

Drowsiness Detection and Analysis

By

Yi Shen



Flinders
UNIVERSITY

Project supervisor: Dr. Sherry Randhawa

**Submitted to the School of Computer Science, Engineering and Mathematics in
the Faculty of Science and Engineering in partial fulfilment of the requirement for
the degree of Master in Biomedical Engineering at Flinders University-Adelaide
Australia**

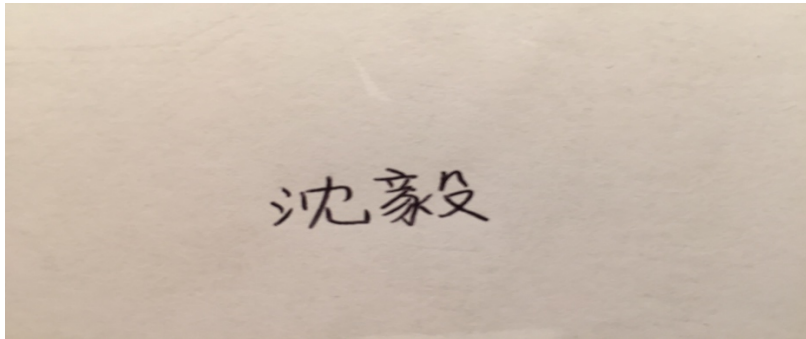
Submitted on: October 2016

Declaration

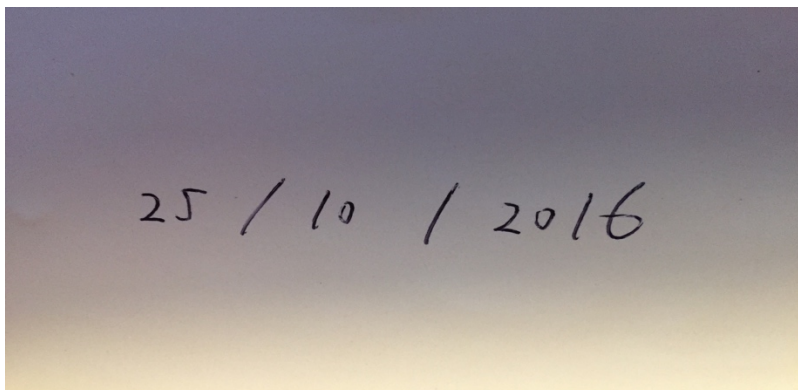
I certify that this work does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief. It does not contain any material previously published or written by another person except where due reference is made in the text.

The Matlab programing had used Matlab Libraries for the Viola-Jones, Camshifit, Surf, KLT, Correlation coefficient algorithm.

Signature:

A photograph of a piece of light-colored paper with the Chinese characters '沈彖' (Shen Hua) written in black ink in the center.

Date:

A photograph of a piece of paper with the date '25 / 10 / 2016' written in black ink in the center.

Acknowledgement

I would like to articulate my deep gratitude to my project supervisor Dr. Sherry Randhawa for her guidance, advice and constant help in the project work. I give my sincere thanks to my friends who have patiently participated in the project experiments.

Abstract

Driver fatigue is one of major reasons which causes vehicle accidents. The drivers' drowsiness detection can be described by ocular measures, such as eye blink duration and blink frequency. The aim of this project is to use ocular measures to detect and analyse drowsiness in real-time. This project consists of four parts which are eye detection, eye tracking, eye blink detection and drowsiness detection respectively. Eye detection used the Viola-Jones algorithm which is based on cascade Haar classifiers to identify region of interest from images. Eye tracking combined the Camshift algorithm with KLT algorithm to track eye pair of participants in the real-time detection. Eye blink detection used the correlation coefficient algorithm to calculate correlation scores between the open eyes template image and the region of interest of every frame. The analysis of eye blink detection measured the blink duration, blink frequency in drowsy, alert, normal state. Also, it compared the differences of blink duration and frequencies in different drowsiness states. Furthermore, it analysed factors which affect the accuracy of drowsiness detection.

Key words: *Drowsiness detection, Classifier, Camshift, KLT, Blink frequency, Blink duration.*

Content

Introduction.....	7
Chapter 1 Literature review	10
1.1 Parameters for the evaluation of eye blink detection	10
1.2 Algorithms for object tracking.....	12
1.2.1 The Meanshift algorithm.....	12
1.2.2 The Camshift algorithm	13
1.2.3 Improvements based on Camshift algorithm	14
1.2.3.1 Motion segmentation method.....	14
1.2.3.2 Combination with Kalman filter	14
1.3 Feature detection	15
1.3.1 The basic principal of feature detection.....	15
1.3.2 Face and eye detection.....	16
1.3.3 Eye blink detection	20
1.4 Summary and evaluation of previous works	22
1.5 Material chosen and further considerations.....	23
Chapter 2 Introduction of software and hardware for this project.....	25
2.1 Software: MATLAB.....	25
2.2 Hardware: Raspberry Pi Board and Raspberry Pi Camera.....	25
Chapter 3 Procedures of project.....	27
3.1 Face detection.....	27
3.2 Eye detection	29
3.2.1 Single eye detection.....	29
3.2.2 Separate eye detection	31
3.2.3 Eye pair detection	32
3.2.4 The comparison of different types of eye detection	33
3.2.5 The improvement of eye detection	35
3.2.6 Real-time eye detection	37

3.3 Eye tracking.....	38
3.3.1 Histogram based object tracking algorithm.....	38
3.3.2 Continuous adapt mean shift object algorithm.....	40
3.3.3 Kanade-Lucas-Tomasi (KLT) algorithm.....	42
3.3.4 Speeded up robust features (SURF) algorithm.....	43
3.3.5 The combination between KLT and Camshift algorithm.....	46
3.4 Eye blink detection.....	49
3.5 Drowsiness detection.....	51
Chapter 4 Results analysis.....	52
4.1 Methodology of eye blink experiment.....	52
4.2 Cases of eye blink experiments.....	52
4.3 Results of drowsiness detection.....	57
4.4 Error analysis.....	61
4.4.1 Factor cases.....	61
4.4.2 Factor analysis.....	66
Chapter 5 Discussion.....	68
Chapter 6 Conclusion and further work.....	71
Reference.....	72
Appendix.....	74

Introduction

Drowsiness is a normal physical reaction when people feel fatigue or have a lack of sleep. This physical reaction substantially increases the risk of serious accidents when people are driving. According to the statistical analysis which reported by National Highway traffic safety Administration of America, about 10 thousand car accidents happened in 2015 which resulted in 1,550 deaths, 71,000 injuries, \$12.5 billion in monetary losses. Car accidents should be avoided by detecting and alarming drivers in a drowsy state.

The study of drowsiness detection can be achieved by drivers' behavior measurement, physiological measurement and ocular measurement. Drivers' behavior measurement includes the pressure on the acceleration pedal, movements of steel wheels and deviations from lane position. They are constantly monitored and any changes of these measurements which over the special threshold value, indicated that the driver is in drowsy. While, Sahayadhas *et al* (2012) proposed that drivers' behavior measurement, such as steering wheel movements and deviations of lane position are hard to estimate the drowsiness level of drivers. The steering wheel movement depends on the geometric characteristic of road. However, it is less depended on the kinetic characteristics of vehicles. Moreover, the deviation of lane position depends on external factors, such as lighting conditions, road marks and climate. In addition, the deviation of lane position could be caused by any types of impaired driving such as drivers are under the influence of drug and alcohol. The drivers' behavior measurement cannot be used in this project as many of them are also affected by factors such as geometric characteristic of road, lighting conditions, drivers' health.

Physiological measurement includes physiological signals, such as EOG (Electrooculogram), EEG (electroencephalogram), ECG (electrocardiogram), is used to evaluate the correlation score between drivers' drowsiness levels. However, Sahayadhas *et al* propose that EEG and EOG are good measurement to estimate sleep states of drivers and their signal changes with brain physical reactions. While, EEG and EOG signals are difficult to be measured in field settings. Meanwhile, the method of measuring EEG/EOG signals cannot provide a real-time monitoring for levels of drowsiness all the time. Anderson (2013) refers that EEG/EOG measurement present a snapshot of performance decrement and cannot provide a continuous

marker of dynamic drowsiness. EEG/EOG measurement also require a quiet and sterile environment to reduce potential confounds from noise. Therefore, physiological measurements cannot be used in this project as they cannot provide continuous signal for real-time detection. Moreover, he claimed that the requirement of physiological measuring cannot be easily achieved, expect in a certain experimental environment.

Ocular measurement includes eyelids' movements, eye closures, eye blinks. These symptoms of drowsiness can be monitored through a camera. Ocular behaviors, such as eye movements, eye blinks are directly linked to the central nervous system function which plays an important role in attention and vigilance. Drowsiness is unstable and rapidly fluctuating state of drivers' eye motions. Any changes in frequency, duration, speed of eye blinks, angle and amplitude of eyelid movements are good parameters to reflect the drowsiness levels of drivers. Based on these parameters of drowsiness, there are two common methods to detect drowsiness levels. One is to use high speed camera to capture fast eye blink reaction and measuring the duration and frequency of eye blinking. Another one is to use advanced assistance system (like Eye tracking II system) for measuring amplitude and angle eyelid movement. However, Benedetto (2011) proposes that most systems can deal with motion analysis and they require the use of rather expensive equipment and high expenditure video cameras. Meanwhile, high speed camera is not advisable due to its expensive price. Chau (2005) proposes that using a microprocessor (like Raspberry pi, Arduino) with a USB camera to create a drowsiness detection system for providing real-time monitoring is an inexpensive and more effective approach. In this project, ocular measurement is easier to be measured by detection system which integrate a microprocessor with a USB camera. The price of a microprocessor with a USB camera is much cheaper than that of advanced assistance system (such as the Eye tracking II system).

In the current, the Optalert Company has already designed and manufactured the "Eagle" product to detect the drowsiness level of drivers during driving. The technology of Eagle is based on detecting the JDS (Johns Drowsiness Scale) of drivers. The "Eagle" product is incorporated in vehicle. Drivers can see their drowsiness scores on the dashboard. Seeing machine is used to measure people's facial expression and head movements and it is designed based on image processing technologies. It consists of an intelligent camera which can detect

the facial movements of drivers and monitors driver fatigue and distraction events in real-time. The CRC Mining Company has designed Smart Cap which is a baseball cup with an integral sensor. The Smart Cap detects EEG signal of drivers to determine their drowsiness levels during driving. These three types of drowsiness products are good references to this project as they provide information about drowsiness detector's development.



Figure 1-1 Smart cap (Coxworth 2012)

The aim of this project is to use ocular measures to detect and analyse drowsiness of people in real time.

Chapter 1 Literature Reviews

1.1 Parameters for the evaluation of eye blink detection

There are 8 parameters for the evaluation of eye blink detection which are Inter Event Duration (IED), percent time with eye closed (%TEC), Blink Total Duration (BTD), Amplitude Velocity Ratio (AVR), Percent Long Closures (%LC), Duration of Ocular Quiescence (DOQ), Johns Drowsiness Scale (JDS) respectively. The Inter Event Duration is blink duration measured from the point of maximum closing speed to maximum opening speed. Percent time with eye closed is the proportion of time eyes are closed which determined from the velocity of eyelid during eye closure. Blink Total Duration is measured from the start of eye closing to the re-opening of eye. Regarding to Amplitude Velocity Ratio, it is the ratio of the maximum amplitude to maximum speed of eyelid movement for eye closing or eye opening. Percent Long Closure is proportion time of eye fully closed. Duration of Ocular Quiescence is duration of no movement between eye lid and ocular movements. John drowsiness Scale uses the weighted value of ocular measures to indicate the level of drowsiness.

Wilkinson and Jackson (2011) implement an Oxford Sleep Resistance Test (OSLER) which used as a gold standard to indicate drowsiness level and determine the ocular measurement cut-off value.

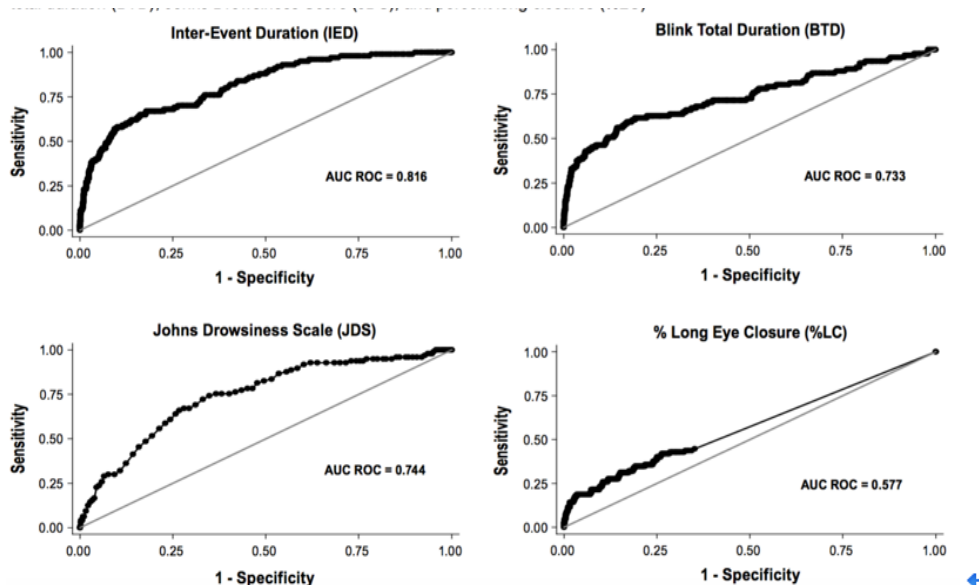


Figure 1-2 ROC curves of ocular variables for identify frequent missed signals (Wilkinson 2011)

As for the statistical analysis for derivation of cut-off value, the Receiver Operating Characteristic (ROC) curve analysis which was conducted for each ocular variable to assess the ability of each parameters to identify missed OSLER signals occurring during any 1-min bin. Sensitivity measures the proportion of positives that are correctly identified. Specificity measures the proportion of negatives that are correctly identified.

They find that IED parameter accurately identified drowsiness-related errors with ROC curve area under the curve (AUC) of over 0.8 when detecting frequent missed signal. While other parameters do not have same performance as that of IED as shown in Figure 1-2. With regards to BTD, the drowsiness-related errors with ROC curve area under the AUC is 0.733, which is nearly same as that of JDS (0.744). The %LC is poor to identify frequent missed signal during the test, AUC is 0.577.

Moreover, the proportion of time with eye closed (PERCLOS) is founded as a variable to accurately identify behavioral lapses in drowsy participants with high accuracy for long-time test. Therefore, PERCLOS is considered as a better measurement for real-time monitoring of drowsiness in previous studies. However, they found that the measurement of eye blink duration such as IED and BTD could be better predictors of behavioral lapses than the PERCLOS as IED and BTD produce the highest sensitivities and specificity.

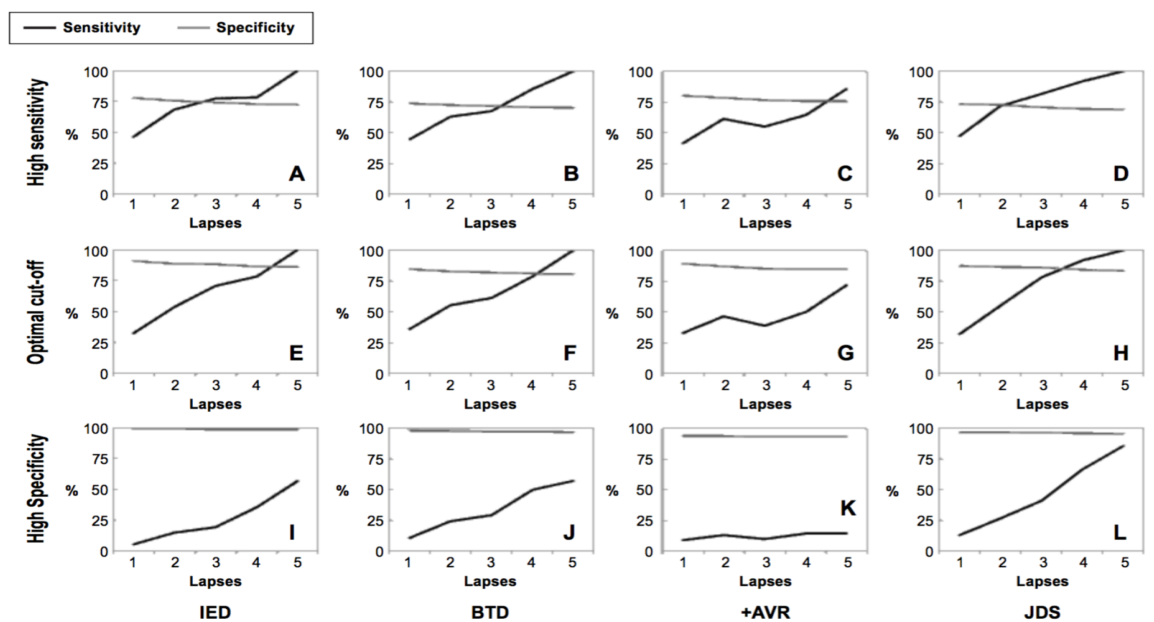


Figure 1-3 Ocular variables correctly identifying increasing numbers of drowsiness-related PVT lapses (1–5) using OSLEDERIVED cutoff values at high sensitivity, high specificity. (Wilkinson 2011)

The Figure 1-3 shows the sensitivity and specificity of the ocular variables (IED, BTM, AVR, JDS) correctly identify increasing numbers of lapses. Wilkinson and Jackson proposed that high sensitivity with lower specificity is important factor in real-time monitoring of drowsiness, as it reduces the risk of missing episodes of drowsiness. The optimal cutoff value for IED achieve a sensitivity of 71% for detecting 3 lapses on the task, as shown in A and E of Figure 1-3. Increasing to 100% for detecting 5 lapses and maintain a lower specificity (66%) as shown in I of Figure 1-3. Similarity, the optimal cutoff value for BTM achieve a sensitivity of 70% for detecting 3 lapses on the task, as shown in B and F of Figure 1-3. Increasing to 100% for detecting 5 lapses and maintain a lower specificity (67%) as shown in I of Figure 1-3. Moreover, JDS has similar area of ROC under the area of curve to BTM as shown in Figure 1-2, it maintains good sensitivity (81%)for detecting 3 lapses. While it has high specificity (76%) for detecting lapses at range of cutoff value, as shown in the D, H, L of Figure 1-3. Furthermore, AVR has lower sensitivity than other measurement for detecting lapses at a range of cutoff value when maintaining a good specificity, as shown in the C, G, K of Figure 1-3. Therefore, they proposed that eye blink duration such as IED and BTM are more reliable measurements to detecting drowsiness.

1.2. Algorithms for object tracking

Object tracking is a process of following every movement of interest object in real time. Two common algorithms are widely used for object tracking which are Meanshift algorithm and Camshift algorithm respectively. The Camshift algorithm is an adaption of Meanshift algorithm for object tracking such as face tracking, eye tracking.

1.2.1 The Meanshift algorithm

Fukunaga and Hostetler proposed that the Meanshift algorithm could be used for the interest object tracking. Mean shift is a procedure of locating the maximal density of model. The Meanshift algorithm consists of kernel function. As for the kernel function, it is a function which estimate the density of model detected. The formula of kernel function as follows:

$$f(x) = \sum_i K(x - x_i) = \sum_i k\left(\frac{\|x - x_i\|^2}{h}\right) \quad (1.1) [23]$$

where x_i are input samples, h is the bandwidth of kernel function. k is the function of model density estimation.

The Meanshift algorithm creates a confidence map in the new image which is based on the histogram of the interest object of the previous image. The confidence map is a probability density function on the new image and it assigns possibilities to each pixel of the new image. These probabilities represent pixel color occurring in the interest object of the previous image. Find the peak of confidence map near the old position of the interest object. As object moves, the location and representation of object is calculated by the probability of the interest object and estimated by kernel function (Fukunga 2000).

The major advantage of Meanshift algorithm is that the detection of tracking object is not affected by the motion of camera and fast movement of object. However, a major drawback of this algorithm is that it does not adapt to movements of target, especially when target experiences significant change in color, size, and shape, as the Meanshift algorithm is depend on the static pixel distribution (Allen 2004).

1.2.2 The Camshift Algorithm

Allen (2004) proposes the Camshift algorithm for object tracking which used the continuously adaptive probability distribution rather than static pixel distribution. The Camshift algorithm consists of 3 major components which are back-projection, ratio histogram and computation of image moments respectively. In the back-projection part, it sets the region of interest and calculate its pixel distribution to the whole image by using the histogram back-projection. In the ratio histogram part, it uses the ratio histogram formula with kernel function to calculate the new pixel distribution of the whole image in consecutive frames. As for the computation of image moments, it calculates the movement and orientation of target, linking motions of target until the end of tracking.

Allen claims that the Camshift algorithm has better performance than the Meanshift algorithm in real-time tracking the object, because it recalculates the pixel distribution to the whole image if target has any motions. However, it easily fails to track target in a various color background and influences driver drowsiness monitoring, as the Camshift algorithm relies on pixel possibility calculation. "Tracking target in a various color background may fail to detect and

track target, as the possibility of each pixel of target is similar with that of background” is claimed by Allen.

1.2.3 Improvements Based on Camshift Algorithm

1.2.3.1 Motion segmentation method

Emami (2011) claims that the performance of Camshift algorithm for tracking object could be improved by combining with the motion segmentation method for eliminating the background color effect on tracking performance. The motion segmentation method is based on Camshift algorithm which uses the imaging differencing to detect the target object. This method computes the pixel by pixel intensity difference of the current frame and the reference background model and then extract the frame’s foreground from its background for the result image. The result image shows temporal changes. After some simple morphological operations (dilation and erosion) on consecutive images, the target region can be extracted from background of continuous images.

Emami proposes that this method reduces the undesirable background pixels’ effect on the tracking object. Most of them are included in the search window of Camshift and they are removed after the motion segmentation phase which increases the accuracy of tracking. Therefore, using the Camshift algorithm combined with motion segmentation can get rid of the background color influence on tracking performance.

1.2.3.2 Combination with Kalman filter

“Combing the Camshift algorithm with Kalman filters could increase the tracking performance of object tracking” which is proposed by Zhang (2009). He suggested that the accurate tracking rate can be improved by using the measurement and state-space equation. One Kalman filter is used to search the true maximum over the local one for accurately search the position of target. Using Kalman filter to define the measurement and state-space equation. The formula of state-

space equation is defined as follows:

$$\begin{bmatrix} x_{n'} \\ y_{n'} \\ u_{x'} \\ u_{y'} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x_n \\ y_n \\ u_x \\ u_y \end{bmatrix} + w_n \quad (1.2) [19]$$

The formula of measurement as follows:
$$\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} * \begin{bmatrix} x_n \\ y_n \\ u_x \\ u_y \end{bmatrix} + v_n \quad (1.3) [19]$$

Where $k \geq 1$, x_n and y_n are coordinates of the center point of search window, u_x and u_y are displacements of the target. T is the interval between frames. w_n is white Gaussian noise with variance Q , v_n is white Gaussian noise with variance R . Q, R is constant value.

Another Kalman filter is used to define the width and height of the search window. We can accurately find out the centroid and size of the detection window by combining these two Kalman filters.

The state-space formula defined as:
$$\begin{bmatrix} b_{n'} \\ h_{n'} \\ r_{b'} \\ r_{h'} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} b_n \\ h_n \\ r_b \\ r_h \end{bmatrix} + U_n \quad (1.4) [19]$$

The formula of measurement as follows:
$$\begin{bmatrix} b_c \\ h_c \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} * \begin{bmatrix} b_n \\ h_n \\ r_b \\ r_h \end{bmatrix} + V_n \quad (1.5) [19]$$

Where b_n and h_n represent width and height of the search window respectively, r_b and r_h represent scale ratio of width and height to the size of detection window. b_c and h_c represent current width and height of the detection window respectively. V_n and U_n represent white Gaussian noise.

He proposes that this improvement algorithm enhances the tracking performance. However, it is not effective to track the target with orientation change, such as drivers have head movements during driving, because this algorithm does not consider the orientation of detection window has variation which accompanies with tracking (Zhang 2009).

1.3. Feature detection

1.3.1 The basic principal of Viola-Jones algorithm

Viola and Jones (2001) proposes an algorithm for rapid object detection. Viola-Jones algorithm is a machine learning algorithm which can process images rapidly and has a higher object detection rate. The Viola-Jones algorithm consists of 3 components which are the region of interest identification, adaptive boosting algorithm, the cascade classifier system respectively.

There are two steps of the identification of the region of interest. Firstly, it calculates the pixel sum within feature region of images. Secondly, it compares the sum pixel within feature regions and calculate their differences. The region of interest is located by calculating differences of sum pixel between feature regions.

Adaptive boosting algorithm is a kind of machine learning algorithm which help classifier to identify the region of interest from images. Given a training set of positive and negative images and feature settings, this algorithm is designed to select and separates the positive and negative examples of features.

The cascade classifier system consists of cascading classifiers in series. Meanwhile, it discards background regions of images and reduces time on computing the object-like regions. To be exact, the cascade classifier system achieves a higher detection performance as classifiers reject many of the negative sub-windows and select positive instances during detection. The positive result from the first classifier triggers the next classifier which has been adjusted (setting threshold to minimize false negative) to have a higher detection rate. The positive result of previous classifier passes one by one until it reached to the last classifier. The overall decision process of the cascade classifier as shown in Figure 1-4.

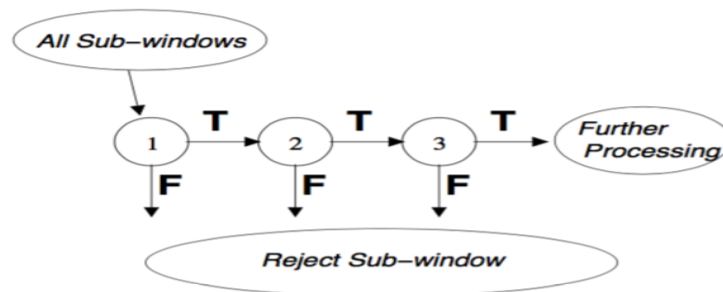


Figure 1-4 The decision process of the cascade classifiers (Viola&Jones 1998)

They claim that the Viola-Jones algorithm minimizes computation time on the object region and achieves a higher detection accuracy, it is also an effective algorithm for detecting interest objects as it does not need to scale the image itself for detecting feature regions. Moreover, generic detection scheme is trained for detecting other types of features such as stop sign, aircraft.

1.3.2 Face and eye detection

Guerrero (2006) proposes that the face recognition performance is obtained by using the face detector which eliminate the background region and cropping the face region from video frames. He used the Viola-Jones algorithm to detect the face region on the Lena image. The cascade classifiers have been trained to classify the face region and return as a sequence to the bounding box. Moreover, cascade classifiers also obtain the location of face region which are collected and identified. As the result, the bounding box in the image that are likely to contain the region of interest (face region). The result of face detection as shown in Figure 1-5.



Figure 1-5 The face detection on Lena image (Paolo 2012)

Moreover, Vukadinovic (2006) builds a detector system which labels the facial feature points in the face image automatically. The detector is based on the Viola-Jones algorithm and it consists of cascade classifier which is trained by GentleBoost algorithm. One advantage of Gentleboost algorithm is that it can shift and scale chosen filter by pixel. Another advantage is that it is simple to implement and numerically robust. It had been experimentally shown to outperform other boosting variants for the face detection.

Bhaskar (2003) proposes that eye detection can be achieved by marking feature points on the eyes and tracking their positions in real-time. This method is based on the Kanade-Lucas-Tomasi (KLT) algorithm. The principal of this algorithm is to use the spatial intensity information to direct search the position that yields the best match. It generalizes usage in dealing with the arbitrary linear distortion of the image, such as head rotation.

However, Thiyagarajan (2014) proposes that the KLT algorithm only relies on the local information that derived from small window surroundings each of points of interest. The feature points will move out of the local window during a large motion of target which make the KLT algorithm hard to find feature points. The KLT algorithm rests on three assumptions which are

brightness constancy, temporal persistence, spatial coherence respectively. As for brightness constancy, the pixel of an object in the image does not change as the object move from frame to frame. As for temporal persistence, the scale motion of object in the image relative to the temporal increment that does not move too much from frame to frame. As for spatial coherence, neighbor points which in the same object belong to same surface, have same motion and project to the neighbor points on the image plane.

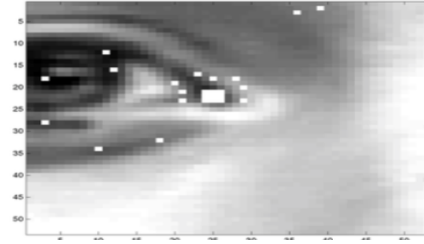


Figure 1-6 Eye detection by using the KLT (Vukadinovic 2006)

Guerrero proposes that eyes exhibit strong vertical edge between eye white and iris. He applied the Sobel operator into a face image for obtain the vertical edge. The Sobel operator calculates gradients of the intensity of image at each point. As the result, the region of constant image intensity set to zero and pointed out the edge by the transition between dark and bright image intensity Afterwards, applying the image projection function to detect the vertical edge. The projection function is defined as follows:

$$IPF_v(x) = \int_{y_2}^{y_1} I(x, y) dy$$

$$IPF_h(y) = \int_{x_1}^{x_2} I(x, y) dx$$

where: $I(x, y)$ is the intensity of a pixel with location (x, y) , $IPF_v(x)$ and $IPF_h(y)$ are the vertical integral projection function and horizontal integral projection function in intervals $[y_1, y_2]$ and $[x_1, x_2]$ respectively.

Given a threshold to define the vertical boundary in the image:

$$\theta_v = \max \left\{ \left\| \frac{dIPF_v}{dx} \right\| > 255 \right\} \quad (1.6) \quad [17]$$

Where: θ_v is the set of vertical points which vertically divides the image into different feature regions.

Applying the integral function to detect the vertical locations of eyes in the image, as eye is located at the upper half of the image the y_2 , y_1 are middle and top of image. The max value of integral projection function represents the vertical location of eye. The x coordinates of eye located defined as:

$$cte = scale\ factor * width_image \quad (1.7) [8]$$

$$x_{left1} = cte, \quad x_{left2} = \frac{width_image}{2} - cte, \quad (1.8) [8]$$

$$x_{right1} = \frac{width_image}{2} + cte, \quad x_{right2} = width_image + cte, \quad (1.9) [8]$$

The scale factor is decided by head rotations. After the eyes location confirmed, designing a window to enclose the region of interest (a pair of eyes).

The shape, texture, animation parameters have changes during the head rotation. Animation parameters consist of local and global motion parameters. The global parameters deal with the location, rotation and translation of the face, from which the angle of head rotation can be obtained. The distance between the mid-point of two eyes is calculated by the following formula: $d_0 = f * \sin \theta$. (1-10) [8]

Where f is constant value which is related to the radius of the head.

$$\text{The middle position of eye in the video frame defined as; } d = \frac{width_image}{2} + d_0. \quad (1-11) [8]$$

The horizontal and vertical position of eye will also have a change as the head movement and defined as follows:

$$z_0 = 0.2 * \left(\frac{width_image}{2} - d_0\right), \quad z_1 = 0.75 * \left(\frac{width_image}{2}\right) \quad (1-12) [8]$$

$$x_{left1} = \begin{cases} z_0 & \text{if } \theta < 0 \\ z_1 + d_0 & \text{if } \theta > 0 \end{cases} \quad x_{left2} = d - z_1 \quad (1-13) [8]$$

$$x_{right1} = d + z_1, \quad x_{right2} = \begin{cases} width_image - z_1 - d_0 & \text{if } \theta < 0 \\ width_image - z_0 & \text{if } \theta > 0 \end{cases} \quad (1.14) [8]$$

Guerrero claims that this method is not very suitable for eye detection especially when people wearing glasses, because the glasses display very strong edge and cause a strong reflection of the light. It is hard for detect the vertical edge of eye as it relies on the gradient of image intensity. Therefore, the detection accuracy rate for this method is not very high.

1.3.3 Eye blink detection

Polatsek (2013) proposes the histogram projection method for detecting eye blink. This method uses the histogram to represent the skin color of participants and it create a possibility map over the image. When people close or open their eyes, the color distribution of histograms are different. The higher sum pixel value within eye region considered as eye closed otherwise eye opened, as shown in Figure 1-7.



Figure 1-7 Back-projection image of eye open and eye close (Polatsek 2013)

The procedure of this method as follows: Firstly, the person's face is detected by using Haar cascade classifiers. Secondly, calculate the skin color histogram from the image of face region. Afterwards, normalize the histogram and keep updating and calculate the back-projection with the histogram for every input image. Using morphological operations (open and erode) to modify resulted back-projection images. Setting the threshold to increase the difference between open and closed eyelids, $threshold_{HS} = 10$ in hue-saturation and $threshold_S = 25$, where S presents 1D saturation histogram, HS presents 2D hue-saturation histogram. Lastly, calculating the average value of the probability in eye regions.

He claims that this method is not reliable as it much relies on lighting conditions of environment. Especially, many false detections may happen in the poor lighting condition.

Soukupova (2016) refers that eye blink detection can be achieved by designing a real time facial land marker detector to measure the position of characteristic points and calculating the eye aspect ratio (EAR). The characteristic points as shown in Figure 1-8.

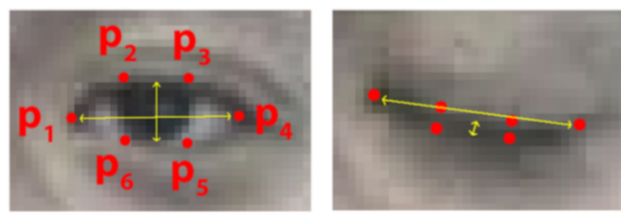


Figure 1-8 Landmarks on the opened eye and closed eye (Soukupova 2016)

The formula of EAR as follows: $EAR = \frac{\|p_2-p_6\|+\|p_3-p_5\|}{2\|p_1-p_4\|}$ (1.15) [13], where: p_1, \dots, p_6 are positions of landmarks shown in Figure 1-8. The EAR is mostly constant value when the eye is opened and closed. This parameter has a change during eye blinking. The threshold value of EAR is 0.2. If the value of EAR is less than 0.2, it means eye blink. This method gets rid of false tracking caused by facial expression, yawning or people close their eyes intentionally. The result of detection as shown in Figure 1-9.

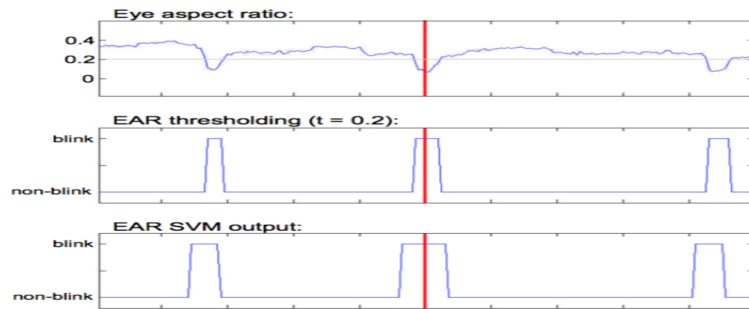


Figure 1-9 The graph of EAR and EAR thresholding and EAR SVM output (Soukupova 2016)

One limitation of this method is that the detection is partially sensitive to the head rotation. Another is that the eye blink duration need to fixed, as the duration of every user's blink differently.

Chau (2005) refers that the template matching method could be used for eye blink detection. The procedure as follows: extracting eyes region from a static facial image to create an open eyes image as a template at the beginning, then using the template image to compare with the eyes image at every frame. Afterwards, using the correlation coefficient algorithm which is proposed by Grauman to calculate the correlation score between the template image and eye

image at every frame, the formula as follows:
$$\frac{\sum_{x,y}[f(x,y)-f_{u,v}][t(x-u,y-v)-\bar{t}]}{\sqrt{\sum_{x,y}[f(x,y)-f_{u,v}]^2} \sqrt{\sum_{x,y}[t(x-u,y-v)-\bar{t}]^2}} \quad (1.16) [5]$$

Where: $f(x,y)$ is the brightness of the video frame at the point (x,y) , $t(x,y)$ is the pixel value of template image at the point (x,y) , \bar{t} is the average value of template image, $f_{u,v}$ is the average pixel value of video frame in the current search region.

The result of correlation score indicates that the similarity between open eye template images and search region in every frame. The correlation score normally between -1 and 1. If correlation score is close to 1, it indicates a high similarity. If correlation score is close to 0, it

indicates a low similarity between template images and the detecting region at every frame. In the normal open state of eyes, the correlation score is about 0.85 to 1.0. The correlation score drops down to 0.5-0.55. during eye blinking. The result of eye blinking detection as shown in Figure 1-10.

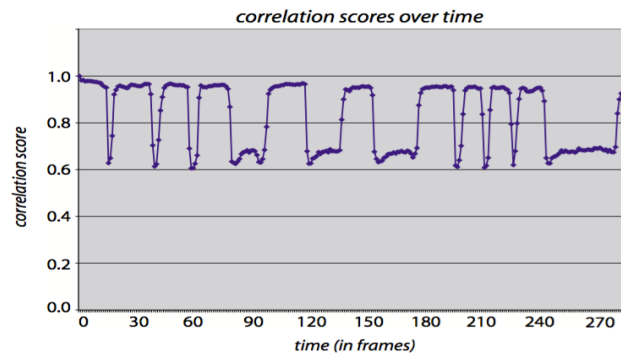


Figure 1-10 The result graph of eye blinking detection (Chau 2005)

Chau suggests that this method performs a higher detection accuracy in normal lighting condition. However, people who are wearing glasses may disturb eyes' tracking performance.

1.4. Summary and evaluation of previous works

Drowsiness driving can be decreased by designing a portable machine to detect the drowsiness levels of drivers in real time. Ocular measurement directly reflects drowsiness symptoms of drivers and easy to be detected. Compared with ocular measurement, driver's behavior measurement, such as movements of steel wheels, deviations from lane position cannot directly reflect drowsiness levels of drivers. As they can be caused by any types of impaired driving or drivers under the influence of alcohol or drugs. Physiological measurement cannot provide continuous monitoring for drowsiness detection as signal artifact. Total eye blink duration and the percentage of eye closure are good parameters to reflect the drowsiness levels of drivers which has been demonstrated by Wilkinson and Jackson.

To achieve drowsiness detection, it needs to detect eye blinks and track movements of eyes. Regards to eye tracking, the Meanshift algorithm for feature tracking is not useful when the tracking object experiences size, color, orientation changes. Compared with the Meanshift algorithm, the Camshift algorithm has a good tracking performance when the tracking object experiences color or size changes. However, it easily fails to track the object in a various color

background. This problem was solved by motion segment method which is proposed by Emami. However, it ignores the undesirable background pixels' effect on the tracking object which may decrease tracking accuracy.

The Viola-Jones algorithm is useful for detect facial features with a high accuracy (about 95%). As it can calculate the pixels within the interest object, then learning and classify target from the background in less time. The KLT algorithm which is referred by Vukadinovic is to make facial feature points and track their positions. However, this method has 3 limitations which are brightness constancy, temporal persistence, spatial coherence respectively. The back-projection method which is proposed by Polatsek is influenced by lighting conditions. If the lighting condition of environment is too strong or too weak, it will cause many false detections. The method of Soukupova which uses facial points to calculate the eye amplitude rate. The accuracy rate of eye blink detection will be low if drivers have a head rotation. The template matching method is a good for detecting eye blinks with a high detection rate. The correlation score is computed from frame to frame which reflects the blink duration of drivers.

1.5. Material chosen and further considerations

Two types of portable microprocessor are commonly used for making a detector which are Raspberry Pi broad and Arduino broad. There are 2 major advantages of Raspberry Pi over Arduino broad. Firstly, the CPU of Raspberry Pi is 700MHZ which is much higher than that of Arduino (20MHz). Secondly, Raspberry Pi broad has well performance in video recording and imaging processing aspect, as it has good video controller which can emit TV solution such as HD, Full HD. This project is achieved by using a Raspberry pi broad equips with a USB camera (Raspberry pi Camera) as hardware and MATLAB as software, because the Raspberry pi is a kind of card-size functional computer board which can promote programming languages like C, C++, Java, MATLAB... The Raspberry pi as shown in Figure 1-11. The Raspberry pi camera is a kind of USB camera which is compatible with the Raspberry pi board. The raspberry pi camera as shown in Figure 1-12.



Figure 1-11 Raspberry pi board



Figure 1-12 Raspberry pi camera

It is a good choice to select the Viola-Jones algorithm for facial feature detection. To get rid of noise effect on the feature tracking, combining the Kalman filter with the Camshift algorithm. The template matching method for eye blink detection can compute the correlation score between the open eye image and eye images at every moment, it can directly reflect the drowsiness level of drivers in real time. The distance between drivers and the camera is an important factor which affect the accuracy of eye blink detection. If the driver moves close or away from the camera, the eye image captured will be different size. This problem could be solved by converting every eye image at each moment into the same size. Furthermore, head rotation issue is also need to consider, it could be solved by using the method which is proposed by Guerrero [8]. The detection rate is also influenced by lighting conditions. This issue will be considered in further works.

Chapter 2 Introduction of software and hardware for this project

2.1. Software: MATLAB

The MATLAB software provides a Multi-paradigm numerical computing environment and programming language for solving engineering and mathematical problems. In numerical computing aspect, it allows matrix manipulation, plotting of function and data, creation of user interfaces, implementation of algorithms. In engineering programming aspect, it provides optional toolboxes, such as the Simulink toolbox which can simulate the designing dynamic and embedded systems. In this project, for software aspect, it mainly used image processing toolbox and computer vision toolbox of MATLAB.

As for image processing toolbox, it provides a set of reference-standard algorithms, functions and applications for image processing. It can perform image segmentation, image enhancement, geometric transformation, image registration, noise reduction of image. Moreover, the image processing toolbox supports a various set of image types, including high dynamic range, gigapixel resolution image, JPEG images, bmp images etc. the visualization function of this toolbox can explore images and videos, examine a region of pixels, adjust color and contrast, create contours or histograms, and manipulate region of interests (ROIs).

As for computer vision toolbox, it provides algorithms, functions and applications for designing and simulating video processing systems. It can achieve the performance of feature detection, object detection, object tracking, video processing. Therefore, these two toolboxes of MATLAB are very helpful for this project.

2.2 Hardware: Raspberry Pi Board and Raspberry Pi Camera

The Raspberry Pi Board is a credit card-size functional computer which can be written programming code into the board for project. The Raspberry pi 3 board mainly consists of a 1.2GHZ 64-bit quad-core CPU, 1GB RAM, 4 USB ports 40 GPIO pins, Full HDMI port, Ethernet port, Camera and Display interface, Micro SD card slot, Video Core IV 3D graphics core. It has well performance in image and graphical processing, video showing. The Raspberry pi 3 is equipped with 1.2 GHz 64-bit quad-core ARM Cortex-A53 processor which has 10

times computing performance of that of Raspberry pi 1. The video controller of Raspberry pi 3 can emit modern TV solution such full HD, HD or lower monitor resolution which is benefit for video recording.

Raspberry Pi camera is compatible with the raspberry pi board and it is used to take high definition video and still images. The camera module has a five-megapixel fixed-focus camera which support 1080p30,720p60 and VGA90 video modes as well as still image capture. The video recording is achieved by attaching a ribbon cable to the CSI port on the Raspberry pi board.

Chapter 3 Procedures of the project

This project is divided into 3 major sections which are facial features detection, eye tracking and real-time eye blink detection respectively. In facial feature detection section, it mainly explains how to detect people's face, eyes in static images, problems associate with eye detection (false detection), reasons of false detection. In the eye tracking section, having tried 4 algorithms for eye tracking which are Histogram based algorithm, Continuous adaptive mean shift algorithm, Kanade-Lucas-Tomasi algorithm, SURF algorithm respectively. It discusses advantages and disadvantages of each method. In the real-time eye blink detection section, it uses the correlation coefficient algorithm to detect eye blink and it discusses limitations of this algorithm. In the drowsiness detection section, it uses the threshold of drowsiness blink duration to make judgement for drowsiness states. Procedures of the project as shown in the following flow chart.

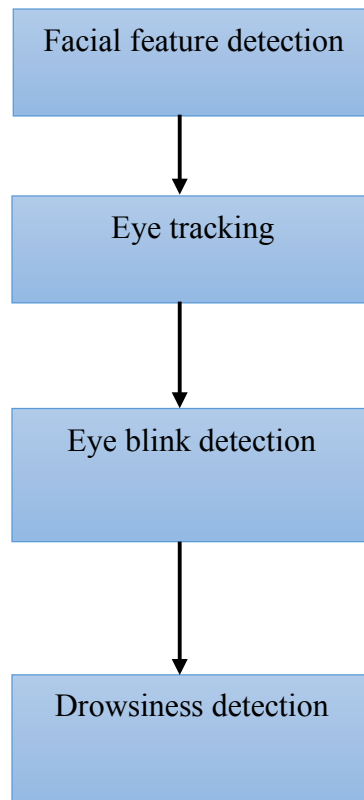


Figure 3-0 Procedures of the project

3.1 Facial feature detection

The first step of eye blink detection is detecting the face and eyes in static images. The purpose of this procedure is to let computer to identify human face and eyes in digital images. Face detection is regarded as a special case of object-class detection which is to find the locations and sizes of objects in the image belong to a given classification. The face detection is shown in Figure 3-1. The face region is within the blue bounding box.

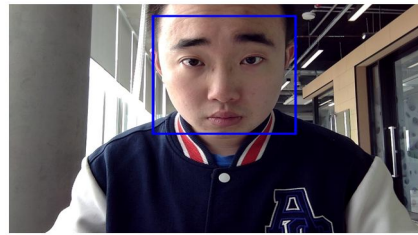


Figure 3-1 Face detection

The principal of face detection is based on the Viola-Jones algorithm. The Viola-Jones algorithm has 3 components which are feature selection, Adaptive boosting training and cascading classifiers respectively. In the feature selection, as human face all has same properties which could be matched by using Haar Features. Common properties to human face such as: the eye region is darker than the upper cheek region, nose region is brighter than the eye region, the similar location and size of nose, mouth, oriented gradients of pixel intensities. The principal of feature selection is based on calculating and comparing the sum of pixels within the region of interest and its adjacent region.

The value of rectangles is calculated with 4 array references as shown in Figure 3-2.

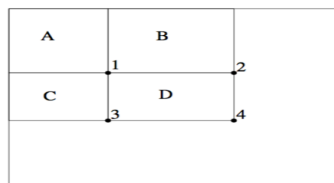


Figure 3-2 The calculation of the sum pixel within the region of interest (Viola&Jones 1998)

The value of the image at point 1 is the sum of the pixels in rectangle A. The value at point 2 is calculated as A+B and the value at point 3 is calculated as A+C, the value at point 4 is

A+B+C+D, the sum of original image values within the rectangle can be calculated as sum = A-B-C+D. The feature detection is based on the value classification of object feature.

Using the same principal, the rectangular pixel sum of image is computed in 4 array references.

$$I(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \quad (3.1) [16]$$

where $I(x, y)$ is the sum of pixel of image at the (x, y) location. $i(x, y)$ is the original image.

Using following pair of recurrences:

$$s(x, y) = s(x, y - 1) + i(x, y), I(x, y) = I(x - 1, y) + s(x, y) \quad (3.2) [16]$$

where $s(x, y)$ is the sum of cumulative row, $s(x, -1) = 0$, and $I(-1, y) = 0$.

Adaptive boosting algorithm is a kind of learning algorithm which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers. Given a training set of positive and negative image and feature, this algorithm is designed to select and separates the positive and negative examples of the rectangle feature. By setting the optimal threshold of classification, the learner determines how much samples are misclassified or classified. The threshold format as follows:

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j < p_j \theta_j \\ 0 & \text{otherwise} \end{cases} \quad (3.3) [16]$$

where f_j feature, θ_j is the threshold, p_j is parity which indicates the direction of the inequality sign.

The principal of classifier to identify the image whether includes region of interest. The cascade classifier consists of stages and each stage classifiers are weak learner. The classifier labels a sliding window which enclosed the detecting region, it checks whether the region of interest (ROI) is included in the detecting region.

If target is found in the detecting region, it is labeled as positive sample, the classifier passes the result to the next stage. Otherwise it is labeled as negative sample, the classification is completed and the detector slide the window to next location. The detector continues to slide the window until the it reports object is found and the final stage classifies the region as positive. The cascade classifiers must have a low false positive detection rate (negative sample misclassified as positive sample) and false negative detection rate (positive samples classified as negative samples). However, each stages of cascade classifiers have high false positive rate. In cascade classifier system, each classifier makes a judgement whether the image includes the

region of interest, if it contains the region of interest the result will be positive and then pass it to the next classifier to make a further judgement until to be end. Otherwise, the result is negative and put the image into reject windows. Cascaded classifiers have ability to achieve higher detection rate and low false detection rate. For example, single feature classifier achieves 100% detection rate while it has 50% false positive rate. Five feature classifiers in cascade achieve 100% detection rate and the false positive rate will drop down to 40%. Moreover, twenty feature classifiers in cascade achieve 100% detection rate with only 10% false positive rate, it is mean that more classifiers in cascade, lower false positive rate will be achieved.

3.2 Eye detection

3.2.1 Single eye detection

The second step of this section is to detect eyes, which is the fundamental step of eye blink detection. There are three methods of eye detection which are single eye detection, eye pair detection and separate eye detection respectively. Single eye detection detects single eye (left eye or right eye) in static images. The single eye detection is also based on Viola-Jones algorithm which trained a set of cascade classifier to identify left or right eye region of human face.

The left single eye detection as shown in Figure 3-3. The blue bounding box labeled around the left eye region and eyebrow region.

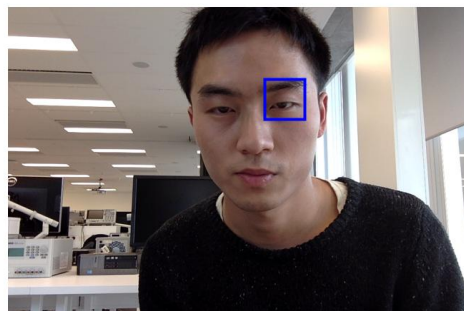


Figure 3-3 Single eye detection (open)

When eye is closed, the single eye detection as shown in Figure 3-4. It shows that the blue bounding box is around left eye region. The bounding box also includes left eyebrow region. However, eyebrow region is the irrelevant region to eye detection.

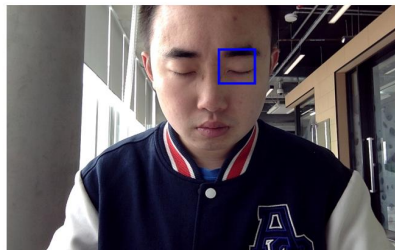


Figure 3-4 Single eye detection (closed)

The Figure 3-4 shows that the false detection about right single eye detection. The set of classifiers fail to identify the right single eye on the face, it identifies left single eye on the face instead. This drawback of single eye detection may affect eye blink detection (especially for eye tracking) as it fails to distinguish positions of left eye and right eye.

The problem associates with single eye detection is that it may fail to distinguish left and right eye during eye detection. The first reason of this result is that the classifier of Viola-Jones algorithm is weak. Secondly, as the image pixel of left and right eye region are nearly same and the feature identification of cascade classifier is based on the sum of pixel within feature regions. In the computer vision view, they are very similar with each other. Therefore, it is hard for classifier to distinguish between left eye and right eye in static images.

3.2.2 Separate eye detection

Separate eye detection detects left and right eye separately. This type of eye detection avoids false detection of left or right eye in static images. The separate eye detection as shown in Figure 3-5.

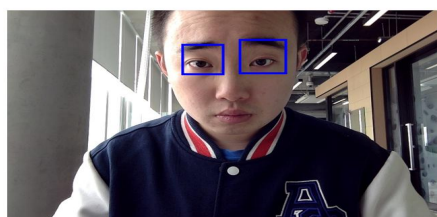


Figure 3-5 Separate single eye detection

One problem associates with separate eye detection is that it identifies eyebrow region as single eye region during separate single eye detection when eyes widely opened, it is shown in Figure 3-6.

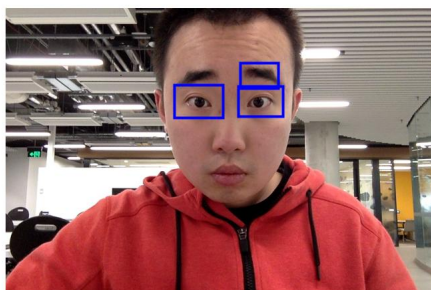


Figure 3-6 False detect eyebrow region

Another problem is same as that of single eye detection. Separate single eye detection fails to distinguish left eye and right eye separately. As it shown in Figure 3-7, the red bounding box around left eye region, the blue bounding box around right eye region. However, in the MATLAB programming, it requires that the red bonding box should around right eye region and the blue bounding box should around left eye region.

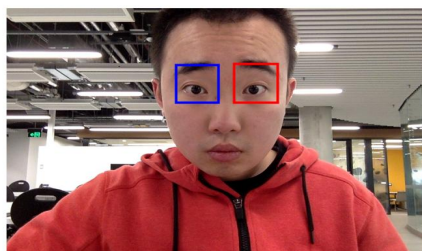


Figure 3-7 False separate eye detection

3.2.3 Eye Pair detection

Eye pair detection detects a pair of eyes on human face, as shown in Figure 3-8. It shows that the blue bounding box is around a pair of eye region which does not include eyebrow region. One advantage of eye pair detection is that it is more stable than other types of eye detection, such as single eye detection, separate eye detection, as classifiers identify eye pair more accurately than single eye detection, separate eye detection and the false detection rate of eye pair detection is very low (less than 5%). Another advantage is that the eye pair detection does

not detect eyebrow region which is irrelevant region for eye blink detection (it discusses in Discussion part). The Figure 3-8 shows that the result of eye pair detection.

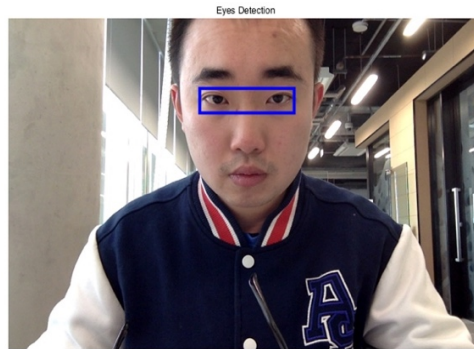


Figure 3-8 Eye pair detection

3.2.4 The Comparison of different types of eye detection

Detecting different people's eyes in the same lighting condition and comparing the accuracy of single eye detection, separate eye detection, eye pair detection. Each type of eye detection is experimented 50 times. The result is shown in Table1.

Types of eye detection	Single eye detection	Separate eye detection	Eye pair detection
Times of false positive detection	5	6	2
Times of false negative detection	3	5	3
Accurate detection rate	84%	78%	90%

Table 1 The comparison of accurate detection rate, times of false positive detection and false negative detection between different type of eye detection.

Types of eye detection	Eye pair detection		
	1	2	3
Lighting conditions			
Accurate detection rate	86.7%	93.3%	83.3%
Error rate	13.3%	6.7%	16.7%

Table 2 The result of eye pair detection in different lighting conditions

It shows that the accuracy of eye pair detection (90%) is higher than that of other eye detection types. Compared with eye pair detection, single eye detection and separate eye detection are more likely to fail to detect ROI and have false detection. Meanwhile, the times of failing detect ROI and false detection of eye pair detection is only 2 and 3 times separately.

Detecting different people's eye pairs in different lighting conditions and comparing their detection accurate. Eye pair detection is experimented 30 times in each lighting condition. The result as shown in Table 2.

The following Figures show eye pair detection in different lighting conditions.

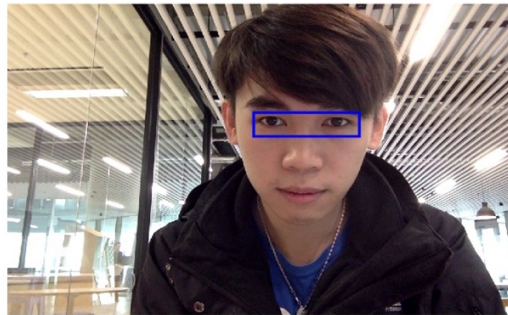


Figure 3-9 Eye pair detection in lighting condition 1

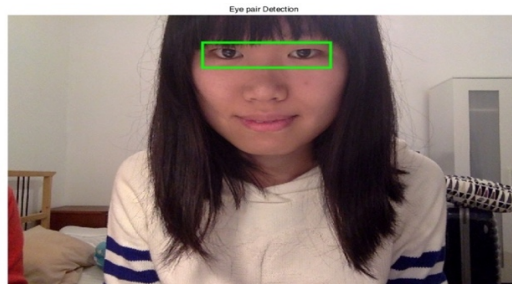


Figure 3-10 Eye pair detection in lighting condition 2

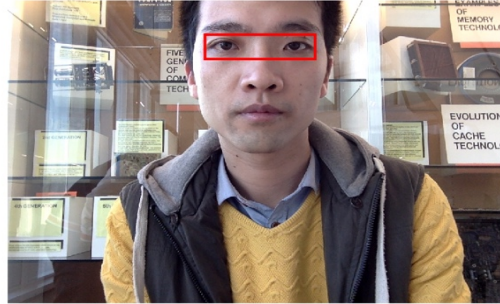


Figure 3-11 Eye pair detection in lighting condition 3

From the result of Table 2, the accuracy of eye pair detection is varied by lighting conditions, it means that eye pairs may easily fail to be detected in some lighting conditions. Therefore, the accuracy of eye pair detection in different lighting conditions need to be improved.

3.2.5 The improvement of eye detection

High accurate detection rate of eye pair detection can be achieved by training cascade classifiers to identify the interest object. Firstly, the training of cascade classifiers needs to be provided a set of positive samples and a set of negative samples. Positive samples include the region of interest (ROI). On the contrary, negative images do not include the ROI. Secondly, using the ‘Training Image Labeler’ is to label region of interest with bounding boxes. The output of Training Image Labeler used as positive samples. Lastly, acceptable detection accuracy is achieved by setting the number of stages, feature type, other function parameters.

There are two ways to supply positive images. One method is to a specify region which contains the region of interest in a whole image. Another method is to crop out the ROI from a whole image and save it as a separate image. As for negative images supply, the detector system automatically generates negative images which do not include the region of interest. As for parameters setting, increasing the number of stages or decrease the false alarm rate per stage to reduce false positive detection rate. Increasing the true positive rate to reduce the possibility of missing the detection of interest object. If the detector system need a large training set, it needs to increase the number of stage and have low false positive rate for the cascade classifier system. Otherwise, decreasing the number of stage which will increase false positive rate for the cascade classifier system.

The cascade object detector training can increase the accurate detection rate of detector system and reduce the possibility of missing the interest object or the false detection rate. However, it may take time for large training set of cascade object detector system.

In this project, using eye pair images of different people in different lighting conditions as positive images and using other images (not include eye pair regions) as negative images. Selecting bounding box for eye pair regions. Afterwards, generate a folder contains eye pair images (eye close and eye open) and added them to the pathway of MATLAB. Meanwhile, generate a folder contains negative images and store them to the same pathway as that of positive image folder. Set the false alarm rate of classifiers and cascade 5 classifiers in series. Lastly, provide positive and negative images to the cascade classifier system. The result of training needs to wait for minutes, as the process of training cascade classifier system needs to take times. A positive image sample and a negative image sample as shown in Figure 3-12 and Figure 3-13.

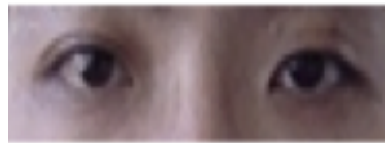


Figure 3-12 Positive image sample



Figure 3-13 Negative image sample

After, the cascade classifier is trained to recognize positive images and negative images, detecting different people's eye pairs in different lighting conditions and comparing their detection accurate. Eye pair detection is implemented 40 times in each lighting condition.

The Table 3 shows the improvement result of eye pair detection in different lighting conditions.

Types of eye detection	Eye pair detection		
	1	2	3
Lighting condition	1	2	3
Detection accuracy	92.5%	97.5%	90%
Error rate	6.5%	2.5%	10%

Table 3 The improvement result of eye pair detection in different lighting conditions

From Table 3, it shows that the accuracy of eye pair detection is higher than that of before. For example, the accuracy of eye pair detection in the lighting condition 1 (92.5%) is higher than that of before (86.7%).

3.2.6 Real-time eye detection

This step is to make a real-time eye detection system, install a 'wecam' camera software on MATLAB for capturing eyes action at each frame. However, it exists problems which associate with real-time eye detection. During the real-time detection, false detection appears and generates extra bounding boxes around face as frames go on, it is shown in Figure 3-14.

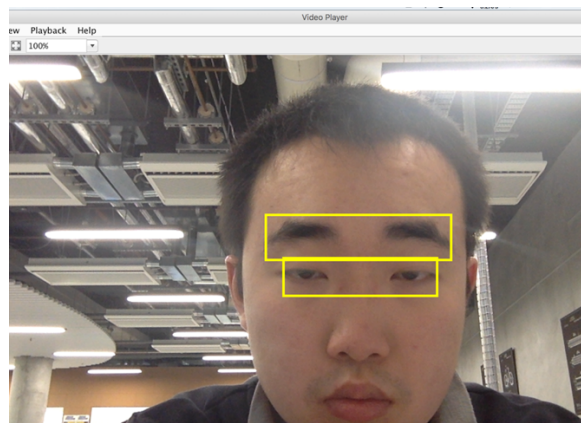


Figure 3-14 False detection on video

When head has a movement, the eye pair detector will not only false detect the object but also lose tracking the region of interest (ROI).

Without the tracking function, the eye pair detector cannot catch up with the speed of video moving as the cascade classifier of detection system need time to make a judgment whether each frame has the region of interest. The time interval from frame to frame is too small to make a judgement. In addition, the detection system need more time to relocate the position of interest object and recalculate pixel value within of ROI, that is a reason of failing detect the interest object when head has a movement. Moreover, eyebrow region which has been false detected looks like eye closed image in the computer vision view. Each Haar classifier is weaker learner to identify and distinguish region of interest from each frame. Furthermore, when head has a rotation, the detect system may fail to detect the interest object as it moves out of the view of camera. Therefore, the possibility of miss target and false detection rate will increase, if the eye pair detector do not have a tracking function.

3.3 Eye tracking

As the result of real time eye detection (without tracking function), real-time eye detection is likely to fail to identify the interest object as video frames move on. It is known that the eye detection without tracking function is not stable for real-time eye detection. Four algorithms of object tracking are experimented and compared which are histogram based object tracking algorithm [1], Camshift algorithm [1], Kanade-Lucas-Tomasi (KLT) algorithm [14] [20] [22], Speed up robust feature (SURF) algorithm [9] [25] respectively. Their advantages and disadvantages are discussed and evaluated. Eye tracking function is achieved by using KLT algorithm incorporate the Camshift algorithm.

3.3.1 Histogram based object tracking algorithm

The histogram based object tracking algorithm is based on the pixel histogram of the characteristic object of interest object to track the interest object. The first step of histogram based tracking algorithm is that load initial frame of image sequence and then choose the region of interest (ROI) in the initial frame. Using bounding box to select the ROI which is represented in the appropriate feature space.

Feature space represents the pixel value of the target. The procedure of representing the target as follows: Firstly, finding the bin of histogram which belongs to the ROI. Each pixel of ROI

has its own RGB value and they are quantized by the size of each bin. For example, the size of target is n . As the pixel value of red, green, blue range from 0 to 255, the size of red, green, blue component is equal to $\frac{255}{n}$. The 2D histogram of each pixel's bin index is equal to $\frac{r}{n} + \frac{256g}{n^2} + \frac{256^2b}{n^3}$. After the bin index of each pixel is found, adding the associated weight value to the correspond bin. The weight value of pixel relates to the distance it to the center point of the ROI. The distance between boundary point and the center point of the target is d . The distance between the pixel point and center point is m_i . If $m_i \leq d$, the weight value $w_i = \frac{d^2 - m_i^2}{d^2}$, whereas $m_i > d$, $w_i = 0$. Lastly, adding each pixel's weighted value to its associated bin to construct the histogram of ROI.

Afterwards, loading the next frame and locating the position of the ROI in this frame. All possible candidates are represented as same way as the target, then calculating and comparing the histogram of target and all possible candidates to find out a pair of maximal similarity. Using the Bhattacharyya coefficient to calculate the similarity value between histograms of target and possible candidates. The formula as follows:

$$\Phi_b(S) = \sum_{t,c=1}^n \sqrt{\frac{h_t}{\sum_{t=1}^n h_t} * \frac{h_c}{\sum_{c=1}^n h_c}} \quad (3.4) [1]$$

Where Φ_b represents the Bhattacharyya coefficient, it ranges from 0 to 1. h_t represents the histogram of initial target at the beginning frame. h_c represents the histogram of candidate. S represents the state of target when it is obtained. t and c are t bin index of target and candidate model respectively. As known from the formula, h_c value varies from frame to frame, h_t is constant value.

If the position of ROI changed significantly from one frame to another frame, it needs to be updated. In principal, the size of bounding box is assumed to be fixed. While, the size of bounding box is varied by the distance from target to the camera during tracking. Once the interest target is found, keep loading it in the next frame until the last frame.

In this part, eye pair is being tracked in the real-time detection. The detection bounding box do not change its size when the distance between camera and eye pair had changed. The reason of this result is that the histogram based tracking algorithm considered the pixel histogram belong to the tacking target, compared with possible candidates and found out the target. It does not

consider the size of bounding box is varied by movements of target. Therefore, it is likely to have over detect or not fully detect the region of interest.

3.3.2 Continuous adapt mean shift algorithm (Camshift algorithm)

The continuous adapt Meanshift algorithm (Camshift) is the adaption of Meanshift algorithm. It is used for tracking interest objects and adapt to their motion during real-time detection. Moreover, it can offset the drawback of histogram-based tracking algorithm.

The steps of Camshift tracking algorithm as follows: detecting the region of interest of initial frame and calculating the color pixel distribution to whole image of initial frame. Afterwards, locating the position of detection windows which contained the region of interest.

The location formula as follows:

$$M_i = \sum_x \sum_y I(x, y), \quad M_x = \sum_x \sum_y x * I(x, y), \quad M_y = \sum_x \sum_y y * I(x, y), \quad x_c = \frac{M_x}{M_i}, \quad y_c = \frac{M_y}{M_i}.$$

Where $I(x, y)$ is the intensity of the new probability image at (x, y) . M_i is the initial moment of the detection window, M_x M_y are the first moment in x, y direction respectively, x_c , y_c are x, y coordinates of the location of detection window.

Then calculate the pixel distribution of center region of window and store the distribution area by using the histogram back-projection which describes pixel values in the images correspond to histogram bins.

The formula of histogram calculation as follows:

$$q_u = \sum_{i=1}^n \delta[c(x_i) - u] \quad (3.5) [1]$$

where q_u represents u-bin histograms, $c(x_i)$ is a function which associates to the pixel at location x_i with the histogram bin index.

Using the following formula to rescale the histogram bin value to the range from 0 to 255.

$$\{p_u = \min\left(\frac{255}{\max(q_u)}, 255\right)\}_{u=1\dots m} \quad (3.6) [1]$$

where p_u represents new value range histogram.

Following frames calculate the new probability by using ratio histogram which can help to extract region of interest from image background. The following formula shows the weight of histogram of ROI to the histogram of whole image.

$$p'_u = w_u * \sum_{i=1}^n k(\|x_i\|^2) \delta[c(x_i) - u] \quad (3.7) [1]$$

where $w_u = \frac{o}{o_u}$ its value range from 0 to 1. O represents the histogram of region of interest. O_u represents the histogram of whole image.

where k is kernel function which used to compute histogram for region out of the normalized target location $k(x_i) = \begin{cases} a * x_i & 1 < r \leq h \\ 0 & \text{otherwise} \end{cases}$, a represents scaling factor, h represents the width of new detection window.

Afterwards, calculate the orientation and scale of the detection window and locate its position in consecutive frames. Orientation calculation is based on the moment of detection window. Second moment defined as follows:

$$M_{y'} = \sum_x \sum_y y^2 * I(x, y) \quad M_{x'} = \sum_x \sum_y x^2 * I(x, y), \quad M_{xy} = \sum_x \sum_y xy * I(x, y)$$

where $M_{x'}$, $M_{y'}$ are second moment in x , y direction respectively. M_{xy} is the first moment.

The orientation of the detection box calculated as the following formula:

$$\theta = \frac{1}{2} \tan^{-1} \left(\frac{2 * \left(\frac{M_{xy}}{M_i} - x_c y_c \right)}{\frac{M_{x'}}{M_i} - x_c^2 - \frac{M_{y'}}{M_i} + y_c^2} \right) \quad (3.8) [1]$$

The scale of the detection box can be determined by the following formula:

$$l_1 = \sqrt{\frac{\left(\frac{M_{x'}}{M_i} - x_c^2 + \frac{M_{y'}}{M_i} - y_c^2 \right) + \sqrt{4 * \left(\frac{M_{xy}}{M_i} - x_c y_c \right)^2 + \left(\frac{M_{x'}}{M_i} - x_c^2 - \frac{M_{y'}}{M_i} + y_c^2 \right)^2}}{2}} \quad (3.9) [1]$$

$$l_2 = \sqrt{\frac{\left(\frac{M_{x'}}{M_i} - x_c^2 + \frac{M_{y'}}{M_i} - y_c^2 \right) - \sqrt{4 * \left(\frac{M_{xy}}{M_i} - x_c y_c \right)^2 + \left(\frac{M_{x'}}{M_i} - x_c^2 - \frac{M_{y'}}{M_i} + y_c^2 \right)^2}}{2}} \quad (3.10) [1]$$

where l_1 and l_2 are distances from the center point of detection box.

Link motions of detection window in successive frames and keep updating scale and orientation of the detection box until the last frame.

In this project, using the Camshift algorithm to track the interest object, the detection box will adapt to motions of the interest object. The drawback of Camshift algorithm is that it may lose tracking the interest object from a various color background.

3.3.3 Kanade–Lucas–Tomasi (KLT) algorithm

The Kanade–Lucas–Tomasi (KLT) algorithm which make use of the spatial intensity of interest objects to directly search for its position yields best match. The steps of KLT tracking algorithm as follows:

1. Detect feature points on the region of interests.
2. Estimate feature points motion between consecutive frames.
3. Link motion vectors in successive frames.
4. Update the position of feature points.
5. Repeat steps 1-4 until the last frame.

Feature points' detection is based on the Harris corner detection algorithm. The principal of Harris algorithm is shown as follows: assume a grayscale image I, the position of image patch over the area is (x, y) and shifting it by (u, v). The weighted sum of squared differences between these two patches represents as S:

$$S(u, v) = \sum_u \sum_v w(u, v) (I(x + u, y + v) - I(x, y))^2 \quad (3.11). [20]$$

Where w is the Gaussian windows function. For $I(x + u, y + v)$, it can be approximated by Taylor expansion. I_u and I_v are partial derivatives of I, so that $I(x + u, y + v) \approx I(x, y) + I_u(x, y) * u + I_v(x, y) * v$ and the formula can be converted to be: $S(x, y) =$

$\sum_u \sum_v w(x, y) (I_x(u, v) * x + I_y(u, v) * y)^2$. It can be written as the matrix form: $S(x, y) =$

$$(x \ y) A \begin{pmatrix} x \\ y \end{pmatrix}. \text{ Where } A = \sum_u \sum_v w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}, \quad (3.12) [20]$$

it is the structure tensor. $\begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$ is Harris matrix. The large variation of S in all direction of vector (u, v) characterized the structure tensor A which should has two large eigenvalues (λ_1, λ_2) for an interest point.

If $\lambda_1 \approx 0$ and $\lambda_2 \approx 0$ then there do not have feature points in the region of interest. If $\lambda_1 \approx 0$ and λ_2 has some large positive value, then the edge of the region of interest can be detected.

If λ_1 and λ_2 have large positive value, then a feature point of the region of interest is detected.

As for feature points motion estimation, the aim of this step is to estimate parameters of coordinates. Assume the initial estimate of parameter of coordinate is known which is p . Then find out the change of parameter Δp :

$$\sum_x [I(W(x; p + \Delta p) - T(x))]^2 \quad (3.13) \quad [22]$$

Where $T(x)$ presents whole image include the image background. $I(x)$ represents the ROI. W is warp function. Find the Taylor series: $\sum_x [I(W(x; p) + \nabla I * \frac{\partial W}{\partial p} \Delta p - T(x))]^2$. Using the Hessian matrix for calculating the translation motion.

$$H^{-1} = \sum_x \left[\nabla I \frac{\partial W}{\partial p} \right]^T \left[\nabla I \frac{\partial W}{\partial p} \right], \quad H^{-1} = \sum_x \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (3.14) \quad [22]$$

where I_x and I_y are partial derivatives of I .

$$\Delta p = H^{-1} \sum_x \left[\nabla I \frac{\partial W}{\partial p} \right]^T [T(x) - I(W(x; p))] \quad (3.15) \quad [22]$$

The drawback of KLT tracking algorithm is that it may lost tracking objects, if objects have out of plane rotation or experienced the variation of light condition.

3.3.4 Speeded up robust features (SURF) algorithm

The surf tracking method consists of two major parts which are SURF detection and tracking SURF feature detected. For SURF detection, it is based on SURF algorithm. For SURF algorithm, it has three major parts which are feature points detection, local neighborhood description, matching respectively.

For feature points detection, this algorithm uses square-shaped filters as an approximation of Gaussian smoothing which can filter the integral image with a square much faster. The sum of original image within a square can be calculated as follows: $S(x, y) = \sum_{i=0}^x \sum_{j=0}^y I(i, j)$. This formula requires to evaluate four corners of rectangular.

This algorithm uses a blob detector which is based on the Hessian matrix to find out feature points on the region of interest. Using the determinant of the Hessian matrix to measure the local change around the point and points are chosen where the determinant is maximal. The determinant of Hessian matrix also can be used to select the scale. This can be calculated as the following formula:

$$H(p, \sigma) = \begin{pmatrix} L_{xx}(p, \sigma) & L_{xy}(p, \sigma) \\ L_{xy}(p, \sigma) & L_{yy}(p, \sigma) \end{pmatrix}. \quad (3.16) \quad [25]$$

Where $H(p, \sigma)$ is the Hessian matrix at point p . σ is the scale of Hessian matrix. $L_{xx}(p, \sigma)$, $L_{xy}(p, \sigma)$, $L_{yy}(p, \sigma)$ are the second order derivative of the grayscale image I .

Searching and comparing the correspondences of images when they are in different scale to find out interest points at different scales. The scale space of image is realized as a pyramid of

the image. As for image pyramid, it is a type of multi-scale signal presentation which is used for image processing, computer vision. Images are repeated smoothing and subsampling. The Figure 3-15 shows an image pyramid with 6 levels.

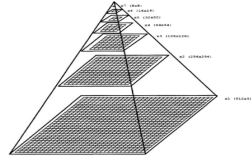


Figure 3-15 An image pyramid with 6 levels (Wikipedia)

There are three types of image pyramid which are Gaussian pyramid, Laplacian pyramid and Steerable pyramid respectively. In this tracking algorithm, it uses Gaussian pyramid which using Gaussian average and scale parameter to weight subsequent images. Each pixel of original image contains local average which corresponds to pixel of subsequent image on the lower level of image pyramid. The scale of several levels of subsequent images can be calculated as following formula: $\sigma = cfs * (\frac{bfs}{bfs'})$ (3.17) [25]

Where cfs represents current filter size of subsample image. bfs represents the based filter scale of former image. bfs' represents the based filter size of former image.

Applying different size of box filter to implement scale spaces in SURF.

As for local neighborhood description, it provides a robust description of image feature. To be exact, it describes the intensity distribution of pixels within the neighborhood point of interest. The descriptor of SURF is affected by the appearance variations which may not provide sufficient discrimination and gives too much false positives. Therefore, the orientation of point of interest need to be found for achieving rotational invariance. The steps of determine orientation as follows: calculating the Haar wavelet response in x, y direction within a circular neighborhood region which around the point of interest. Then, using Gaussian function centered at point of interest to weight the obtained responses. Afterwards, calculating the sum of all responses within a sliding orientation window to estimate the dominant orientation. The sum response yields a local orientation vector which define the orientation of point of interest. In matching part, matching pairs can be found by comparing descriptor obtained from different images.

After the SURF is detected, extracting the SURF feature from initial frame and get the position of SURF in initial frame. Calculating the 2D affine transformation of SURF feature between consecutive frames by using the following formula: $c = (u_x, u_y, \varphi, \rho)$ (3.18). Where u_x, u_y represents spatial translation. φ represents the rotational change of feature between consecutive frames. ρ represents the scale change of feature between consecutive frames. Afterwards, link the vector of motion in successive frames. Using the following formula to calculate the vector: $v_{f,t} = w(c_t, p_f)$. (3.19)

where c_t represents the transformation. p_f represents the expectation of the motion of feature located at p_f . w is the wrap function which process any features of image has significantly distorted.

Lastly, keep updating the position of SURF feature until the last frame.

In this project, using the open eye pair image as the region of interest. Detecting the feature points on the region of interest by using the SURF algorithm. The result is shown in the red frame of Figure. Afterwards, dilating the eye pair image to be 1.2 times of original size of image and rotating the image about 30 degrees as shown in the blue frame of Figure 3-16.

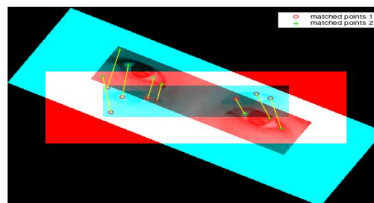


Figure 3-16 SURF matching

It finds that feature points of ROI can still match to the feature points of ROI which have changed image scale and image rotation.

However, if the angle of eyelid has a little change as shown in Figure 3-17. Feature points of ROI cannot match to the ROI which had feature points position change. The result as shown in Figure 3-18. It found that feature points of ROI cannot match to each other.



Figure 3-17 Comparison images for eyelid position have changed



Figure 3-18 The result of feature points matching

In this project, SURF tracking algorithm is likely to lose tracking if feature points of object do not have relative change. Therefore, it is not good for eye blink detection as feature points of eye pair have position change during eye blinking.

3.3.5 The combination between KLT and Camshift algorithm

The advantage of Camshift algorithm for tracking object is that it adapts to movements of interest object during tracking. The advantage of KLT algorithm for tracking object is that it can accurately locate the position of interest object during tracking. In this project, it combines KLT algorithm with Camshift algorithm together for tracking the eye pairs. The combination tracking algorithm has a higher performance of tracking object than other algorithms. Moreover, it can accurately locate the position of the interest object and adapt to the motion of the tracking object during tracking.

The procedure of the combination algorithm as follows:

1. Using the Viola-Jones algorithm to detect the ROI (eye pair region) in the initial frame and it was shown within the bounding box.
2. Using Harris corner detection algorithm to detect the feature points of ROI in the initial frame.
3. Calculate the pixel distribution of ROI with respect to the whole image in the initial frame by using histogram-projection to describe pixel values in the image correspond to histogram bins.
4. Recalculate the new pixel distribution of ROI in the following frames by using histogram. The formula of histogram weights the pixel distribution of ROI to the whole image. The formula of histogram is same as that in Camshift algorithm.
5. Estimate and calculate the scale of bounding box in successive frames as it will have change during tracking.

6. Match feature points within ROI by calculating translations of feature points in successive frames.
7. Repeat step1-6 until the last frame.

The scaling of bounding box is based on the image transformation. Four types of image transformation which are translation, scaling, shearing and rotation.

For example, assume the 2D image is represented as an image matrix I, where $I = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$,

$I' = I * T$, T is image transformation matrix.

The image transformation matrix of scaling is that: $T_{sc} = \begin{bmatrix} x & 0 & 0 \\ 0 & y & 0 \\ 0 & 0 & 1 \end{bmatrix}$

where x represents scaling factor along the x axis, y represents the scaling factor along the y axis.

Regards to image rotation, the image transformation matrix of scaling is that:

$T_r = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$, θ is the angle of object had rotated.

The image transformation matrix of shearing is that: $T_{sh} = \begin{bmatrix} 1 & sy & 0 \\ sx & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Where sx represents scaling factor along the x axis, sy represents the scaling factor along the y axis.

The image translation matrix presents as: $T_t = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ tx & ty & 1 \end{bmatrix}$

Where tx represents the displacement along the x axis, ty represents the displacement along the y axis.

Regards to match feature points, it estimates the image transform of bounding box and calculates the Euclidean distance between points in continuous frames. For example, the coordinate of feature point 1 in initial frame is (x_1, y_1) , the coordinate of feature point 1 in second frame is (x'_1, y'_1) . The distance between two points in successive frames represents as:

$$d = \sqrt{(x'_1 - x_1)^2 + (y'_1 - y_1)^2} .$$

Comparing the tracking accuracy of each algorithm for tracking eye pair.

The Table 4 shows the comparison of detection accuracy and speed of implementing SURF algorithm and the combination algorithm for eye pair tracking.

Tracking algorithm	SURF algorithm	The combination algorithm (KLT and Camshift algorithm)
Total frames of detection	300	300
Frames of false positive detection	2	3
Frames of false negative detection	7	2
Detection accuracy	97%	98.3%
Time of spend	24s	24.5s
Speed	12.5frames/s	12.31frames/s

Table 4 The comparison of detection accuracy between two algorithms

From Table 4, it indicates that the detection accuracy of the combination algorithm (98.3%) is higher than that of SURF algorithm (97%) for tracking eye pair. Meanwhile, the speed of SURF algorithm for eye pair tracking is about 12.5frames/s which is little faster than that of the combination algorithm (12.31frames/s). However, the detection accuracy of tracking algorithm is an important factor of the choice of tracking algorithm, as the requirement of detection accuracy in this project.

Therefore, the combination algorithm has better tracking performance compares with the SURF algorithm and the detection accuracy of real-time detection is improved by the tracking function.

The Figure 3-19 shows the procedure of eye tracking.

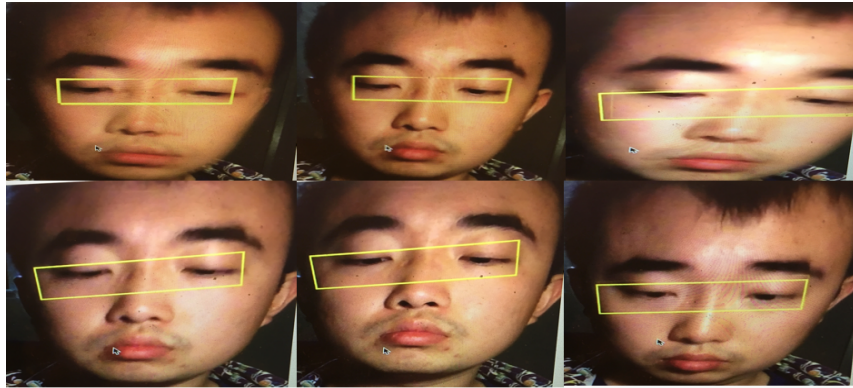


Figure 3-19 The procedure of eye tracking

3.4 Eye blink detection

The eye blink detection is achieved by using the correlation coefficient algorithm which applied for comparing the similarity between two images. The formula of this algorithm as follows:

$$r = \frac{\sum_m \sum_n (A_{mn} - \bar{A})(B_{mn} - \bar{B})}{\sqrt{(\sum_m \sum_n (A_{mn} - \bar{A})^2)(\sum_m \sum_n (B_{mn} - \bar{B})^2)}} \quad (3.20) [4]$$

Where r is the correlation score and the value range of r is from 0 to 1. A_{mn} represents the pixel of the video frame at the point (m, n) . B_{mn} represents the pixel of the template image at the point (m, n) . \bar{A} and \bar{B} are the average pixel value of video frame and the template image respectively. The correlation score r is a measure of match between all points of ROI in each frame and the open eye template.

In this project, it uses the open eye pair image as the template image, as shown in Figure 3-20.



Figure 3-20 The open eye pair template image

Afterwards, compute the correlation coefficient score between the template image and the region of interest (eye pair region detected) of every frame. In principal, the correlation score between the template image and open eye pair image is about 0.95-0.85. The correlation score between the template image and open eye pair image is about 0.60-0.50. However, the correlation coefficient algorithm has two limitations. Firstly, the correlation coefficient algorithm requires that the ROI in every frame should have same color space as the template

image. As the color space of template image and the ROI in every frame are not same, so that they need to be converted into gray images or binary images. Secondly, the correlation coefficient algorithm also requires that the image size of template image must be same as that of the ROI in every frame. In this project, the ROI are detected in every frame are not in the same image size due to eye blink and the change of distance from eye pairs to the camera. As the requirement of coefficient relation algorithm, they are need to be resized to be the same image size as that of template image. The image scaling transformation (mentioned in section 3.3.5) is used for resize image works. The following Figure 3-21 and Figure 3-22 show results of converting the template image and open eye image closed eye image to be the same color space and same image size.



Figure 3-21 Closed eye pair (left) and opened eye pair (right)



Figure 3-22 The template image after convert color space.

The correlation score between the template image and the closed eye pair image is about 0.86. The correlation score between the template image and opened eye pair image is about 0.94. It indicated that the closed eye pair results are not conform to the ideal coefficient value range of eye close. There are two reasons cause this result. One reason is that the pixel value of template image and closed eye image changed as images are converted into gray image. Another reason is that some pixel value of image lost as closed eye pair image resize its shape to be same size as the template image. However, the correlation score between the template and open eye pair image is higher than that between the template image and closed eye pair image. If the correlation score has a significant change, it represents eye pair have a blink. If the correlation score relatively keeps the same, it represents eye pair in open state or close state.

3.5 Drowsiness detection

The threshold blink duration of people in drowsy state is 400ms. When people in drowsy state, the blink duration is over 400ms. The Figure 3-23 shows the blink duration in alert state is different from that in drowsy state. If the period of correlation score changed from the value range of eye opened to the value range of eye opened over 400ms, it presents the participant in drowsy. Otherwise, it presents the participant in alert.

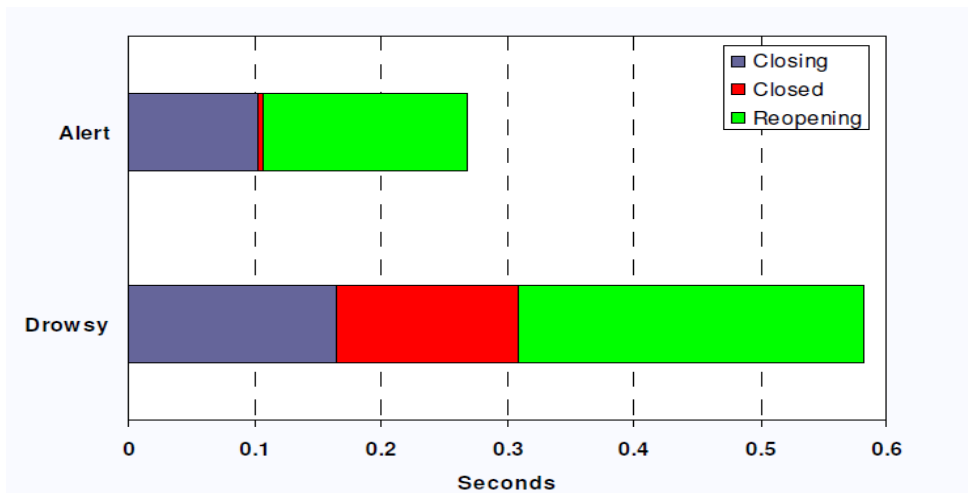


Figure 3-23 The blink duration in alert state and drowsy state (Tucker 2005)

Chapter 4 Result analysis

Two major factors which affect the correlation score are: pixel distribution within ROI and the distance between the participant and camera. The correlation score between the open eye template image and each input frame have variation in different experiment conditions.

4.1 Methodology of eye blink experiment

Results of eye blink detection are verified by the following 8 cases of experiments. Two participants took part in eye blink detection experiments. One participant took part in different cases of eye blink experiments and his or her behaviors (eye state, times of eye blink, blink interval) observed, recorded by another participant.

4.2 Cases of eye blink experiments

Case 1 Opened eyes state

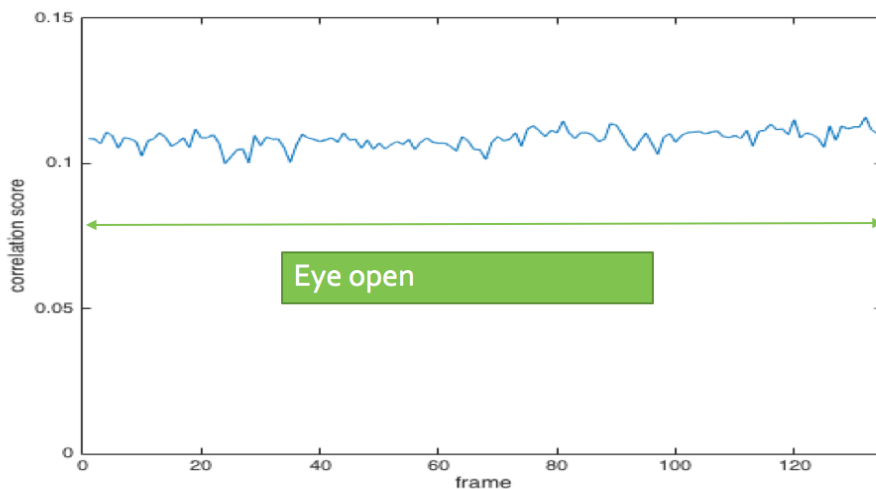


Figure 4-1 The result of case 1

As shown in Figure 4-1, the curve from the initial frame to the 130th frame represents that the participant in opened eyes state, correlation scores between opened eyes template image and ROIs in each frame are around 0.11.

Case 2 Closed eye state

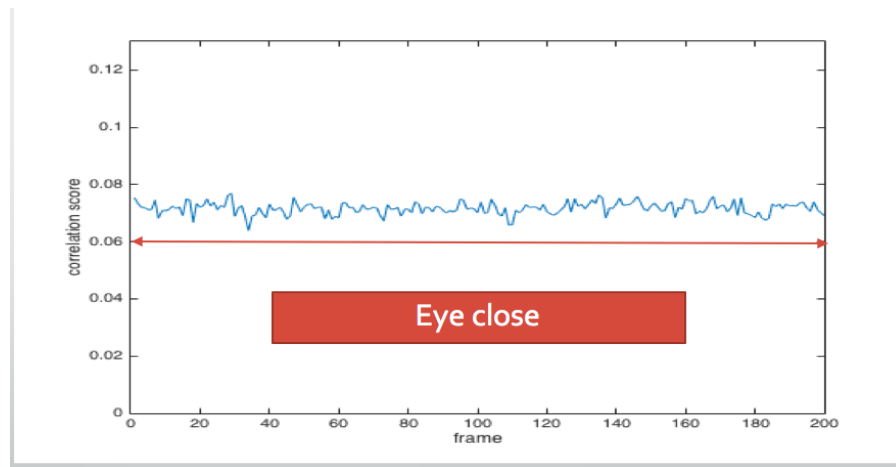


Figure 4-2 The result of case 2

As shown in Figure 4-2, the curve from the initial frame to the 200th frame represents that the participant in eyes closed state, correlation scores between open eyes template image and the ROI in every frame are around 0.07. Compared with the result of case 1, it shows that correlation scores of the participant in eyes closed state are less than that of the participant in opened eyes state. Implemented two experiment cases and record their results for verify the correlation score of participant in open and close eye state,

Case 3 Eye open-close

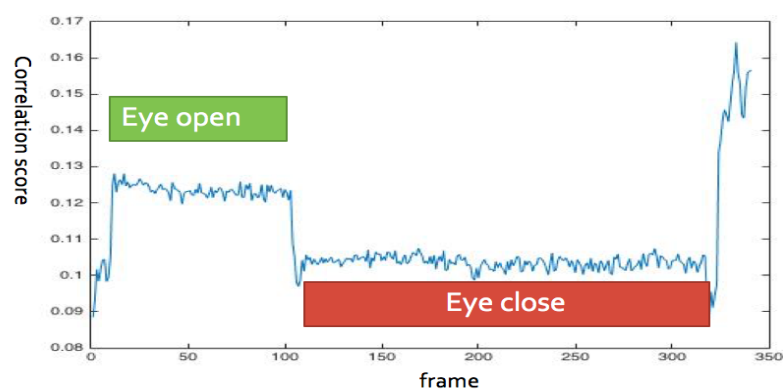


Figure 4-3 The result of case 3

As shown in Figure 4-3, the curve from the 10th to the 100th frame represents that the participant in open eyes state, correlation coefficient scores between open eyes template image and the ROI in each frame are around 0.125. At the 100th frame, the value experienced a dramatically

decrease, it dropped down to 0.1. From the 100th to the 300th frame, the participant in close eyes state, correlation scores between close eye template image and each frame are around 0.10.

Case 4 Long eyes closed and short eyes opened

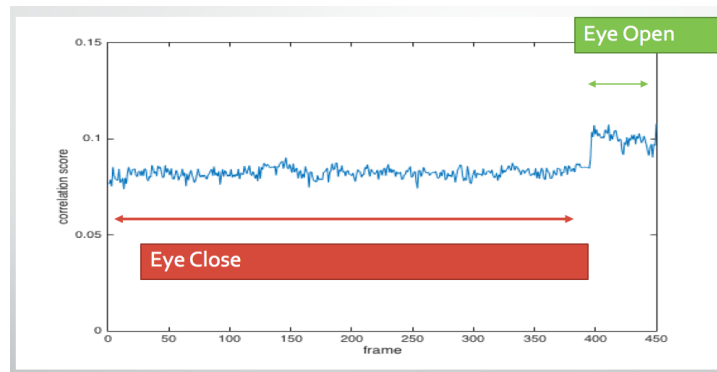


Figure 4-4 The result of case 4

As shown in Figure 4-4, the curve from the initial frame to the 400th frame represents that the participant in eye close state, correlation scores between open eye template image and the ROI in each frame are around 0.08. At the 400th frame the value experienced a dramatically increase, it up to the 0.10. The curve from the 400th to the 450th frame represents the participant in close eye state, correlation scores between open eye template image and ROI in each frame are around 0.10.

The variation of correlation score from close eyes to open eyes state is verified by implemented an additional experiment which required the participant to keep close eyes and then open eyes in several phases.

Case 5 Eyes closed- eyes opened (4 loops)

In this case, the participant kept closed-open eye states in 4 phases. Correlation scores of open eyes state and close eyes state are around 0.09 and 0.06 respectively. The result as shown in Figure 4-5.

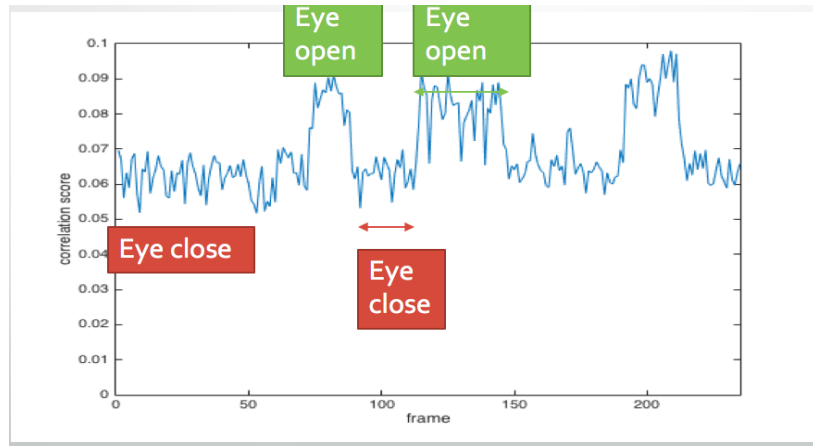


Figure 4-5 The result of case 5

Case 6 Eye blink once then closed-opened eyes phases

In this case, the participant blinked once during the closed eyes period from the initial frame to the 100th frame, a peak among the values from the initial frame to the 200th frame. It represents the participant blinked in close eye state. Correlation scores of opened eyes state and close eyes state are around 0.11 and 0.09 respectively, as shown in Figure 4-6.

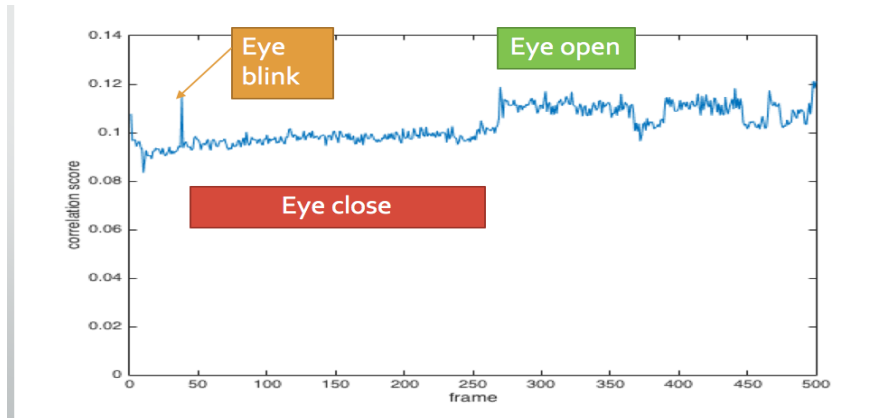


Figure 4-6 The result of case 6

Case 7 Eye blink not frequently

In this case, the participant kept open eye state from the initial frame to the 39th frame, and correlation scores of opened eyes state are near to 0.10. Afterward, the participant had blink several times from the 40th frame to the 90th frame. It shows that the correlation score drop down and then increase up during this period. The waveform of this case is different from that of other cases. It depends on the initial state of eye had been detected, if the initial eye state is

open, when eye had a blink, the correlation score goes down and then goes up. Otherwise, it goes up and then goes down. Correlation scores of closed eyes state are around 0.02. A long blink happened on the period of the 80th frame to the 100th frame. After the 110th frame, the video is turned off, the correlation score significantly increased.

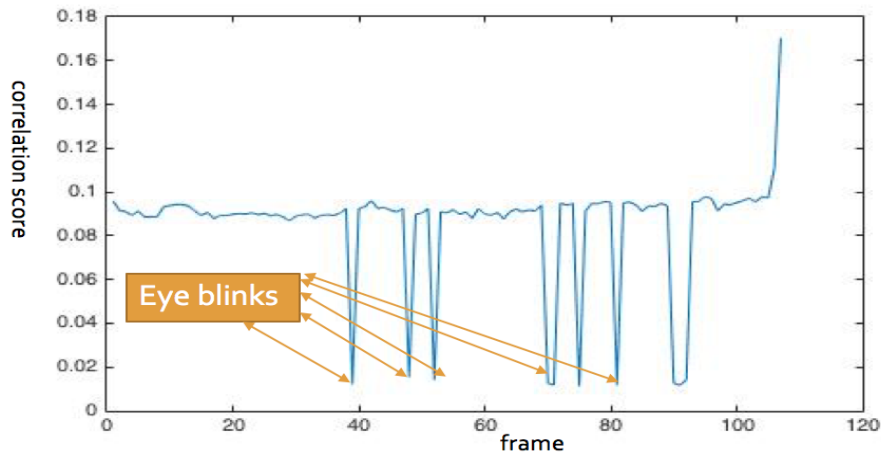


Figure 4-7 The result of case 7

Case 8 Eye blink frequently

In this case, correlation scores of opened eyes state and closed eye state are around 0.18 and 0.14 respectively. The participant blinked three times which is presented in the curve from the initial frame to the 200th frame. From the 250th to the 450th frame, the curve in this period represents the participants blinked frequently.

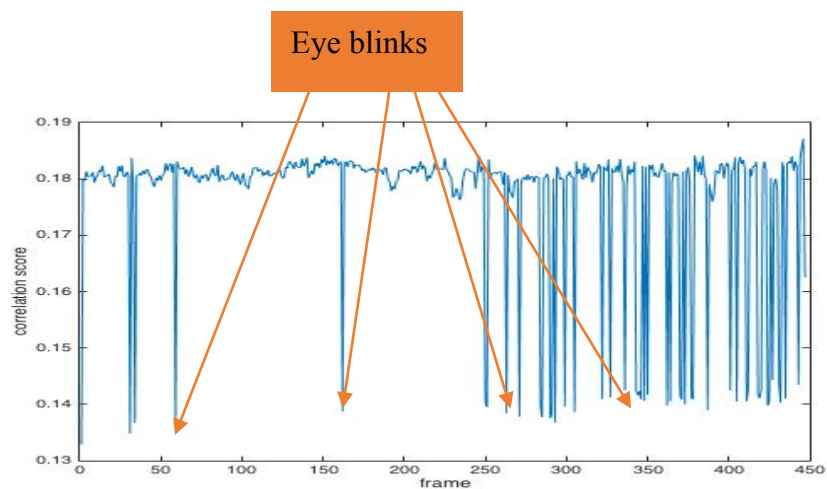


Figure 4-8 Eye blink frequently

4.3 Results of drowsiness detection

Case1 Blink slowly (Drowsy state)

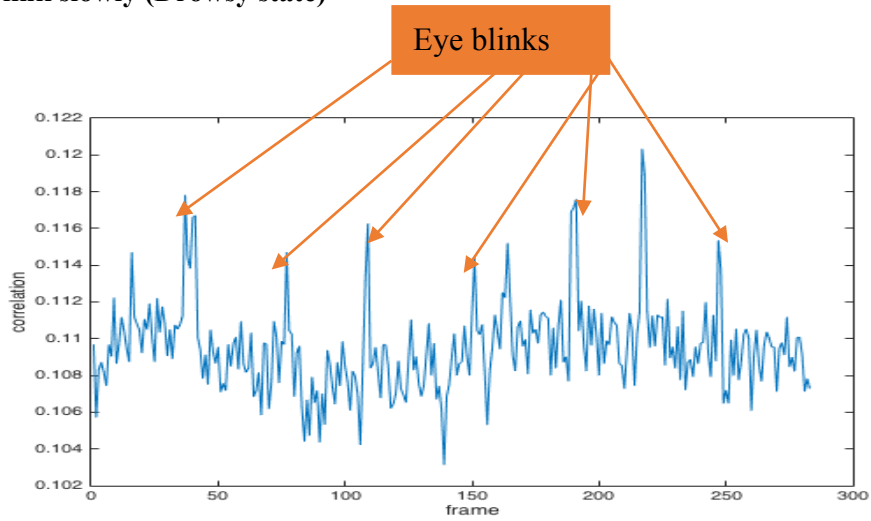


Figure 4-9 Eye blink slowly

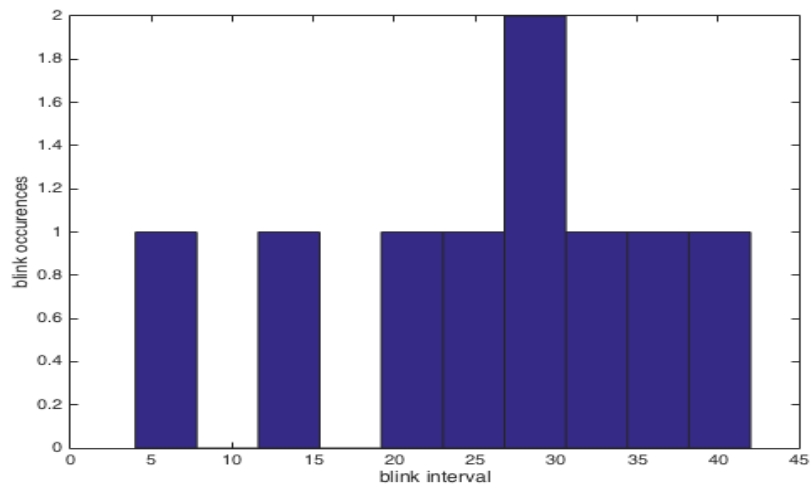


Figure 4-10 The histogram of eye blink interval in drowsy state

The result of drowsiness detection in this case as shown in Figure 4-9. The histogram graph presents the occurrence of blink interval. It shows that the blink interval of eye blink times is very fluctuant. The blink duration is measured from peak to peak values. The interval of eye blinks presents eye blink duration (t_b), the frequency of eye blink calculated as: $f_b = \frac{1}{t_b}$

One frame is equal to $\frac{1}{30}$ s.

The participant blinked 10 times during the blink detection. The frequency of eye blink in the period of eye blink detection is about 1.16Hz. The blink duration is about 0.86s which is equal to 860ms. It is over than the threshold value of drowsy state which is 400ms. Therefore, the state of participant is drowsy during this period.

Blink times	10
Blink frequency	1.16Hz
Blink duration	860ms
Average distance of peak value	25.667 (frames/time)
State	Drowsy

Table 5 The evaluation of eye blink parameters in drowsy state

Case 2 Blink frequently (alert state)

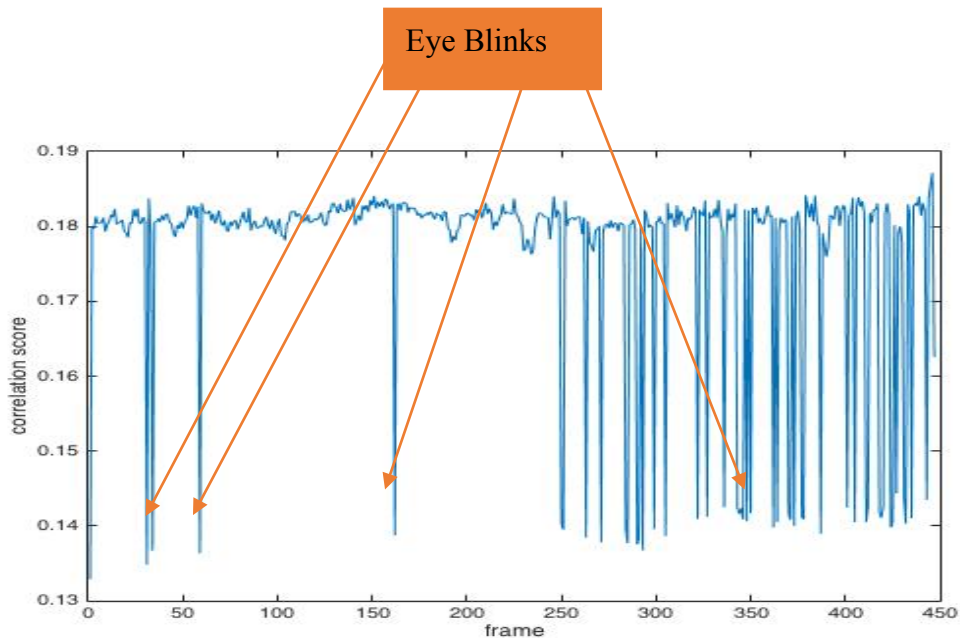


Figure 4-11 Eye blink frequently

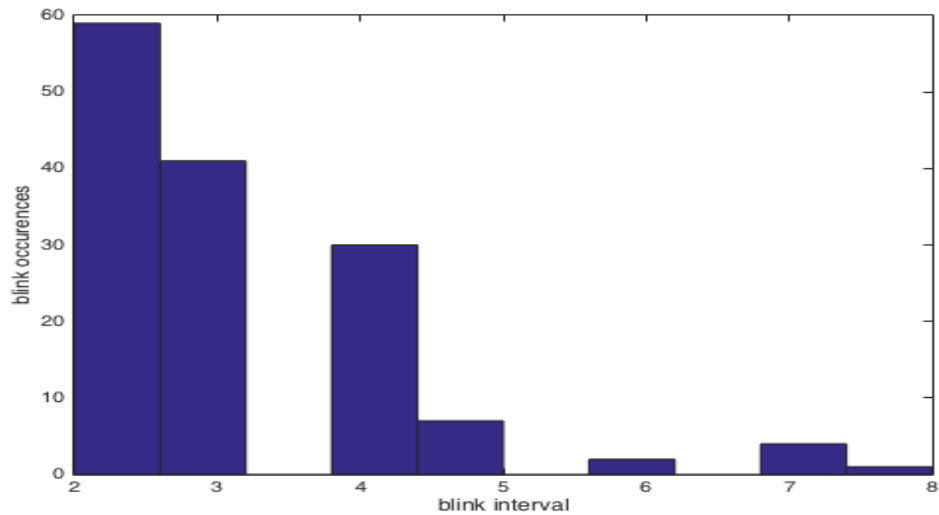


Figure 4-12 The histogram of eye blink interval in alert state

Blink times	145
Blink frequency	92.5 Hz
Blink duration	11ms
Average distance of peak value	3.083 (time/frame)
State	Alert

Table 6 The evaluation of eye blink parameters in alert state

The result of drowsiness detection in this case as shown in Figure 4-11. The histogram graph presents the occurrence of blink interval. Most of blink interval are about 2 frame time which is equal to 0.067s. In this case, the interval of eye blinks presents eye blink frequency. From Table 6, the participant blinked 145 times during the detection. The blink frequency of participant is about 92.5Hz and blink duration is about 11ms. The blink duration of participant is much less than the threshold value of drowsy state (400ms). The blink speed very fast and the participant in alert state during this period.

Case 3 Normal blink (alert state)

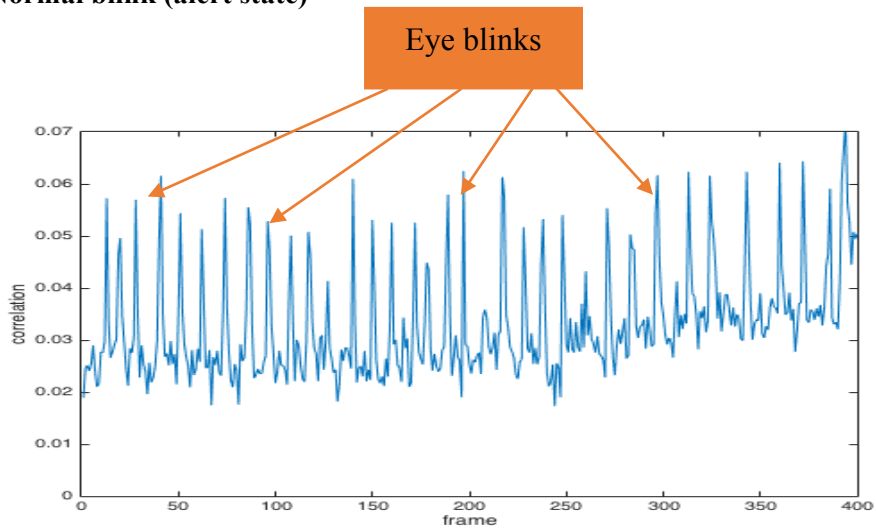


Figure 4-13 Normal eye blink

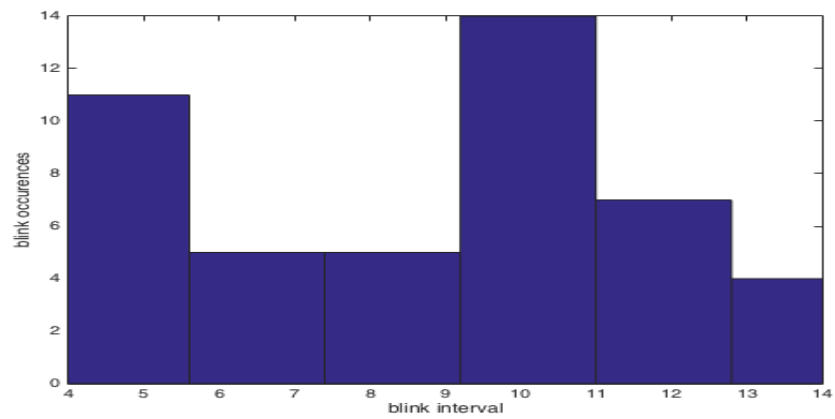


Figure 4-14 The histogram of Blink interval in normal state

Blink times	35
Blink frequency	2.63 Hz
Blink duration	380ms
Average distance of peak value	11.4 (frame/time)
State	Alert

Table 7 The evaluation of eye blink parameters in alert state.

The result of drowsiness detection in case 3 as shown in Figure 4-13. The histogram graph presents the occurrence of blink interval. It indicates that the blink interval of eye blink times is very fluctuant. The participant had blinked 35 times during this blink detection. The frequency of eye blink in the period of eye blink detection is about 2.63Hz. The blink duration is about 0.38s which is equal to 380ms. It is less than the threshold value of drowsy state (400ms). Therefore, the state of participant is alert during this period.

4.4 Error analysis

There are 5 major factors which affected the drowsiness detection are: head plane movements (horizontal and vertical), head rotations, iris movements, body shakings, light variation. For explore the influence of these factors to eye blink detection, 7 cases of experiments were implemented.

4.4.1 Factor cases

Case 1 Open eye with horizontal head movement (only)

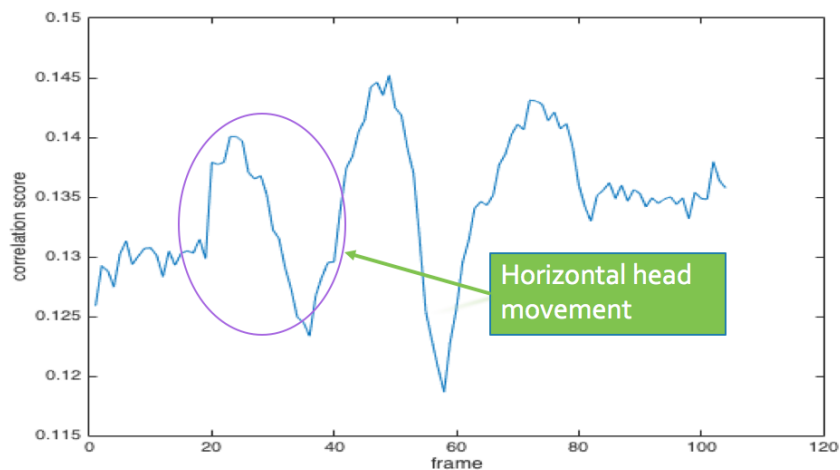


Figure 4-15 The result of horizontal head movement

As shown in Figure 4-15, the correlation coefficient score experienced several fluctuations from the 20th frame to the 80th frame. It represents that the participants had several significant horizontal head movements during this period.

Case 2 Open eye with vertical head movement (only)

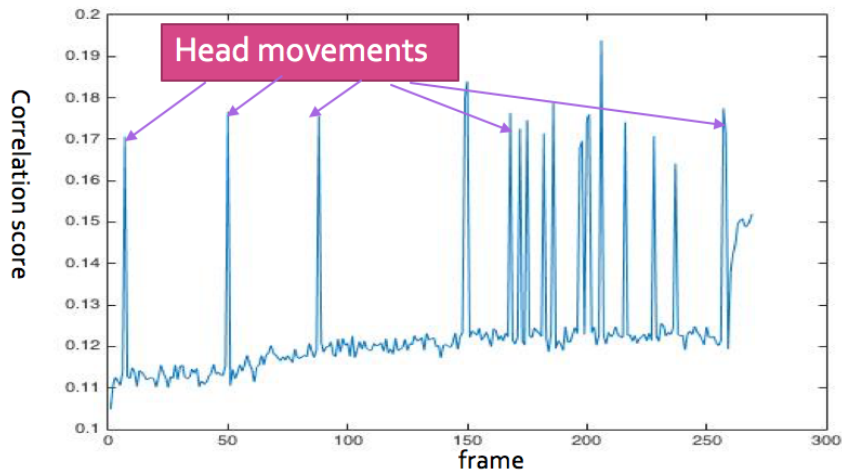


Figure 4-16 The result of vertical head movement

As shown in Figure 4-16, it shows that several peak values appear from the initial frame to the 250th frame. These mountain peak values represent that the participant had quick significant head movements in vertical direction during eye blink detection.

Case 3 Open eye with head rotation and blinks

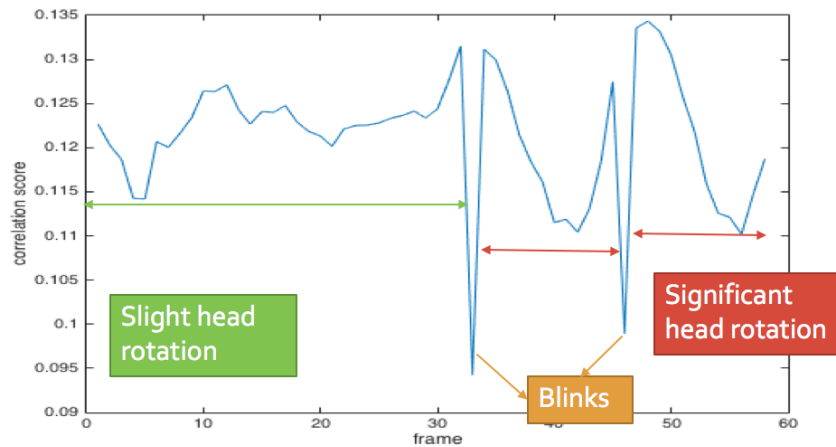


Figure 4-17 The result of horizontal head rotation

As shown in Figure 4-17, the correlation score experiences slight fluctuation from the initial frame to the 30th frame. It represents that the participants had slight head rotation during this period. However, correlation score experienced significant fluctuation from the 45th frame to 58th frame. It represents that the participants had significant head rotations during this period.

Case 4 Open eye with iris movements (only)

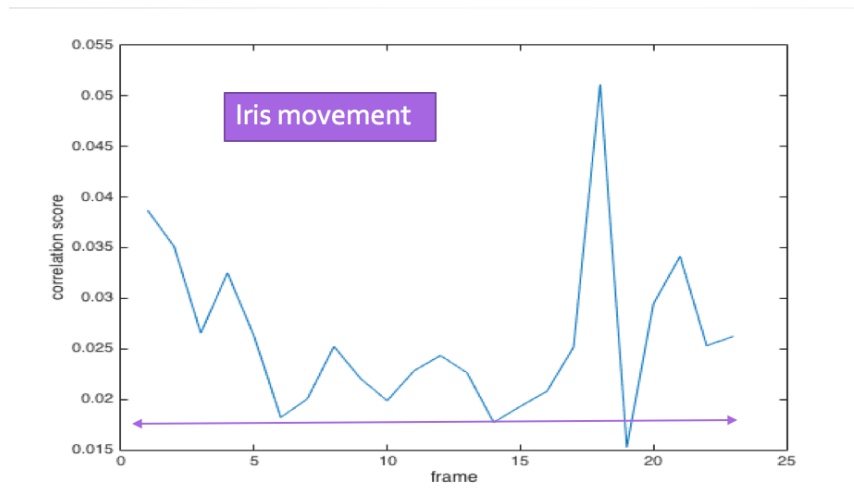


Figure 4-18 The result of iris movements

In this experiment case, the participant kept eye open state with iris movements. The correlation score experiences fluctuation from the initial frame to the last frame.

Case 5 Open eye with body shakings (only)

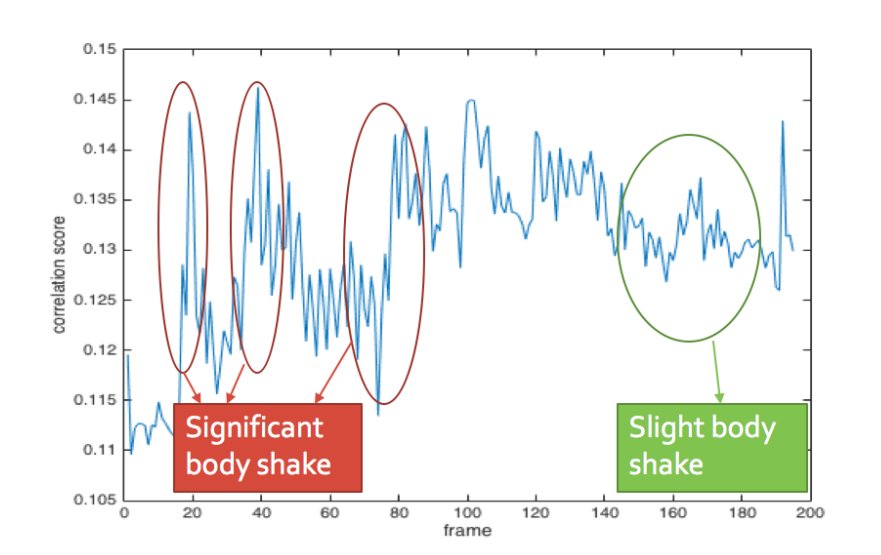


Figure 4-19 The result of body shakings

In this experiment case, the participant is required to shake his body during detection and it simulates the situation of driving on a rough road. From the 20th frame to the 80th frame, the correlation score experiences several significant fluctuations. It represents that the participant

had significant body shake. From the 100th frame to the last frame, the correlation score experiences slight fluctuations. It represents that the participant had slight body shakings.

Case 6 Open eye in different lighting intensities (without any head, iris movements and body shakings)

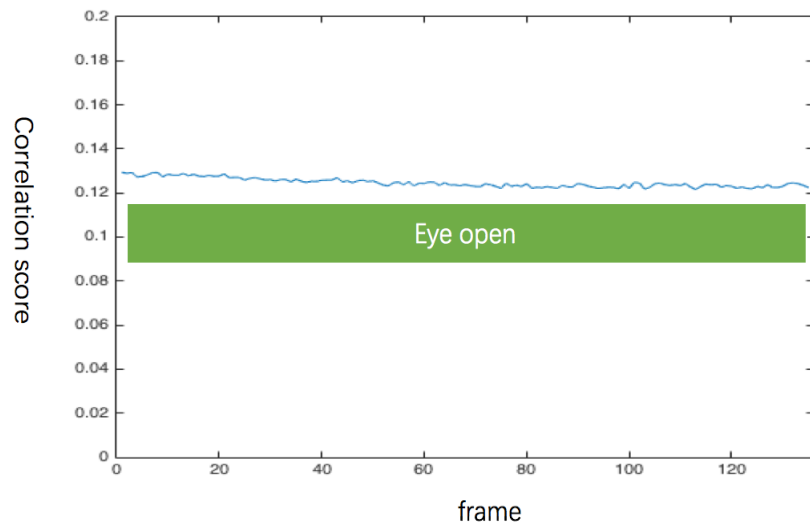


Figure 4-20 Eye open in sunlight (outdoors) intensity

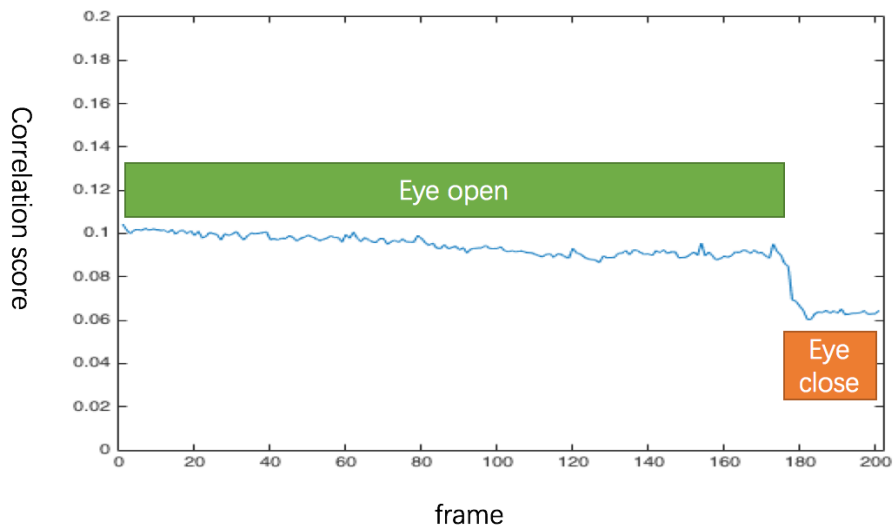


Figure 4-21 Eye open with eye close in sunlight (indoors) intensity

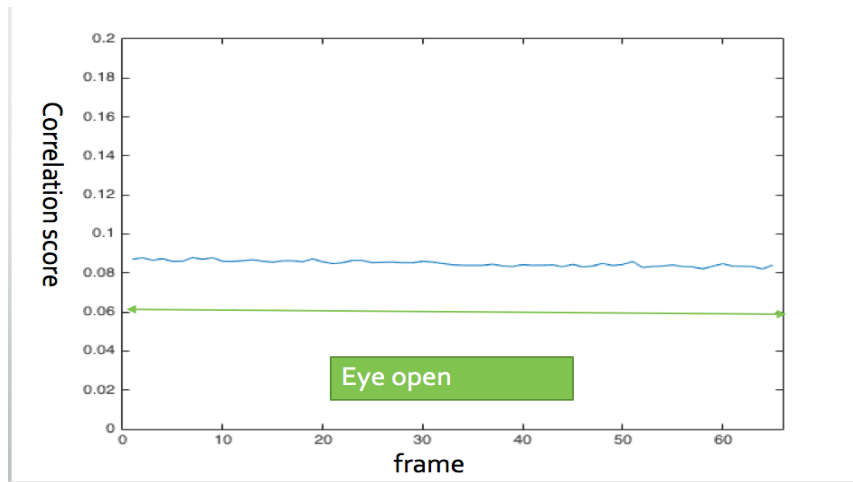


Figure 4-22 Eye open in artificial lighting intensity (lamp)

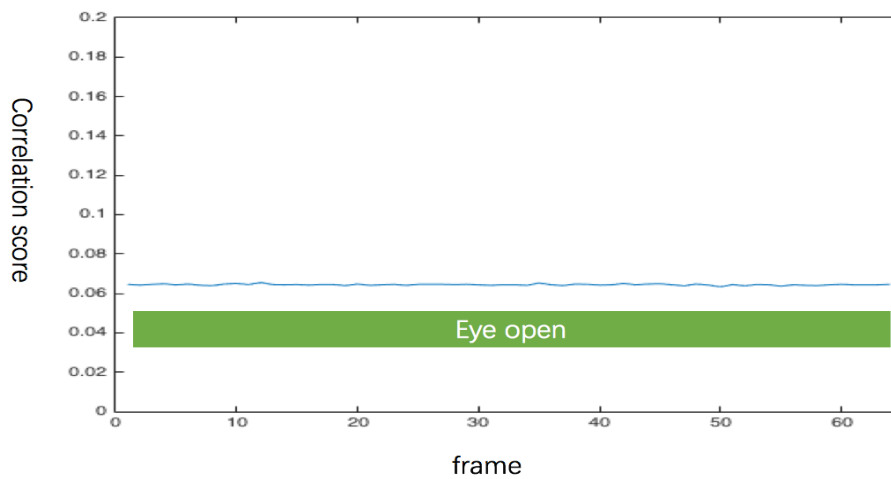


Figure 4-23 Eye open in moonlight intensity

Figure 4-20, 4-21, 4-23 and 4-24 show the result of eye opening in different lighting intensity. It indicates that correlation scores of eyes opened state in different lighting intensities are different to each other. With the intensity of lighting decrease, the correlation score becomes smaller. The lighting intensity of sunlight in outdoors is about 60000 Lux. The lighting intensity of sunlight in indoors is about 500 Lux. Moreover, the lighting intensity of artificial light (lamp with a 40w tube) is about 10.76 Lux. Furthermore, the lighting intensity of moonlight is about 0.2 Lux. Lux represents the unit of lighting intensity. Knowledge about lighting intensities is learning from Wikipedia websites.

Case 7 Eye opening from different distances to the camera

In this experiment case, participants are required to seat in different distances to the camera.

The following Figures show results of eye opening from different distances to the camera.

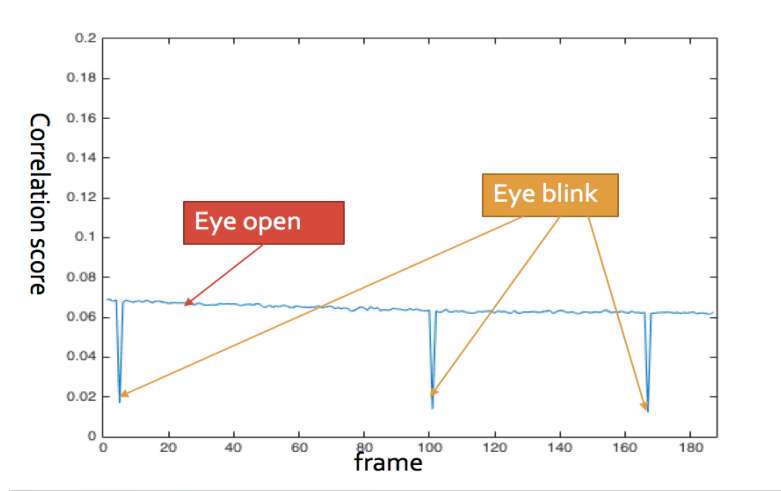


Figure 4-25 Eye opening from distance 30cm to the camera

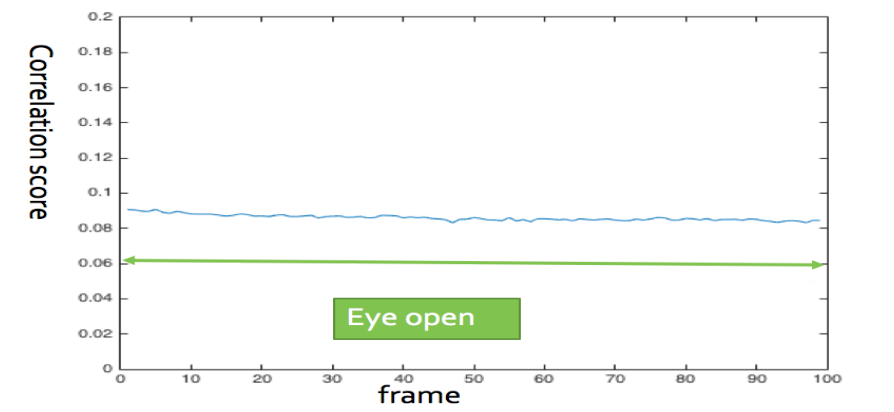


Figure 4-26 Eye opening from distance 20cm to the camera

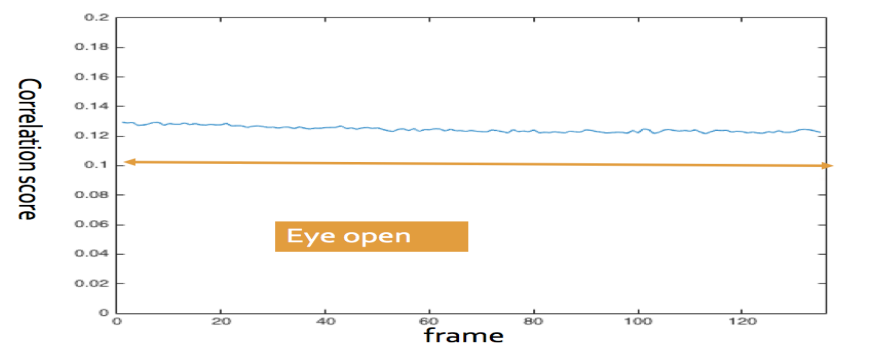


Figure 4-27 Eye opening from 10cm to the camera

It shows that correlation scores of eyes opened state is varied by the distance between the participant and the camera. If the participant seat is closer to the camera, the correlation score will become larger. Otherwise, it will become smaller.

4.4.2 Factor analysis

From results of experiments, it can be found that the correlation score is varied by head rotations, head movements, body shakings. The reason is that the perpendicular distance between the eye pair of participant and the camera have variation during head movements, head rotations and body shakings. The correlation score is varied by the perpendicular distance between participant and the camera, it had proved in the case 7 of factor experiment. In another aspect, the detecting box adapt to the motions of target (eye pair) during tracking. If the eye pair of participant move closer to the camera, the ROI will become bigger to show on the video frame and vice versa. The size of pixel matrix of ROI varied from frame to frame. As the requirement of correlation coefficient algorithm, the ROI need to be resize its image size to be same as that of the template image. It causes some pixels lost during image resize and pixel distribution within ROI have variations from frame to frame. Therefore, the correlation score has variation when the participant has head rotations, head movements, body shakings in real-time drowsiness detection.

Iris movements is another factor which affects eye blink detection. The correlation score is varied by iris movements. The reason is that the pixel distribution of ROI has a variation during iris movements. The pixel distribution of ROI has changed during iris movements. However, the size of pixel matrix of ROI is not varied by iris movements. The iris movement as shown in Figure 4-28.

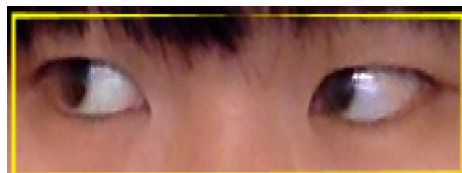


Figure 4-28 Iris movement

Furthermore, lighting intensity variation also affects the correlation score. The influence of lighting intensity variation to the correlation score cannot be neglected. As the result of

experiment in case 6, correlation scores of opened eyes are totally different in different lighting conditions. The sum pixel of ROI nearly is varied by the variation of lighting intensity, some pixel values within ROI has little change. Also, the pixel distribution of ROI has been changed by the lighting intensity variation. Therefore, the variation of lighting intensity has great effect on the correlation score.

The Table 8 summary the effect of head movements, head rotation, body shakings, iris movements, lighting condition to correlation score and drowsiness detection.

Factors	Head movements	Head rotation	Body shakings	Iris movements	Lighting intensity variation
Effect to correlation score	Significant	Significant	Significant	Little	Heavily
Effect to drowsiness detection	Significant	Significant	Significant	Little	Heavily

Table 8 Effect of factors to correlation score and drowsiness detection

Head movements, head rotations, body shakings have significant effect on the performance of drowsiness detection, it will be hard to determine the drowsy state of participant if he or she has head movements, head rotations or body shakings during blink detection. Iris movements and lighting variation have little and great effect on eye blink detection respectively. If the participant has blinks with head rotation (or other types of motion), it is hard to distinguish the variation of correlation score caused by blink and motions. Therefore, it is hard to use certain quantity number to describe factors' effect on drowsiness detection.

Chapter 5 Discussion

According to the analysis of eye blink detection results, it can be clearly found that the correlation score varies from different cases of experiments. The perpendicular distance from the eye pair to the camera is an important factor effect the correlation score, the correlation score will be changed by the variation of different perpendicular distance from eye pair to the camera. The reason of this result is that the size of ROI detected from images or video frames varied by the distance between eye pair and the camera. The correlation score is very sensitive to the perpendicular distance between eye pair and the camera which caused the change of pixel distribution within ROI. If the participants have any head moments, it will cause the change of distance between eye pair and the camera. If the participants have a significant motion (head movements, head rotations, body shakings) in the drowsiness detection, the correlation score will experience a significant change. The effect of body movements on eye blink detection is significant.

Lighting intensity variation is another factor which affects the performance of drowsiness detection. Correlation scores in different lighting intensity are different from each other as shown in Figure 4-20, 4-21, 4-22 and 4-23. The reason of that is the pixel distribution of ROI is affected by lighting intensity variation. Therefore, the effect of lighting variation to correlation score cannot be neglected.

Iris movements also affect drowsiness detection. The correlation score is varied by iris movement as the pixel distribution of ROI has a variation during iris movement. Compared with the change of pixel distribution which cause by body motion, the effect of iris movement on eye blink detection is very little.

In addition, correlation score in blink detection are much lower than the ideal value range of eye blink detection. Firstly, as the image color space requirement of correlation coefficient algorithm, the template image and the ROI within detection boxes had to convert into same color space. Images are likely to lost some pixel values during the image color space transformation. Secondly, ROI within detection boxes had to resize its image size to be same as that of the template image. some pixels within ROI lost during image transformation.

Moreover, the algorithm selection for eye tracking also affects the accuracy of drowsiness detection. The Table 9 summaries advantages and disadvantages of algorithms, which had been experimented in this project.

Methods	Advantages	Disadvantages
Histogram-based algorithm	Accurate detect interest objects	Restrict to lighting conditions. Not adapt to movements of object.
KLT algorithm	High performance in tracking	Lost tracking due to lighting variation, object has out of plane rotation
Camshift algorithm	Adapt to the movement of interest object during tracking	Easily fails to track objects in various color background
SURF algorithm	Interest objects can be tracked and detected no matter change of size noise, illumination	The interest object will be not detected if its relative position of feature points had changed.

Table 9 Summary of advantages and disadvantages of 4 algorithms

From Table 9, it indicates that SURF algorithm are likely to lose tracking as the position of feature points on the eye region will have changed during blinking. Histogram-based algorithm cannot be used for eye blink detection in the dark. The tracking performance of KLT algorithm is limited by the variation of light condition and target's out of plane rotation. Camshift algorithm cannot be used for eye blink detection in various color background. The tracking performance is improved by incorporating Camshift algorithm and KLT algorithm to be a new tracking algorithm. The new algorithm has combined advantages of Camshift and KLT algorithm. Therefore, the tracking performance of new algorithm is higher than other 4 algorithms.

Furthermore, the chosen of ROI (region of interest) also affects the accuracy of drowsiness detection. Selecting the single eye region as region of interest is likely to have false detections. It also affects the accuracy of eye blink detection, because the cascade classifier system is trained to recognize single eye images. Selecting separate eyes as region of interest may make eye blink detection much harder as it need to detect and track two separate eyes. If one of them fail to detect, the eye blink detection will not be detected. Selecting the eye pair region as ROI have relatively higher detection accuracy than other two types of eye detection. However, the accuracy of eye pair detection is also varied by different lighting conditions. The accuracy of eye pair detection is improved by training the cascade classifiers system to recognize positive images and negative images (contain ROI or not).

Regards to drowsiness detection, when the participants in the drowsiness state, their eye blink duration is over the threshold value of drowsy state (400ms). When the participants in the alert state, their eye blink duration is less than the threshold value of drowsy state (400ms). When participants feel uncomfortable at eyes, they are likely to blink their eyes unconsciously. It is hard to distinguish between involuntary and voluntary blink as their blink duration are quite similar with each other. Therefore, involuntary eye blink also affects the drowsiness detection. In addition, the project design also exists some limitations. If participants are wearing glasses, the tracking performance will be very poor. The tracking algorithm is used in this project is also based on detecting feature points of ROI. The feature points of glasses are detected instead that of eye pair region. The eye tracking performance for wearing glasses as shown in Figure 5-1.

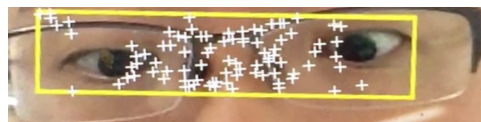


Figure 5-1 Eye tracking performance for wearing glasses

Furthermore, more than one participants' eye blink cannot be detected, this limitation is not important as the aim of project is to detect the drowsiness of single participant.

Chapter 6 Conclusions and further works

In this project, using the Viola-Jones algorithm for detecting eyes from any images. Selecting eye pair region as region of interest and training the cascade classifier system to recognize positive images and negative images for improving the accuracy of eye pair detection. Afterwards, using the KLT algorithm combine with the Camshift algorithm to track the eye pair during the real-time detection as the combination algorithm has higher tracking performance than other tracking algorithms. The eye blink detection is achieved by using the correlation coefficient algorithm which calculates the correlation score between the template image and ROI of every frame. Lighting intensity variation, head movements, head rotations, iris movements, body shakings are 5 major factors affect the eye blink detection. They also affect the accuracy of drowsiness detection.

The drowsiness detection is achieved by calculating the duration of eye blink, making judgement of blink duration, it accurately detects people drowsiness except when people have head movements, head rotations, body shakings during detection.

The contribution of this project is that it has increased the eye detection accuracy and improved the performance of eye tracking, detected eye blink and drowsiness in different drowsiness states, analyzed factors which affect the evaluation of drowsiness detection.

In the future, the influence of light variation to the real-time eye blink detection will be avoided by using Infra-red lighting technology. The effect of head movements, head rotations, body shakings will be avoided by designing a new algorithm which measure the distance between eyelids for drowsiness detection. The accuracy of drowsiness detection will be improved by training the cascade classifiers system to recognize more sets of eye pair images in different lighting conditions and more sets of negative images such as eyebrow images.

Reference

- [1] Allen et al, 'Object Tracking Using CamShift Algorithm and Multiple Quantized Feature Spaces', 2004, Vol 36.
- [2] Anderson et al, 'Sept 2013', 'Assessment of Drowsiness Based on Ocular Parameters Detected by Infrared Reflectance Oculography Journal of Clinical Sleep Medicine', Vol. 9, No. 9.
- [3] Benedetto et al, '2011', 'Driver workload and eye blink duration', Psychology and Behaviour, Vol.14(3), pp. 199-208.
- [4] Bhaskar et al, '2003', 'Blink Detection and Eye Tracking for Eye Localization', TENCON 2003, Vol.2, pp.821-824
- [5] Chau, 'Real Time Eye Tracking and Blink Detection with USB Cameras', American 12th May, viewed 18th June, <http://www.cs.bu.edu/techreports/pdf/2005-012-blink-detection.pdf>.
- [6] Emami&Fathy, 'Nov 2011', 'Object Tracking Using Improved CAMShift Algorithm Combined with Motion Segmentation', Machine Vision and Image Processing, pp.1-4
- [7] Grauman&Betke&Gips&Bradski, 'Communication via eye blinks-detection and duration analysis in real time', IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol.1, pp.I-I.
- [8] Guerrero, 'Model-Based Eye Detection and Animation', 2006, viewed 7th July, http://www.dis.uniroma1.it/~frat/tesi_svolte/sguerrero/%5BSandra%20Trejo%20Guerrero%5D%20-%20Model-Based%20Eye%20Detection%20and%20Animation.pdf.
- [9] He&Yamashita&Lu&Lao, Sept. 2009, 'SURF tracking', '2009 IEEE 12th International Conference on Computer Vision (ICCV)', pp.1586-1592.
- [10] POLATSEK, 'Eye Blink Detection', Bratislava, pp. 1-8.
- [11] L. Itti&C. Koch& E. Niebur. A model of saliency-based visual attention for rapid scene analysis. IEEE Patt. Anal. Mach. Intell, Vol 20(11), pp. 1254-1259.
- [12] Sahayadhas et al, 'Detecting Driver Drowsiness Based on Sensors: A Review', Sensors, Vol 12, pp. 16937-16953.
- [13] Soukupova, 'Real-Time Eye Blink Detection using Facial Landmarks', 21st Computer Vision Winter Workshop, Slovenia, Rimske Toplice.

- [14] Thiagarajan, 'Face Detection and Facial Feature Points' Detection with the Help of KLT Algorithm', International Journal of Advance Research in Computer Science and Management Studies, Vol 2, pp 49-54.
- [15] Tucker, 2005, 'The Duration of Eyelids Movements During Blinks: Changes with Drowsiness', viewed 15th March 2016, <http://www.mwjohns.com/>
- [16] Viola&Jones, 'Rapid Object Detection using a Boosted Cascade of Simple Features', computer vision and pattern recognition, Vol.1, pp. I-I.
- [17] Vukadinovic, 2006, 'Fully Automatic Facial Feature Point Detection Using Gabor Feature Based Boosted Classifiers', IEEE International Conference on Systems, Man and Cybernetics, Vol.2, pp.1692-1698.
- [18] Wilkinson et al, 'The Accuracy of Eyelid Movement Parameters for Drowsiness Detection', Journal of Clinical Sleep Medicine, Vol. 9, pp. 1315-1324.
- [19] Zhang, 'An Improved CamShift Algorithm for Target Tracking in Video Surveillance', Software Engineering and Applications, Vol.7(13), pp.1065-1073.
- [20] Corner detection, viewed 23th June 2016, https://en.wikipedia.org/wiki/Corner_detection
- [21] Drowsiness Detection, viewed 25th June 2016, <<http://www.optalert.com/>>.
- [22] Kanade-Lucas-Tomasi feature tracker, viewed 26th June 2016, https://en.wikipedia.org/wiki/Kanade-Lucas-Tomasi_feature_tracker
- [23] Mean shift, viewed 26th June 2016, https://en.wikipedia.org/wiki/Mean_shift.
- [24] Seeingmachines, viewed 25th June 2016, <https://www.seeingmachines.com/solutions/>.
- [25] Speeded up robust features, viewed 26th June 2016, https://en.wikipedia.org/wiki/Speeded_up_robust_features
- [26] SmartCap monitors workers' fatigue levels by reading their brain waves, viewed 25th June 2016, <http://newatlas.com/smartcap-measures-fatigue-brain-waves/21271/>
- [27] Train a Cascade Object Detector, 1994, The MathWorks, USA, viewed 25th June 2016 <https://au.mathworks.com/help/vision/ug/train-a-cascade-object-detector.html>.

Appendix

Face detection

```
%% Read the input image

I = imread('4.1.jpg');

I = imresize(I,0.6);

%% Detect face by using Viola-Jones algorithm

FaceDetect=vision.CascadeObjectDetector();

%% Returns a bounding box

BB=step(FaceDetect,I);

%% Define the rectangular frame for the bounding box, line width and style, edge color.

rectangle('Position', BB ,'LineWidth',4,'LineStyle','-','EdgeColor','b');

imshow(I);
```

Left eye detection

```
Clear

EyeDetect1 = vision.CascadeObjectDetector('LeftEyeCART');

%Read the input Image

I = imread('4.2.jpg');

I = imresize(I,0.6);

BB1=step(EyeDetect1,I);

figure,imshow(I);

for i = 1:size(BB1,1)

rectangle('position',BB1(i,:), 'LineWidth',4,'LineStyle','-','EdgeColor','b');

end
```

Right eye detection

Clear

%% Detect the right eye by using Viola-Jones algorithm

EyeDetect = vision.CascadeObjectDetector('RightEye');

%Read the input Image

I = imread('6.3.jpg');

%% resize the input image, as the input image is too big to how on the screen.

I = imresize(I,0.6);

%%BB=step(EyeDetect,I);

figure,imshow(I);

for i = 1:size(BB,1)

rectangle('position',BB(i,:), 'LineWidth',4, 'LineStyle','-', 'EdgeColor','b');

end

Eye pair detection

%To detect Eyes

EyeDetect = vision.CascadeObjectDetector('EyePairBig');

%Read the input Image

I = imread('2.1.jpg');

BB=step(EyeDetect,I);

figure,imshow(I);

rectangle('Position',BB, 'LineWidth',4, 'LineStyle','-', 'EdgeColor','b');

```
title('Eyes Detection');
```

Separate eye detection

```
>> clear

>> EyeDetect = vision.CascadeObjectDetector('RightEye');

%Read the input Image

I = imread('6.3.jpg');

I = imresize(I,0.6);

BB=step(EyeDetect,I);

figure,imshow(I);

for i = 1:size(BB,1)

rectangle('position',BB(i,:), 'LineWidth',4, 'LineStyle','-', 'EdgeColor','b');

end
```

Training eye pair detector

```
load('eye.mat');

>> Training Image Labeler app.

positiveInstances =

    imageFilename: '/Users/Apple/Documents/MATLAB/10.9.jpg'
  objectBoundingBoxes: [451 288 344 70]
% add image to the pathway
positiveFolder = fullfile(matlabroot,'toolbox','vision','visiondata',...
    'EyePair');

>> imDir = fullfile(matlabroot,'toolbox','vision','visiondata',...
    'EyePair');

addpath(imDir);

negativeFolder = fullfile(matlabroot,'toolbox','vision','visiondata',...
    'nonEyePair');

negativeImages = imageDatastore(negativeFolder);

trainCascadeObjectDetector('EyePairBigDetector.xml',positiveInstances, ...
```

```
negativeFolder,'FalseAlarmRate',0.1,'NumCascadeStages',5);
```

Eye blink detection without tracking function

```
Clear
```

```
% Read the template image
```

```
I = imread('2.3.bmp');
```

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

```
I = imresize(I,[128 344]);
```

```
% Create the eye detector object.
```

```
EyeDetect = vision.CascadeObjectDetector('EyePairBig');
```

```
% Create the webcam object.
```

```
cam = webcam();
```

```
% Capture one frame to get its size.
```

```
videoFrame = snapshot(cam);
```

```
frameSize = size(videoFrame);
```

```
% Create the video player object.
```

```
videoPlayer = vision.VideoPlayer('Position', [100 100 [frameSize(2), frameSize(1)]+30]);
```

```
frameCount = 0;
```

```
Count=0;
```

```
while frameCount < 8000
```

```
    % Get the next frame.
```

```
    videoFrame = snapshot(cam);
```

```
    videoFrameGray = rgb2gray(videoFrame);
```

```
    frameCount = frameCount + 1;
```

```
    % Detection mode.
```

```

bbox = EyeDetect.step(videoFrameGray);

% Display a bounding box around the detected face.
videoFrame = insertShape(videoFrame, 'rectangle', bbox, 'LineWidth', 4);

% Display the annotated video frame using the video player object.
step(videoPlayer, videoFrame);

%while Count < 8000
    Count= Count + 1;
    %Resize ROI to be same size as the template.
    bbox=imresize(bbox,'OutPutsize',[128 344]);
    R(Count)=corr2(I,bbox);
%end
end
% Clean up.
clear cam;
release(videoPlayer);
plot(R);

```

SURF feature matching

```

I = imread('2.3.bmp');
I= rgb2gray(I);
>> I1=imresize(imrotate(I,-30),1.2);
>> points1 = detectSURFFeatures(I);
>> points2 = detectSURFFeatures(I1);
>> [f1,vpts1] = extractFeatures(I,points1);
>> [f2,vpts2] = extractFeatures(I1,points2);

```

```

>> indexPairs = matchFeatures(f1,f2) ;
matchedPoints1 = vpts1(indexPairs(:,1));
matchedPoints2 = vpts2(indexPairs(:,2));
>> figure; showMatchedFeatures(I,I1,matchedPoints1,matchedPoints2);
>> legend('matched points 1','matched points 2');

```

SURF tracking

```

% Create an eye pair detector
EyeDetect = vision.CascadeObjectDetector('EyePairBig');

% Create a point tracker.
pointTracker = vision.PointTracker('MaxBidirectionalError', 2);

% Create a webcam object.
cam = webcam();

% Capture one frame to get its size.
videoFrame = snapshot(cam);
frameSize = size(videoFrame);

% Create the video player object.
videoPlayer= vision.VideoPlayer('Position', [100 100 [frameSize(2), frameSize(1)]+30]);

FeaturePoints = 0;
frameCount = 0;
Count=0;
while frameCount < 8000

    % Get the next frame.
    videoFrame = snapshot(cam);

    % convert the frame to be grayscale
    videoFrameGray = rgb2gray(videoFrame);

    frameCount = frameCount + 1;

    % Detection mode.

```



```

bbox = EyeDetect.step(videoFrameGray);
if FeaturePoints < 10

    % Display a bounding box around the detected eye pair region.
    videoFrame = insertShape(videoFrame, 'rectangle', bbox, 'LineWidth', 4);
    if ~isempty(bbox)

        % Find feature points within region of interest.
        points = detectSURFFeatures(videoFrameGray, 'ROI', bbox(1, :));

        % Re-initialize the point tracker.
        xyPoints = points.Location;
        FeaturePoints = size(xyPoints,1);
        release(pointTracker);
        initialize(pointTracker, xyPoints, videoFrameGray);

        % Save previous points.
        oldPoints = xyPoints;

        % Convert coordinates of corners
        bboxPoints = bbox2points(bbox(1, :));

        % Convert the box corners space

        bboxPolygon = reshape(bboxPoints', 1, []);

        % Display a bounding box around the detected eye.
        videoFrame = insertShape(videoFrame, 'Polygon', bboxPolygon, 'LineWidth', 3);
    end
else
    % Tracking mode.
    [xyPoints, isFound] = step(pointTracker, videoFrameGray);
    visiblePoints = xyPoints(isFound, :);
    oldInliers = oldPoints(isFound, :);

```

```

FPts = size(visiblePoints, 1);

if FPts >= 10
    % Estimate the transformation between the old points and new points

    [xform, oldInliers, visiblePoints] = estimateGeometricTransform(...
        oldInliers, visiblePoints, 'similarity', 'MaxDistance', 4);

    % Apply the transformation to the bounding box.
    bboxPoints = transformPointsForward(xform, bboxPoints);

    % Convert the box corners space
    % format required by insertShape.
    bboxPolygon = reshape(bboxPoints', 1, []);

    % Display a bounding box around the eye being tracked.
    videoFrame = insertShape(videoFrame, 'Polygon', bboxPolygon, 'LineWidth', 3);

    % Reset the points.
    oldPoints = visiblePoints;
    setPoints(pointTracker, oldPoints);
end

end

% Display the video frame.
step(videoPlayer, videoFrame)

end

% Clean up.
clear cam;
release(videoPlayer);
release(pointTracker);

```

Real time- eye blink detection with tracking function (combined KLT and Camshift)

Clear

% Read the template image

I = imread('9.2.jpg');

I = rgb2gray(I);

% Create an eye pair detector

EyeDetect = vision.CascadeObjectDetector('EyePairBig');

% Create a point tracker.

pointTracker = vision.PointTracker('MaxBidirectionalError', 2);

% Create a webcam object.

cam = webcam();

% Capture one frame to get its size.

videoframe = snapshot(cam);

frameSize = size(videoframe);

% Create the video player object.

videoPlayer = vision.VideoPlayer('Position', [100 100 [frameSize(2), frameSize(1)]+30]);

FeaturePoints = 0;

frameCount = 0;

Count=0;

while frameCount < 8000

 % Get the next frame.

 videoframe = snapshot(cam);

 % convert the frame to be gray

 videoframegray = rgb2gray(videoframe);

 frameCount = frameCount + 1;

 % Detection mode.

```

bbox = EyeDetect.step(videoframegray);

if FeaturePoints < 10

    % Display a bounding box around the detected eye pair region.
    videoframe = insertShape(videoframe, 'rectangle', bbox, 'LineWidth', 4);
    if ~isempty(bbox)

        % Find feature points within region of interest.
        points = detectHarrisFeatures(videoframegray, 'ROI', bbox(1, :));

        % Re-initialize the point tracker.
        xyPoints = points.Location;
        FeaturePoints = size(xyPoints,1);
        release(pointTracker);
        initialize(pointTracker, xyPoints, videoFrameGray);

        % Save previous points.
        oldPoints = xyPoints;

        % Convert coordinates of four corners
        bboxPoints = bbox2points(bbox(1, :));

        % format required by insertShape.
        bboxPolygon = reshape(bboxPoints', 1, []);

        % Display a bounding box around the detected eye.
        videoframe = insertShape(videoframe, 'Polygon', bboxPolygon, 'LineWidth', 3);
    end
else

    % Tracking mode.
    [xyPoints, isFound] = step(pointTracker, videoframegray);
    visiblePoints = xyPoints(isFound, :);
    oldInliers = oldPoints(isFound, :);

```

```

FPts = size(visiblePoints, 1);

if FPts >= 10
    % Estimate the transformation between the old points and new points

    [xform, oldInliers, visiblePoints] = estimateGeometricTransform(...
        oldInliers, visiblePoints, 'similarity', 'MaxDistance', 4);

    % Apply the transformation to the bounding box.
    bboxPoints = transformPointsForward(xform, bboxPoints);

    % Convert the box corners space
    bboxPolygon = reshape(bboxPoints', 1, []);

    % Display a bounding box around the eye being tracked.
    videoframe = insertShape(videoframe, 'Polygon', bboxPolygon, 'LineWidth', 3);

    % Reset the points.
    oldPoints = visiblePoints;
    setPoints(pointTracker, oldPoints);
end

end

% Display video frame.
step(videoPlayer, videoframe)

Count= Count + 1;

% Resize the bounding box to be same size as the template image
bbox=imresize(bbox,'OutPutsize',[135 411]);

% Calculate the correlation score
R(Count)=corr2(I,bbox);

plot(R);

% Let frames move
axis([0,frameCount+1,0,0.2]);

```

```

end

% Clean up.

clear cam;

release(videoPlayer);

release(pointTracker);

```

Drowsiness detection

```

>> [~,peak] = findpeaks(R,'MinPeakHeight',0.13);

>> peakInterval = diff(peak);

>> AverageDistance_Peaks = mean(diff(peak));

```

Implemented in Raspberry pi

```

clear
% set up the raspberry pi
mypi=raspi();

% set up camera

mycam=cameraboard(mypi,'Resolution','640*480');

mysnap=snapshot(mycam);

% Read the template image

I = imread('9.2.jpg');

I= rgb2gray(I);

% Create an eye pair detector

EyeDetect = vision.CascadeObjectDetector('EyePairBig');

% Create a point tracker.

pointTracker = vision.PointTracker('MaxBidirectionalError', 2);

% Capture one frame to get its size.

videoframe = snapshot(mycam);

frameSize = size(videoframe);

% Create the video player object.

videoPlayer= vision.VideoPlayer('Position', [100 100 [frameSize(2), frameSize(1)]+30]);

```

```

FeaturePoints = 0;

frameCount = 0;

Count=0;

while frameCount < 8000

    % Get the next frame.

    videoframe = snapshot(mycam);

    % convert the frame to be gray
    videoframegray = rgb2gray(videoframe);

    frameCount = frameCount + 1;

    % Detection mode.

    bbox = EyeDetect.step(videofamegray);

    if FeaturePoints < 10

        % Display a bounding box around the detected eye pair region.
        videoFrame = insertShape(videoframe, 'rectangle', bbox, 'LineWidth', 4);

        if ~isempty(bbox)

            % Find feature points within region of interest.

            points = detectHarrisFeatures(videoframegray, 'ROI', bbox(1, :));

            % Re-initialize the point tracker.

            xyPoints = points.Location;

            FeaturePoints = size(xyPoints,1);

            release(pointTracker);

            initialize(pointTracker, xyPoints, videoframegray);

            % Save previous points.

            oldPoints = xyPoints;

            % Convert coordinates of four corners

            bboxPoints = bbox2points(bbox(1, :));

```

```

    % format required by insertShape.

    bboxPolygon = reshape(bboxPoints', 1, []);

    % Display a bounding box around the detected eye.

    videoframe = insertShape(videoframe, 'Polygon', bboxPolygon, 'LineWidth', 3);
end
else
    % Tracking mode.

    [xyPoints, isFound] = step(pointTracker, videoframegray);

    visiblePoints = xyPoints(isFound, :);

    oldInliers = oldPoints(isFound, :);

    FPts = size(visiblePoints, 1);

    if FPts >= 10

        % Estimate the transformation between the old points and new points

        [xform, oldInliers, visiblePoints] = estimateGeometricTransform(...
            oldInliers, visiblePoints, 'similarity', 'MaxDistance', 4);

        % Apply the transformation to the bounding box.

        bboxPoints = transformPointsForward(xform, bboxPoints);

        % Convert the box corners space

        bboxPolygon = reshape(bboxPoints', 1, []);

        % Display a bounding box around the eye being tracked.

        videoframe = insertShape(videoframe, 'Polygon', bboxPolygon, 'LineWidth', 3);

        % Reset the points.

        oldPoints = visiblePoints;

        setPoints(pointTracker, oldPoints);

```



```

        end
    end
    % Display video frame.
    step(videoPlayer, videoframe)
    Count= Count + 1;
    % Resize the bounding box to be same size as the template image
    bbox=imresize(bbox,'OutPutsize',[135 411]);
    % Calculate the correlation score
    R(Count)=corr2(l,bbox);
    plot(R);
    % Let frames move
    axis([0,frameCount+1,0,0.2]);
end
% Clean up.
clear cam;
release(videoPlayer);
release(pointTracker);

```