ISSN 1605-8607

REAL-TIME DRIVER'S VIGILANCE DETECTION SYSTEM

Umair Ali Khan[*], Syed Raheel Hassan[**], Intesab Hussain Sadhayo[***], Zahid Hussain Abro[****]

ABSTRACT

The number of road accidents caused by non-vigilant drivers is increasing rapidly all over the world. Therefore, driver's vigilance detection is an important area of research in intelligent transportation systems. This paper presents an image processing based approach using minimum hardware to detect a driver's fatigue and generate an alarm if the driver is inattentive to driving. The system works in separate modules on each video frame from a smart, low-power, Near Infra-Red (NIR) camera installed at the vehicle's dash board. Our proposed system is based on two trained Haar classifiers to detect driver's face and eyes and then tracking the iris with the method of template matching. Our algorithm updates statistics of the iris's tracking and the direction of gaze in successive video frames. An alarm is generated if the frequency of iris detection in a specific time period goes below a certain threshold or the driver does not concentrate straight to the road. The search area of every module is restricted to a particular region of interest (user-selectable) which increases the detection speed. Fatigue detection through iris tracking by template matching provides a fast means of detecting driver's fatigue. Another module, running in parallel with the face, eyes and iris detection modules tracks the head of the driver. In the situations where the driver is wearing sun glasses, eyes cannot be detected and hence iris cannot be tracked. In this case, head tracking helps to analyze driver's fatigue. We tested our algorithm in the scenarios very close to the real cases (different lighting conditions, subjects, etc). The results presented in this paper show that our algorithm is able to maintain an adequate level of detection accuracy and its minimum architecture makes it best suited for implementation in real scenarios.

1. INTRODUCTION

The lacks of vigilance from drivers pose an imminent threat to road's safety and causes significant number of accidents. Almost 10-20% of the road accidents all over the world result from driver's fatigue [1]. While in the heavy vehicles (e.g., trucks), driver's lack of vigilance causes 60% of the fatal accidents [2]. Moreover, the accidents caused by driver's declined level of attention are far worse than those caused by other factors because the fatigued drivers usually do not take any preventive actions before an accident. Therefore, many countries have invested heavily in building intelligent transportation systems to provide secure transportation and researchers are paying more attention to the driving safety problem to decrease road accidents [26]. Developing systems for detecting drivers' vigilance level and thereby warning them to become attentive are gaining interest among the scientific communities related to intelligent transportation systems. However, an effective solution for this problem should be cost-effective, portable, small-sized, easily implementable, appealing to the end-users, and motivational for the traffic authorities in the enforcement perspective [27]. Such a system should be able to work autonomously in different environmental conditions. Additionally, it must not pose any serious threat to the driver's health.

Among various other techniques, image processing is considered to be more viable and user-friendly approach for determining driver's vigilance level than other approaches such as those based on Electro Encephalon Graph (EEG) which are considered to be intrusive [2]. The most important feature of an image processing based system is that it does not pose any significant challenge in terms of large hardware design, change in road's or vehicle's infrastructure, or a special training to the end-users. Images contain a lot of details which can be manipulated to obtain the required information about a scene. Apart from that, the image processing based

[*] Institute of Networked and Embedded Systems, Alpen-Adria Universität, Klagenfurt
[**] Deptartment of Computer Systems Engineering, Quaid-e-Awam University, Nawabshah
[***] LIPADE, University Paris Descartes, Paris
[****] Department of Information Technology, Quaid-e-Awam University, Nawabshah

systems just need an image sensor with a built-in processing hardware for running image analysis.

The motivation of this research is based on our development of a small-scale, portable, image processing based system for detecting a driver's vigilance level. The proposed system uses a smart NIR camera to cope with the problem of changing light conditions. The algorithm proposed to perform driver's fatigue detection is based on detecting driver's eyes by using our (trained) Haar classifier and then detecting driver's iris using a template matching scheme. Our minimum hardware (a single smart camera comprising 1 GHz ARM Cortex A8 processor and 512 MB RAM) and the computationally-efficient algorithm provide a good potential of implementing our system in real scenarios.

The contribution in this paper is summarized as follows:

1 We performed an analytical survey on the existing methods of driver's fatigue detection and highlight their limitations.
2 We trained two detectors based on Haar features to detect the driver's face and then eyes within a region of interest.
3 We proposed a template matching algorithm for tracking the driver's iris within the detected eye and obtain statistics of the iris detection frequency and the direction of driver's gaze. We used this information to determine the driver's attention level and to generate an alarm in case of declined vigilance level.
4 We have completely implemented our algorithm on our selected hardware and performed long-term experiments in real scenarios to demonstrate the effectiveness of the algorithm.

The rest of the paper is structured as follows. In section 2, we provide an extensive survey and critical analysis of the existing approaches for driver's fatigue detection. Section 3 gives a detailed insight into the theoretical background of Haar features and describes the method of data collection and preparation for training a Haar classifier for face and eye detection. Section 4 describes the method of AdaBoost training for generating a trained classifier and section 5 provides details of iris detection with template matching. Section 6 presents the overall algorithm with the evaluation results. Section 7 concludes the paper with a potential future work.

2. RELATED WORK

The existing approaches for Driver's Fatigue Detection (DFD) can be broadly classified into three categories: (i) Studying the driver's mental state pertaining to driving safety by psychologists and real-time monitoring of several physiological conditions, e.g., brain frequency, heartbeat, pulse beat rate, rate of respiration, etc [3]; (ii) Devising auxiliary equipments for improving driving safety by designing special car seats, monitoring grip force change on the steering wheel, or analyzing EEG recordings from sensors attached to the human body [4]; (iii) Computer vision techniques to monitor the driver's expressions. Figure 1 depicts the classification of the existing techniques for determining driver's vigilance level.

2.1. PSYCHOLOGICAL AND PHYSIOLOGICAL ANALYSIS

Analyzing driver's physiological conditions [5][6][7] deliver the most accurate results. Measuring different psychological and physiological conditions such as brain frequency, heartbeat, pulse-beat rate, respiration rate, facial expressions, body postures, and head nodding, etc provide adequate information to determine the driver's attention towards driving. However, getting all these measurements is not straightforward, cost-effective and nonintrusive. It requires a set of special sensors and hardware to be worn by the drivers which causes annoyance for most of the people. Hence these techniques are not practically acceptable [8].

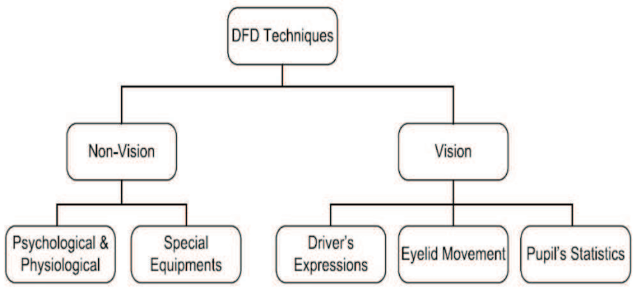


Fig. 1: Classifications of DFD Systems

2.2. SPECIALIZED VEHICLES

Some techniques for driver's fatigue detection use special vehicles with a number of sensing and processing elements installed in the vehicle to observe driver's control over the vehicle [9][10][11]. The specialized hardware used in these vehicles include electroencephalogram, devices to send a visual stimulus to the driver and receiving a stimulus-response, and the sensors to monitor the steering wheel angle and the lateral position, etc. The information obtained from the sensing hardware includes steering wheels movements, acceleration, braking, lateral position and gear changing, etc. This information is continually processed by the processing hardware to determine the level of driver's control over the vehicle. While these techniques deliver an optimal level of performance and accuracy, they are limited to vehicle type and driving conditions [12].

2.3. COMPUTER VISION

Image processing techniques are the most flexible and cheaper solution for driver's fatigue detection. Several computer vision techniques for driver's fatigue detection are studied in the literature. The technique proposed in [13] uses cascades of classifiers to detect driver's mouth and the yawning expressing. The classifiers are trained to detect mouth and yawning and used in real-time to detect yawning expressing in successive video frames. A drawback of this approach is that the driver may not necessarily yawn when he is tired. Therefore, this technique is limited to a specific psychological behavior.

The approaches proposed in [14][15] are based on the statistics of eyelid movement. In these methods, the face area is segmented from the image based on a mixed skin tone model. The process of crystallization is simulated to obtain the location of eyes within face area. Later, eye area, average height of the pupil and width-to-height ratio are used to analyze the eye's status. Finally, the driver fatigue is confirmed by analyzing the changes of eye's states. Nevertheless, the eyelid statistics used in these approaches cannot be used in a generalized way for people belonging to diverse races and having diverse eye shapes.

Another approach presented in [16] first locates the driver's face by a Haar features based object detection algorithm. The driver's eye is detected and eyelid distance is computed. By analyzing the successive changes in eyelid distance over time, driver's fatigue can be detected and a warning can be issued. However, this approach shares the same limitations as in [14][15].

In another approach [17], the authors propose a real-time eye tracking based fatigue detection system that uses a dynamically generated eye template and exploits a correlation technique to detect the driver's eye. The frequency of eye blinking is calculated and a cross-correlation based classification technique is used to determine if the eye-blinking frequency is below a certain threshold and the driver should be warned. However, detecting eyes with template matching instead of a trained classifier results in high rate of false positives.

After a critical analysis of the existing DFD techniques, we can safely conclude that the non-vision based techniques are either expensive or are intrusive. On the other hand, the vision based DFD techniques proposed in the literature are either computationally expensive or have limited accuracy.

3. OUR WORK IN COMPARISON WITH RELATED WORK

We are committed to overcome the issues pertaining to existing vision-based DFD systems. For this purpose, we combine iris and head tracking in two different modules running in parallel so that the system is effective for the situations where the driver is wearing sun glasses. Instead of directly detecting driver's eyes and thus getting numerous false positives, we train and implement two classifiers for face and eyes detection which minimize the search space for iris tracking with template matching and help to avoid large number of false positives. In contrast to the existing approaches which mainly focus on the frequency of eye blinking or the open-eyes status in successive video frames, iris tracking also helps to detect driver's direction of gaze (straight, left, right, up, down) to determine his vigilance level. Our proposed algorithm is simple, computationally-efficient, robust and requires simple infrastructure. Our implementation hardware comprises a single NIR smart camera to overcome the problems related to different light conditions.

4. THEORETICAL BACKGROUND

Our face and eye detection algorithms are based on two (trained) Haar-classifiers adapted from [18]. Haar classifiers make use of Haar features which represent the unique set of an object's characteristics (face and eye in our case) in an image and can be expressed by the concept of Haar wavelets [19].

Haar features perceive an image as a combination of small sub-images and represent the properties of individual sub-images. The set of features, in combination, reports the availability or non-availability of an object of interest within an image. For example, the separation between a black and a white region can be found by a bi-rectangular feature.

Using the features confined in rectangular areas, the authors in [20] propose summing up the pixel intensities within these regions to locate an object of interest in an image. Having an image database, the sum of the pixels corresponding to the region of an object of interest would be quite high and low for other objects of non-interest. For each object of interest, we can set a threshold value for pixel sum and compare the Haar features in a certain rectangular area with the threshold value. A pixel sum greater than the threshold value confirms the presence of an object of interest in an image [21]. This approach divides the image database into the images having objects of interest and those having different objects. Figure 2 shows the rectangular Haar features. A feature in this figure represents a scalar quantity calculated by summing up the pixels in the white region and subtracting those in the dark.

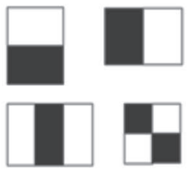


Fig. 2: Example rectangular Haar features

For each object in an image, there exists a unique set of features which represents the object classifier. There are two types of classifiers: *weak classifier* and *strong classifier*. A weak classifier, which performs naïve object detection, is obtained by subtractions of individual blocks computed within small rectangular regions of an image [22]. If a Haar feature is represented by *hj*, a weak classifier *wj(z)* for a certain threshold intensity T is defined as,

$$w_j(z) = \begin{cases} 1 & if\ p_j h_j(z) < p_j T_j \\ 0 & otherwise \end{cases}$$

where z is a sub-window in an image and p_j represents the direction of the inequality sign. A strong classifier $s(z)$ represents the combination of n weak classifiers and is defined as,

$$s(z) = sign\sum_{j=1}^{n} \alpha_j w_j(z)$$

Where α_j denotes the weight of a weak classifier $w_j(z)$ which is adjusted during the process of training strong classifier. The output of a strong classifier is 1 if the object of interest is found, or 0 otherwise. In order to train strong classifiers for face and eye detection, we use AdaBoost training [23] which finds optimal set of weak classifiers and further combines them into strong classifiers for face and eye detection. The AdaBoost training algorithm starts out with a binary labeled dataset of input feature vectors from the weak classifiers. In several rounds of learning, the weights of training samples are adjusted in order to correct the misclassifications. The convergence of the training algorithm results in a strong classifier which is the sum of the (correctly) weighted values of weak classifiers [23]. Figure 3 depicts the basic mechanism of AdaBoost training.

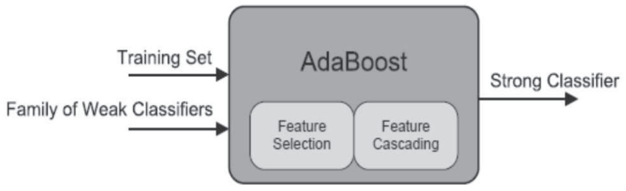


Fig. 3: Process of AdaBoost Training

5. TRAINING FACE AND EYE CLASSIFIERS

Before the AdaBoost training process, the first step involves collecting data (images) containing human faces and eyes. Two types of training samples are required: positive samples that contain the objects of interest, and negative samples that contain any type of objects other than the object of interest. We collected 2500 images each for training face classifier and eye classifier respectively from different online sources. Our image database contains images of the people from different nationalities and races. The collected data also contains 2500 negative samples for each classifier. The example positive samples for face and eyes detection are shown in Figure 4.

Fig. 4a: Positive samples for face classifier

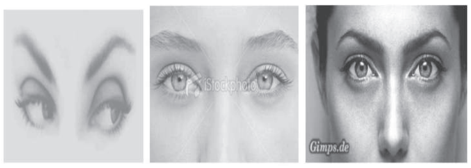


Fig. 4b: Positive samples for eye classifier

Fig. 4: Dataset for AdaBoost training

The next step includes labeling the faces and eyes in the images and cropping the images. We used an open-source utility [24] for labeling and cropping the images.

Once the dataset is ready, the next step involves feature selection from the labeled data, creating the weak classifiers and then using AdaBoost training to select the best weak classifiers and combine them into a strong classifier for both face and eye detection separately. For extracting face and eye features from the images and preparing data for AdaBoost training to generate a strong classifier, we used an OpenCV utility [25] to create training vectors comprising the face and eye feature values, respectively. This process generates 650 weak classifiers for face, and 629 weak classifiers for eye.

For AdaBoost training of face (strong) classifier, we create two sets each of *n* images: (i) the images having faces, (ii) the images having miscellaneous objects other than images. The same procedure is followed for training eye classifier with *n* images. The object and non-object images in both the cases are associated with the label O_i, where $O_i = \{1, 0\}$. The AdaBoost algorithm for training a strong classifier with n weak classifiers is described in Algorithm 1. In this algorithm, z_n represents the nth training sample and O_n represents the object or non-object label.

ALGORITHM 1: AdaBoost training for face and eye classifiers

```
* Input: training data: $(z_1, O_1), \dots, (z_n, O_n)$
* Specify training weights $\alpha_{1,i} = \frac{1}{2n_1}, \frac{1}{2n_2}$ where $n_1$ and $n_2$ represent the number of negative and positive samples, respectively.
* For every classifier indexed by $t = 1, \dots, n$
* Normalize each weight with respect to the sum of all the weights, $\alpha_{t,i} = \frac{\alpha_{t,i}}{\sum_{j=1}^{n} \alpha_{t,j}}$
* Pick the classifier with the minimum error, $\delta_t = \min_{h,p,T} \sum_i \alpha_i \mid w(z_i, h, p, T) - O_i \mid$
* Set $w_t(z) = w(z, h_t, p_t, T_t)$ where $h_t, p_t$ and $T_t$ minimize $\delta_t$
* Adjust weights: $\alpha_{t+1,i} = \alpha_{t,i} \gamma_t^{1-\mu_i}$ where $\mu_i = 0$ for correct classification, $\mu_i = 1$ for misclassification, and $\gamma_t = \frac{\delta_t}{1-\delta_t}$
* Output: the strong classifier:
    $s(x) = \begin{cases} 1 & if \sum_{t=1}^{n} \log \frac{1}{\beta_t} w_t(x) \geq \frac{1}{2} \sum_{t=1}^{n} \log \frac{1}{\beta_t} \\ 0 & otherwise \end{cases}$
```

Figure 5 shows the results of the final strong classifiers of face and eye detection. We tested the detectors on a large number of image database and found an accuracy of 99% and 98.5% for face and eye detection, respectively.

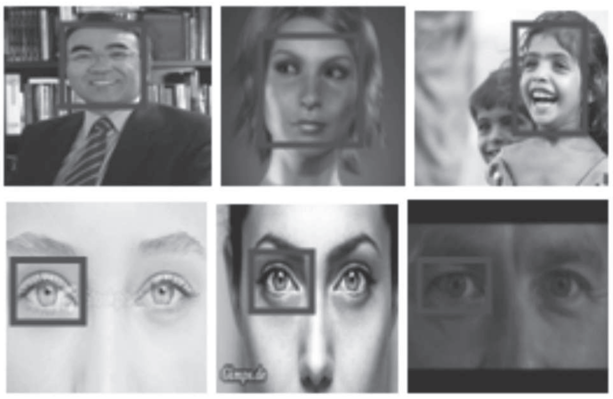


Fig. 5: Face and eye detection using the strong classifiers

6. IRIS TRACKING AND DETERMINING DIRECTION OF GAZE

We use a template matching technique for iris tracking and determining the direction of the driver's gaze. Given a sub-image (called template), the template matching technique tries to find the specified template within a large image. Template matching is an effective technique to find an object with fixed features and its position in an image. In our case, the template comprises a small pre-selected image of an iris. Since the shape of an iris does not vary significantly among different people, our template matching algorithm for iris tracking is not limited to specific physical characteristics. Once the driver's face and then eyes are detected, we have a very small search space in the image to match the iris template within the region of the detected eye. The small search space not only improves the speed of the template matching, but also helps to avoid false positives in case of low-light conditions or the presence of other similar features in the image.

Our template matching algorithm for iris tracking works as follows. We apply a template on a given image as a convolution mask and find the Sum-Of-Products (SOPs) of the neighboring pixels with the center pixel of the template. Starting with the left topmost pixel of the image, we move the template over the entire image and find the SOPs at each location. The SOPs and the template locations are temporarily stored in a look-up table. At the end of the search, all the SOPs are compared to find the one with the maximum value. The corresponding SOPs mark the location of the object (iris) in the image. Figure 6 shows the result of our template matching algorithm for iris tracking after face and eye detection.

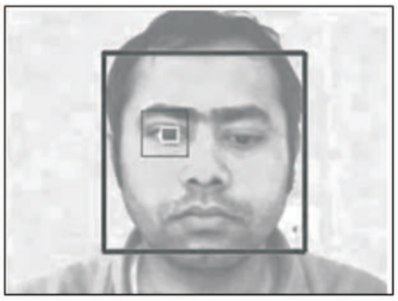


Figure. 6: Iris tracking after face and eye detection

For speeding up the iris tracking, we prefer to detect only one eye. It is natural that if the driver feels sleepy, both the eyes will begin to close at the same time. By tracking iris, we can also determine the direction of gazing. The whole algorithm works in following steps:

The iris is detected within a specific rectangular area of the eye detection. Hence, we know the rectangular coordinates of the eye area. We also know the area surrounded by the iris and the coordinates of iris position returned by the template matching algorithm. With this information, we can compute the direction of gaze by comparing the iris coordinates with the eye coordinates. Figure 7 shows the bounding box of eye and relative position of iris from the coordinates of the eye's bounding box. The black circle represents iris's position while looking straight. The shaded circles represent iris's position while looking up, down, left or right. The parameters a, b, c, d represent iris's distance from the eye's bounding box while looking straight. Whereas a', b', c', d' represent iris's distance from the eye's bounding box while looking up, right, left and down, respectively. The direction of gaze is determined by comparing the iris's relative position with respect to the coordinates of the eye's bounding box. For example, if $a \leq a'$ the driver is looking upward.

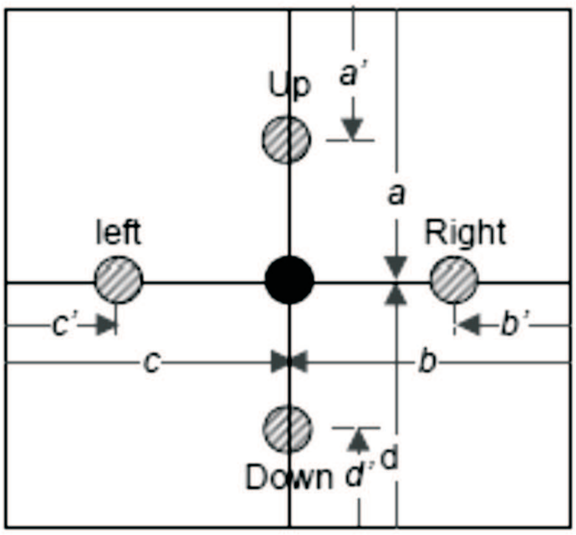
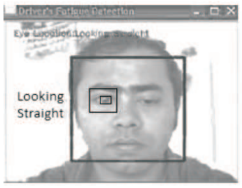


Figure. 7: Iris's relative distance from the eye's bounding box

Figure 8 shows the result of detecting direction of the driver's gaze. The directions mentioned in the figure are relative to the reader's view.



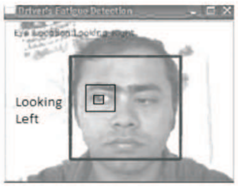
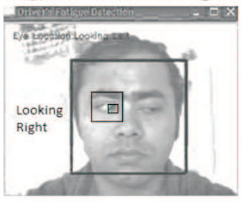


Figure 8a. Driver looking straight
Figure 8b: Driving looking left



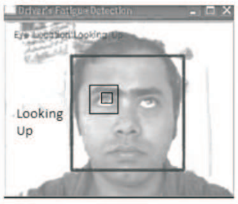


Figure 8c: Driver looking right
Figure 8d: Driver looking up

Figure 8: Directions of driver's gaze determined by the iris tracking

The face detector also returns the head coordinates that are located at the upper part of the face's bounding box. This information is useful to track the head in situations where the driver is wearing sun glasses. At the startup, the algorithm obtains the current coordinates of the head from the face detector and updates these coordinates after regular intervals in time with the information obtained from the iris detection module. If the driver is sleepy, his head will begin to go down. Therefore, when required, the algorithm compares the head coordinates with the lower part of the image (bottom coordinates of the image) to check if there is a significant difference in the vertical coordinates. This situation is depicted in Figure 9. If the difference between the *y* coordinates of previous (head coordinates updated last time) and current bounding box is found to be equal to or greater than Δy, the head is assumed to be going down and a warning is generated. The steps involved in the algorithm are listed in Algorithm 2.

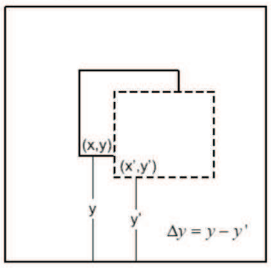


Figure 9: Difference of the last-updated and current head coordinates

ALGORITHM 2: Iris and head tracking

```
Start:
1.  Detect driver's face
2.  Track head for certain number of frames
      2.1. If driver wearing glasses?
      2.1.1. Update statistics of head coordinates.
      2.1.2. Generate alarm on the basis of updated statistics
           else
              Go to step 3.
      3. Detect driver's face
1.  Detect driver's eye
2.  Track iris by template matching
      4.1. Determine direction of gaze
      4.2. Determine frequency of iris
      4.3. Update statistics of gaze direction and iris appearance
      4.4. Generate alarm based on the updated statistics
         4.4.1. If driver wearing glasses?
                    Go to step 2.
                      else
                    Go to step 3.
      end.
```

7. OVERALL ALGORITHM AND EXPERIMENTAL RESULTS

Our experimental setup comprises a single NIR smart camera that not only has an image sensor to capture images, but is also capable of executing image processing tasks. The whole algorithm runs on the camera and the system does not require any additional processing hardware.

Our algorithm, running on the camera, starts with loading four modules, i.e., face detector, eye detector, head detector and iris tracker. At startup, the algorithm gets the head coordinates from the first instance of face detection. It then decides if the driver is wearing sun glasses by capturing and analyzing a certain number of images. If it doesn't detect eyes and iris for the next successive images, the algorithm assumes that the driver is wearing sun glasses and starts working with head tracking module only. Otherwise, for each new image, the algorithm first detects face and eye, and then makes sure that the iris is detected. It then determines the direction of gaze as mentioned in section 6. In case the iris is not detected or the driver is not looking straight, the algorithm monitors the next successive images for 3 seconds to check if this situation persists. If the algorithm finds the same behavior in the next successive frames for 3 seconds, it generates an alarm. If the driver wears sun glasses during driving, the eye detector and iris tracker stop working for the next successive images. The algorithm again starts working with head tracking module only. Figure 10 and 11 show the results of fatigue detection and generating alarm. Our algorithm is able to process images at a speed of 20 frames per second.

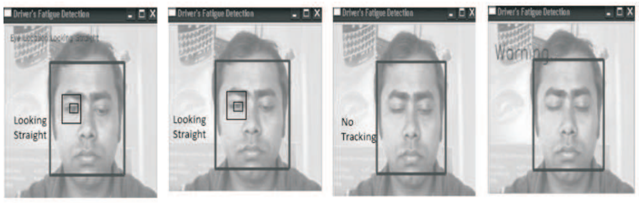


Fig. 10: Iris tracking and alarm generation in case of iris disappearance for a certain amount of time

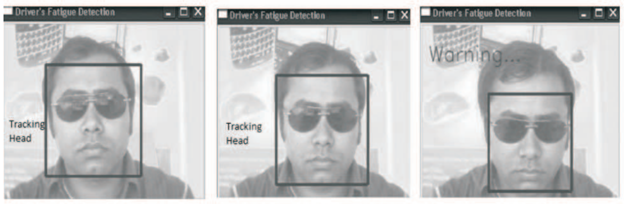


Fig. 11: Fatigue detection through head tracking and alarm generation

We tested our algorithm on a number of subjects at different times of the day under varying light conditions. During the test recordings, our subjects closed eyes on various occasions for a duration longer than a normal eye blink. We analyzed the impact of this behavior for false positives. The subjects were also asked to pretend to be sleepy in different ways. We noted down their behaviors and compared it with the alarms generated by our algorithm. Table I shows the results of our DFD algorithm.

	Test 1 (Afternoon)	**Test 2 (Evening)**	**Test 3 (Night)**
Total images processed	5000	5583	5760
Number of times eyes closed	15	18	20
Real sleep behavior	7	8	9
Alarms	7	8	8
False positives	0	0	1
False negatives	0	0	0
True positives	7	8	8
True negatives	0	0	0
Accuracy	100%	100%	88.9%
Average accuracy	95.8%		

Table 1: Evaluation results of the DFD algorithm

The results show that our algorithm is able to deliver an optimal accuracy. The only false positive detected in all the tests was during the experiment at night due to very low light.

8. CONCLUSION

In this paper, we have proposed a real-time driver's fatigue detection system based on iris tracking and (when required) head tracking. We constructed two detectors for eye and face detection. For iris tracking, we used a template matching based technique which speeds up the overall image processing pipeline. Before iris tracking, driver's face and eyes are detected and the search space is minimized for iris tracking. The statistics of iris tracking are continuously updated and if the iris tracker loses iris for a certain amount of time, the driver is assumed to be either wearing glasses or fatigued. In both the cases, an alarm is generated and a head tracking module is initialized which keeps track of driver's head and generates an alarm if the driver's head goes down below a certain threshold. In case the iris is tracked perfectly, but the driver is not found to be looking straight, an alarm is generated. Due to its simplicity and faster image processing with Haar-features based detectors and template matching, our system is capable of delivering an optimal performance with minimum hardware requirements.

Our future work in this direction includes image enhancement for extremely low-light conditions. The image enhancement module incorporated to our DFD system will first analyze the contrast level of each image and determine if any enhancements are required. For this purpose, we are working on an adaptive and computationally-efficient image enhancement algorithm. Furthermore, we are also looking for migrating our algorithm on a camera having a dedicated Digital Signal Processor (DSP) for image processing which can further improve the speed of our algorithm.

REFERENCES

[1] Bergasa L.M., Nuevo J., Sotalo M.A., and Vazquez M., *Real-time system for monitoring driver vigilance*, In Proc. of Intelligent Vehicle Symposium, pp.78-83, 2004.

[2] Awake Consortium, *System for effective assessment of driver vigilance and warning according to traffic risk estimation (AWAKE)*, Sep. 2001–2004, [Online] Available: http://www.awake-eu.org.

[3] Qiang J., Zhiwei Z., and Lan P., *Real-time nonintrusive monitoring and prediction of driver fatigue,* In IEEE Trans. on Vehicular Technology, vol.53, no.4, pp. 1052- 1068, 2004.

[4] Horng, Wen-Bing et al, *Improvements of Driver Fatigue Detection System Based on Eye Tracking and Dynamic Template Matching*, In Trans. on Information Science and Applications, Issue 1, volume 9, 2012.

[5] Healey J., and Picard R., *SmartCar: Detecting driver stress*, In Proc. of 15th International Conference on Pattern Recognition, pp. 218–221, 2000.

[6] Kircher A., Uddman M., and Sandin J., *Vehicle control and drowsiness*, Swedish National Road and Transport Research Institute, Linkoping, Sweden, Technical Report VTI-922A, 2002.

[7] Anon, *Perclos and eyetracking: Challenge and opportunity*, Applied Science Laboratories, Bedford, MA, 1999, [Online] Available: http://www.a-s-l.com.

[8] Bergasa L.M., Nuevo J., Sotelo. M.A, Barea, R, Lopez, M.E., *Real-time system for monitoring driver vigilance*, In IEEE Trans. on Intelligent Transportation Systems, vol.7, no.1, pp.63-77, 2006.

[9] Artaud P., Planque S., Lavergne C., Cara H., Lepine P., Tarriere C., and Gueguen B., *An on-board system for detecting lapses of alertness in car driving,* In Proc. of 14th International Conference on Enhanced Safety of Vehicles, pp. 350-359, 1994.

[10] Mabbott N., Lydon M., Hartley L., and Arnold P., *Procedures and devices to monitor operator alertness whilst operating machinery in open-cut coal mines. Stage 1: State-of-the-art review*, ARRB Transport Res. Rep. RC 7433, 1999

[11] Lavergne C. et al, *Results of the feasibility study of a system for warning of drowsiness at the steering wheel based on analysis of driver eyelid movements*, In Proc. of 15th International Technical Conference on Enhanced Safety Vehicles, pp. 282-291, 1996.

[12] Ueno, H., Kaneda, M., Tsukino, M., *Development of drowsiness detection system*, In Proc. of Vehicle Navigation and Information Systems Conference, pp.15-20, 1994.

[13] Sarada D., Mandalapu, and Preeti B., *Driver fatigue detection using mouth and yawning analysis,* International Journal of Computer Science and Network Security vol. 8, no. 6, pp.183-188, 2008.

[14] Du, Yong, et al. *Driver Fatigue Detection based on Eye State Analysis*, In Proc. of the 11th Joint Conference on Information Sciences. pp. 1-6, 2008.

[15] Fan, Xiao, Bao-Cai Yin, and Yan-Feng S., *Yawning detection for monitoring driver fatigue*. In Proc. of IEEE Conference on Machine Learning and Cybernetics, pp. 664-668, 2007.

[16] Ma H., Yang Z., Song Y., and Jia P. *A Fast Method for Monitoring Driver Fatigue Using*

Monocular Camera. In Proc. of Joint Conference on Information Science, pp. 1-4, 2008.

[17] Khan M., and Mansoor A., *Real time eyes tracking and classification for driver fatigue detection*, In Proc. of conference on Image Analysis and Recognition, pp. 729-738, 2008.

[18] Viola P., and Jones M., *Rapid object detection using a boosted cascade of simple features*, In Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 511-518, 2001.

[19] Stollnitz E.J., DeRose A.D., and Salesin, D.H, "*Wavelets for computer graphics: a primer.1*, In IEEE Trans. on Computer Graphics and Applications, vol.15, no.3, pp.76-84, 1995.

[20] Papageorgiou, Constantine P., Michael O., and Tomaso P., *A general framework for object detection.* In Proc. of IEEE Sixth International Conference on Computer Vision, pp. 555-562, 1998.

[21] Sang-Hyeon J., Kue-Bum L., Kwang-Seok H., *An implementation of multimodal Gaze Direction Recognition System using Image and EOG*, In Proc. of 6th International Conference on Digital Content, Multimedia Technology and its Applications (IDC), pp.229-234, 2010.

[22] Eng-Jon O., Bowden R., *A boosted classifier tree for hand shape detection*, In Proc. of Sixth IEEE International Conference on Automatic Face and Gesture Recognition, pp. 889- 894, 2004.

[23] [23] Viola P., and Michael J., *Robust real-time face detection*, International journal of computer vision vol. 57, no. 2, pp.137-154, 2004.

[24] http://sourceforge.net/projects/opencv /CV_TrainingPede.zip

[25] Bradski G., and Adrian K.., *Learning OpenCV: Computer vision with the OpenCV library*, O'Reilly Media, Incorporated, 2008.

[26] Figueiredo, Lino, et al. *Towards the development of intelligent transportation systems.* In IEEE Proceedings on Intelligent Transportation Systems, pp. 1206-1211, 2001.

[27] Khan, Umair Ali, and Bernhard Rinner. *A Reinforcement Learning Framework For Dynamic Power Management of a Portable, Multi-Camera Traffic Monitoring System.* In IEEE Proceedings on Green Computing and Communications (GreenCom), pp. 557-564, 2012.