Drowsy Driver Detection and Warning System on Raspberry Pi

Yoonchul Nam

MEng (Electrical and Electronic)

ENGR9700 (A-D) Masters Thesis (18 Units)

Project supervisor: Dr. Sherry Randhawa

Submitted to the College of Science and Engineering in partial fulfilment of the requirements for the degree of Master of Engineering (Electrical and Electronic) at Flinders University – Adelaide Australia

Submitted in August 2022

DECLARATION

I certify that this thesis:

1. does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university
2. and the research within will not be submitted for any other future degree or diploma without the permission of Flinders University; and
3. to the best of my knowledge and belief, does not contain any material previously published or written by another person except where due reference is made in the text.

Signature of student	[illegible]
Print name of student	Yoonchul Nam
Date	09/08/2022

I certify that I have read this thesis. In my opinion [✓ it is]/is not fully adequate, in scope and in quality, as a thesis for the degree of Master of Engineering in Electrical and Electronic. Furthermore, I confirm that I have provided feedback on this thesis and the student has implemented it [✓ minimally]/partially/fully.

Signature of Principal Supervisor	[illegible]
Print name of Principal Supervisor	Dr. Sherry Randhawa
Date	09/08/2022

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my project supervisor Dr. Sherry Randhawa to enabling me to complete this report on drowsiness detection on Raspberry Pi. Without her kind direction and proper guidance this study would have been a little success.

EXECUTIVE SUMMARY

Fatigue is a loss of alertness that reduces human performance and may or may not end up in sleep or micro-sleeps. It is one of the leading factors contributing to road crashes and has several problematic effects on driving performance. In order to reduce accidents caused by drowsy driving, various algorithms have been developed to detect driver's eye movements and sound alarms, and the information of the practical equipment and its experimental data in this study is offered to the drivers who want to apply this technique to their cars. In this study, camera performance is a large part of the work, and the author used two different types of cameras (general webcam and night vision camera) to check the accuracy of facial feature recognition, which can be identified whose face is in the image. He also studied factors that can affect facial recognition, such as the amount of light and the angle of the camera, to find out which environment shows the highest accuracy. There was a limit to the hardware such as low frames per second (fps) because it had to consider a reasonable price.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1. Motivation

Drowsiness during driving is the risk of falling asleep for a moment by closing eyes for a long period of time. This state can be described as an intermediate between awake and asleep. This is caused by sleep restriction by various coincidences, having baby waking every couple of hours, an abnormal shift at work, hanging out with friends too much time, or a long, monotonous drive. It is one of the leading factors contributing to road crashes and has several problematic effects on driving performance.

According to the Transport Accident Commission (Department of Infrastructure, Transport, Regional Development and Communications, 2018), 20% of all fatal road crashes in Victoria involve driver fatigue, while estimates in Queensland from the Centre for Accident Research and Road Safety, Queensland are that sleepiness contributes to 20–30% of all deaths and severe injuries on the road. Also, National Road Safety Strategy shows that fatigue is four times more likely to contribute to impairment than drugs or alcohol. Unlike drunk driving, which is carried out arbitrarily, drowsy driving is an instantaneous accident (Department of Infrastructure, Transport, Regional Development and Communications, 2018).

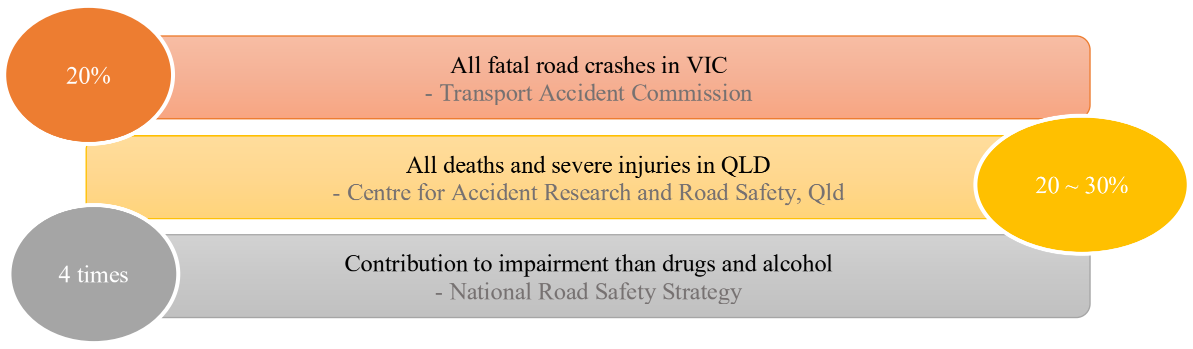

20%
All fatal road crashes in VIC
- Transport Accident Commission
All deaths and severe injuries in QLD
- Centre for Accident Research and Road Safety, Qld
20 ~ 30%
4 times
Contribution to impairment than drugs and alcohol
- National Road Safety Strategy

Figure 1 Estimated severe accident rate due to drowsy driving

Driver's drowsiness can be detected by three different approaches: physiological (Hendra et al., 2019), vehicle-based (Zhang & Zhang, 2006), and driver-based approaches. First, Sahayadhas et al (2012) shows that physiological signals from brains, such as EEG (Electroencephalogram), are considered as great measurement to estimate fatigue level. However, this approach requires specific experimental conditions due to possible distracting noise, which cannot be satisfied by normal driving circumstance. Second, vehicle-based approach such as Advanced Driver Assistance systems can detect whether the driver is having trouble with driving when the lane departure sensors realize the car does not follow the lane properly. However, this system does not detect the driver's fatigue level directly. Lastly, driver-based approach uses camera capturing driver's face and eyes and analyzing how long the eyes are closing. This approach requires cameras and microcontroller such as raspberry pi, so it is the most convenient way for normal drivers to install and use.

The most effective and fundamental solution is enough rest, but sleepiness can come at unexpected moments for various causes, no matter how well the driver prepares. As an engineer, the author came to think about solutions as an action on this social issue. It is a way to wake up with strong noisy sound when drivers look sleepy, and the author thought it would be very helpful to make and install a simple portable device in cars that have already been produced in the market.

1.2. Aims

The primary aim of this study was to compare differences in camera circumstances. The specific aims were to:

Aim 1: Compare Night Vision camera and normal webcam at different amount of lights.

Aim 2: Find proper angles to install the camera facing the driver's eyes.

Aim 3: Analyze the influence of the number of cameras on the system's accuracy.

1.3. Thesis Outlines

To achieve the aims above, the thesis is organised as following chapters:

Chapter 1 introduces the background of the thesis such as motivation and aims.

Chapter 2 provides the review of literature on the knowledge of the eyes recognition and blink detection.

Chapter 3 addresses the methodology of the study including hardware and software setup and experimental conditions.

Chapter 4 presents the main findings and results of the study.

Chapter 5 discusses the results identifying appropriate camera setting.

Finally, **Chapter 6** provides the overall conclusion of the thesis and suggests future work to develop the system.

CHAPTER 2: LITERATURE REVIEW

2.1. Driver-based Approach for Drowsiness Detection

Schleighher et al. (2008) investigated how changes in various eye movements in 129 participants were related to sleepiness and claimed that blinking was the most promising biological notification signal in the car. Participants self-evaluated the decreasing alertness through six steps and compared and analyzed each step with parameters of eye movement such as blink duration, eyelid opening time, and blinking interval. He argued that the most notable variable to determine drowsy driving is the duration of the blink. During awakening, there is little delay between closing and reopening the eyelids, so the value of the eyelids ranges from 1 to 4 milliseconds, while a delay of 100 ms is possible during micro-sleep (Schleighher et al., 2008). When fully asleep, reopening is completely stopped, suggesting a close relationship between increased reopening delays and the urge to fall asleep. Thus, the resume process can be combined with the closure process by a separate neural process, maintained only while awake and then stopped during sleep. This parameter may represent a marker indicating the beginning of sleep and the end of sleep (Schleighher et al., 2008).

2.2. Parameters for the Eye Blink Detection

Wilkinson and Jackson (2013) also measured eye movement parameters as fatigue indicators: Inter-Event Duration (IED), Percent Time with Eyes Closed (%TEC), Blink Total Duration (BTD), and Amplitude-Velocity Ratio (AVR). IED is measured from the point of maximum closing velocity to maximum opening velocity of the eyelid. %TEC is a proportion of time eyes are closed. BTD is duration of blinks from start of closing to the end of reopening. AVR is the ratio of the maximum amplitude to maximum velocity of eyelid movement. The average duration of eye closure (IED and BTD) provided the best discrimination in the primary measurement, and the ratio of amplitude to eyelid movement speed during eyelid closure also provided good discrimination. They showed that the results support the data from the ocular measurements to identify people who are impaired by drowsiness (Wilkinson et al., 2013).

2.3. Feature Detection

Viola and Jones (2001) introduced a rapid object detection algorithm. This algorithm is based on machine learning which allows rapid image processing and more accurate results of object detection. The Viola-Jones algorithm consists of three parts: region of interest (ROI) identification, adaptive boosting algorithm, and the cascade classifier.

ROI can be identified by two steps. The program sums all pixels in facial feature region of images, then the sum pixels are compared to find the differences among them. Adaptive boosting helps classifier by identifying the ROIs from images. A training set of images and feature settings are given in order to select and separate the positive and negative feature area. The cascade classifier discards background regions of images in terms of saving time to compute the object like regions. The cascading classifiers can reject negative regions and choose positive instances. This can lead higher detection rate.

Figure removed due to copyright restriction.

Figure 2 The process of the cascade classifiers [14]

This process of Viola-Jones algorithm can reduce computation time and achieve more accurate object detection. It is also an effective algorithm for detecting region of interest because the image itself does not need to be resized to detect feature areas.

2.4. Face and Eye Detection

In Guerrero's study (2006), face recognition is executed by distinguishing a cropped face region from background region. The face was detected by the Viola-Jones algorithm and the cascade classifiers were trained to distinguish face area. As shown in Figure 3, the rectangle was left on the image by surrounding the face.

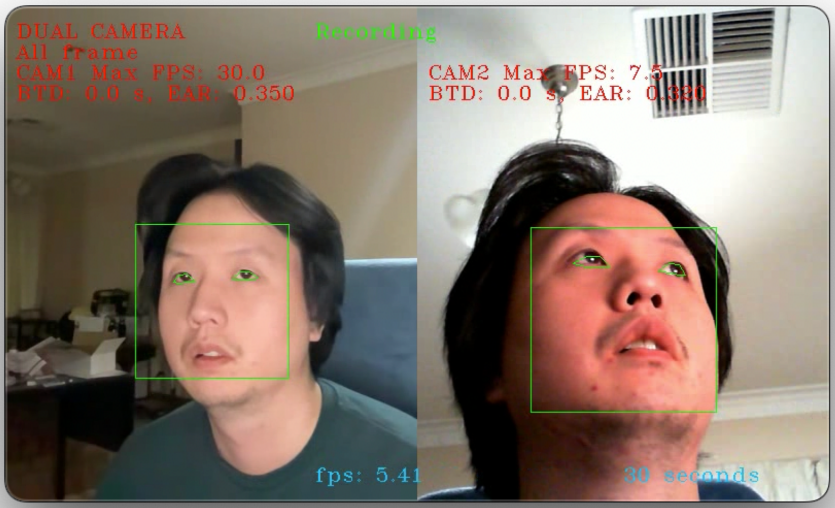


Figure 3 Face Recognition with bounding box

Vukadinovic (2006) also introduced a facial feature detector based on the Viola-Jones algorithm. Its cascade classifier is trained by GentleBoost, which has advantages that can shift and scale the chosen filter depending on the pixels.

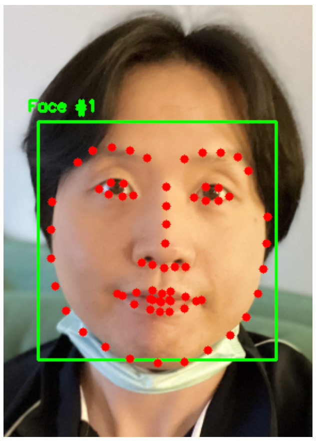


Figure 4 Facial Landmarks

Eyes can be detected by the method proposed by Bhaskar (2003), which marks feature points on the eyes and tracks their movements in real-time. This is based on the Kanade-Lucas-Tomasi (KLT) algorithm, which uses the spatial intensity data to find the best matching area. However, this approach has a limitation insisted by Thiyagarajan (2014), which is the KLT algorithm deals with limited data from only peripheral area around ROI. The KLT algorithm is based on three assumptions: brightness constant, time persistence, and spatial consistency. Firstly, the brightness constancy is that the pixel of an object does not change as the object moves from frame to frame. Second, temporal persistence is that the scale motion of the object does not move too much from frame to frame in an image proportional to the temporal increment. Lastly, spatial coherence is that adjacent points of the same object belong to the same area and have the same motion for the adjacent points of the image.

Figure removed due to copyright restriction.

Figure 5 KLT Eye Detection [15]

Guerrero (2006) insisted that eyes have bold vertical edge between iris and whites of eye. He used the Sobel operator in the face recognition to find the vertical edge. This approach is not proper for eyes wearing glasses.

2.5. Eye Blink Detection

Soukupova and Cech (2016) introduced an equation of Eye Aspect Ratio (EAR), which determine whether the eye is open or closed by comparing distances among the eye landmarks. The Eye Aspect Ratio is a constant value when the eye is open, but rapidly falls to 0 when the eye is closed. As shown in Figure 6 and Equation 1, the numerator computes the distance between the vertical eye landmarks while denominator computes the distance between horizontal eye landmarks.

$$\text{EAR} = \frac{|p_2 - p_6| + |p_3 - p_5|}{2\,|p_1 - p_4|} \qquad \text{(Equation 1)}$$

Figure removed due to copyright restriction.

Figure 6 Eye Landmarks for EAR equation [11]

According to the value of EAR, the eye can be recognized as closed or open, and the duration of the eye states determines whether the eye is drowsy.

CHAPTER 3: METHODOLOGY

3.1. Hardware Configuration

The hardest part in the experiment in a car is to choose appropriate equipment, which can handle different light conditions and lack of positioning. To install the system in a car, a microcomputer is required as the core part. The microcomputer should be small and located in an area not blocking the driver's view. Also, the purpose of this study is to offer an affordable system, so the cost is the major factor to be chosen. As a microcomputer, Raspberry Pi board and Arduino board (*Buy a Raspberry Pi 4 Model B – Raspberry Pi*, n.d.) are universally used. In this study, the author selected Raspberry Pi 4B due to the fact that it has two major advantages over Arduino. Firstly, the CPU of Raspberry Pi is 700MHz, while Arduino has CPU of 20MHz. This allows it to execute computation faster. Furthermore, Raspberry Pi board has better video controller in video recording and imaging processing, which are main functions of this study.

To capture the driver's face, two kinds of cameras were chosen. 5MP Night Vision Raspberry Pi Camera (*Buy a Pi NoIR Camera v2 – Raspberry Pi*, n.d.) and MS LifeCam HD-3000 webcam (*Microsoft Webcam: LifeCam HD-3000*, n.d.) were chosen to find out the effect of Night Vision. The Night Vision camera shows normal colourful picture with enough bright light, and it also has IR light to capture object in the dark. The MS webcam has 720p resolution, which can capture the appropriate image with enough light but cannot take an image in the dark. This feature is great to be used to compare with Night Vision camera.

The author wanted to achieve more accurate face recognition using dual camera set, which can capture each side of driver's face. However, there was only one CSI port on the board, so an external CSI hub was needed. However, the price was considered not affordable, so the author chose a different camera with reasonable cost and another connection, which is USB. The Night Vision camera was connected to the board through CSI connection, while the MS webcam was connected through USB port.

Figure removed due to copyright restriction.

Figure removed due to copyright restriction.

Figure removed due to copyright restriction.

Figure 7 Raspberry Pi 4B [4]

Figure 8 Night Vision Pi Camera [3]

Figure 9 MS LifeCam HD-3000 [8]

Potable USB battery was also used to give power to the board, for the experiment conducted in the car. Even though the experiment was not conducted in the running car due to the safety issue, the study required the experiment to be conducted in the car due to the different light condition.

3.2. Software Configuration

The program was made in Python, which is simple, famous programming language. For detecting and localizing facial landmarks, various libraries were also used in the program: DLIB, OpenCV, NumPy, imutils, SciPy, and xlwt (Zhang et al., 2020). DLIB is a suitable library for facial landmarks detection. To find facial landmarks from images, the program firstly has to extract the region of interest (ROI) of the face. For this job, the author used a Haar cascade classifier, which returns a multidimensional NumPy array, which is a Python library used for working with arrays. OpenCV was used to manipulate the images from the camera and to record a video of the experiments. To calculate the EAR by computing the Euclidean distance between the facial Eye landmarks, the SciPy package in python was also needed. Imutils was also needed for image processing and computer vision functions to support OpenCV. Comparatively, Matlab is also universally used, but it is hugely generic use and slow. Finally, the xlwt library was used to write the results of the experiment as an Excel spreadsheet.

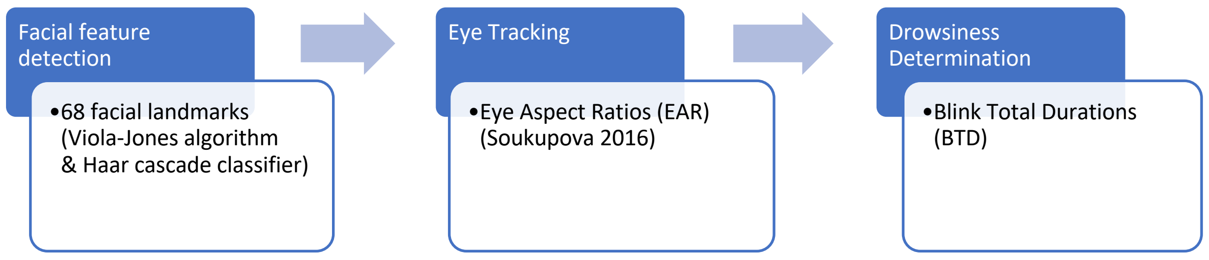

Facial feature detection
•68 facial landmarks (Viola-Jones algorithm & Haar cascade classifier)
Eye Tracking
•Eye Aspect Ratios (EAR) (Soukupova 2016)
Drowsiness Determination
•Blink Total Durations (BTD)

Figure 10 Procedures of drowsiness detection

3.3. Experiment 1: Eye Aspect Ratio (EAR) Threshold Setting

EAR threshold is used as a standard to distinguish whether the eye is closing, and the duration of closing eyes is the major parameter to determine drowsiness level (Soukupova and Cech, 2016). This means that, depending on the EAR threshold setting, the accuracy of drowsiness detection would be affected significantly. To figure out the appropriate value of the threshold, the author captured his face with closing eyes, and analyzed the data from the experiment.

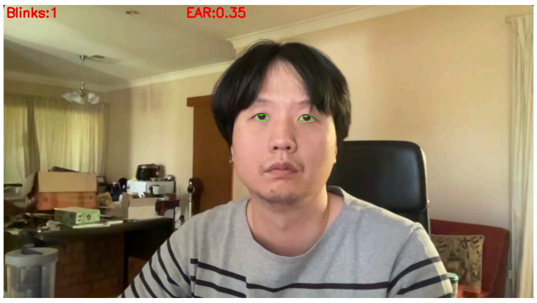


Figure 11 When the eyes were opening

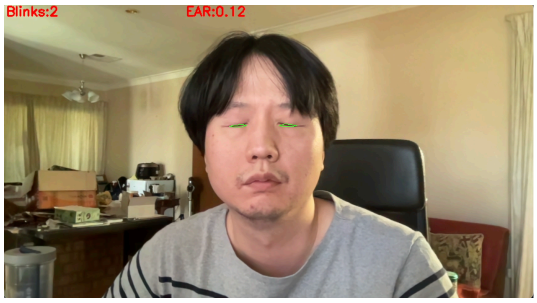


Figure 12 When the eyes were closing

The EAR threshold was calculated by averaging EAR of camera images captured by camera facing a participant's closed eyes directly in three different times: Morning (7AM ~ 8AM), Afternoon (12PM ~ 1PM), and Night (6PM ~ 7PM). In the morning, the sun was facing driver's face through the front windscreen. In the afternoon, the sunshine was above the car, so the car's ceiling was a bit darker than that in the morning. Lastly, it was totally dark in the night.

3.4. Experiment 2: Various Angles of Capturing Face

The ratio of facial landmarks and shadow patterns are changed according to the camera angles. Furthermore, it is hard to capture the driver's front face in a car due to the location restriction. In this experiment, the author monitored the impact of change of camera angles in order to find an appropriate angle to capture the driver's face. Both cameras captured the participant's facial movement with the angles of 0º, 20º, 40º, 60º, and 80º. The participant also pretended drowsy by nodding his head in order to monitor whether the program could detect his face with nodding.

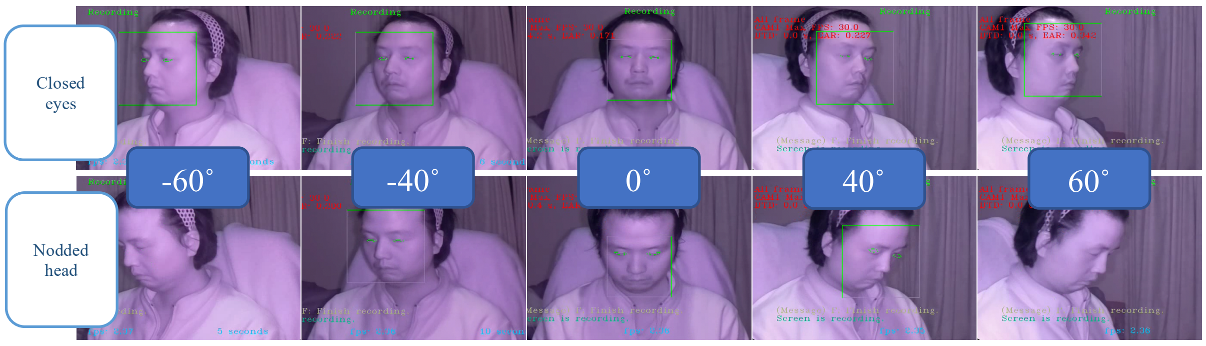
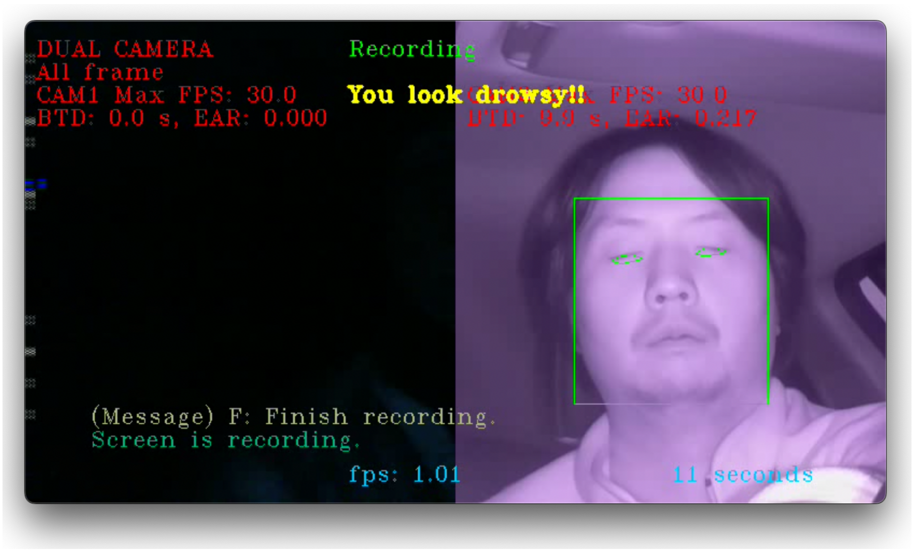


Figure 13 Screenshots showing the view of closed eyes and noddled head of Night Vision camera with various angles

3.5. Experiment 3: Capturing Face at Daytime and Night-time

Most of the time, the light in the car is not as bright as the system detecting the facial landmarks. Especially at night, the captured image by the camera in a car was totally dark as shown in Figure 14. In this experiment, the author compared between MS webcam and Night Vision Pi camera to observe the benefits of IR function of Night Vision camera in the dark by running the detection program at night.



```
DUAL CAMERA                Recording
All frame
CAM1 Max FPS: 30.0         You look drowsy!!  FPS: 30.0
BTD: 0.0 s, EAR: 0.000           BTD: 9.9 s, EAR: 0.217

   (Message) F: Finish recording.
   Screen is recording.
                           fps: 1.01            11 seconds
```

Figure 14 MS webcam vs Night Vision at low light

3.6. Experiment 4: Single Camera Set and Dual camera Set

In previous experiments, the author monitored the results of different environmental conditions such as the amount of light and different angle of face. In this experiment, the author observed whether the accuracy of using both cameras was improved rather than that of using single camera in the real driving environment. Both cameras were installed below eye level due to dealing with nodding head, one of acts that drivers do in the drowsy state.

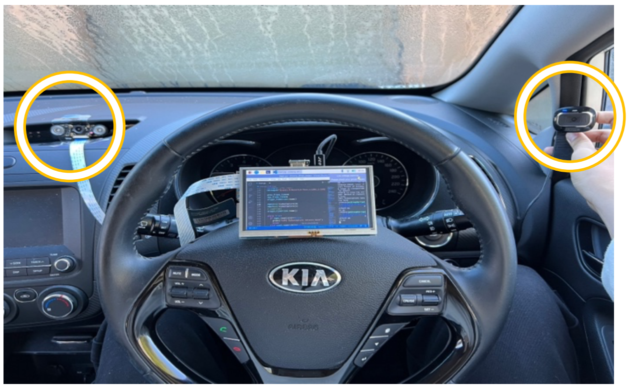


Figure 15 Experimental environment showing the position of the two cameras

The angles should be less than the value observed in previous experiments, which is the maximum angle that the system can recognize the face. If the angle is too big, the accuracy would drop to zero, and the dual camera set is meaningless.

CHAPTER 4: RESULTS

4.1. Eye Aspect Ratio Threshold

In Soukupova and Cech's report (2016), they also measured the EAR of opening or closed eyes. The EAR of opening eyes was bigger than 0.2 and that of closed eyes was smaller than 0.1 as shown in Figure 16. In this study, the EAR was also calculated by the Equation 1 which is the same equation in Soukupova and Cech's report.

Figure 16 The EAR plotted for several frames of a video sequence [11]

Both MS webcam and Night Vision Pi camera showed the similar result of the average of EAR in the same conditions, which means that the amount of light and the location of the camera were the same. However, the average EAR of both cameras at night was a bit bigger than that in different time as shown in Table 1.

Table 1 Average EAR of Open and Closed Eyes

Time	Open Eyes avg. EAR	Closed Eyes avg. EAR
7AM ~ 8AM	0.35	0.18
12PM ~ 1PM	0.34	0.17
6PM ~ 7PM	0.37	0.19

The EAR of opening eyes was bigger than 0.2 which was the same result with Soukupova's outcome, while that of closed eyes was bigger than 0.1 which was bigger than her result. For the following experiments, the author set EAR of 0.2 as a threshold that determined whether the eyes were closed, and the total time of closing eyes could be counted as the drowsiness parameter (BTD) during the time that captured EAR was under the threshold.

4.2. Impact of Different Angle on the Accuracy

The program detected the state of eyes, such as closed or opening, and it noted 1 in a excel file when the eyes on the frame were determined as closed, while it noted 0 when the eyes on the frame were determine as opening. Every frame could be compared with the images and the data from the excel in order to calculate the accuracy of the program. The accuracy of the program was determined by whether the program detected the state of eyes correctly.

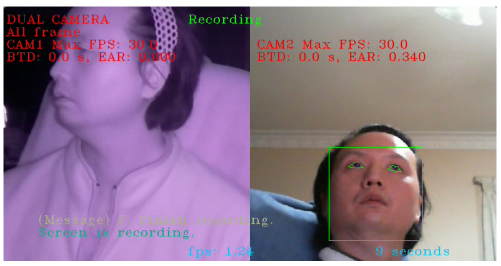
Overall, Night Vision Pi camera showed more accurate results, which means the program detects the correct state of eyes such as closed or opening, rather than MS webcam with every different angle except 80º. When the face angle was 80º, both cameras could not detect facial landmarks.

Table 2 Accuracy and Avg. EAR of Single Camera Set with angles (MS webcam and Night Vision camera)

Angles of camera	-80º	-60º	-40º	-20º	0º	20º	40º	60º	80º
MS face Accuracy	0%	92%	88%	100%	100%	100%	100%	0%	0%
MS closed EAR	0.00	0.31	0.19	0.17	0.18	0.17	0.21	0.00	0.00
MS face Accuracy (Nod)	0%	20%	43%	97%	100%	71%	45%	0%	0%
MS closed EAR (Nod)	0.00	0.36	0.18	0.19	0.22	0.19	0.21	0.00	0.00
NV face Accuracy	0%	100%	100%	100%	100%	100%	100%	90%	100%
NV closed EAR	0.00	0.28	0.21	0.18	0.17	0.19	0.24	0.36	0.36
NV face Accuracy (Nod)	0%	77%	100%	100%	100%	100%	100%	62%	32%
NV closed EAR (Nod)	0.00	0.33	0.18	0.19	0.19	0.19	0.19	0.41	0.35

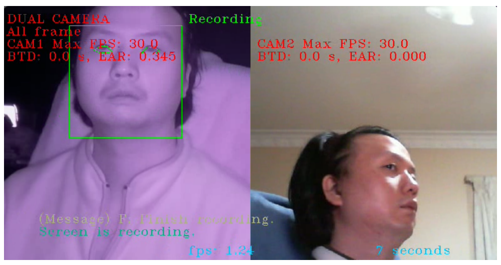
This showed that Night Vision camera is more suitable than MS webcam to detect face landmarks. According to Table 2, the face angle to camera should be less than 40º.

4.3. Single Camera and Dual Camera System



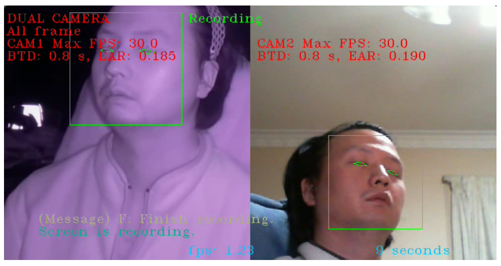
```
DUAL CAMERA                 Recording
All frame
CAM1 Max FPS: 30.0          CAM2 Max FPS: 30.0
BTD: 0.0 s, EAR: 0.000      BTD: 0.0 s, EAR: 0.340
(Message) F: Finish recording.
Screen is recording.
                 fps: 1.24               9 seconds
```

Left look
(to MS camera)



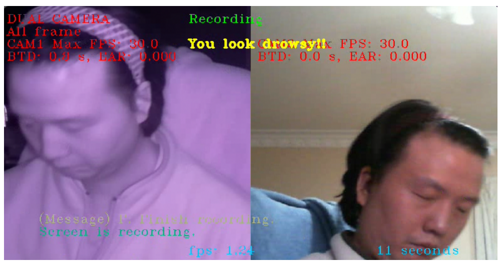
```
DUAL CAMERA                 Recording
All frame
CAM1 Max FPS: 30.0          CAM2 Max FPS: 30.0
BTD: 0.0 s, EAR: 0.345      BTD: 0.0 s, EAR: 0.000
(Message) F: Finish recording.
Screen is recording.
                 fps: 1.24               7 seconds
```

Right look
(to NV camera)



```
DUAL CAMERA                 Recording
All frame
CAM1 Max FPS: 30.0          CAM2 Max FPS: 30.0
BTD: 0.8 s, EAR: 0.185      BTD: 0.8 s, EAR: 0.190
(Message) F: Finish recording.
Screen is recording.
                 fps: 1.23               9 seconds
```

Front look



```
DUAL CAMERA                 Recording
All frame
CAM1 Max FPS: 30.0          You look drowsy!! FPS: 30.0
BTD: 0.8 s, EAR: 0.000      BTD: 0.0 s, EAR: 0.000
(Message) F: Finish recording.
Screen is recording.
                 fps: 1.24               11 seconds
```

Front look (nod)

Figure 17 Screenshots of dual camera set

When both cameras were 75cm far away from a participant's face and 40º wide to capture all his face properly, they could detect his facial landmarks unless he turned his face to the left or right. When he turned his face to the right, where Night Vision camera was, NV camera could detect the landmarks, while MS webcam could not. On the other hand, Night Vision camera could not detect the landmarks when he turned to the left, where MS webcam was, while MS webcam could detect them. In this experiment, the dual camera set could improve the accuracy of face recognition by complementing each other. When a camera could not detect the face, the other camera could detect it.

Table 3 Accuracy of Dual Camera Set (NV at -40˚ & MS at 40˚)

	Night Vision	**MS webcam**
Closed Eyes	100%	82%
Nodding	81%	56%
left turn	0%	100%
right turn	100%	0%

According to Table 3, even though the participant nodded his head as the act of drowsiness and the cameras set up with 40 degrees wide, Night Vision camera could show a fairly high level of reliability of face recognition. However, if the head was turned, the camera could not detect them due to the fact that they could not see his face properly, as shown in Figure 16.

CHAPTER 5: DISCUSSION

There are three points to discuss in the results section. Firstly, the parameter determining driver's fatigue is blink total duration, so it is the most important for the system to recognize face and detect closed eyes. However, the MS webcam does not have any features to filter the image for better face recognition, while the Night Vision Pi camera has IR filtering feature able to capture objects in the tough condition such as low light. It was obvious to affect the accuracy of face recognition significantly. The accuracy of the MS webcam was sensitive to the amount of light and facial angles that may occur shadow interfering face recognition.

Second, even though the Night Vision Pi camera has IR feature to capture objects in the dark, if the face was turned and part of landmarks was hidden or vague to see, the program could not recognize the face. According to Table 2, the cameras showed appropriate accuracy within 40º. However, the NV camera showed that it could detect face and eyes during nodding head. That is, the NV camera has a great advantage in detecting the driver's drowsy driving. If the CSI hub is used, it will be possible to detect drowsiness with higher accuracy using several NV cameras.

Figure removed due to copyright restriction.

Figure 18 Autofocus Synchronized Quad-Camera Kit for Raspberry Pi [1]

Finally, there were hardware limitation. The Raspberry Pi has low performance of CPU and GPU, which is important for Computer Vision, so the frames per second (fps) was 1.23 (dual camera set) and 2.34 (single camera set), respectively. If the fps is lower than 10, the object movement on the image can be lagging, and this can affect the accuracy of face recognition significantly. For this issue, the microcontroller should have better CPU and GPU to obtain more accurate results.

For decades, there have been studies on blinking using facial recognition and eye movement, and there have been many projects to detect drowsy driving by quantifying blinking [2, 4, 12, 13, 16]. However, even though the equipment that can be purchased by individuals was not enough to be used in practice, this study was able to obtain good data for direct installation in cars on the market.

CHAPTER 6: CONCLUSIONS AND FUTURE WORK

6.1. Conclusions

Drowsy driving has long been considered a social problem and as an engineer, the writer has a mission to contribute to solving the problem. A program was created to detect the driver's sleepiness by grasping the driver's eye blink and sleepiness correlation. It was difficult to grasp the driver's face with one camera in the car, so two cameras were configured to improve accuracy and observed what changes were made. He not only increased the number of cameras, but also found out which angle has the highest accuracy. It was also investigated whether the difference in the basic performance of the camera has a great influence on accuracy under various light conditions such as a small amount of light. It was confirmed that the Night Vision camera captured the face well even in a completely dark environment, but the normal camera did not detect it at all. The limited performance of hardware such as microcomputers also greatly affected the results.

6.2. Future Work

Since MS webcam does not have its own filtering function, it has shown low accuracy in facial recognition and sensitive to environmental changes. However, in future studies, if the captured screen is modified using Computer Vision technology, good results can be obtained even with programs using low-performance, inexpensive webcams. The expected technology is the histogram of gradient (HOG). HOG is a vector that divides a target area into cells of a certain size, obtains a histogram of the direction of edge pixels such as pixels with a gradient magnitude greater than or equal to a certain value for each cell, and connects these histogram bin values in a row. That is, the HOG may be described as a direction histogram template of the edge. HOG is a kind of template matching, so it is difficult to detect when an object is rotated, or the shape change is severe. However, drowsy driver detection captures driver’s face, which does not usually have severe change, so this technique might be good trial for the next study.

Furthermore, in this study, the Blink Total Duration (BTD) was used as a drowsiness detection parameter. However, this parameter was calculated by the total time of closing eyes, so it might not appropriate in specific conditions such as itching eyes. For this perspective, Inter Event Duration (IED) and Amplitude to velocity ratio (AVR) can be replaced. IED and AVR are calculated by the total time and the speed of closing or opening eyes, respectively. With this parameter, even though the driver closed his eyes for a certain time, if the closing speed is short, the system would not determine he is drowsy.

References

[1] *16MP IMX519 autofocus Synchronized Quad-Camera Kit for Raspberry Pi, Nvidia Jetson Nano/Xavier NX-Arducam*. (n.d.). Arducam. https://www.arducam.com/product/arducam-16mp-imx519-autofocus-synchronized-quad-camera-kit-for-raspberry-pi-nvidia-jetson-nano-xavier-nx/

[2] Bhaskar, T. N., Keat, F., Ranganath, S., & Venkatesh, Y. V. (2003). Blink detection and eye tracking for eye localization. *TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region*. https://doi.org/10.1109/tencon.2003.1273293

[3] *Buy a Pi NoIR Camera V2 – Raspberry Pi*. (n.d.). Raspberrypi.org. https://www.raspberrypi.org/products/pi-noir-camera-v2/

[4] *Buy a Raspberry Pi 4 Model B – Raspberry Pi*. (n.d.). Raspberrypi.org. https://www.raspberrypi.org/products/raspberry-pi-4-model-b/

[5] Department of Infrastructure, Transport, Regional Development and Communications. (2018). *Fatigue*. National Road Safety Strategy. https://www.roadsafety.gov.au/action-plan/2018-2020/fatigue

[6] Guerrero, S. T. (2006). *Model-Based Eye Detection and Animation*.

[7] Hendra, M., Kurniawan, D., Chrismiantari, R. V., Utomo, T. P., & Nuryani, N. (2019). Drowsiness detection using heart rate variability analysis based on microcontroller unit. *Journal of Physics: Conference Series*, *1153*, 012047. https://doi.org/10.1088/1742-6596/1153/1/012047

[8] *Microsoft Webcam: LifeCam HD-3000*. (n.d.). Microsoft Accessories. https://www.microsoft.com/en-ww/accessories/products/webcams/lifecam-hd-3000?activetab=overview%3aprimaryr2.

[9] Sahayadhas, A., Sundaraj, K., & Murugappan, M. (2012). Detecting Driver Drowsiness Based on Sensors: A Review. *Sensors*, *12*(12), 16937–16953. https://doi.org/10.3390/s121216937

[10] Schleicher, R., Galley, N., Briest, S., & Galley, L. (2008). Blinks and saccades as indicators of fatigue in sleepiness warnings: looking tired? *Ergonomics*, *51*(7), 982–1010. https://doi.org/10.1080/00140130701817062

[11] Soukupova, T., & Cech, J. (2016) *Real-Time Eye Blink Detection Using Facial Landmarks*.

[12] Thiyagarajan, B. V. (2014). Face Detection and Facial Feature Points Detection with the Help of KLT Algorithm. *International Journal of Advance Research in Computer Science and Management Studies*, *2*(8), pp 49-54.

[13] Tucker, A., & Johns, M. (2005). *The duration of eyelid movements during blinks: Changes with drowsiness*.

[14] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer*

Vision and Pattern Recognition. CVPR 2001. https://doi.org/10.1109/cvpr.2001.990517

[15] Vukadinovic, D., & Pantic, M. (2006). Fully Automatic Facial Feature Point Detection Using Gabor Feature Based Boosted Classifiers. *2005 IEEE International Conference on Systems, Man and Cybernetics*. https://doi.org/10.1109/icsmc.2005.1571392

[16] Wilkinson, V. E., Jackson, M. L., Westlake, J., Stevens, B., Barnes, M., Swann, P., Rajaratnam, S. M. W., & Howard, M. E. (2013). The Accuracy of Eyelid Movement Parameters for Drowsiness Detection. *Journal of Clinical Sleep Medicine*, *09*(12), 1315–1324. https://doi.org/10.5664/jcsm.3278

[17] Zhang, Z., & Zhang, J. (2006). Driver Fatigue Detection Based Intelligent Vehicle Control. *18th International Conference on Pattern Recognition (ICPR'06)*. https://doi.org/10.1109/ICPR.2006.462

[18] Zhang, D., Li, J., & Shan, Z. (2020). Implementation of Dlib Deep Learning Face Recognition Technology. *2020 International Conference on Robots & Intelligent System (ICRIS)*. https://doi.org/10.1109/icris52159.2020.00030

Appendices

A. CODE

```
from scipy.spatial import distance as dist
from imutils import face_utils
import numpy as np
from datetime import datetime
import dlib
import cv2
import time
from xlwt import Workbook

font=cv2.FONT_HERSHEY_COMPLEX_SMALL

# Display Spec
disp_height=480
disp_width=800

# GLOBAL FLAGS
FLAG_DUAL_CAMS=True
FLAG_CROP_DRIVER=False
isCountingBTF=False
BTF1=0
BTF2=0
# EAR Detection
EAR_THRESH=0.25
BTD_THRESH=800

# Landmark Detection Initialization
detector=dlib.get_frontal_face_detector()
predictor=dlib.shape_predictor("shape_predictor_68_face_landmarks.dat
")
(lStart,lEnd)=face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(rStart,rEnd)=face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]

def eye_aspect_ratio(eye):
  A=dist.euclidean(eye[1],eye[5])
  B=dist.euclidean(eye[2],eye[4])
  C=dist.euclidean(eye[0],eye[3])
  ear=(A+B)/(2.0*C)
  return ear

def getCapFrame(cap):
  global disp_width,disp_height
  disp_ratio=disp_height/disp_width
  dim=(disp_width,disp_height)
  cam_width=cap.get(cv2.CAP_PROP_FRAME_WIDTH)
  cam_height=cap.get(cv2.CAP_PROP_FRAME_HEIGHT)
  cam_ratio=cam_height/cam_width
  ret,frame=cap.read()
  if frame is not None:
    # Resizing frame to display size
    if disp_ratio>cam_ratio:
      frame_width=cam_height/disp_ratio
      frame_height=cam_height
      frame=frame[0:int(cam_height),int(cam_width/2-
frame_width/2):int(cam_width/2+frame_width/2)]
      frame=cv2.resize(frame,dim,interpolation=cv2.INTER_AREA)
      frame=cv2.flip(frame,1)
    elif disp_ratio<cam_ratio:
      frame_width=cam_width
      frame_height=cam_width/disp_ratio
      frame=frame[int(cam_height/2-
frame_height/2):int(cam_height/2+frame_height/2),0:int(cam_width)]
      frame=cv2.resize(frame,dim,interpolation=cv2.INTER_AREA)
      frame=cv2.flip(frame,1)
    else:
      frame_width=cam_width
      frame_height=cam_width/disp_ratio
      frame=cv2.resize(frame,dim,interpolation=cv2.INTER_AREA)
      frame=cv2.flip(frame,1)
    return frame

def detectBlink(BTF,frame):
  global font,isCountingBTF,EAR_THRESH

  rects=detector(frame,0)
  if rects:
    rect=rects[0] # The biggest rectangle would be on the driver's
face
    (x,y,w,h)=face_utils.rect_to_bb(rect)
    shape=predictor(frame,rect)
    shape=face_utils.shape_to_np(shape)
    leftEye=shape[lStart:lEnd]
    rightEye=shape[rStart:rEnd]
    leftEyeHull=cv2.convexHull(leftEye)
    rightEyeHull=cv2.convexHull(rightEye)
    cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0))
    cv2.drawContours(frame,[leftEyeHull],-1,(0,255,0),1)
    cv2.drawContours(frame,[rightEyeHull],-1,(0,255,0),1)
    leftEAR=eye_aspect_ratio(leftEye)
    rightEAR=eye_aspect_ratio(rightEye)
    ear=(leftEAR+rightEAR)/2.0
    if ear<EAR_THRESH:
      if isCountingBTF: BTF+=1
      else:
        isCountingBTF=True
        BTF=1
    else:
      isCountingBTF=False
      BTF=0
    if FLAG_CROP_DRIVER:
      margin=50
      f_height,f_width=frame.shape[:2]
      cv2.rectangle(frame,(0,0),(f_width,y-
margin),(0,0,0),cv2.FILLED)
      cv2.rectangle(frame,(0,y-margin),(x-
margin,y+h+margin),(0,0,0),cv2.FILLED)
      cv2.rectangle(frame,(x+w+margin,y-
margin),(f_width,y+h+margin),(0,0,0),cv2.FILLED)

cv2.rectangle(frame,(0,y+h+margin),(f_width,f_height),(0,0,0),cv2.FIL
LED)
    if FLAG_DUAL_CAMS: frame=crFrame(x,w,frame)
    dt_rect=1
  else:
    if FLAG_DUAL_CAMS: frame=crFrameNoRect(frame)
    ear=0
    dt_rect=0
  return BTF,ear,dt_rect,frame

def mergeCapFrame(frame1,frame2):
  global disp_width,disp_height
  full_frame=np.zeros((disp_height,disp_width,3),np.uint8)
  full_frame[0:,0:int(disp_width/2)]=frame1
  full_frame[0:,int(disp_width/2):int(disp_width)]=frame2
  return full_frame

def crFrameNoRect(frame):
  cr_frame=frame[0:int(disp_height),0:int(disp_width/2)]
  return cr_frame
```

```
def crFrame(x,w,frame):
  global disp_width,disp_height
  cr_frame=frame[0:int(disp_height),0:int(disp_width/2)]
  if x+w/2<disp_width/4:
    cr_frame=frame[0:int(disp_height),0:int(disp_width/2)]
  elif x+w/2>disp_width*3/4:
    cr_frame=frame[0:disp_height,int(disp_width/2):int(disp_width)]
  else:
    cr_frame=frame[0:disp_height,int(x+w/2-
disp_width/4):int(x+w/2+disp_width/4)]
  return cr_frame

def main():
  global FLAG_DUAL_CAMS,FLAG_CROP_DRIVER,font,BTF1,BTF2,BTD_THRESH
  sh_row=1
  FLAG_RECORDING=False
  BTD=0
  BTD1=0
  BTD2=0
  message1="(Key Maps)"
  message2="Q:quit,R:Record,O:face,1:CAM1,2:CAM2,D:Dual,S:Single"

  prev_frame_time=0
  new_frame_time=0
  origin_time=time.time()

  cap1=cv2.VideoCapture(1)
  cap2=cv2.VideoCapture(0)
  origin_time=time.time()
  while True:
    new_frame_time=time.time()
    fps=1/(new_frame_time-prev_frame_time)
    prev_frame_time=new_frame_time

    frame1=getCapFrame(cap1)
    BTF1,ear1,dt_rect1,dt_frame1=detectBlink(BTF1,frame1)
    BTD1=BTF1/fps
    cv2.putText(dt_frame1,"CAM1 Max FPS:
{:.1f}".format(cap1.get(cv2.CAP_PROP_FPS)),(10,70),font,1,(0,0,255),1
)
    cv2.putText(dt_frame1,"BTD: {:.1f} s, EAR:
{:.3f}".format(BTF1/fps,ear1),(10,90),font,1,(0,0,255),1)

    frame2=getCapFrame(cap2)
    BTF2,ear2,dt_rect2,dt_frame2=detectBlink(BTF2,frame2)
    BTD2=BTF2/fps
    cv2.putText(dt_frame2,"CAM2 Max FPS:
{:.1f}".format(cap2.get(cv2.CAP_PROP_FPS)),(10,70),font,1,(0,0,255),1
)
    cv2.putText(dt_frame2,"BTD: {:.1f} s, EAR:
{:.3f}".format(BTF2/fps,ear2),(10,90),font,1,(0,0,255),1)

    if int(BTD1)>0 & int(BTD2)>0: # Averaging the dual cams' BTDs
      BTD=(BTD1+BTD2)/2
    elif BTD1>BTD2:
      BTD=BTD1
    elif BTD1<BTD2:
      BTD=BTD2
    else:
      BTD=0

    if FLAG_RECORDING: # Writing data on spreadsheet
      if FLAG_DUAL_CAMS:
        sheet1.write(sh_row,0,dt_rect1)
        sheet1.write(sh_row,1,ear1)
        sheet1.write(sh_row,2,BTF1)
        sheet1.write(sh_row,3,dt_rect2)
        sheet1.write(sh_row,4,ear2)
        sheet1.write(sh_row,5,BTF2)
        sheet1.write(sh_row,6,BTD)
        sheet1.write(sh_row,7,fps)
      else:
        sheet1.write(sh_row,0,dt_rect1)
        sheet1.write(sh_row,1,ear1)
        sheet1.write(sh_row,2,BTF1)
        sheet1.write(sh_row,3,BTD)
        sheet1.write(sh_row,4,fps)
      sh_row+=1

    full_frame=np.zeros((disp_height,disp_width,3),np.uint8)
    if FLAG_DUAL_CAMS:
      full_frame=mergeCapFrame(dt_frame1,dt_frame2)
    else:
      full_frame=dt_frame1

    # Information on the screen
    if FLAG_DUAL_CAMS:
      cv2.putText(full_frame,"DUAL
CAMERA",(10,30),font,1,(0,0,255),1)
    else:
      cv2.putText(full_frame,"Single
CAMERA",(10,30),font,1,(0,0,255),1)
    if FLAG_CROP_DRIVER:
      cv2.putText(full_frame,"Crop Driver's
face",(10,50),font,1,(0,0,255),1)
    else:
      cv2.putText(full_frame,"All frame",(10,50),font,1,(0,0,255),1)
    if FLAG_RECORDING:
      cv2.putText(full_frame,"Recording",(300,30),font,1,(0,255,0),1)
    else:
      cv2.putText(full_frame,"Not
Recording",(300,30),font,1,(0,0,255),1)
    if BTD >= BTD_THRESH/1000:
      cv2.putText(full_frame,"You look
drowsy!!",(300,70),font,1,(0,255,255),2)
    cv2.putText(full_frame,"fps:
{:.2f}".format(fps),(300,460),font,1,(255,200,0),1)
    cv2.putText(full_frame,"{} seconds".format(int((time.time()-
origin_time))),(600,460),font,1,(255,200,0),1)
    cv2.putText(full_frame,message1,(60,410),font,1,(130,180,170),1)
    cv2.putText(full_frame,message2,(60,430),font,1,(130,180.170),1)
    if FLAG_RECORDING: writer.write(full_frame)
    cv2.imshow("Masters Thesis",full_frame)

    # Command key maps
    if cv2.waitKey(10)&0xFF==ord("r"): # Recording a video of frames
and writing data
      if FLAG_RECORDING==False:
        FLAG_RECORDING=True
        message1="(Message) F: Finish recording."
        message2="Screen is recording."
        origin_time=time.time()
        wb=Workbook()
        if FLAG_DUAL_CAMS:
          sheet1=wb.add_sheet('Dual Face Recognition')
          sheet1.write(0,0,'Rect1')
          sheet1.write(0,1,'EAR1')
          sheet1.write(0,2,'BTF1')
          sheet1.write(0,3,'Rect2')
          sheet1.write(0,4,'EAR2')
          sheet1.write(0,5,'BTF2')
          sheet1.write(0,6,'BTD')
          sheet1.write(0,7,'FPS')
          sheet1.write(0,8,'DUAL')
          sheet1.write(0,9,FLAG_DUAL_CAMS)
          sheet1.write(1,8,'CAM1 Max fps')
          sheet1.write(1,9,cap1.get(cv2.CAP_PROP_FPS))
          sheet1.write(2,8,'CAM2 Max fps')
          sheet1.write(2,9,cap2.get(cv2.CAP_PROP_FPS))

writer=cv2.VideoWriter('videos/dual_'+datetime.now().strftime("%d%m%Y
_%H%M%S")+'.mp4',cv2.VideoWriter_fourcc(*"mp4v"),fps,(disp_width,disp
_height))
        else:
          sheet1=wb.add_sheet('Single Recognition')
          sheet1.write(0,0,'Rect')
          sheet1.write(0,1,'EAR')
          sheet1.write(0,2,'BTF')
          sheet1.write(0,3,'BTD')
          sheet1.write(0,4,'FPS')
          sheet1.write(0,5,'CAM Max fps')
          sheet1.write(0,6,cap1.get(cv2.CAP_PROP_FPS))

writer=cv2.VideoWriter('videos/single_'+datetime.now().strftime("%d%m
%Y_%H%M%S")+'.mp4',cv2.VideoWriter_fourcc(*"mp4v"),fps,(disp_width,di
sp_height))
        sh_row=1
    if cv2.waitKey(10)&0xFF==ord('f'): # Finishing the recording
      if FLAG_RECORDING:
        FLAG_RECORDING=False
        message1="(Message) Q/R/D/S/1/2/O"
        message2="Screen recording is done."
        origin_time=time.time()
        if FLAG_DUAL_CAMS:
wb.save('xls/dual_'+datetime.now().strftime("%d%m%Y_%H%M%S")+'.xls')
        else:
wb.save('xls/single_'+datetime.now().strftime("%d%m%Y_%H%M%S")+'.xls'
)
        writer.release()
    if cv2.waitKey(10)&0xFF==ord("q"): # Quit the program
      if FLAG_RECORDING: message2="the screen is still recording,
finish the recording before exit the program."
      else:
        message1="(Message)"
        message2="Key Q is pressed. Exit."
        break
    if FLAG_RECORDING==False: # When recording, the keys are avoided.
      if cv2.waitKey(10)&0xFF==ord("o"): # cropping driver's face
        message1="(Message) Q/R/D/S/1/2/O/A"
        message2="Only driver's face will be displayed"
        FLAG_CROP_DRIVER=True
      if cv2.waitKey(10)&0xFF==ord("a"): # without cropping
        message1="(Message) Q/R/D/S/1/2/O/A"
        message2="All frame will be displayed"
        FLAG_CROP_DRIVER=False
      if cv2.waitKey(10)&0xFF==ord("1"): # Cam '1' is the main
        message1="(Message) Q/R/D/S/1/2/O"
        message2="Cam1 is selected."
        cap1=cv2.VideoCapture(1)
        cap2=cv2.VideoCapture(0)
      if cv2.waitKey(10)&0xFF==ord("2"): # Cam '2' is the main
        message1="(Message) Q/R/D/S/1/2/O"
        message2="Cam2 is selected"
        cap1=cv2.VideoCapture(0)
        cap2=cv2.VideoCapture(1)
      if cv2.waitKey(10)&0xFF==ord("d"): # 'D'ual Cam Activated
        message1="(Message) Q/R/D/S/1/2/O"
        message2="Dual Cams are activated"
        FLAG_DUAL_CAMS=True
        origin_time=time.time()
      if cv2.waitKey(10)&0xFF==ord("s"): # 'S'ingle Cam Activated
        message1="(Message) Q/R/D/S/1/2/O"
        message2="Single Cam is selected"
        FLAG_DUAL_CAMS=False
        origin_time=time.time()

  if cap1: cap1.release()
  if cap2: cap2.release()
  cv2.destroyAllWindows()

main()
```