. This model has a high accuracy. Jibo He et al. [10] utilized an application of Google Glass to detect driver's eye blinks. This glass has many benefits like GPS navigation and texting. Nevertheless, the glass based drowsiness detection scheme is restricted with the fast battery consumption and the difficulty of eye blinks detection when the driver put sunglasses or lenses. Rateb Jabbar et al. [11] proposes an

 ISSN: 2075 - 2954 (Print)

embedded drowsiness detection system based on the facial landmark that extracted from the driver's image and the acquired data is delivered to the trained model for identifying the state of the driver. The size of the model is small and the accuracy rate is relatively good. F. Vicente et al. [12] proposed a system can detect the head pose and the eye gaze direction (if it is out of the road) along the day with the consideration of various situations like wearing glasses or age. Practically, there is no excellent warning scheme is applicable to the drivers' drowsiness-fatigue detection since every proposed scheme has different restrictions in the concepts of flexibility, effectiveness, implementation, or cost.

3. The Proposed Scheme

The proposed scheme includes four fundamental parts, namely the image pre-processing, the face detection part, the eye region detection part, and finally, the eye gaze estimation part. The main components of the proposed scheme are given in figure [2].

In this paper a simple and reliable face detection module to simplify the working on eyes has been developed . Based on the location of the detected face, the eyes location is able to be found out easily. After that, the eye analyzing process is going to determine whether the eye gaze is looking forward to the center, left, right, up, or down of the camera.

3.1 Image Pre-Processing

The first part of the proposed scheme is the image pre-processing which consists of several steps as follows;

Step 1: Input an image.

Step 2: Color spaces used for skin modeling. It is a difficult problem because there are a wide variety of skin tones, and many background colors may be misinterpreted as skin. In this step, the color input images have been converted into the HSV (Hue, Saturation, and Value) color space to help for finding the skin area in the next step. The following equations have been defined the transformation of RGB to HSV as:

$$\begin{cases} H = arccos\dfrac{\frac{1}{2}\left((R-G)+(R-B)\right)}{\sqrt{\left((R-G)^2+(R-B)(G-B)\right)}} \\ S = 1-3\dfrac{minimum\,(R,G,B)}{R+G+B} \\ V = \dfrac{1}{3}(R+G+B) \end{cases} \quad \ldots\ldots\ldots\ldots\ldots\ldots \ (1)$$

Step 3: Use the skin color bounding rules. In this step, the non-skin color is separated from the image. So, the image may contain only segmented color skin areas. Here, the face will be isolated from the background and the image will be converted into a black and white image based on the following specified rules:

$$If\ (H>0)\ \&\ (H<40)\ \&\ (S>30)\ \&\ (S<160)\ \&\ (V>150)\ \&\ (V<255)\ Then\ HSV_{Skin}=255, Else\ HSV_{Skin}=0 \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ 2)$$

ISSN: 2075 - 2954 (Print)

Step 4: Apply median filter to remove the noise from skin regions.

3.2 Face Detection Part

The several processes that are necessary for detecting the face and preparing it for the next part as:

Step 1: Find the localization points (left, right, top and bottom) for the face. To extract the region of the face, the first thing in this process is finding the skin area for the face by searching for the first white point from both sides (left, right) and also (top, bottom) sides and save the location of the four points for using it in the next step. This step is illustrated in algorithm 1:

Algorithm 1. Point Localization Left & Right, Top & Bottom
Input: *Filtered Bounded face* ***call(****Abmp)* ***Output:*** *The (left, right) and (top, bottom) points*
Step 1 *MaxLeftX ← 0 : MinRigthX ← Abmp.Width* *MaxTopy ← 0 : MinBottomy ← bmp.Height* ***Step 2*** *Point Localization Left Right* *for all pixle in image do* *2-1 Get PixelValue* *2-2 If (Rred =255) Then* *2-2-1 If(Rows >=Max_LeftX)Then Max_LeftX←I* *2-2-2 Max_Lefty←j* *2-3-3 ForAll (krow=Abmp.Width-1; krow>0; krow--)* *If (Rred=255) Then* *If(krow<=Min_RigthX) Then Min_RigthX←krow* *Min_Rigthy←Coll* *Endif* *Endfor* *Endif* *Endfor* ***Step3*** *Point Localization Top Bottom* *for all pixle in image do* *3-1 Get PixelValu* *3-5 If (Rred=255) Then* *3-5-1 If (j-col >=Max_Topy) Then Max_Topy←j-col* *3-5-2 Max_Topx←i-row* *3-5-3 ForAll (k-col=Abmp.Height-1; k-col>0; k-col--)* *If (Rred =255) Then*

```
                    If(k-col<=Min_Bottomy) Then (MinBottomy=k-col)
                    Min_Bottomy =i-row
                  Endif
            3-5-4 Endfor
         3-6 Endif
      Endfor
```

Step 2: Find the correct face region by removing the extra un-useful areas. The skin area in the face can be found by searching on both sides (left, right) and (top, down) of the image to find the ratio between the white color and the black color areas, so if the white color ratio is larger than the black, it should be the points of finding the correct region on both sides of the face. After that, the face region can be cropped based on the found points around the face. This process performed in order to get the first initial area of the face by exploiting the four points around the face then the X and Y points have to be found at those points.

Step 3: Crop the face region based on the founded (X, Y) points around the face. The area of the face should be easy detected and cropped, and this process is very useful to get an accurate face area to be used in the next steps.

Step 4: Apply the Gaussian Filter. This step will slightly smooth the image to reduce the effects of obvious noise on the detected face.

Step 5: Convert the Face region image into the grayscale image.

Step 6: Find the contrast of the grayscale image. This process is the most important aspect to find and reach the eye

ISSN: 2075 - 2954 (Print)

region in the face image in the next part.

3.3 Eye Region Detection Part

The output of the face detector, the eye region detection part represent in algorithm 2 as:

Step 1: Find the eyes region based on geometrical ratios. This step is demonstrated in the following algorithm;

Algorithm 2. Finding The eye Region
Input: *The Face Region Image* ***Output:*** *The Eyes Region Image*

Step1

```
cryMidleft
Midx   = ABmp.Width \ 2
 Midy   = ABmp.Height \ 2
 xleft   = Integer.MinValue
lenBlack   = 0:  MovPoit   = 0:        xBoolen = False
For All (j = 0; j< ABmp.Height-1; j++)
     MovPoit = 0: xBoolen = False:   lenBlack = 0
      For All (i = Midx;i< (ABmp.Width - (Midx / 2)) - 1;i++)
           PixelValueA = Abmp.GetPixel(i, j)
              BblueA = PixelValueA \ (MaxColor ^ 2)
              GgreenA = (PixelValueA - BblueA * MaxColor ^ 2) \ MaxColor
               RredA = (PixelValueA - BblueA * MaxColor ^ 2 - GgreenA * MaxColor)
             PixelValueB = Abmp.GetPixel(Midx - MovPoit, j)
              BblueB = PixelValueB \ (MaxColor ^ 2)
              GgreenB = (PixelValueB - BblueB * MaxColor ^ 2) \ MaxColor
               RredB = (PixelValueB - BblueB * MaxColor ^ 2 - GgreenA * MaxColor)
         If Not (RredA = 255 And RredB = 255) Then
               If (RredA= 0    And RredB = 0) Then
                    xBoolen = True
                    lenBlack = lenBlack +1
```

```
                    Else If (RredA = 0 And RredB = 255) Then If xBoolen Then Exit For
                Else If xBoolen Then Exit For
                MovPoit = MovPoit + 1
              End for
         End for
         MidWy = Midx – xleft
Step 2   cryMidRight
         Midx = im.Width \ 2
          Midy = im.Height \ 2
         xRight = Integer.MinValue
         lenBlack = 0: MovPoit = 0
          xBoolen = False
         For All (j = 0; j< ABmp.Height-1; j++)
             MovPoit = 0:  xBoolen = False:          lenBlack = 0
             For All (i= Midx;i< (im.Width - (Midx / 2)) - 1;i++)
                  PixelValueA = Abmp.GetPixel(i, j)
                    BblueA = PixelValueA \ (MaxColor ^ 2)
                    GgreenA = (PixelValueA - BblueA * MaxColor ^ 2) \ MaxColor
                     RredA = (PixelValueA - BblueA * MaxColor ^ 2 - GgreenA * MaxColor)
                  PixelValueB = Abmp.GetPixel(Midx - MovPoit, j)
                   BblueB = PixelValueB \ (MaxColor ^ 2)
                    GgreenB = (PixelValueB - BblueB * MaxColor ^ 2) \ MaxColor
                   RredB = (PixelValueB - BblueB * MaxColor ^ 2 - GgreenA * MaxColor)
                 If Not (RredA = 255 And RredB = 255) Then
                        If (RredA = 0 And RredB = 0) Then
                           xBoolen = True:   lenBlack= lenBlack +1
                        Else If (RredA = 255 And RredB = 0) Then
                              If xBoolen Then Exit For
                        End If
                     MovPoit = MovPoit +1
            End for
              xRight = Math.Max(xRight, lenBlack)
         End for
         MidWy = Midx – xRight
```


ISSN: 2075 - 2954 (Print)

Step 2: Split the rectangular eyes region, by checking in two sides and finding the exact middle of this region. Then, crop the clear eye (master) and find its coordinates.

3.4

Eye Gaze Estimation Part

This is the final part of the proposed scheme, which is exploiting the output of the previous steps to determine the direction of the estimated eye gaze, represented in algorithm 3 as:

Step 1: Convert the cropped eye region image into the grayscale image.

Step 2: Apply the histogram algorithm on the selected eye region.

Step 3: Apply the K-mean clusters to collect the darkest points in the eye region image into two classes. The biggest cluster is representing the pupil and the center of the pupil. As demonstrated in the pseudocode 3.

Step 4: Use the circle equations to find the angle of the eye direction depending on the center of the pupil's cluster class. As demonstrated in pseudocode 4.

Algorithm 3. Pupil or iris center	
Input: *The Histogram from Step 2* ***Output:*** *x, y*	
Step1	***Size of hisgram =30***
Step2	***Convert All Point in the histogram to xx, yy*** ***Maxx=0: maxy =0*** *For all (i=0; i< **Size_of_hisgram**; i++)* *If maxx< xx(i) then maxx=xx(i)* *If maxy <yy(i) then maxy=yy(i)* *End for*
Step3	*totalData=0* *CounPixelCuster1=0* *CounPixelCuster2=0* *For all (i=0; i< size_of_hisgram; i++)* *totalData = totalData + 1* *Data (0 To 2, 1 To totalData)* *If i = 0 Then* *Data (1, totalData) = Maxx* *Data (2, totalData) = Maxy* *Else* *Data (1, totalData) = xx (i - 1)* *Data (2, totalData) = yy (i - 1)* *End If* *Call kMeanCluster (Data, numCluster, Centroid) 'do k-mean clustering* *For all (j=0; j< totalData; j++)* *TyperCluster = Data (0, j)* *X = Data (1, j)* *Y = Data (2, j)* *If TyperCluster = 0 Then* *CounPixelCuster1= CounPixelCuster1 + 1* *Else* *CounPixelCuster2 = CounPixelCuster2+ 1* *End If* *End for* *End for*
Step4	*Xclusrer1== Centroid (1, 1): Yclusrer1== Centroid (1, 2)* *Xclusrer2== Centroid (2, 1): Yclusrer2== Centroid (2, 2)* *If CounPixelCuster1> CounPixelCuster2 then* *Return Xclusrer1, Yclusrer1* *Else* *Return Xclusrer2, Yclusrer2*

Algorithm 4. Eye's direction	
Input: *Xecnter,Ycenter, Xclusrer, Yclusrer* ***Output:*** *Angle*	
Step1	*DistanceBetween = Sqr ((Xcenter - Xclusrer) ^ 2) + (Ycenter - Yclusrer) ^ 2))* *Horizontal = Abs (Xclusrer - Xcenter)* *Vertical = Abs (Ycluster - Ycenter)* *XDistanceBetween = Sqr((Horizontal * Horizontal) + (Vertical * Vertical))*
Step 2	*For all (i=0; i< **360**; i++)* *xx = Xcnter + Maxradius * Cos (i * pi / 180#))* *yy = Ycenter + Maxradius*Sin (i * pi / 180#))* *If xx = Xclusrer And yy = Yclusrer Then* *Angele = i* *Exit For* *End If* *End for*

ISSN: 2075 - 2954 (Print)

4. Experimental Results

The requirements of the eye gaze estimation system include hardware tools; these tools will speed up the system performance and have a positive effect on this performance, especially when it has a high specification. The system has many steps executed sequentially, start by image acquisition and ends with eye gaze estimation as illustrates in the next tables and figure 3. In addition, the scheme implementation is tested in a simulation environment platform.

In order to evaluate the performance of the proposed eye gaze estimation scheme with fixed head movements, the term of accuracy has been used. The degree of angular (*Da*) is a common measurement for finding the accuracy of the proposed system. *Da* can be determined as:

$$Da = arctan\left(\frac{dis_d}{dis_g}\right) \quad \dots\dots\dots\dots (3)$$

Where dis_d represents the distance between the detected position of eye-gaze and the ones in the real; and dis_g represents the distance between the subject and the screen plane. The lower Da measurement is, the greater is the accuracy of the eye-gaze detecting. The average accuracy that is obtained for the proposed system is about 1º.

Also, the detection success count (DSC) is utilized as the number of correctly detected faces over the actual number of faces images.

$$DSC = \frac{no.of\ correctly\ detected\ faces}{Total\ number\ of\ faces} \quad \dots\dots\dots (4)$$

The proposed scheme achieved high detection rates (93.33%) to localize and extract a human face clip.

Conclusions

In this paper, the major concentrate is to utilize eye gaze estimation scheme for detecting the driver's drowsiness-fatigue whilst driving to reduce the vehicle accidents happening daily. This scheme has been implemented on the Arduino board for creating a comfortable mobile system. If the driver's eyes are closed and the direction of driver's eye gaze is out of the road for a time more than two seconds, then the driver's drowsiness-fatigue is detected and the driver is warned by an audible sound. Best results can be acquired when the camera and light source are placed in front of the driver. The proposed scheme shows a great accuracy with high achieved detection rates. In future work, we will focus on detecting other facial features like the head movement.

References

[1] Sayani Ghosh, Tanaya Nandy and Nilotpal Manna, "Real Time Eye Detection and Tracking Method for Driver Assistance System", Advancements of Medical Electronics, Proceedings of the First International Conference, ICAME, pp. 13-25, 2015.

[2] Qiang Ji, Xiaojie Yang,-Real-Time Eye, Gaze, and Face Pose Tracking for Monitoring Driver Vigilance, Real-Time Imaging, Vol. 8, No. 5, pp. 357-377, 2002.

[3] C. V. Anilkumar, M. Ahmed, R. Sahana, R. Thejash-

 ISSN: 2075 - 2954 (Print)

wini and P. S. Anisha, "Design of drowsiness, heart beat detection system and alertness indicator for driver safety," 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, 2016, pp. 937-941.

[4] S. M. R. Noori and M. Mikaeili, "Detecting driving drowsiness using EEG, EOG and driving quality signals," 2015 22nd Iranian Conference on Biomedical Engineering (ICBME), Tehran, pp. 210-215, 2015.

[5] I. Belakhdar, W. Kaaniche, R. Djmel and B. Ouni, "Detecting driver drowsiness based on single electroencephalography channel," 2016 13th International Multi-Conference on Systems, Signals & Devices (SSD), Leipzig, pp. 16-21, 2016.

[6] Jichi Chen, Hong Wang, Chengcheng Hua, "Assessment of driver drowsiness using electroencephalogram signals based on multiple functional brain networks", International Journal of Psychophysiology, Vol. 133, pp. 120-130, 2018.

[7] Aamir Saeed Malik, Hafeez Ullah Amin,"Chapter 13 - Drivers' Drowsiness", Designing EEG Experiments for Studying the Brain, Academic Press, pp. 181-190, 2017.

[8] H. Singh, J. S. Bhatia and J. Kaur, "Eye tracking based driver fatigue monitoring and warning system," India International Conference on Power Electronics 2010 (IICPE2010), New Delhi, 2011, pp. 1-6.

[9] Xuesong W., Chuan X., "Driver drowsiness detection based on non-intrusive metrics considering individual specifics", Accident Analysis &

Prevention, Vol. 95, Part B, pp. 350-357, 2016.

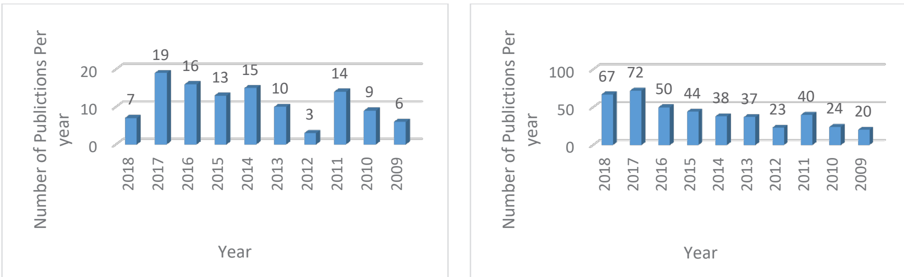
[10] Jibo He, William Choi, Yan Yang, Junshi Lu, Xiaohui Wu, Kaiping Peng, "Detection of driver drowsiness using wearable devices: A feasibility study of the proximity sensor", Applied Ergonomics, Vol. 65, pp. 473-480, 2017.

[11] Rateb Jabbar, Khalifa Al-Khalifa, Mohamed Kharbeche, Wael Alhajyaseen, Mohsen Jafari, Shan Jiang, "Real-time Driver Drowsiness Detection for Android Application Using Deep Neural Networks Techniques", Procedia Computer Science, Vol. 130, pp 400-407, 2018.

[12] F. Vicente, Z. Huang, X. Xiong, F. De la Torre, W. Zhang and D. Levi, "Driver Gaze Tracking and Eyes Off the Road Detection System," in IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 4, pp. 2014-2027, 2015.

ISSN: 2075 - 2954 (Print)

Figure



(a) (b)

Figure 1: The publications distribution data between the years 2009-2018 have been retrieved from (a) "IEEE Explore website"; (b) "."Science Direct website".

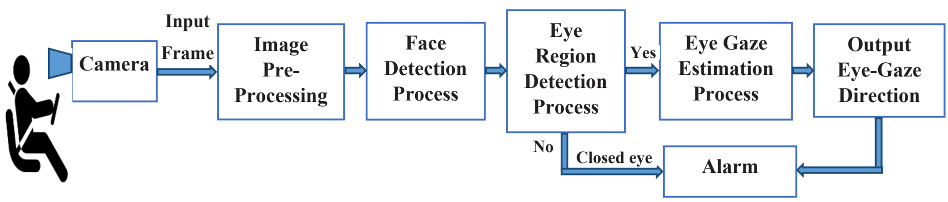


Figure (2): The main components of the proposed scheme.

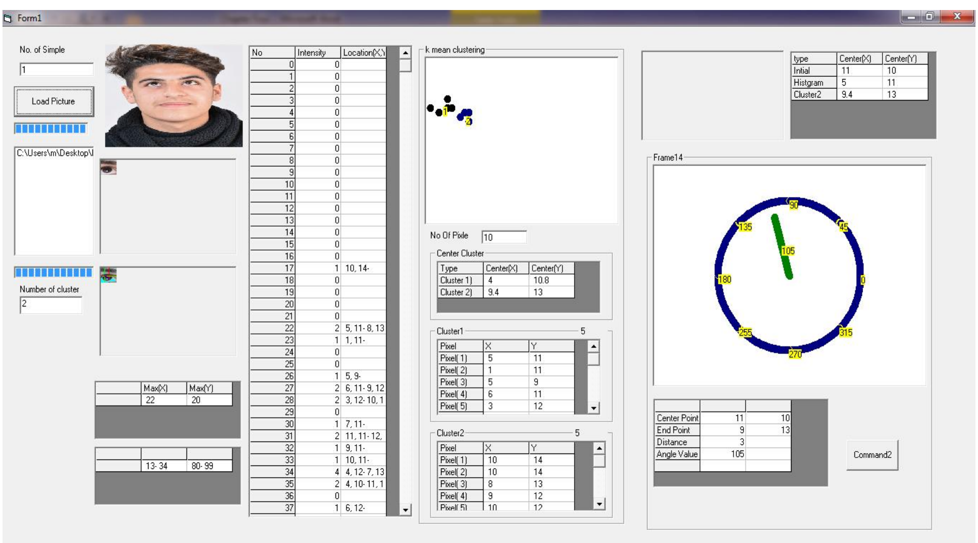
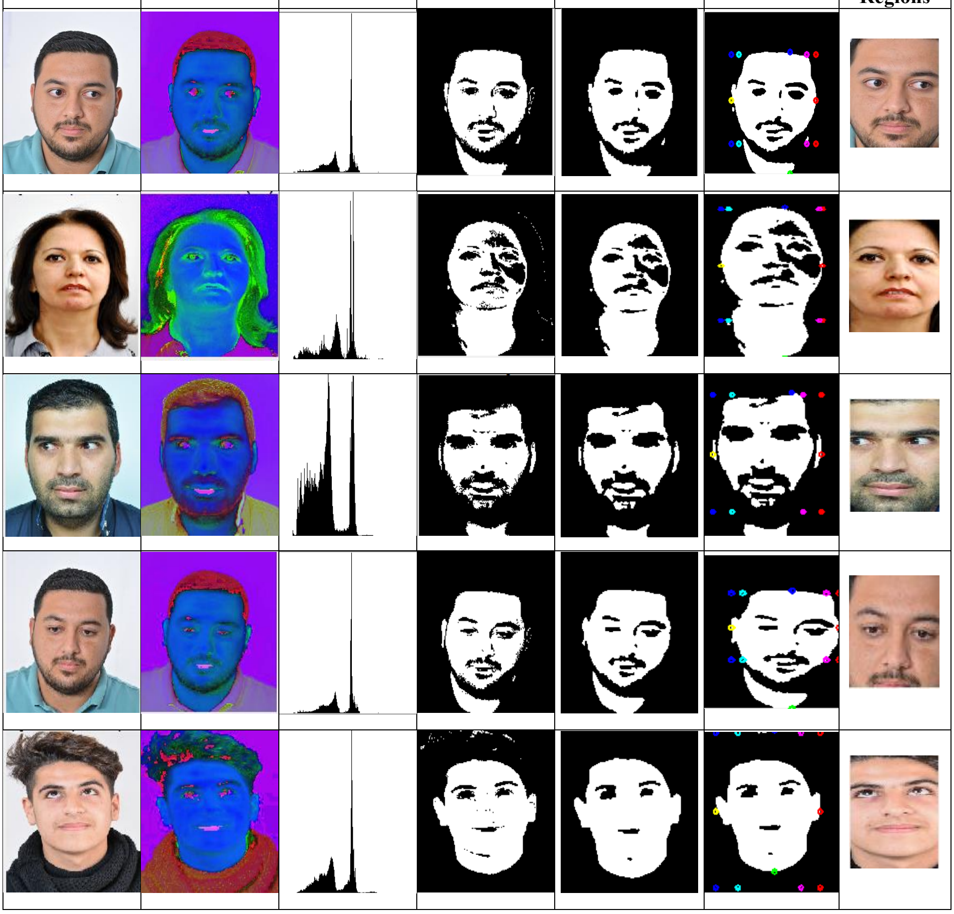


Figure (3): The overall system processes to estimate the angle of the eye gaze.

Table :

Table (1): The main processes for the 1st and 2nd steps of the proposed scheme.

The Original Images	HSV Conversion Process	The Histogram for HSV Images	HSV Skin Color Region	HSV filtered Based on Median Filter	The Correct Points (X & Y)	The Cropped Face Regions



ISSN: 2075 - 2954 (Print)

Table (2): The main steps for finding the clearest eye region.

Applying Gaussian Filter	Grayscale	Contrast Grayscale	Finding Eyes Region	Finding Clearest eye