International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 10 | October - 2022 Impact Factor: 7.185 ISSN: 2582-3930

DRIVER DROWSINESS DETECTION SYSTEM

Ms. JAYA DHARSHINI.G
Student, Department of Computer Science and Engineering, Ramco Institute of Technology

ABSTRACT

A technology for detecting driver sleepiness is intended to warn them, preventing accidents. If the driver is fatigued or not, the detection mostly relies on the outcome of the face/eye detection or the pupil of the eye. The majority of earlier research make the assumption that the driver does not wear glasses. However, spectacles can readily affect eye detection, which lowers the proper detection ratio. In this study, a tiredness detection approach with eyeglass removal is suggested in order to counteract the impact of the spectacles. First, the OpenCV library's functions are used to identify the facial area. Then, utilising morphological processes, the spectacles are taken out. Following the removal of spectacles, the eye regions are found using the OpenCV library's functions, and they are tracked by utilising a template matching technique. The horizontal projection and Kalman filter are used to achieve the binarization result of the ocular area. Next, the open/closed state of the eyes is established, and based on the series state of the eyes, weariness is established. The effectiveness of the suggested strategy is assessed using four testing movies. The fatigue detection rate can approach 100%, while the average correct detection ratio for eye state is 88.5 percent. The proposed method is workable, according to the preliminary results.

Keywords—Face detection, Eye detection, Eye tracking, Kalman filter, Template matching

CHAPTER 1
INTRODUCTION

1.1.INTRODUCTION

Long-distance driving can make a person quickly exhausted, which increases the likelihood of a traffic collision. Because it alerts the driver, the fatigue detection method is helpful in preventing auto accidents. Real-time tiredness detection is required. The primary area of focus for fatigue detection right now is the eyes. The motorist is fatigued if their eyes close for an extended period of time.

Face identification is thus the first step in the fatigue detection process. It uses the "integral image," a robust face detection method devised by Viola and Jones that is based on an encoded image. Then, using the face area, the eyes area is found. In this report, a method is proposed to detect the fatigue of the driver. The method can remove the eyeglasses in the real-time image to improve the accuracy of the fatigue alarm and reduce the number of false alarms.

The rest of the paper is organized as follows. Chapter 2 presents the related work. Chapter 3 presents the fatigue detection method. Chapter 4 presents the experimental study and Chapter 5 is the conclusion.

1.2.OBJECTIVE OF THE WORK

As a specific countermeasure to decrease collisions brought on by driver exhaustion, fatigue warning systems (FWS) have been proposed. For detecting driver fatigue while operating a vehicle and alerting a driver when critical drowsiness levels are reached, these gadgets use a number of methodologies. However, it is still difficult to identify driver weariness using reliable, intrusive, and objective methods.

Lane departure, activity at the steering wheel, and ocular or facial traits can all be used as detection methods. Of course, drivers also have a responsibility to follow speed limits, work-permitting limits, and rest-period regulations. Additionally, those in the chain of command are required to take reasonable action to avoid situations that could result in driver weariness or speed limit violations. It offers in-depth knowledge about drivers' attentiveness, driving abilities, physiological conditions, and subjective states.

CHAPTER 2 LITERATURE

SURVEY

To understand the potential causes of auto accidents, in-vehicle cameras are frequently deployed. The driver's level of weariness can also be determined with such a camera. The following is a description of a few studies that pertain to fatigue detection.

The amount of pixels in the eye picture was used by authors Horng et al. [4] and Sharma et al. to identify whether the eyes were open or closed. In order to locate the eyes, Horng et al. [4] created an edge map, and the eye state is ascertained using the HSL colour space of the eye image. The position of the eyes affects how accurate it is. The facial image was converted to YCbCr colour space by Sharma and Banga. The pixel number in the image's average and standard deviation the binarisation is computed.

Then, fuzzy rules are used to determine the eye state. Liu et al. and *Tabrizi et al.* [12] proposed methods to detect the upper and lower eyelids based onthe edge map. The distance between the upper and lower eyelids is then used to analyse the eye state. Besides, Dong et al.and Li et al.proposed methods by utilizing AAM (Active Appearance Model) to locate the eyes. Then, a PERCLO (PERcentage of eye CLOsure) was computed to detect the fatigue.

For the above methods, the locating of eye areas was easily influenced by the change of brightness. Circular Hough transform is popular method to overcome the influence of brightness. Several studies proposed methods to locate the pupil of eyes by using circular Hough transform. Then, the eye state was analysed according to the locations of pupils.

The above related studies didn’t address the issue of eyeglasses. Eyeglasses may reduce the correct ratio of the fatigue detection. A driver wearing eyeglasses is often seen. Therefore, a fatigue detection method should considerate such a condition.

CHAPTER 3

METHODOLOGY

The process of the proposed method is depicted in Figure 3.1. When the face area is detected successfully in the real-time image frame, the eyeglasses removal step is applied. If the template of the eye area is not established yet, an eye detection step is performed to get the initial area of eyes. Otherwise, the eye tracking step is performed to keep tracking the eyes. When the eyes area is detected, the RGB color space of the area is converted to HSL color space. The S-channel is then projected horizontally. The projected results are then filtered by Kalman filter to analyse the eye state. Then, the fatigue is detected according to the successive eye states.

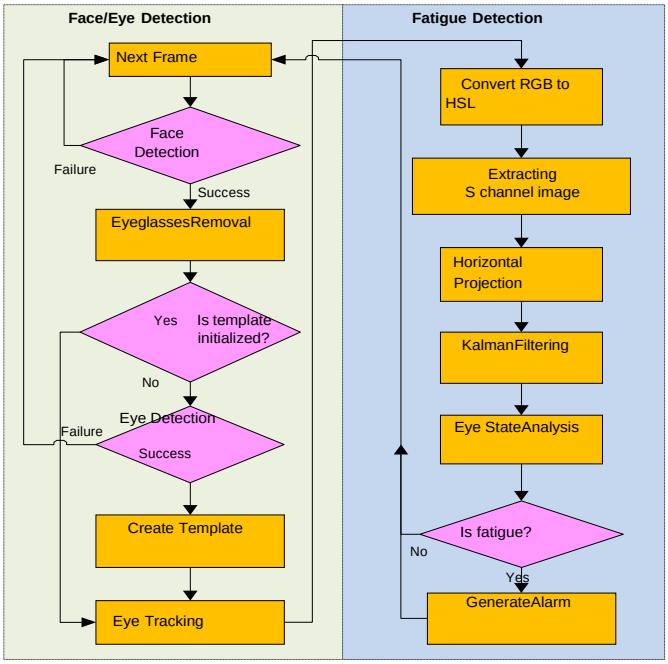


Figure 3.1:The fatigue detection

3.1.FACE AND EYE DETECTION

In this subsection, the methods of face detection, eyeglasses removal, and eye detection are presented in turns.

3.1.1.Face detection

The initial stage of fatigue detection is face detection. This phase makes use of an OpenCV library standard function. The detection result that is closest to the region in the previous frame in terms of distance is the face area in the current frame.

Based on the examination of the face database reported in the prior study, the ratio of eyes to face area has been determined.Therefore, assume the face area is denoted as *FR* and the left, top, width, height of *FR* are denoted as FR_{left}, FR_{right}, FR_{top} and FR_{bottom}.

$$RI_{left} = FR_{left} + FR_{width}/8$$
$$RI_{top} = FR_{left} + FR_{height}/4$$
$$RI_{right} = FR_{left} + FR_{width} - FR_{width}/8$$
$$RI_{bottom} = FR_{top} + FR_{height}/2$$

3.1.2.Eye detection

Eyes are very important to the fatigue detection. The area should be as accurate as possible. The sub-steps of eye detection are listed below.

S1: The cascade classifier of OpenCV library is used to generate the eye areas in the eye area *RI*.

S2: The areas of two eyes are recorded as the initial templates for eye tracking.

S3: Two eyes are tracking separately using the above templates. If the face area cannot be detected in the previous face detection step, the template is discarded and back to S1.

3.1.3. Tracking and Detection of Eyes

The ability to identify weariness relies heavily on the eyes. The region ought to be as precise as feasible. The following is a list of the eye detection sub-steps.

S1: The eye regions in the eye area RI are generated using the OpenCV library's cascade classifier.

S2: The first templates for eye tracking are two eyeballs' surface regions.

S3: Using the aforementioned templates, two eyes are tracking independently. The template is deleted and returned to S1 if the facial area cannot be detected in the previous face detection phase.

3.2.FATIQUE DETECTION

The fatigue detection consists of two sub-steps. The first one is the extraction of the S-channel image from HSL color space of eye image. The second sub-step is the analysis of eye state, i.e. open or close, for fatiguedetection.

3.2.1.Extraction of S-channel image

The colour space for the two eye regions is changed from RGB to HSL. The histogram equalisation technique is then applied to the S-channel to boost ocular contrast. The S-channel image is then binarized using a threshold of 30 that has been predetermined. Additionally, it makes the outcome of fatigue detection unstable. The value is stabilised using a Kalman filter. When the value, or the greatest number of consecutive white pixels, exceeds 0.45 times the eye area, RIL or RIR, the eye state is considered to be "near" after Kalman filter calculation. If not, the eye condition is classified as "open." Sometimes, just one eye can be accurately evaluated due to the head's rotation. Consequently, if one eye is deemed to be "near,"

3.2.2.Eye state Analysis

By looking at the binarization output, one can see that the eyeball is almost spherical. The blob resembles a flat shape when the eye is close. As a result, the binarization image is projected horizontally. For each horizontal line, the maximum number of consecutive while pixels is calculated.

CHAPTER 4

EXPERIMENTAL STUDY

A fatigue detection system based on the above method was implemented by using Visual C++. An experiment was also designed to evaluate the performance of the implemented system. Four test videos about 5~10 minutes are recorded. Two of them are the subjects wear eyeglasses and two of them are the subjects don't wear eyeglasses.

The proposed method is compared with the method presented by Flores et al.The results are listed in Table 4.1. For every test videos, the total frames are marked in the parentheses. The number of frames with eye open and close area also marked below. The overall ratio is also listed at the bottom of the table.

Table 4.1. THE COMPARISON OF EYE STATE DETECTION

	Our method			**Flores et al.**		
Videos (total)open + close	**Open**	**Close**	**Ratio(%)**	**Open**	**Close**	**Ratio(%)**
V1 (686) 493+193	471	180	94.9	488	171	96.1
V2 (1621) 1041+580	974	552	94.1	1027	544	96.9
V3 (1217) 545+672	450	599	86.2	433	574	82.7
V4 (1319) 361+958	350	710	80.4	316	696	76.7
Overall	4286/4843=**88.5%**			4249/4843=**87.7%**		

The following conclusions can be drawn from Table 4.1.

- For the method presented in[13], the correct ratio of eye open is higher than that of eye close.
- For the test videos with eyeglasses (V3 and V4), our method has higher correct ratio than the method presented in[13].
- When a subject in the test video rotates his head about 45 degrees. The eyeglasses will be failed to remove by our method. It causes the mis judgement of eye state.

Next, the fatigue detection is also evaluated. The results are listed in Table 4.2.

The abbreviations used in the table are described below:

- EB: the number of eye blinking
- RD: the number of real dozing
- AG: the number of alarm generated by the system
- NF (Negative False): the number of false alarm.
- PF (Positive False): the number of real dozing without alarm
- CA (Correct Alarm): the number of real dozing with alarm generated
- CR (Correct Ratio): the correct percentage of fatigue alarm

Table 4.2. THE COMPARISON OF FATIGUE DETECTION

Methods	Test Videos							
	V1		V2		V3		V4	
	Our	[19]	Our	[19]	Our	[19]	Our	[19]
EB	8		11		9		5	
RD	2		6		4		3	
AG	2	2	6	6	4	6	3	4
NF	0	0	0	0	0	2	0	1
PF	0	0	0	0	0	0	0	0
CA	2	2	6	6	4	4	3	3
CR (%)	100	100	100	100	100	100	100	100
PR (%)	100	100	100	100	100	66.7	100	75

According to the results listed in Table 4.2, our method and the method presented in can generate alarm with 100 percent. However, the methodpresented in generates false alarms when the subjects wear eyeglasses. It causes the precision ratio is decreased to 66.7 and 75 percent.

The average execution time of every steps of the proposed method is measured. The result is listed in *Table 4.3.*

Table 4.3. THE AVERAGE EXECUTION TIME

Steps	Elapsed time (ms)
Face detection	15.885
Eyeglasses removal	1.393
Eye detection and tracking	1.722
Fatigue detection	0.439
Overall of the method	19.439
A frame processed by the system	61.995

According to the results listed in Table 4.3, face detection is the most time- consuming step since the whole frame must be processed. The rest of the steps are limited in the ROI, the execution time is shorten substantially. So, the overall execution time of the proposed method is about 19 milliseconds. Besides, the system consumes about 62 milliseconds for processing a frame, including the display of the GUI (Graphical User Interface). It means that the system can handle 15~16 frames per second.

4.1.OUTPUT

Figure.4.1.1. Drowsiness detection with frames. Here the eyes are completely opened,Hence the driver status is normal. The frames are taken into consideration.

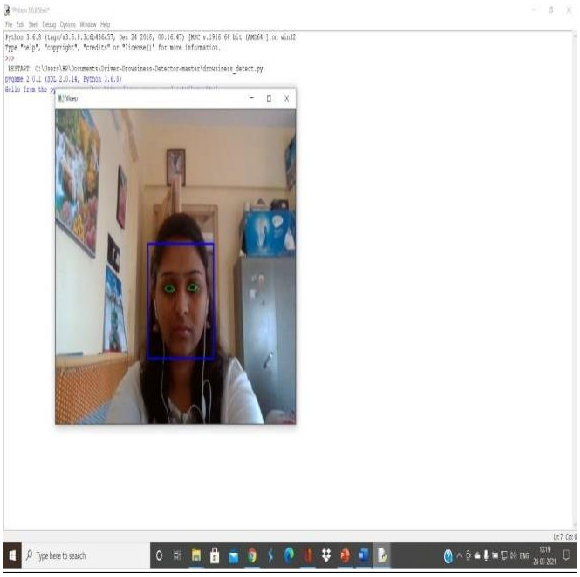
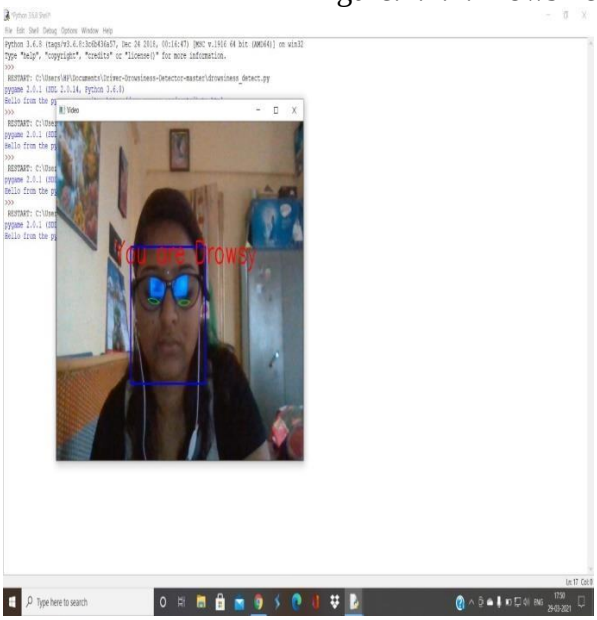


Figure.4.1.1.Drowsiness detected with eyeglasses and reported



CHAPTER 5

CONCLUSION

In this paper, a fatigue detection method with eyeglasses removal is proposed. The removal of eyeglasses increases the precision of eye detection and thus reduces the false alarms. According to the results of experimental study, the correct ratio of the eye state detection can reach 94 and 86 percent when the subjects don't and do wear eyeglasses, respectively. The overall correct ratio can reach 90 percent.Although the method seems promising,as some issues discussedbelow.

1. Eyeglasses removal may be fail when the rotation of the driver's head is over 30 degrees. Therefore, the install of the camera is suggested in front of the driver to prevent such a problem.

2. When the driver is back to the light, it causes the face and eyes area is dark.The detection of eyes area may be failed and causes the generation of false alarms.

APPENDIX I

WORKING ENVIRONMENT

Hardware Components

Raspberry Pi 3
Pi Camera Module
Micro USB Cable
Buzzer

Software and Online Services

OpenCV
Dlib
Python3

APPENDIX II

CODE

```
import face_recognition

import cv2

import numpy as np

import time

import cv2

import eye_game

import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

BUZZER= 23

GPIO.setup(BUZZER, GPIO.OUT)

previous ="Unknown"

count=0

video_capture = cv2.VideoCapture(0)

#frame = (video_capture, file)

file = 'image_data/image.jpg'

# Load a sample picture and learn how to recognize it.
```

```
img_image = face_recognition.load_image_file("img.jpg")

img_face_encoding = face_recognition.face_encodings(img_image)[0]

# Create arrays of known face encodings and their names

known_face_encodings = [

    img_face_encoding

]



known_face_names = [

    "Ashish"

]



# Initialize some variables

face_locations = []

face_encodings = []

face_names = []

process_this_frame = True

while True:

    # Grab a single frame of video

    ret, frame = video_capture.read()

    # Resize frame of video to 1/4 size for faster face recognition processing
```

```
small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)

# Convert the image from BGR color (which OpenCV uses) to RGB color (which face_recognition uses)

rgb_small_frame = small_frame[:, :, ::-1]

# Only process every other frame of video to save time

if process_this_frame:

    # Find all the faces and face encodings in the current frame of video

    face_locations = face_recognition.face_locations(rgb_small_frame)

    face_encodings=face_recognition.face_encodings(rgb_small_frame, face_locations)

    cv2.imwrite(file, small_frame)

    face_names = []

    for face_encoding in face_encodings:

        # See if the face is a match for the known face(s)

        matches=face_recognition.compare_faces(known_face_encodings, face_encoding)

        name = "Unknown"

        face_distances=face_recognition.face_distance(known_face_encodings, face_encoding)

        best_match_index = np.argmin(face_distances)

        if matches[best_match_index]:

            name = known_face_names[best_match_index]

            direction= eye_game.get_eyeball_direction(file)

            print(direction)
```

```
            #eye_game.api.get_eyeball_direction(cv_image_array)
            if previous != direction:
                previous=direction
            else:
                print("old same")
                count=1+count
                print(count)
                if (count>=10):
                  GPIO.output(BUZZER, GPIO.HIGH)
                  time.sleep(2)
                  GPIO.output(BUZZER, GPIO.LOW)
                  print("Alert!! Alert!! Driver Drowsiness Detected")
                  cv2.putText(frame, "DROWSINESS ALERT!", (10, 30),cv2.FONT_HERSHEY_SIMPLEX,
0.7, (0, 0, 255), 2)
        face_names.append(name)
    process_this_frame = not process_this_frame
    # Display the results
    for (top, right, bottom, left), name in zip(face_locations, face_names):
        # Scale back up face locations since the frame we detected in was scaled to 1/4 size
        top *= 4
        right *= 4
```

```
        bottom *= 4
        left *= 4
        # Draw a box around the face
        cv2.rectangle(frame, (left, top), (right, bottom), (0, 255, 0), 2)
        # Draw a label with a name below the face
        cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255), cv2.FILLED)
        font = cv2.FONT_HERSHEY_DUPLEX
        cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (0, 0, 255), 1)
        #cv2.putText(frame, frame_string, (left + 10, top - 10), font, 1.0, (255, 255, 255), 1)
    # Display the resulting image
    cv2.imshow('Video', frame)
    # Hit 'q' on the keyboard to quit!
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
# Release handle to the webcam
video_capture.release()
cv2.destroyAllWindows()
```

REFERENCES

[1] C. Du and G. Su, "Eyeglasses removal from facial images," Pattern Recognition Letters vol. 26, no. 14, 2005, pp. 2215-2220.

[2] Chenyu Wu, Ce Liu and Heung-Yueng Shum, "Automatic Eyeglasses Removal from Face Images**,"** IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 3, March 2004, pp. 322-336**.**

[3] J. S. Park, Y. H. Oh, S. C. Ahn, and S. W. Lee, "Glasses Removal from Facial Image Using Recursive Error Compensation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 5, May 2005, pp. 805-811.

[4] W. B. Horng, C. Y. Chen, Y. Chang and C. H. Fan, "Driver Fatigue Detection Based on Eye Tracking and Dynamic Template Matching," Proceedings of International Conference on Networking, Sensing & Control, Taipei, Taiwan, 2004, pp. 7-12.

[5] H. Z. Dong, M. Xie, "Real-Time Driver Fatigue Detection Based on Simplified Landmarks of AAM," Proceedings of the ICACIA, Chengdu, China, 2010, pp. 363-366.

[6] M. I. Khan and A. B. Mansoor, "Real Time Eyes Tracking and Classification for Driver Fatigue Detection," Lecture Notes in Computer Science, vol. 5112, 2008, pp. 729-738.

[7] I. Garcia, S. Bronte, L. M. Bergasa, N. Hernandez, B. Delgado, M. Sevillano "Vision-based Drowsiness Detector for a Realistic Driving Simulator," Proceedings of the 13th International IEEE ITSC, Funchal, Portugal,
2010, pp. 887-894.

[8] P. C. Yuen and C. H. Man, "Human Face Image Searching System Using Sketches," IEEE Transactions on System, Man, and Cybernetics Part A: Systems and Humans, vol. 37, no. 4, 2007, pp. 493-504.

[9] Simon, "Kalman Filtering," Embedded Systems Programming, vol. 14, no.6, June 2001, pp. 72-79.

[10] M. J. Flores and J. M. Armingol, A. Escalera, "Driver Drowsiness Warning System Using Visual Information for Both Diurnal and Nocturnal Illumination Conditions," EURASIP Journal on Advances in Signal Processing, vol. 2010, 19 pages.

[11] P. Viola and M. Jones. "Rapid Object Detection Using a Boosted Cascade of Simple Features," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 12, 2001, pp. I- 511-I-518.

[12] P. R. Tabrizi and R. A. Zoroofi, "Open Closed Eye Analysis for Drowsiness Detection," Proceedings of the First Workshops on Image Processing Theory, Tools and Applications, Sousse, Tunisia 2008, pp. 1-7. L.

[13] Li, M. Xie, H. Dong, "A Method of Driving Fatigue Detection Based on Eye Location," Proceedings of the IEEE 3rd ICCSN, Xian, China, 2011, pp. 480-484.

[14] D. Liu, P. Sun, Y. Q. Xiao, Y. Yin, "Drowsiness Detection Based on Eyelid Movement," Proceedings of the Second International Workshop on ETCS, Wuhan, China, 2010, pp. 49-52.

[15] J. S. Park, Y. H. Oh, S. C. Ahn, and S. W. Lee, "Glasses Removal from Facial Image Using Recursive Error Compensation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 5, May 2005, pp. 805-811.

[16] Wen-Chang Cheng," A fatigue detection system with eyeglasses removal" Advanced Communication Technology (ICACT), 2013.