Drivers Drowsiness Detection in Embedded System

Tianyi Hong , Huabiao Qin
College of Electronic and Information Engineering
South China University of Technology
Guangzhou, China
hongtianyister@gmail.com, eehbqin@scut.edu.cn

***Abstract*—It is a difficult problem to make drivers drowsiness detection meet the needs of real time in embedded system; meanwhile, there are still some unsolved problems like drivers' head tilted and size of eye image not large enough. This paper proposes an efficient method to solve these problems for eye state identification of drivers' drowsiness detection in embedded system which based on image processing techniques. This method break traditional way of drowsiness detection to make it real time, it utilizes face detection and eye detection to initialize the location of driver's eyes; after that an object tracking method is used to keep track of the eyes; finally, we can identify drowsiness state of driver with PERCLOS by identified eye state. Experiment results show that it makes good agreement with analysis.**

Keywords—drowsiness detection,face detection,eye location,object tracking, adaboost

I. INTRODUCTION

Many researches have been done on measuring the fatigue degree of drivers' physiology. After development of decades, researches based on image processing and pattern-recognition technology [1,2,3, 4, 5, 6] have been widely adopted.

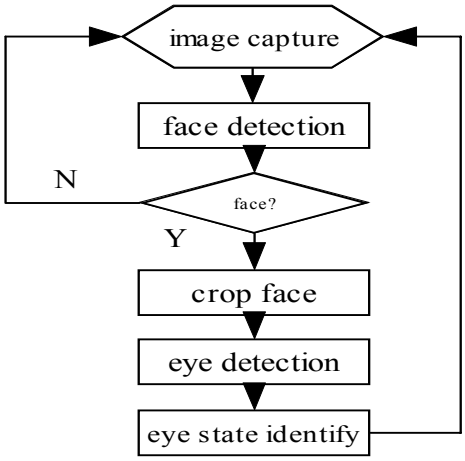

Figure 1. former drowsiness detection flowchart

Totally speaking, when identifying drivers' drowsy state by measuring PERCLOS, we have to do the following stage: 1. Face detection and face tracking. 2. Eye location and eye tracking. 3. Identification of eye state. 4. Calculation of PERCLOS. 5. Identification of drivers' drowsy state. (As in Fig.1) It is an effective way for drivers' drowsiness detection, but when it comes to applications in embedded system, Conception has to be changed. For drowsiness detection, the most important problems are accuracy and real time detection (according to survey, the shortest time of per eye blink is 0.1s, so our detection has to be limited to it). Of all the methods adopted image processing and pattern-recognition, most of them built their system basing on measuring PERCLOS (as in Fig.1). In this flowchart, however, in order to meet real time some methods[7] only detect erect face, and they can not get an effective way to get the eye image when the driver's head is tilted, some use complex methods [8] to detect eyes of the driver, but far away from requirement of real time in embedded system. In order to solve these problems, we have to adjust the whole flowchart (As in Fig.2). We can see that because drivers

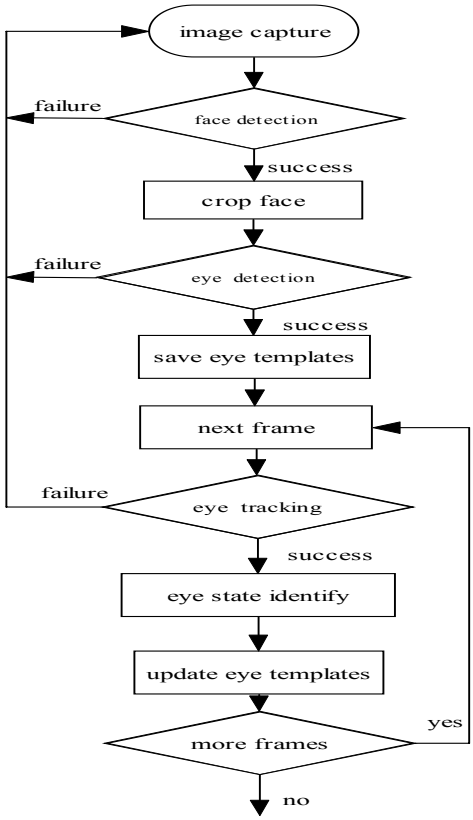

Figure 2. new drowsiness detection flowchart

do not move their head a lot while driving, after eyes location is initialized with erect face an object tracking method [9] is applied to keep track of the eyes which will save much time for eye state identification. So more accurate methods can be used for eyes states identification in embedded system. Although such conception has been adopted in [10, 11], but with more experiments on different people results seems to be not so perfect since individual eye size difference, more reasonable eye state identification method is needed. This paper uses robust tracking method, gets eye image that is large enough and finds some way to solve the problem and identify drowsiness state with PERCLOS which is a more credible criterion.

This work was sponsored by Key Science Project of Guangdong Province NO.2006A10503002 and Fundamental Nature Science Program of Guangdong Province NO.04020073

The rest of the paper unfolds as follows: Section 2 introduce methods to initialize the eye location, Section 3 talks about eyes tracking and eyes states identification, Section 4 shows the experiment results, and finally conclusion is in section 4.

II. EYE LOCATION INITIALIZED

According to Fig.2, we have to initialize the eyes location with erect face by face detection and eye located on detected face, in that case we can track the eyes by using mean shift; finally, We can calculate PERCLOS from statistical results of eyes state, after that drivers' drowsiness state is identified. Details are described in the following subsections.

A. Face Detection

In order to get higher accuracy of face detection , we adopted feature based method rather than other methods. Of all the features haar-like feature is considered to be simple and effective for face detection. There are several simple haar-like features(shown in Fig.3) that we can use. Given the haar-like features of the face, we used the face detection method basing on the cascade of classifiers trained by adaboost technique. There are several techniques to get better performance of face detection, such as integral image and cascade.

The integral image at location x,y contains the sum of the pixels above and to the left of x,y , inclusive:

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x',y') \qquad (2.1)$$

Where $ii(x,y)$ is the integral image and $i(x',y')$ is the original image, Using the following pair of recurrences:

$$s(x,y) = s(x,y\text{-}1) + i(x,y) \qquad (2.2)$$

$$ii(x,y) = ii(x-1) + s(x,y) \qquad (2.3)$$

(where $s(x,y)$ is the cumulative row sum , $s(x,-1) = 0$ and $ii(-1,y) = 0$) the integral image can be computed in one pass over the original image.

Cascade method bases on the trained data. The trained data is constructed by 22 stages, every stage is a strong classifier which consisting of numbers of weak classifiers(Fig.5) .

Fig.4 shows how to calculate haar-like feature using integral image

Fig.5 represents flowchart of the method to get face region by cascade.

After the face is detected, we copy it out to get the location of eye on it.

B. Eye Location

According to experiments eye location gets an obvious feature on the horizontal projection curve of the face. So we contrive the new eye region detection method basing on horizontal projection. And we ought to get an validation of the feature of eye on the curve of horizontal projection.

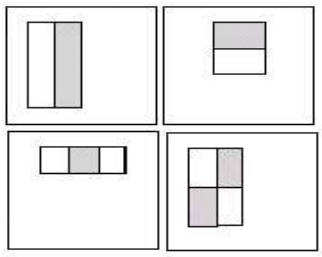

Figure 3. haar-like feature

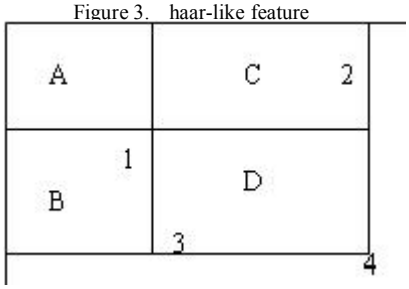

Figure 4. The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is A+B, at location 3 is A+C, and at location 4 is A+B+C+D. The sum within D can be computed as 4+1-(2+3).

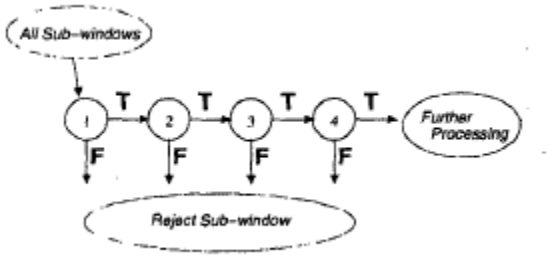

Figure 5. cascade flowchart

Our horizontal projection function (as in 2.4) is based on the gray image.

$$PV(y) = \sum_{x=1}^{m} I(x,y) \qquad (2.4)$$

Where we suppose size of the image is $m \times n$, $PV(y)$ is the horizontal projection value, $I(x,y)$ is the gray value.

We get the horizontal projection curves as shown below (Fig.6)

Figure 6. horizontal projection curves of open eyes and close eyes

In order to get the eye region we have to make an analysis of the horizontal projection curves (Fig.7).

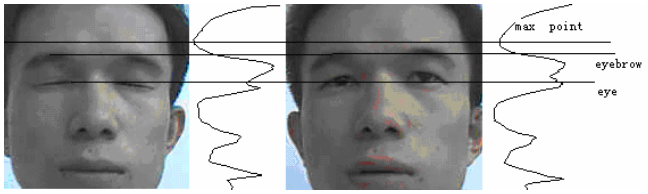

Figure 7. analysis for horizontal projection curves of open eyes and close eyes

As our analysis on the image the max point of projection is just a bit up of the eyebrow. According to the face feature, the eye was about 1/8 of face height down the max point, the eye height is about 1/6 of the face height. So we can get the eye region image by such relationship.

After that we crop eye region image out and save it as templates for eye tracking .

III. EYES TRACKING AND STATE IDENTIFICATION

After the eyes images are captured, we need to set it as a template for future tracking. Basing on mean shift arithmetic in [9], we can track the eyes, after that eye state identification method is applied.

A. Eye Tracking

A robust and nonparametric technique [9] is used in this paper. it implements the CAMSHIFT algorithm which uses a one-dimensional histogram to track an object with known hue in color images sequences. Given a color image and a color histogram, the image produced from the original color image by using the histogram as a look-up table is called back-projection image. If the histogram is a model density distribution, then the back projection image is a probability distribution of the model in the color image. CAMSHIFT detects the mode in the probability distribution image by applying mean shift while dynamically adjusting the parameters of the target distribution. In a single image, the process is iterated until convergence (or until an upper bound on the number of iterations is reached).

A detection algorithm can be applied to successive frames of a video sequence to track a single target. The search area can be restricted around the last known position of the target, resulting in possibly large computational savings. This type of scheme introduces a feed-back loop, in which the result of the detection is used as input to the next detection process. The detection algorithm can be described as following:

1. Initialize size and position of the search window.

2. Calculate the mass center (X_C, Y_C) of the window as (3.4).

$$M_{00} = \sum_{x' \le m, y' \le n} i(x', y') \qquad (3.1)$$

$$M_{10} = \sum_{x' \le m, y' \le n} x \times i(x', y') \qquad (3.2)$$

$$M_{01} = \sum_{x' \le m, y' \le n} y \times i(x', y') \qquad (3.3)$$

$$X_C = \frac{M_{10}}{M_{00}}, \quad Y_C = \frac{M_{01}}{M_{00}} \qquad (3.4)$$

3. Adjust center of the window to mass center.

4. Repeat 2 and 3 until distance of the two centers (center of the window and the mass center) is less than some threshold.

Base on the CAMSHIFT algorithm above. The tracking process is as following:

1. Get the eyes as initialized search window.
2. Convert color space to YCrCb, calculate histogram of Y and calculate back projection of the histogram;
3. Run CAMSHIFT to get the new search window.
4. In the next video frame, use the updated window as the initialized search window size and position, and return to step 2.

The only difference between this paper and [9] is the color space, we convert the color space to YCrCb instead of HSV, as density distribution H of the eyes and the face is nearly the same. Because of its adaptability so we can track the eyes when the eyes state change (close or open, the only thing we do is adjust the threshold). As in fig.8

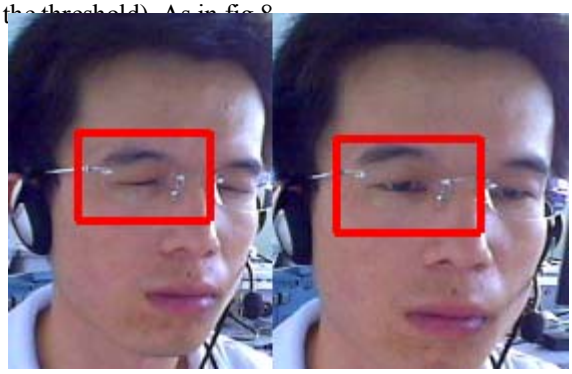

Figure 8. eye tracking results of close eye and open eye

B. Identification of eye state

Before we propose our arithmetic for identification of eye state we need to make an analysis of binary image of close eyes and open eyes (Fig.8). We can see that there is difference between complexity of the two. Open eyes' complexity are much higher than close eyes'

Figure 9. binary image of close eyes and open eyes

Basing on the eye region image, we apply a new complexity function to identify eye state. Firstly, we adopt to get the binary image (as in 2.5) by dynamic threshold (pseudo code below):

$$grayvalue = \begin{cases} 255, if\ value > threshold \\ 0, \quad if\ value < threshold \end{cases} \qquad (3.5)$$

(where value is the pixel value of the eye region image)

PSEUDO CODE:

```
Initialize threshold1
While(1)
{
Get binary image by threshold;
```

Calculate black-num (black point of the binary image);

Get sum of the four corner;

If(black-num>threshold2)

threshold1-- ;

Else

Break;

}

After that, we detect the edge of the binary image by using laplacianoperator, the operator is as below (TABLE I):

TABLE I. LAPLACIAN OPERATOR

-1	-1	-1
-1	8	-1
-1	-1	-1

Finally, we apply the improved complexity functions [12] (the functions identify one of the eyes) to identify the eye state, as below:

$$com\_x[k] = \sum_{j=1}^{m}\sum_{i=1}^{n-1} b(i,j) * k(j) \qquad (3.6)$$

$$com\_y[k] = \sum_{j=1}^{m}\sum_{i=1}^{n-1} b(i,j) * k(i) \qquad (3.7)$$

$$com\_sum = com\_x[k] + com\_y[k] \qquad (3.8)$$

(where image size is $m \times n$, $b(i,j)$ is the value of eye image after canny)

$$k(j) = \begin{cases} j, j < m/2; \\ m - j, m/2 < j < m; \end{cases}$$

$$k(i) \begin{cases} i, i < n/2; \end{cases}$$

Figure 10. corner features of close eyes and open eyes

In order to make sure the results are more accurate. We add another feature which is called corner feature of open eyes and close eyes to identify the eye state. We can see from the picture there are less corners in close eyes than in open eyes.

Considering 3×3 neighborhood S(p). we calculate covariation matrix of derivatives over the neighborhood as follows:

$$M = \begin{vmatrix} sum_{S(p)}(dI/dx)^2 & sum_{S(p)}(dI/dx \cdot dI/dy) \\ sum_{S(p)}(dI/dx \cdot dI/dy) & sum_{S(p)}(dI/dy)^2 \end{vmatrix}$$

Figure 11. covariation matrix of derivatives

Then we get eigenvalues of the resultant matrix(λ_1, λ_2), and store to the eigimage. after all the eigenvalues are fetched. We reject the corners with the eigenvalue less than Δt (Δt is related to coordinate of eigimage), and the eye corners are remained.

After that we identify eye state by com_sum and corner feature using the following function.

$$d(x) = 0.001 \times com\_sum + corners - 10; \qquad (3.9)$$

(where $corners$ is the number of eyes corner features)

$$\begin{cases} d < 0, close; \\ d \geq 0, open; \end{cases}$$

From the figure we can see that the eyes are on the top part of the face, and it is no need to search the whole face, so we can search the top part of the face to reduce computation. Also, as in Fig.8 1/9 of the face width can be cut off to get the eyes, so we can crop off unwanted part of the detected eye region to reduce computation for the next step.

IV. EXPERIMENT

The whole system has been validated on GENE-8310 embedded platform with 600MHZ, 256 M memory. Frame series in our experiment are about nine persons with 320×240 resolutions. When we test the speed that the whole system performs image frames were captured real time. However when we simulated the arithmetic accuracy, we got static image frames. The arithmetic simulation results are as followed[TABLE II]:

As in the table, the average accuracy can be about 90%, but the face and camera have to be kept in a certain distance, when the distance change the accuracy will fall down. Considered that drivers would not move their face tilted too much when they are driving, the arithmetic of the system can be supposed to be feasible.

TABLE II. ARITHMETIC SIMULATION RESULTS

Total frames		False	Miss	Correct
kinds	num	Stat. of face detection		percentage(%)
Stat of eye state identification				
Open eye	207	22		89.3
(non-glasses) Close eye	99	14		85.9
(non-glasses) Open eye	68	20		70.6
(with-glasses) Close eye	21	7		66.7
(with-glasses)				

The system processed at the average of 12 frames per second. It surely met the needs of real time.

V. CONCLUSION

This paper presents a arithmetic based on new conception to solve the problem of drowsiness detection in embedded system, the arithmetic comprised three stages: 1. Face detection

using cascade[7] and eye position detection basin on horizontal projection; 2. Eyes tracking by mean shift [9]; 3. Identify eye state by complexity function and eye corner feature. And this method is more efficient than other ones: 1.it adopted a new conception to make the whole system real time; 2. It identify eye state using complexity function to make higher accuracy of the identification result. All results show it an efficient way for drivers' drowsiness detection. However results are not good when detected eyes images are too small to identify eyes state. So future work will be done based on making it feasible to detect drivers' eye state when eyes images are too small

REFERENCES

[1] Perez, C.A.; Palma, A.; Holzmann, C.A.; Pena, C.. Face and eye tracking algorithm based on digital image processing. *Systems, Man, and Cybernetics, 2001* IEEE International Conference on Volume 2, 7-10 Oct. 2001 Page(s):1178 - 1183 vol.2.

[2] Tianjian Liu, Shanan Zhu, Eyes detection and tracking based on entropy in particle filter. *Control and Automation, 2005. ICCA '05.* International Conference on Volume 2, 26-29 June 2005 Page(s):1002 - 1007 Vol. 2

[3] Ito, T., Mita, S., Kozuka, K., Nakano, T., Yamamoto, S.. Driver blink measurement by the motion picture processing and its application to drowsiness detection. *Intelligent Transportation Systems, 2002. Proceedings.* The IEEE 5th International Conference on 2002 Page(s):168 - 173

[4] Perez, C.A.; Lazcano, V.A.; Estevez, P.A.; Estevez, C.M.. Real-time iris detection on faces with coronal axis rotation. *Systems, Man and Cybernetics, 2004* IEEE International Conference on Volume 7, 10-13 Oct. 2004 Page(s):6389 - 6394 vol.7

[5] Hayashi, K.; Ishihara, K.; Hashimoto, H.; Oguri, K.; Individualized drowsiness detection during driving by pulse wave analysis with neural network. *Intelligent Transportation Systems, 2005.* Proceedings. 2005 IEEE 13-15 Sept. 2005 Page(s):901 - 906

[6] Ilkwon Park; Jung-Ho Ahn; Hyeran Byun. Efficient Measurement of Eye Blinking under Various Illumination Conditions for Drowsiness Detection Systems. *Pattern Recognition, 2006.* ICPR 2006. 18th International Conference on Volume 1, 20-24 Aug. 2006 Page(s):383 - 386

[7] Paul Viola , Michael Jones Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, 2001.* CVPR 2001

[8] Huang, J.; Wechsler, H.; Visual routines for eye location using learning and evolution. *Evolutionary Computation, IEEE Transactions* on Volume 4, Issue 1, April 2000 Page(s):73 - 82

[9] Comaniciu, D.; Meer, P.; A robust approach toward feature space analysis; *Pattern Analysis and Machine Intelligence, IEEE Transactions* on Volume 24, Issue 5, May 2002 Page(s):603 - 619

[10] Ling Gan; Bing Cui; Weixing Wang;Driver Fatigue Detection Based on Eye Tracking;Intelligent Control and Automation, 2006. WCICA 2006.

[11] Wen-Bing Horng; Chih-Yuan Chen; Yi Chang; Chun-Hai Fan;Driver fatigue detection based on eye tracking and dynamk, template matching;Networking, Sensing and Control, 2004 IEEE International Conference

[12] Zhichao Tian; Huabiao Qin; Real-time driver's eye state detection. *Vehicular Electronics and Safety, 2005. IEEE International Conference* on 14-16 Oct. 2005 Page(s):285 - 289