Technology and Health Care 23 (2015) S453–S463 S453
DOI 10.3233/THC-150982
IOS Press

# Driver fatigue detection based on eye state

Lizong Lin*, Chao Huang, Xiaopeng Ni, Jiawen Wang, Hao Zhang, Xiao Li and Zhiqin Qian
*Laboratory of ROBOT, School of Mechanical and Power Engineering, East China University of Science and Technology, Shanghai, China*

**Abstract.**
**BACKGROUND:** Nowadays, more and more traffic accidents occur because of driver fatigue.
**OBJECTIVE:** In order to reduce and prevent it, in this study, a calculation method using PERCLOS (percentage of eye closure time) parameter characteristics based on machine vision was developed. It determined whether a driver's eyes were in a fatigue state according to the PERCLOS value.
**METHODS:** The overall workflow solutions included face detection and tracking, detection and location of the human eye, human eye tracking, eye state recognition, and driver fatigue testing. The key aspects of the detection system incorporated the detection and location of human eyes and driver fatigue testing. The simplified method of measuring the PERCLOS value of the driver was to calculate the ratio of the eyes being open and closed with the total number of frames for a given period.
**RESULTS:** If the eyes were closed more than the set threshold in the total number of frames, the system would alert the driver.
**CONCLUSION:** Through many experiments, it was shown that besides the simple detection algorithm, the rapid computing speed, and the high detection and recognition accuracies of the system, the system was demonstrated to be in accord with the real-time requirements of a driver fatigue detection system.

Keywords: Driver fatigue detecting, eyes detecting and locating, PERCLOS

## 1. Introduction

Driving fatigue generally refers to the state in which a driver possesses physiology and mental function deficiencies, and where driving skills decline objectively, usually after an extended period of driving. Statistics [1] show that traffic accidents due to fatigued driving accounts for about 10% of the total number of traffic accidents and around 30% of major traffic accidents. The current monitoring systems can be divided into three main categories [2]: The first category includes the monitoring method based on physiological signals such as the comparison between ECG, EEG signal acquisition, and a warning signal value which has been set; the second category consists of the monitoring method based on driver behavior such as determining the strength the driver exerts to control the steering wheel and the steering performance of the car; the third category involves the monitoring method based on the driver's visual behavior, such as yawning, eye characteristic changes, and head posture, although the detection speed and accuracy are not high.

In order to improve on the above-mentioned shortcoming, the primary focus of this paper was on human eye detection and tracking, and eye state recognition. The eye detection method was based on the Adaboost algorithm built on statistical learning and Haar-like cascade classifier. The face tracking [3]

*Corresponding author: Lizong Lin, Laboratory of ROBOT, School of Mechanical and Power Engineering, East China University of Science and Technology, Shanghai 200237, China. Tel.: +86 13166139889; Fax: +86 021 64252108; E-mail: lzlin@ecust.edu.cn.

Table 1
Internal parameter calibration results of the camera

| Internal parameters | $\alpha_x$ | $\alpha_y$ | $u_0$ | $y_0$ |
|---|---|---|---|---|
| Given value | 252.345 | 259.788 | 127.653 | 126.827 |
| Calibration value | 253.433 | 259.466 | 127.927 | 125.102 |

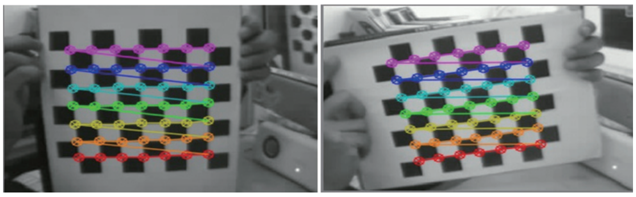


Fig. 1. Planar target images in different positions.

used the prediction combination based on the Camshift and Kalman methods. Finally, using the method based on the Adaboost cascade classifier training template to identify the human eye state, the driver's fatigue state [4] could be determined.

## 2. Detection and location of the human eye

### 2.1. Camera calibration

In order to enable the system to obtain more eye position images accurately from different directions, inner parameters V0 and alpha y of the camera were obtained using the camera calibration technique [5]. In order to obtain the parameters, the method involved the range camera in two or more planar target shooting settings, where the camera and 2D planar target had to move freely, and the camera target plane graph of different positions, is shown in Fig. 1.

In this process, the internal parameters of the camera were always the same to ensure the accuracy of the camera parameter values. The calibration results for the on-board camera parameters are shown in Table 1. Because the camera itself contained nonlinear components, a certain degree of error between the calibrations in the given value and the accurate value existed, and although it affected the distance measurement accuracy to a certain extent, it was an acceptable discrepancy for the vehicle warning system.

### 2.2. A brief description of the human eye detection method

In this study, based on statistical learning methods, the Adaboost algorithm [6,7] and Haar-like cascade classifier were used to detect the human eye with high accuracy and good robustness. The mathematical method of the Harr-like wavelet function was used to transform the expression of the image, and then the eigenvalues of various scales in different locations could be quickly calculated through notation to describe the image feature set of the rectangular area that was used for image representation. And for different target detection, the algorithms were based on a variety of scales of a sliding window of fixed sizes in the range of the specified images. The advantages of the cascade classifier were in the speed with which the results of the eigenvalues were obtained, the ability to combine a number of weak classifiers into a strong cascade classifier, and optimizing the characteristic of the Adaboost learning algorithm from coarse to fine, which fulfilled the requirements of instantaneously monitoring the index of fatigue on board. In respect to the cascade classification method, the Haar-like features of classifiers were regarded as the input and the Adaboost algorithm was regarded as a training algorithm to combine a number of simple classifiers into a strong classifier. Then, these strong classifiers were combined into a complex classifier to detect the targets in the image separately. In the commencement of the study, an abundant number of specimens, including both human eye and human non-eye objects, were collected, and then the optimal classifier could be selected using a method of building a mathematical model to train them. The static image or video stream image was scanned until the position of the human eye was located. This process of detection required a significantly long period of time because each image contained a 30 $\times$ 30 pixel sub-window that contained 200,000 rectangular characteristic features. Therefore, the design of a weak learning algorithm was necessary to select the key features that possessed the strongest classification capabilities in order to minimize classification errors to obtain the optimal threshold of a weak learning algorithm to facilitate the detection of features. As such, the weak classifiers $h_j\left(x\right)$ were introduced:

$$h_j(x) = \begin{cases} 1 & p_j f_j(x) < p_j \theta_j \\ 0 & \text{Others} \end{cases} \tag{1}$$

In the formula, $f_j$ represents the characteristic, $\theta_j$ represents the threshold value, $p_j$ represents inequality sign direction number, and x represents the 30 $\times$ 30 pixel sub-window. In each circuit training process, weak classifiers were introduced into the process, the training specimens were further optimized, and at the same time, the feature selection was also important.

### 2.3. *Eye detection process and results*

The eye detection process [8,9] was divided into two parts: First, it was necessary to detect the face in the input image, and then to detect the position of the human eye on the basis of the previously conducted work. The specific process was done in four steps as follow: 1) Selection and treatment of the human eye and non-eye training set. Human eye samples were found on the internet and the scene pictures were taken with the camera. These pictures were included in different eye positions, angles, scale sizes, etc. Also, eye size normalization, appropriate luminance compensation, and gray level changes were carried out. The non-eyes training set had a wide range, including many images not included in the area of human eyes. Here, the cascade structure classifier was introduced. The weak classifier was introduced into the process to find a strong classifier, and then repeated tests on samples were performed to train the strongest classifier. 2) Image preprocessing. Due to lighting conditions and external disturbances, noises contained in the captured images were removed and the image brightness and contrast were adjusted to reduce the impact of light and surroundings. 3) Classifier detecting images. The image was checked by the classifier from top to bottom and left to right. Because of the variety of images, the position and size of the human eye was uncertain, meaning the detection inspection window needed to be adjusted each time to ensure that different scales of the human eye could be detected to reduce the false negative rate. 4) Rechecking of the results. The classifier was able to find all the detection windows with similar human eye areas. These also included duplicate and missing areas. The experiment combined and averaged the plurality of the windows, and then the detection result was output as the final. The structure of the human eye detection process is shown in Fig. 2.

It was necessary to shoot the video stream continuously for the dozens of pictures, which had the size of 640 $\times$ 480 and occupied 90 KB. The video frame rate was between 25–31 frames per second. To validate the idea of the algorithm, it was necessary to recognize the human eye in three-dimensional situations from different angles and positions and with changes in the position of the head. During the experiment, with real-time processing of the pictures collected, the position of the left and right eye could be located in each frame. The results are shown in Fig. 3.

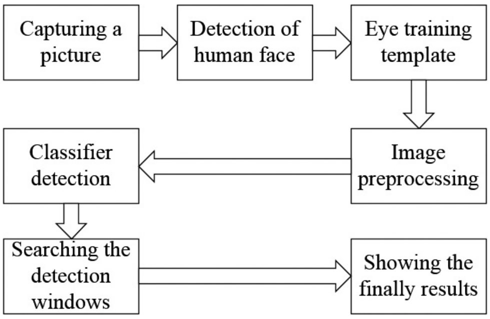
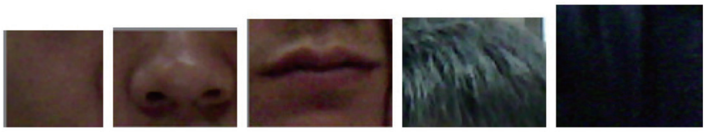
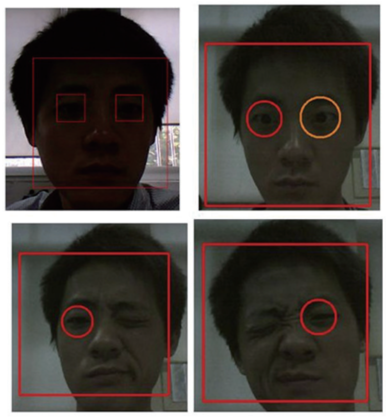



Fig. 2. Human eye detection process.

(a) Human eye samples



(b) Human non-eye samples



(c)

Fig. 3. Results of the human eye detection process.

## *2.4. Localization of the human eye*

A human head rotation model was simplified, as shown in Fig. 4, to simulate movement during driving. It was assumed that the movement of the human head was in the ideal state, i.e., the human head turns according to the center axis without swinging in a vertical direction. Line MN in the graph refers to the connection line of eyes when the driver looks straight ahead. Segment PH is the connection line when the head turns at an angle of $\alpha$. The values of these two lines are equal because the distance between the eyes does not change. PH is the projection of PQ in the forward plane, the distance between the eyes in two-dimensional pictures. According to the geometry relationship, the value of the angle between PH and PQ is equal to the angle of head rotation here. In a right-angle triangle PQH, $\angle a = \arccos\left(\frac{PQ}{PH}\right)$. A negative rotation angle is left and positive rotation angle is right.

Considering the inclination of the head is limited, in Fig. 5, coordinate $(x_1, y_1)$ is defined as the left center of the rectangle and $(x_2, y_2)$ is the right. $\Delta x = x_2 - x_1$, $\Delta y = y_2 - y_1$, and $B = arn\tan\left(\frac{\Delta y}{\Delta x}\right)$. It was divided into the following circumstances: looking left $B > 8°$, looking forward $|B| \leqslant 8°$; looking right $B < 8°$. Different lighting, external lighting interferences, vibrations, background changes, and different face directions affect the detection and location of the eyes.

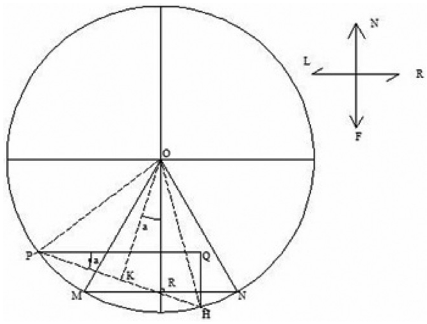



Fig. 4. Model of head motion.

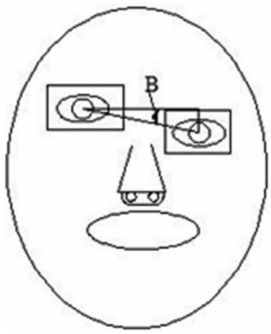



Fig. 5. Graph of eye calibration.

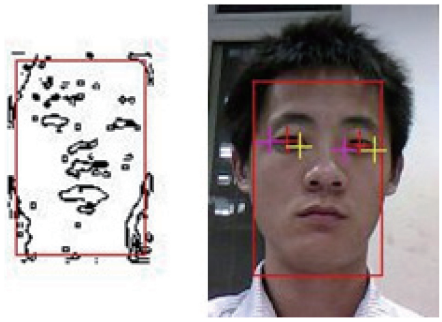


Fig. 6. Results of the eye location.

The human eye location algorithm adopted in this paper combined image pre-processing, histogram equalization, image enhancement, horizontal gray projection, and regional characteristics. After determining the face region in the image, knowing the general distribution rules of geometric features of the face (the eyes are located in the upper part of the face) narrowed the research. There are two common types of image enhancement techniques: the spatial domain method and the frequency domain method. The spatial domain method mainly performs direct calculations of the image pixel gray values in the spatial domain. It is shown in the following formula:

$$g(x, y) = f(x, y) \cdot h(x, y) \tag{2}$$

where $f(x, y)$ refers to the image before treatment; $g(x, y)$ refers to the image after treatment; and $h(x, y)$ represents the spatial computing functions. Experimental results are shown in Fig. 6.

At the same time, during the face tracking experiment combination forecasting based on the Camshift and Kalman methods, the search area in the images was shrinking. Then, the AdaBoost method was used to detect and track eye positions. Considering that the training classifier was time-consuming, the OPENCV classifier haarcascade_eye. xml file was directly adopted and the program loaded the cascade classifier while running. During the driving process, because the driver must move his eyes in different directions to see the road, the sizes of the two eye regions are not the same in the images captured by the camera. Therefore, when the search window scanned different images, the proportion needed adaptive adjustments. From top to bottom and left to right, the cascade classifier found many rectangular regions that included the object, so the rectangular regions were reviewed and then merged. The average regions of the rectangles were extracted, with different colors to mark the original image where the rectangular regions contrasted the circular ones, as shown in Fig. 7.

## 3. Detection of driver fatigue state

In this study, the purpose of face detection, eye detection, and localization and tracking of the human eye was to further identify the state of the eye (open or closed), in order to determine driver fatigue state.

### *3.1. Human eye state recognition based on adaboost cascade classifier training template*

The template matching method pre-determined the size and orientation of a small picture and then searched the target areas with the same features in the entire image through the set algorithm in order

(a) front side (b) head drop

(c) side-looking (d) side angle increase

(e) low angle (f) arm move

Fig. 7. The eye tracking experimental results.

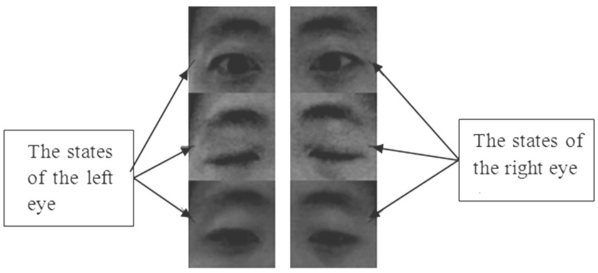



Fig. 8. Samples of eye states.

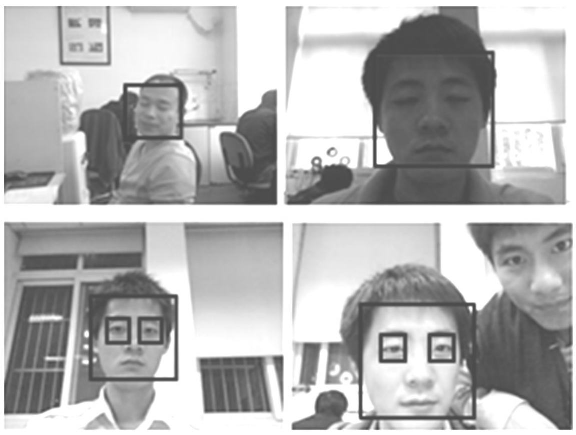


Fig. 9. Eye state recognition.

to mark its specific coordinates in the source image. The template was conducted by shift operations in the image and then compared to the measuring areas by the method of similarity comparison in the source image. When the feature matching degree between the target and the template reached a certain extent, it was determined that the small image was found. The specific steps that were carried out for feature matching included: 1) Selecting the features of the template; 2) Determining the features, size, and coordinates of the basic template; 3) Conducting the similarity calculation; 4) Matching the search strategy of the template.

Referring to the template matching method, it was necessary to first collect a certain amount of human eye samples including closed, half-open, and wide-open states of left and right eyes. Then, the samples were trained through the Adaboost algorithm combined with a multi-level classifier to find the strongest cascade classifier. The specific process was as follows:

(1) Using Haar-1ike characteristics, the human eye image gray scale distribution was described.
(2) Then, the best feature of the human eye image gray scale distribution was chosen from amounts of the Haar-1ike characteristics with the utilization of the Adaboost algorithm.
(3) Next, the same sample set was trained for different levels of the weak classifier. The final strong classifier was combined by amounts of the weak classifiers.
(4) Using the strong classifier to search the whole picture window, the areas to be measured which best matched the template were found and then the goals were determined.

The OPENCV cascade classifier was used here. The OPENCV cascade classifier contained various definitions of features and the structures of the classifier and it set specific forms of the final classifier, which was stored as an XML file. OPENCV contains numerous functions and examples to overcome common problems in computer vision fields including the analysis of movement and tracking of the target, analysis of pictures, the analysis of structures, and identification of targets. Figure 8 shows samples of different eye states measuring $80 \times 80$ pixels. These samples show how images collected from video streams were used to train the system's eye detection function. The human eye in the picture was marked with a red rectangle when the eye was fully open. Similarly, when the eye was half-open, it was also marked. When the eye was closed, the rectangular selection frame became confused, only selecting the face or nothing, assuming there was no target face or human eye in the image. Throughout the entire process, functions of OPENCV were used. The first step was to distribute memory for areas that could possibly be determined to contain human face or eye. Then, the trained cascade classifier used Haar to detect a human face and eyes in the images. Figure 9 shows the results of the former method when detecting different states of faces and eyes.

### *3.2. The PERCLOS principle*

#### *3.2.1. Brief introduction of PERCLOS*

In Chinese, PERCLOS refers to the percentage of eyelid closure over the pupil within a specified time. It originated from an experiment carried out by Walt Wierwille of Virginia University and his colleagues. The results of their experiment provided a solid basis from which to decide whether the driver was in a state of fatigue. PERCLOS was specifically defined by three criteria, P70, P80, and EM. It turns out to be maximum in correlation coefficient between P80 in ve results from fatigue [10]. The definitions of the three PERCLOS measures are as follows:

(1) P70: When the proportion of the pupil shaded by the eyelid is over 70 percent. The ratio of cl time should be found.

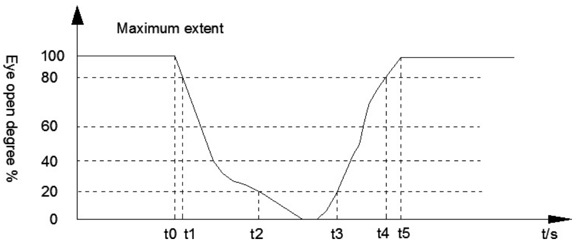



Fig. 10. PERCLOS measurement principle graph.

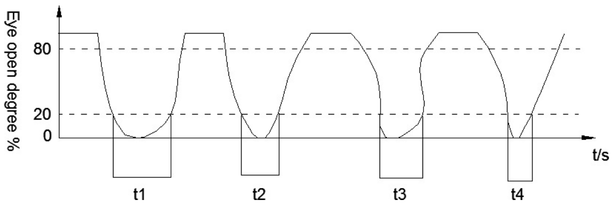



Fig. 11. PERCLOS actual process graph.

(2) P80: When the proportion of the pupil shaded by the eyelid is over 80 percent. The ratio of cl time should be found.
(3) EM: When the proportion of the pupil shaded by the eyelid is over 50 percent. The ratio of cl time should be found.

This thesis did research on the degree of fatigue using the principle of P80.

### 3.2.2. *Testing principle of PERCLOS*

The statistics and research of transportation department from home and abroad say high relieve time and fatigue. The degree of fatigue can be obtained through measuring eyelid closure time. In the actual driving experiments, PERCLOS and closure time were monitored continuously. If:

$$\text{PERCLOS} > 40\% \text{ AND Closure time } (t_3 - t_2) > 3 \text{ s}, \tag{3}$$

then they were considered fatigued and required a warning to brake emergently. A graph of the PERCLOS principles shows the cycle process of eyes from opening to closing. PERCLOS can be obtained within the time of one blink by measuring the value from $t_0 - t_5$ [11]. If the value of PECLOS within this period of time is F, the formula is as follow:

$$F = \frac{t_3 - t_2}{t_4 - t_1} \times 100\% \tag{4}$$

In this formula, F is the ratio of closure time, namely the value of PERCLOS; $t_0$ is the time of the eyes fully open; $t_1$ is the time of 20% closed; $t_2$ is the time from fully open to 80% closed; t3 is the time from fully open to fully closed to 20% opening; $t_4$ is the time from fully open to fully closed to 80% open; $t_5$ is the time from fully open to fully closed to fully open.

The methods introduced above refer to the period of one-time eye closure. In this thesis, to define the value of PERCLOS during driving, the assumed time of eye closure was referred to as n and the time the eyelid covered 80% of the pupil was referred to as Tk on the K time within continuously sampling T. The following formula was used to achieve the value of PERCLOS within the period of sampling time.

$$\mathit{PERCLOS} = \sum_{k=1}^{n} t_k / T \tag{5}$$

Figure 11 shows the actual calculating method of PERCLOS in this research. Firstly, it was necessary to establish a specific time $T$. During the time period from the first second to $T - 1$, the system did not generate a PERCLOS value. From $T$, the system calculated the value of PERCLOS before T every second to determine the final value. The camera captured images in PAL video format with a frame rate of 25–30 frames per second in the research system. With the method introduced in this paper, whether or not the eye was open could be detected. By calculating the closure proportion of the total within a period of time, the fatigue state of the driver could be detected [12,13].

Table 2
Eye state detection results

| | With interference | Without interference |
|---|---|---|
| Total frames | 100 | 100 |
| Failure number of frames detection | 15 | 17 |
| Precision | 85% | 83% |
| Average precision | 84% | |

Table 3
Eye state detection results in database

| Number of trials $N$ | Left eye position $P_1$ | Right eye position $P_2$ | Distance between two eye | Left eye state | Right eye state |
|---|---|---|---|---|---|
| 1 | (253, 292) | (372, 282) | 119.42 | 0 | 0 |
| 2 | (253, 290) | (360, 274) | 108.19 | 0 | 0 |
| 3 | (254, 291) | (372, 283) | 118.27 | 0 | 0 |
| 4 | (255, 291) | (375, 285) | 120.14 | 0 | 0 |
| 5 | (253, 292) | (376, 286) | 123.15 | 0 | 0 |
| 6 | (30, 289) | (376, 285) | 346.02 | 1 | 0 |
| 7 | (252, 286) | (369, 283) | 117.04 | 0 | 0 |
| 8 | (254, 292) | (68, 282) | 186.27 | 0 | 1 |
| 9 | (251, 291) | (363, 274) | 113.28 | 0 | 0 |
| 10 | (253, 292) | (372, 283) | 119.34 | 0 | 0 |
| 11 | (255, 287) | (375, 288) | 120.00 | 0 | 0 |
| 12 | (250, 293) | (0, 0) | 385.61 | 0 | 1 |
| 13 | (0, 0) | (374, 285) | 470.21 | 1 | 0 |
| 14 | (254, 292) | (369, 283) | 115.35 | 0 | 0 |
| 15 | (0, 0) | (372, 285) | 468.62 | 1 | 0 |
| 16 | (253, 292) | (360, 274) | 108.50 | 0 | 0 |
| 17 | (0, 0) | (0, 0) | 0 | 1 | 1 |
| 18 | (254, 291) | (375, 285) | 121.15 | 0 | 0 |
| 19 | (255, 291) | (376, 286) | 121.10 | 0 | 0 |
| 20 | (253, 292) | (376, 285) | 123.20 | 0 | 0 |
| 21 | (254, 289) | (369, 283) | 115.15 | 0 | 0 |
| 22 | (252, 286) | (372, 282) | 120.07 | 0 | 0 |
| 23 | (254, 292) | (0, 0) | 387.01 | 0 | 1 |
| 24 | (251, 291) | (372, 283) | 121.26 | 0 | 0 |
| 25 | (0, 0) | (375, 285) | 471.01 | 1 | 0 |
| 26 | (0, 0) | (0, 0) | 0 | 1 | 1 |
| 27 | (250, 293) | (376, 285) | 126.25 | 0 | 0 |
| 28 | (0, 0) | (0, 0) | 0 | 1 | 1 |
| 29 | (254, 292) | (376, 285) | 122.20 | 0 | 0 |
| 30 | (253, 292) | (369, 283) | 116.35 | 1 | 1 |

Note: Detection method noted 1 for closed eyes, 0 for open eyes

### 3.3. *Driver fatigue state recognition system*

System hardware used in this study included a PC (X86 development board), camera, and a LED light source. The software system was Visual C++6.0 in combination with the visual function library served by OPENCV to program.

The software of this system was established on the basis of Windows, and had the ability to multi-task the coordination process. It ran 32-bit Windows applications, which was upper layer software. The job control application software used was Visual C++ 6.0, which handled a larger number of Windows API functions library, forming the MFC (Microsoft Foundation Class) database. With the help of AppWizard and ClassWizard, the program established an application based on a single text and document dialog box. The class CFormView was the main dialog class of the application program. Through CView and CDialog, a variety of commonly used functions of the system were placed on the operator interface, which was extremely convenient for the information exchange between the researchers and the computer.

Here, for counting the recognition results of eye states within a certain time, an access database was established for storing the coordinates of the left and right eye, the interocular distance, and the state of each eye (when the eye was closed, it was referred to as 1, for an open eye, it was 0). For the database

ADO (ActiveX Data Object) programming, it was first necessary to create an empty table in the Access database to store the coordinates of the detected eye, the interocular distance, and eye states, and then define a connection pointer and a note library pointer in the class: ConnectionPtr, m_pConnection, _RecordsetPtr, and m_pRecordset. For example, there were 90–150 frames of images coming through every 3–5 seconds. With the human face detection method, the results could be dealt with immediately and noted in the table. It was only necessary to calculate the proportions of the images in which the eye was closed. If the result was greater than 80%, the driver was considered to be in a fatigue state and should be alerted [14]. Figure 3(c) shows the principle flow chart of the final recognition of eye state and detection of driver fatigue state in the proposed system.

In order to prove the stability and reliability of this system, many experiments were carried out: first, the driving video images without interferences were collected; next, driving video images with interferences were collected; then, the accuracy of the detection of the human eye states in the images was considered. In Table 2, the test results are summarized.

To note the left eye position $P_1$ $(x_1, y_1)$ and the right eye position $P_2$ $(x_2, y_2)$, the distance between the two eyes using pixels, written as d, the formula was calculated according to the distance from point to point:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{6}$$

In addition to detecting the state of the left and right eyes, calculating this distance can exclude some interference and re-check points to avoid mistakes. According to the experiment, and as shown in Table 3, the results showed that the left eye closed 6 times and the right eye closed 5 times, of which the average was 6 times; then, considering the total number to be 20 percent, the eyes were not found to be in a tired state. The system then updated the data as such. On the contrary, if the proportion had been more than 80 percent, it would consider the eyes tired and the corresponding alarm would be sounded.

## 4. Conclusion

Accidents caused by fatigued driving accounts for a considerable proportion of all traffic accidents. In order to reduce accidents, decrease casualties, and preserve property, driver fatigue monitoring systems have become the focus of much attention.

The purpose of this paper was based on this consideration. To determine whether a driver was in a fatigue state, a driver fatigue detection system based on machine vision was used to calculate the value of PERCLOS to determine the eye closure proportion in a given time. In addition, this paper described the detection and location of the human eye and the driver fatigue detection in detail. Experiments showed the efficiency and robustness of this detection method were very good. The results also proved this method met the requirements of a vehicle fatigue detection system.

The main conclusions of this study are as follows:

(1) Determining the suitable research method. The hardware system was based on an embedded development board as the control system, in combination with an external camera, and near-infrared light. The software system was Visual C++ 6.0, used in combination with the visual functions library served by OPENCV to program.

(2) The method developed for detecting the human eye was based on the Adaboost algorithm and Haar-like cascade classifier. Through a series of image processing functions and experimental verifications of the collected images, the algorithm was proved to have good accuracy using a red box and a green frame to mark the location of a human eye.

(3) Finally, the driver fatigue calculation method using the PERCLOS principle was detailed in this article, summing up the method of real-time calculations. The cascade classifier based on the Adaboost algorithm was trained in closed, open, half-open, and half-closed eye templates to find the strongest classifier in order to accurately recognize different states of the human eye. During the set time period, the state of each eye was detected and the locations of the left and right eyes were recorded in the database. Then, it was only necessary to calculate the proportions of the images in which the eyes were closed. If the result was greater than 80%, the system saw the driver as being in a fatigue state and needed to be alerted.

## References

[1] Fang-hui Li, Fei Wang, Peikun He. The principle and application of TMS320C6000 series DSPs (second edition) [M]. Beijing: electronic industry press, 2003.
[2] Chengling Xie, Caiming Chen, Tangyu Liu. A vehicle-mounted fatigue driving monitor[J]. Modern Electronic Technology, 2003, 36(2):121-23.
[3] M. J. Flores, A. de la Escalera. Real-time warning system for driver drowsiness detection using visual information[J]. J Intell Robot Syst, 2010, (59): 103-125.
[4] Qiong Wang, Jingyu Yang, Mingwu Ren, Yujie Zheng. Driver fatigue detection: A survey[J]. IEEE Intelligent Control and Automation, June 21-23, 2006, 2: 8587-8591.
[5] Wang Baisheng, Gao Younian, Shen Wenzhong. Method on camera calibration based on OpenCV[J]. Journal of Shanghai University of Electric Power, 2010, 26(4).
[6] Wang Xinzuil, Guo Lihong, Yang Limei, Kang Changqin. A face decetion algorithm based on AdaBoost[J]. Journal of Changchun University of Science and Technology, 2009, 32(2).
[7] Liu Huanxi, Liu Yuncai. Face detection based on AdaBoost algorithm[J]. Journal of Shanghai Jiaotong University, 2008, 42(7).
[8] Okamoto, M., Otani, S., Kaitani, Y., Uchida, K. Identification of driver operations with extraction of driving primitives[J]. Control Applications (CCA), 2011, pp: 338-344.
[9] Liying Lang, Weiwei Gu. Study of face detection algorithm for real-time face detection system[J]. Electronic Commerce and Security, 2009, 2: 129-132.
[10] Si-yu Guo, Xu-fang Zhang, Fan Zhang. Adaptive randomized hough transform for circle detection using moving window[C]. Machine Learning and Cybernetics, 2006.8, 3880-3885.
[11] Fei Rong, Cui Du-wu, Hu Bo. A novel hough transform algorithm for multi-objective detection[J]. Intelligent Information Technology Application, 2009, 705-708.
[12] Zuotao Zhang, Jiashu Zhang. A new real-time eye tracking for driver fatigue detection[J]. IEEE International Conference on ITS Telecommunication Proceedings, June 2006, 8-11.
[13] Jie Tang, Zuhua Fang, Shifeng Hu, Ying Sun. Driver Fatigue Detection Algorithm Based on Eye Features[J]. IEEE International Conference on Fuzzy Systems and Knowledge Discovery, 2010, 2308-2311.
[14] Mandalapu Sarada Devi, Dr Preeti R Bajaj. Driver Fatigue Detention Based on Eye Tracking[J]. IEEE International Conference on Emerging Trends in Engeering and Technology, 2008, 649-652.