Alert Driving System

Sushmitha. S,
Department of Information science,
Jyothy Institute of technology,
Bangalore, Karnataka, India

Sowbhagya B. N
Department of Information science,
Jyothy Institute of technology,
Bangalore, Karnataka, India

Abstract: **Road accidents are very common all over the world. It is due to the lack of attention of drivers. Facts on traffic accidents state that driver's mistake is the major reason of loss and harm on roads all over the world every day. Drivers, who do not take regular breaks when driving long distances, run a high risk of becoming drowsy, a state which they often fail to recognize early enough according to the experts. In this paper we described a module for intelligent driver monitoring system to decrease the extent of such losses which can automatically detects the driver distraction. As the distracted driving has been concerned as a causal aspect in many accidents, therefore a real-time driver monitoring system can prevent traffic accidents effectively. This paper expresses the techniques to monitor driver safety by evaluating information associated to the visual features of the driver. This monitoring system normally works based on driving prototype, driver's video and images.**

Keywords: Face detection, Eye Detection, Haar cascade, Image Acquisition.

I. INTRODUCTION

The increasing number of transportation accidents has become a serious problem for society. The traffic accidents will be largely decreased if finding a judging rule to determine whether drivers stay awake or not, and make a warning to the drivers when they begin to fall asleep, so it is meaningful to research fatigue detection algorithm which is also a key technology in smart vehicles driving. The ***driver fatigue*** problem has become an important factor for causing traffic accidents. Driver fatigue is a major cause of car accidents, since sleepy drivers are unable to make rapid decisions, and they may have slower reaction times. As a result, many governments have education program to alert people to the dangers of driving while tired, and drivers are encouraged to avoid conditions which may lead to driver fatigue. Therefore, how to supervise and avoid fatigue driving efficiently is one of the significant problems.

Recently many safety systems are followed to avoid transportation accidents. Passive safety systems such as seat belts, airbags, and crashworthy body structures help reduce the effects of an accident. In contrast, active safety systems help drivers avoid accidents by monitoring the state of the vehicle, the driver, or the surrounding traffic environment and providing driver alerts or control interventions. The proposed system focuses on the detection of drowsiness among fatigue-related impairments in driving based on eye-tracking – an active safety system [1].

Nowadays, there are many fatigue detection methods appearing and the best is tracking eye in real time using webcams to detect the physical responses in eyes. It is indicated that the responses in eyes have high relativity with driver fatigue.

Open CV

OpenCV (*Open Source Computer Vision*) is a library of programming functions mainly aimed at real-time computer vision, developed by Intel Russia research center in Nizhny Novgorod, and now supported by Willow Garage and Itseez. It is free for use under the open-source BSD license. The library is cross-platform. It focuses mainly on real-time image processing. If the library finds Intel's Integrated Performance Primitives on the system, it will use these proprietary optimized routines to accelerate it. OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are now full interfaces in Python, Java and MATLAB / OCTAVE (as of version 2.5). The API for these interfaces can be found in the online documentation. Wrappers in other languages such as C#, Perl,[Ch, and Ruby have been developed to encourage adoption by a wider audience. All of the new developments and algorithms in OpenCV are now developed in the C++ interface. OpenCV runs on Windows , Android, Maemo, FreeBSD, OpenBSD, iOS, Blackberry 10, Linux and OS X. The user can get official releases from SourceForge or take the current snapshot under SVN from there. OpenCV uses CMake.

II. PROPOSED SYSTEM

Alert driving real-time driver fatigue detection for the effective vehicle control is proposed in this paper. The system detects the driver fatigue based on eye tracking which comes under an active safety system. At first, an ordinary color webcam is used to capture the images of the driver for fatigue detection. The first frame is used for initial face detection and eye location. If any one of these detection procedures fails, then go to the next frame and restart the above detection processes. Otherwise, the current eye images are used as the dynamic templates for eye tracking on subsequent frames, and then the fatigue detection process is performed. If eye tracking fails, the face detection and eye location restart on the current frame. These procedures continue until there are no more frames.

The proposed system describes about Eye tracking and Driver Fatigue Detection. The live video is captured and stored in the database. From each of the frame the skin is extracted to detect the face of the driver alone from the frame by eliminating the objects bounding the driver and stored in a separate database. After face detection has been done the eyes are configured by giving the control points around the eyes [3].

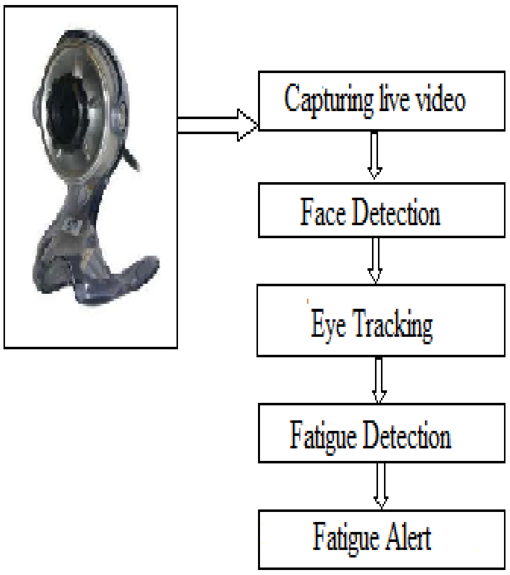


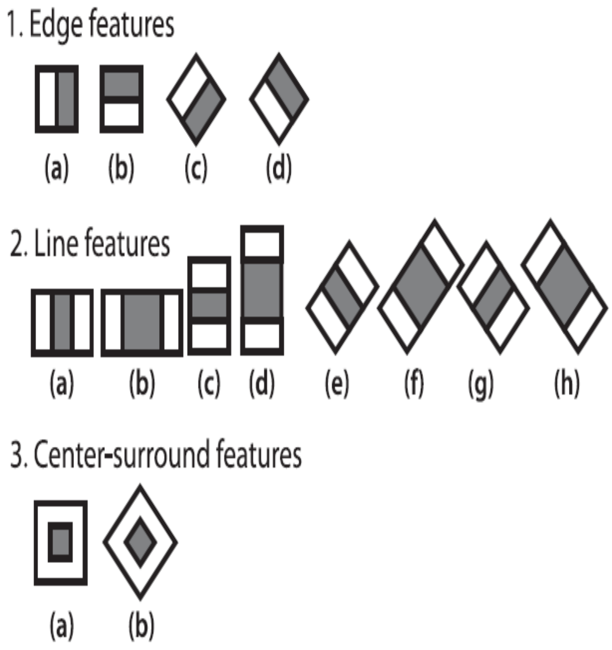
Fig. 2 Block Diagram of Proposed System

The eye template is then cropped and stored. With the stored template as reference we track the eyes of the driver continuously in the live video using dynamic template matching. If the eyes of the drivers are closed for a continuous number of frames then the driver is said to be in fatigue state and an alarm is raised.

The detection is done by a Viola and Jones features called Haar like features. This is called object detection. Object Detection using Haar feature-based cascade classifiers is effective object detection [2]. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Initially for face detection, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. Using Haar feature patterns as shown below.
Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

Viola-Jones Haar Features:



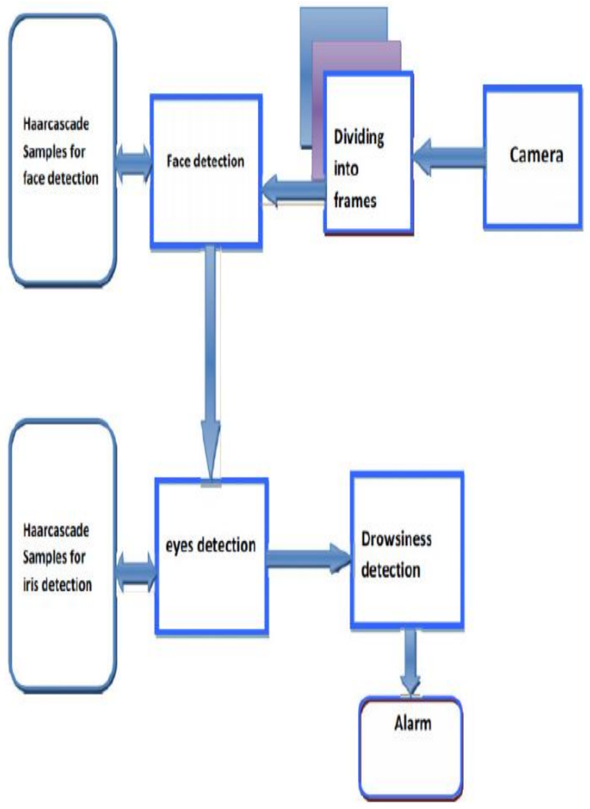
III. SYSTEM DESIGN

The system architecture of the proposed system is represented in Fig 3, Fig 3.1 Showcases the various important blocks in the proposed system and their ts high level interaction. It can be seen that the system consists of 5 distinct modules namely,
(a) Video acquisition,
(b) Dividing into frames,
(c) Face detection.
(d) Eye detection and
(e) Drowsiness detection

In addition to these there are two external typically hardware components namely, Camera for video acquisition and an audio alarm. The functionality of each these modules in the system can be described as follows:

Video acquisition: Video acquisition mainly involves obtaining the live video feed of the automobile driver. Video acquisition is achieved, by making use of a camera.

Dividing into frames: This module takes live video as input and then it converts the video into frames. And it is then preprocessed.



Face detection: The face detection function takes one frame at a time from the frames provided by the frame grabber, and in each and every frame it tries to detect the face of the automobile driver. This is achieved by making use of a set of pre-defined Haar cascade samples.

Eyes detection: Once the face detection function has detected the face of the automobile driver, the eyes detection function tries to detect the automobile driver's eyes. This is achieved by making use of a set of pre-defined Haar cascade samples.

Drowsiness detection: After detecting the eyes of the automobile driver, the drowsiness detection function detects if the automobile driver is drowsy or not , by taking into consideration the state of the eyes , that is , open or closed and the blink rate.

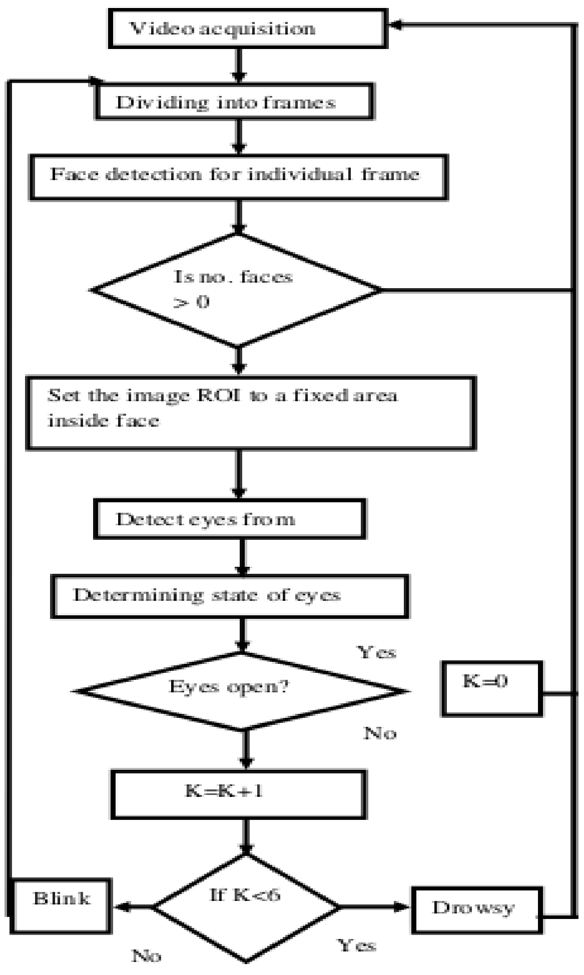
As indicated in Figure 3.2, we start with discussing face detection which has 2 important functions:
(a) Identifying the region of interest, and
(b) Detection of face from the above region using Haar cascade.

To avoid processing the entire image, we mark the region of interest. By considering the region of interest it is possible to reduce the amount of processing required and also speeds up the processing, which is the primary goal of the proposed system.

For detecting the face, since the camera is focused on the automobile driver, we can avoid processing the image at the corners thus reducing a significant amount of processing required. Once the region of interest is defined face has been detected, the region of interest is now the face, as the next step involves detecting eyes.

To detect the eyes, instead of processing the entire face region, we mark a region of interest within the face region which further helps in achieving the primary goal of the proposed system. Next we make use of Haar cascade Xml file constructed for eye detection, and detect the eyes by processing only the region of interest.

Once the eyes have been detected, the next step is to determine whether the eyes are in open/closed state, which is achieved by extracting and examining the pixel values from the eye region. If the eyes are detected to be open, no action is taken. But, if eyes are detected to be closed continuously for two seconds according to [6], that is a particular number of frames depending on the frame rate, then it means that the automobile driver is feeling drowsy and a sound alarm is triggered. However, if the closed states of the eyes are not continuous, then it is declared as a blink.



IV. IMPLEMENTATION

The implementation details of each the modules can be explained as follows:

(a) Video Acquisition

OpenCV provides extensive support for acquiring and processing live videos. It is also possible to choose whether the video has to be captured from the in-built webcam or an external camera by setting the right parameters. As mentioned earlier, OpenCV does not specify any minimum requirement on the camera, however OpenCV by default expects a particular resolution of the video that is being recorded, if the resolutions do not match, then an error is thrown. This error can be countered, by over riding the default value, which can be achieved, by manually specifying the resolution of the video being recorded.

(b) Dividing into frames

Once the video has been acquired, the next step is to divide it into a series of frames/images. This was initially done as a 2 step process. The first step is to grab a frame from the camera or a video file, in our case since the video is not stored, the frame is grabbed from the camera and once this is achieved, the next step is to retrieve the grabbed frame. While retrieving , the image/frame is first decompressed and then retrieved . However, the two step process took a lot of processing time as the grabbed frame had to be stored temporarily. To overcome this problem, we came up with a single step process, where a single function grabs a frame and returns it by decompressing.

(c) Face detection

Once the frames are successfully extracted the next step is to detect the face in each of these frames. This is achieved by making use of the Haar cascade file for face detection. The Haar cascade file contains a number of features of the face , such as height , width and thresholds of face colors., it is constructed by using a number of positive and negative samples. For face detection, we first load the cascade file. Then pass the acquired frame to an edge detection function, which detects all the possible objects of different sizes in the frame. To reduce the amount of processing, instead of detecting objects of all possible sizes, since the face of the automobile driver occupies a large part of the image, we can specify the edge detector to detect only objects of a particular size, this size is decided based on the Haar cascade file, wherein each Haar cascade file will be designed for a particular size. Now, the output the edge detector is stored in an array. Now, the output of the edge detector is then compared with the cascade file to identify the face in the frame. Since the cascade consists of both positive and negative samples, it is required to specify the number of failures on which an object detected should be classified as a negative sample. In our system, we set this value to 3, which helped in achieving both accuracy as well as less processing time. The output of this module is a frame with face detected in it.

(d) Eye detection

After detecting the face, the next step is to detect the eyes, this can be achieved by making use of the same technique used for face detection. However, to reduce the amount of processing, we mark the region of interest before trying to detect eyes. The region of interest is set by taking into account the following:

- ➢ The eyes are present only in the upper part of the face detected.
- ➢ The eyes are present a few pixels lower from the top edge of the face.

Once the region of interest is marked, the edge detection technique is applied only on the region of interest, thus reducing the amount of processing significantly. Now, we make use of the same technique as face detection for detecting the eyes by making use of Haar cascade Xml file for eyes detection. But, the output obtained was not very efficient, there were more than two objects classified as positive samples, indicating more than two eyes. To overcome this problem, the following steps are taken:

- ➢ Out of the detected objects, the object which has the highest surface area is obtained. This is considered as the first positive sample.
- ➢ Out of the remaining objects , the object with the highest surface area is determined. This is considered as the second positive sample.
- ➢ A check is made to make sure that the two positive samples are not the same.
- ➢ Now, we check if the two positive samples have a minimum of 30 pixels from either of the edges.
- ➢ Next, we check if the two positive samples have a minimum of 20 pixels apart from each other.

After passing the above tests, we conclude that the two objects i.e. positive sample 1 and positive sample 2 , are the eyes of the automobile driver.

(e) Drowsiness detection

Once the eyes are detected, the next step is to determine if the eyes are in closed or open state. This is achieved by extracting the pixel values from the eye region. After extracting, we check if these pixel values are white, if they are white then it infers that the eyes are in the open state, if the pixel values are not white then it infers that the eyes are in the closed state. This is done for each and every frame extracted. If the eyes are detected to be closed for two seconds or a certain number of consecutive frames depending on the frame rate, then the automobile driver is detected to be drowsy. If the eyes are detected to be closed in non consecutive frames, then we declare it as a blink. If drowsiness is detected, a text message is displayed along with triggering an audio alarm. But, it was observed that the system was not able to run for an extended period of time, because the conversion of the acquired video from RGB to grayscale was occupying too much memory. To overcome this problem, instead of converting the video to grayscale, the RGB video only was used for processing. This conversion resulted in the following advantages,

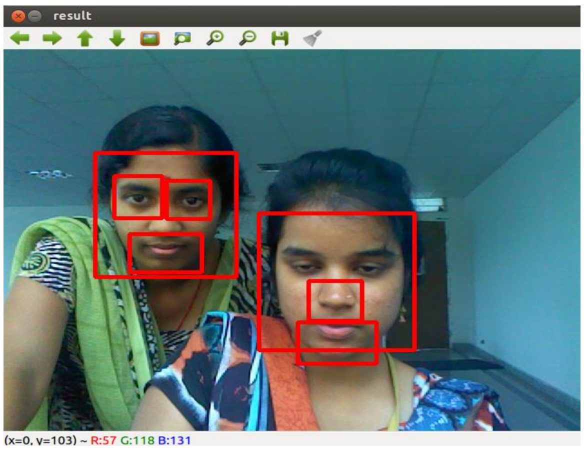
- Better differentiation between colors, as it uses multichannel colors.
- Consumes very less memory.
- Capable of achieving blink detection, even when the automobile driver is wearing spectacles.

Hence there were two versions of the system that was implemented; the version 1.0 involves the conversion of the image to grayscale. Currently version 2.0 makes use of the RGB video for processing.

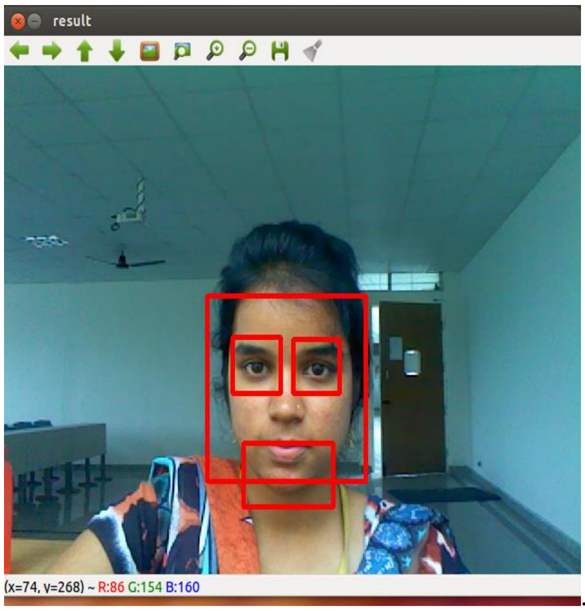
V. TESTING

STEP 1 - Face Detection: The face of driver is detected using Viola Jones Haar extended features [2]. The system uses trained Haar features. We are using haarcascade_frontface_alt.xml for the driver face detection. The XML contains trained sets of positive faces and negative faces (not faces). This is very easy and accurate method to detect real time driver face.

STEP 2 - Eye Detection: The eye of driver is (Region of Interest) ROI detecting fatigue. If eye is blinking normal rates, it means that driver is alert to drive. Whenever the driver feels drowsiness the eye blink rate is decreased (not blinked in 2to 3 seconds). This can cause fatal accidents. The eye detection is also using Viola Jones Haar features for eye detection and tracking. We have used haarcascade_ lefteye_2splits.xml is used for the left eye and haarcascade_ righteye_2splits.xml for the right eye. The two cascades are independent to one another.



Eye Blinking: The detection and tracking of real time blink of an eye is very much important in detection of driver drowsiness. Using the above there cascade i.e. Face cascade and Left eye cascade and right eye cascade



VI. CONCLUSION

The primary goal of this project is to develop a real time drowsiness monitoring system in automobiles. We developed a simple system consisting of 5 modules namely (a) video acquisition, (b) dividing into frames, (c) face detection, (d) eye detection, and (e) drowsiness detection. Each of these components can be implemented independently thus providing a way to structure them based on the requirements. Four features that make our system different from existing ones are:
(a) Focus on the driver, which is a direct way of detecting the drowsiness
(b) A real-time system that detects face, iris, blink, and driver drowsiness
(c) A completely non-intrusive system, and
(d) Cost effective

LIMITATIONS

The following are some of the limitations of the proposed system.

- The system fails, if the automobile driver is wearing any kind of sunglasses.
- The system does not function if there is light falling directly on the camera.

FUTURE ENHANCEMENT

The system at this stage is a “Proof of Concept” for a much substantial endeavor. This will serve as a first step towards a distinguished technology that can bring about an evolution aimed at ace development. The developed system has special emphasis on real-time monitoring with flexibility, adaptability and enhancements as the foremost requirements. Future enhancements are always meant to be items that require more planning, budget and staffing to have them implemented. There following are couple of recommended areas for future enhancements:

• **Standalone product:** It can be implemented as a standalone product, which can be
installed in an automobile for monitoring the automobile driver.

• **Smart phone application:** It can be implemented as a smart phone application, which
can be installed on smart phones. And the automobile driver can start the application after placing it at a position where the camera is focused on the driver.

VII. REFERENCES:

[1] D.Jayanthi, M.Bommy, Vision-based Real-time Driver Fatigue Detection System for Efficient Vehicle Control, International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-2, Issue-1, October 2012.

[2] Jie Tang, Zuhua Fang, Shifeng Hu, Ying Sun, (2010), "Driver Fatigue Detection Algorithm Based on Eye Features," IEEE proceedings of 2010 Seventh International Conference on "Fuzzy Systems and Knowledge Discovery" (FSKD 2010).

[3] Xia Liu; Fengliang Xu; Fujimura, K., "Real-time eye detection and tracking for driver observation under various light conditions," Intelligent Vehicle Symposium, 2002. IEEE , vol.2, no., pp.344,351 vol.2, 17-21 June 2002.