Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 62 (2015) 555 – 564

Procedia Computer Science

The 2015 International Conference on Soft Computing and Software Engineering (SCSE 2015)

Mobile Platform Detect and Alerts System for Driver Fatigue

Maysoon Abulkhair[a] Arwa H. Alsahli[a] Kawther M. Taleb[a] Atheer M. Bahran[a] Fatimah M. Alzahrani[a] Hend A. Alzahrani[a] Lamiaa Fattouh Ibrahim[a,b,] *

[a]*Department of Information Technology, Faculty of Computing and Information Technology, King Abdulaziz University B.P. 42808 Zip Code 21551- Girl Section, Jeddah, Saudi Arabia*

[b]*Department of Computer Science and Information, Institute of Statistical Studies and Research, Cairo University, Cairo, Egypt*

Abstract

Transition state between being awake and asleep is called drowsiness. When driver is in this state he can cause accidents because his reaction time is slower, his attentiveness is reduced, and his information processing is less efficient. Driver Fatigue Detection System (called FDS) has been proposed by the authors in a recent work. The FDS aims to monitor the driver and the alertness to prevent them from falling asleep at the wheel. In the present paper, the FDS software is modified to be run in smartphone instead of Laptop which is very hard to fixed in a car and use all advantages of smartphone like camera and late weight. The proposed system will solve this problem by using a mobile phone camera; the phone will be put on a stand in the car to make the driver feels comfortable. The proposed system has hardware and software components such as mobile camera and Android SDK. Both components are integrated together to record real video for the driver, and then processing it for real-time eye tracking. This system has reserve all advantages in FDS like fast and real-time face and eye tracking, external illumination interference is limited, more robustness and accuracy allowance for fast head/face movement. The Main goals of this system are to ensure that the driver is staying awake during his drive, make the driver feels comfortable and to help decrease the number of accidents.

Peer-review under responsibility of organizing committee of The 2015 International Conference on Soft Computing and Software Engineering (SCSE 2015)

Keywords: Driver fatigue; Image processing; Mobile Platform

* Corresponding author. Tel.: +201063220528 or +966557279106.

E-mail address: lfibrahim@kau.edu.sa

Peer-review under responsibility of organizing committee of The 2015 International Conference on Soft Computing and Software Engineering (SCSE 2015)

doi:10.1016/j.procs.2015.08.531

1. Introduction

For the comfortable drive and the safety of drivers, the intelligent car has been developed[1]. Many algorithms and architectures have been proposed in order to prevent the unexpected accidents during drive[2-7]. The significant factor in a large number of car accidents is driver fatigue. Recent, about 1500 deaths and 40,000 damages can be attributed annually to fatigue related crashes[8].

Three techniques are used to detect Driver fatigue: physiological measurements, driving performance and visual cues. The driver is monitoring directly in Physiological and visual cues, by in driving performance the driver is monitoring indirectly. Techniques of driver fatigue detection are overviewed in[9].

Measure features such as brain waves (EEG)[10], eye movements (EOG) and the electrocardiogram (ECG) signals of driver, which are Physiological measurements, are measured by attaching electrodes to the driver[11]. Attaching electrodes to the driver disturb him.

By monitoring how the driver handles the vehicles, such that the variations in the steering wheel angle, vehicle lateral position or vehicle speed, system can detect driver fatigue[12]. Therefore, trajectory of the vehicle can be used as a metric of fatigue. Sayed and Eskandarian[13] and Eskandarian and Mortazavi[14] proposed prototype systems constructed on this fatigue metric.

E-class Mercedes-Benz vehicles in 2008 has introduced the Attention Assist system[15] which is based primarily on steering wheel movements. This system is sensitive to unfortunate roads, variation in weather and driver expertise. Therefore driver performance is not use in our implementation.

In Wierwille et al. [16] development of the PERCLOS (percentage of eye closure) metric of fatigue which is a Visual cues from a driver's face as an indicator of fatigue. Dinges et al.[17] Confirms the scientific validity of PERCLOS. Many commercially system are implemented using PERCLOS metrics for examples of such systems are AntiSleep developed by Smart Eye AB[18] and Driver State Sensor (DSS) developed by Seeing Machines[19].

Based on the projection of the image many propose systems are invention such as Tabrizi and Zoroofi[20] the vertical projection of the image of both eyes is used to determine the state of an eye. Method based on combination of projection and the geometry feature of iris and pupil is proposed by Zhang and Zhang[21]. Devi and Bajaj[22], a horizontal projection image of an eye is used to determine the interval between eyebrows and eyelids and to recognize the state of an eye. Hong et al.[23] , to determine state of an eye, the horizontal projection of the image of a face is calculated.

Artificial neural networks, SVMs and AdaBoost are three different classification techniques used by Coetzer and Hancke[24]. This work proof that the AdaBoost is suited for eye classification in a real-world application,

In this paper, the developers will focus on measuring the percentage of eyelid closure covering the pupil over time, because this vision-based method is not intrusive and will not cause annoyance to drivers, and it gives accurate results. Section 2 summarizes the design and implementation of the system, Section 3 demonstrates the comparison of WakeApp and other systems. The paper's conclusions are presented in Section 4.

2. Design and implementation of the system

Driver Fatigue Detection System (called FDS) has been proposed by the author in a recent work[25]. The FDS aims to monitor the driver and the alertness to prevent them from falling asleep at the wheel. In the present paper, the FDS software is modified to be run in smartphone instead of Laptop which is very hard to fixed in car and use all advantages of smartphone like camera and late weight. WakeApp application uses image processing technique and Snapdragon library which is a library in Qualcomm Company used to detect driver's fatigue state. Image processing realizes highly accurate and reliable detection of drowsiness. Also it offers a non-disturbing approach for detecting drowsiness without disturbed the driver. The developers have seen some of the popular image processing techniques. These techniques are: yawning detection, head nodding detection and eyelid movement. Yawning technique has several drawbacks such that lip positions are difficult to detect precisely. Head nodding technique also has many drawbacks which are: this method requires equipment to be attached by electrode to the vehicle operator, which can be intrusive. The operator performance has probably already declined to unsafe levels before the head nods forward in a fatigued sleepy state. The system detects drowsy state using eyelid movement technique.

2.1 Algorithms used:

This section explain the different algorithms used in WakeApp system.

2.1.1 Face detection using Haar-like Classifier Cascades

In this system the developers use Data Classification Mining technique for detecting face. Classification is "the task of assigning objects to one of several predefined categories". It is "the task of learning a target function (classification model) that maps each attribute set to one of the predefined class label". The classification model can serve as an explanatory tool to distinguish between object of different classes "[25]. Figure 1 illustrates the face detection procedure used in the WakeApp system.

```
Create empty Array
Copy the data (pixels) from the original image
Detect face using haar-like classifier
If face found then return a pointer to the face rectangular regions
   Else go to detect face
```

Fig. 1 pseudo code of face detection procedure

2.1.2 Eye detection using template matching algorithm

Template Matching is used to detect object. In this template matching method, input image is created by a standard eye pattern manually. To determine the existence of an eye the correlation values with the standard patterns are computed for the eyes and is determined based on the correlation values. This approach is simple to implement[26].

In this work, the new technique from QUALCOMM Company which name Snapdragon is used. This is the highest performing mobile processor with the most advanced 4G LTE connections. It is designed to uses less power than any other mobile processor with integrated 4G LTE and 64-bit computing performance[27,28].

2.2 System Operation

The system will start by the user, by choosing "Start camera" button. Also, it uses high resolution digital video camera to detect the face and eyes in the real time. The developers used several methods to detect the face and eyes:

2.2.1 Snapdragon SDK for Android:

Snapdragon SDK for Android is a package of software libraries, sample code, and documentation designed to make it easy to integrate a host of next-gen technologies into apps that run on Snapdragon processor-powered mobile devices. Now, developers and device manufactures can include capabilities like facial processing, facial recognition, and more to transform the user experience[28].

2.2.2 Facial Processing

Its library in Snapdragon SDK which included intelligent automation of the live camera and auto-filtering of optimal images in a photo library, both based on faces in an image with eyes open and facing the camera. Other concepts have revolved around face and head tracking as user input for hands-free remote control by tilting the head or tracking eye blinks

The facial processing track a variety of facial properties with each frame:

- Blink Detection – measure how open each eye is
- Gaze Tracking – assess where the subject is looking
- Smile Value – estimate the degree of the smile
- Face Orientation – track the Yaw, Pitch and Roll of the head

These capabilities work with both real-time and stored images and videos[29].
The application has many choices that are:

1. Start Camera: This option will start the main goal of the application. When the user chooses this option the system will access phone's front camera by default, to start detect user's face and eyes.
2. Settings: From this option the user can control several features such as: tone of the alarm, tone volume, and choose which camera that the user wants to used (front or back camera).
3. How To Use: This option will help the user to knowing how to use the application by illustrations.
4. About Application: Contains a brief description about the application as well as a "Share" button that allow the user to share through social media.
5. Contact Us: The functionality here opens new window which lets the user to be contacted with the developers by choosing one of the mail applications.

2.3 System Requirements:

This section describes system requirements which include the hardware and software.

2.3.1 Hardware

We will illustrate the hardware needed in this section.

2.3.1.1 Built in Phone Camera

The developers use phone built-in camera to detect the driver's eyes. In order to make the camera works as a monitoring camera, the developers implemented the application used that camera to alert the driver.

2.3.1.2 Stand for Mobile

This tool is purchased by the user, it's used to stand the mobile in a suitable position to allow the application to detect driver's eye and to improve application performance.

2.3.1.3 Car's Charger for the Mobile

WakeApp will be running while the user is driving, thus to prevent loosing the mobile charges, the user needs the charger to charge his mobile phone.

2.3.2 Software:

This section lists the variance software in system.

2.3.2.1 Snapdragon SDK for Android

As mentioned earlier Snapdragon SDK is a package of software libraries, which is working in Android mobile platform. It has sample code, and documentation designed to facilitate the apps development and to integrate a host of next-gen technologies into apps that run on Snapdragon processor-powered the mobile devices. These features help WakeApp developers to include capabilities like facial processing, facial recognition, and more to transform the driver's experience.

2.3.2.2 Android OS:

Android is a software stack for mobile devices that includes an OS, middleware and key applications. It's also a free, open source mobile platform. Android is not a device or a product and it is not even limited to a certain brand of mobile phones - you could build a DVR, a handheld GPS, an MP3 player[29,30].

2.3.2.3 Eclipse:

Eclipse is a platform used for building integrated web and application development tooling. It helps in acceleration of development of integrated features based on a plug-in model. "It provides a common user interface (UI) model for working with tools"[31].

2.3.2.4 Java SDK:

To build, test, and debug the Android application, we used the Android SDK which provides the API libraries and developer tools necessary. It consists of the essential Android SDK components and a version of the Eclipse IDE, with built-in ADT (Android Developer Tools)[32].

2.3.2.5 Android NDK:

"The NDK is a toolset that allows implementing parts of the application using native-code languages"[33].

2.4 Work procedures

In the project there are several files that did the basic work of it - data processing and handling of the system- these files are as follow:

1- CameraPreviewActivity.java This is the key file in the entire project, and it is also the large one in terms of number of code lines.
2- CameraSurfacePreview.java It provides a way to get the contents of the camera, or in other words, the camera turns to a usable surface and works with it directly so that the images can be processed either by reading from or writing on them. This file depends heavily and directly on the Android libraries, especially in dealing with the camera where it inherits directly the class SurfaceView, and uses the interface Surface Holder to follow up on any changes occur on the surface of the camera's image.
3- DrawView.java This file has a simple role. It draws directly on the video, but based on the calls he receives from the key file. All information sent from the key file and this file as we have said hereditary from SurfaceView but here it is used to draw on the surface.
4- AlarmActivity.java This file is responsible for issuing the alarm sound after detecting the closure of the eyes, based on the information in the screen settings file such as voice alarm and sound level.

Once a video frame enters the system it goes through a series of functions and methods. Each function will perform an operation on the image in a real time to detect the face and eyes of the user. The sequence of these operations will be performed in the main class (CameraPreviewActivity) as the following:

1- From *Start Camera* button the camera will be opened by calling it from the main camera class and it will be ready to detect and gather information about user face and eyes.
2- Getting instance from facial processing library.
3- From camera surface preview class the developers initiate new CameraSurfacePreview() object to prepare a good surface for camera to reach the components of the camera and it can uses the camera in easy and direct way to handle pictures either by reading from or writhing on the pictures.
4- After that facial processing sets frame by setFrame() method.
5- Then it uses getNumFaces() to return the number of faces that are displayed on the camera surface. If it returns > 0 that means there is a face to detect so it will continue to the next function, else it will repeat the method until it finds the face.
6- When the face has been detected, the new DrawView() method from DrawView class will be activated to determine right eye, left eye and the face.
7- addView(drawView) will be added to the camera surface preview.
8- Final step is in the alarm activity which it will compare intBlinkCount with detectspeedvaribles, and if intBlinkCount>detectspeed the alarm will be activated.
9- After that stopping the alarm will cause returning to the camera preview activity class.
10- Stopping the camera and finishing the main function.

2.5 Quality requirements

"Quality requirements ensure the system possesses quality attributes such as: usability, efficiency, reliability maintainability and reusability":

1- Response time: WakeApp application is a real time system. So the response time is the time elapsed between the dispatch (time when task is ready to execute) to the time when it finishes its job. In the system, the response time between analyzing the frames and alerting the driver is approximately 40 ms.
2- Resource usage: WakeApp application needs at maximum 45-50 MB RAM. Also the application consume between12-18% of CPU's time.
3- Reliability: WakeApp application may suffer one failure at a year.
4- Availability: WakeApp application must be 99.9999% of the time.
5- Allowances for maintainability and enhancement: WakeApp application can be improved to have more features.
6- Allowances for reusability: In WakeApp application, 20% of analyzing function will be designed generically in order to be reused.

2.6 User interface design

User interface is the space where interaction between humans and machines occurs and allow user in making operational decisions to operate machines[34].

Figure 2 represents the splash page that will firstly present the user, it contains the logo of the application and it will load the application. Figure 3 shows the main interface of the application. There are five basic buttons and the logo of the application.

Fig. 2 Splash screen

Fig.3 main page

2.6.1 Buttons Description of the main page:

All the five buttons are used to transform to another page, this buttons are:

1. **Start Camera button:** This button takes the user to the page that allows the user to start running the application by turning on the camera and start initiate tracking. It has a sub-page that contains *"Stop Alert"* button which allows the user to stop the alarm.
2. **Setting button:** This button takes the user to the page that allows the user to change the default settings of the application; it allows the user to control which tone he/she wants, the size of sound volume, and choose which camera he/she wants.
3. **How to use button:** This button takes the user to the page that shows the user how to use the application in a slide pictures.

4. **About Application button:** This button going to page that contains simple description about the application and it allows the user to share the application with her/his friends by the social media programs, also it contains the *Contact Us* button.
5. **Contact Us button:** This button will display e-mail applications that are installed in the phone of the user and he/she can choose any one of those applications to be connected with the developers.

Pages Description:

In this section we will describe different pages.

1- Start Camera page:

Figure 4 illustrates the first page after pressing the start button, the camera will be turned on directly and it will detect the face and the eyes of the user. Square will be drown around user face plus three dots will identify three important areas on the face [two are on the eyes and the third will be on the mouth].

So, the statue of the eyes will be observed and detected. Then, when the eyes are going to be closed in case of sleeping or drowsiness the application will start creating some noise to alarm the user and it will open the sub-page that contains "Stop Alert" button to stop the alarm.

Fig. 4 Start Camera page

3- Setting page:

The screen, presented in Figure 5, contains options divided into two groups [Screen Settings and Sound Settings], that allow the user to modify the default settings of the application. These options are:

1- Screen Settings:

- *Detecting Speed:* Allows the user to determine the number of seconds he/she wants to be alarmed after his/her eyes are detected in a sleepy mode. He/she can choose between 1 to 50 seconds, this option is very important because it makes the application more flexible and useable.
- *Use Back Camera:* From this option the user can choose if he/she wants to change the default camera - *which is the front camera*- to the back camera and vice versa by checking/un-checking it.
- *Display Information:* From this option the user will be able to choose if he/she wants the application to display some information on the screen of the camera page, these information is: how many number of faces are detected at the same time and the count of the eyes closed.

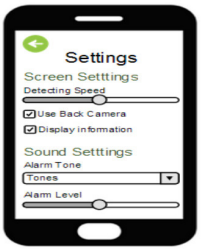

Fig 5 Screen Setting

2- *Sound Settings:*

- *Alarm Tone:* It allows the user to choose the alarm tone from window that contains a list of all device ringtones.
- *Alarm Level:* The user also can control the volume of the alarm sound.

3- *How to use page***:**

Series of screens presented in Figure 6, which are contain instructions that help the user to use the application efficiently.

1. *Back button*: Back to main page..
2. Image illustration containing instructions on how to use the application.

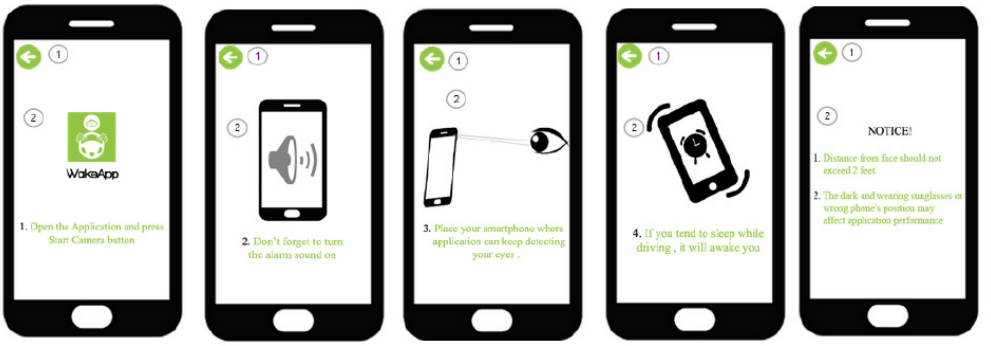

Fig. 6 How to Use page

3. Evaluation performance comparison

To assess the performance of the systems developers use the following criteria:

1. Accuracy: "is the ratio of an error to the range of possible output (full scale output) values"[35].
2. Reliability: "is the average amount of time between failures or the probability of a failure in a given period"[36].
3. Latency (Delays):" which is units that cause a time-shift in the input signals"[35].

Table 1 compares the WakeApp system with DFS system. Table 2 will show the success criteria results for the previous systems. It also shows the success criteria results of WakeApp system.

From the table 1 and 2 the developers conclude the following results:

1. The accuracy of WakeApp is better than three systems (DFM, SmartEye Pro 3.0, Yawning Detection For Monitoring Driver Fatigue).
2. The reliability of WakeApp is better than Yawning Detection for Monitoring Driver Fatigue system, RPI Prototype Fatigue Monitor.
3. WakeApp has the lowest latency comparing with other systems.
4. WakeApp is the only system run in Mobile platform.

Table 1. Comparison between WakeApp and FDS systems

Criteria	The percentage of accurate in WakeApp system	The percentage of accurate in FDS system
In morning	99%	99%
In night	30%	30%
When the head's driver moving frequently	99%	80%
When the driver wearing sunglass	20%	20%
When the driver wearing eyeglass	95%	90%

Table 2. Comparison between WakeApp system and other similar systems.

System name	**Accuracy**	**Reliability**	**Latency**	**Mobile Platform**
Attention Technology Driver Fatigue Monitor (DFM)	Medium	high	Medium	No
SmartEye Pro 3.0	Low	high	High	No
Johns Hopkins APL (DDDS)	High	high	Low	No
RPI Prototype Fatigue Monitor	High	medium	Low	No
Yawning Detection For Monitoring Driver Fatigue	Medium	Low	Medium	No
FDS	High	Medium	Low	No
WakeApp	High	High	Very Low	Yes

4. Conclusion

In this work, Information about the face and eyes status is obtained through various image processing algorithms. During the monitoring, the system is able to decide if the eyes are opened or closed. When the eyes have been closed for certain predefined time, an alarm sound is issued. In case of any type of error, the system is able to recover and properly localize the eyes.

The main components of the system consist of a hardware system which is the smartphone and software implementations for real-time video images of the driver face and eye tracking.

With the developer`s active system, the developers can achieve the followings: Real-time eye and face tracking, minimize external illumination interference and use the capabilities of Smartphone such as camera and small hardware which can fixed in the car which make driver feels comfortable.

References

1. Sangkyun P, Seonyoung L, Soojin K, Kyeongsoon C. Design of AdaBoost classifier circuit using Haar-like features for automobile applications. *International SoC Design Conference (ISOCC)* 17-18 Nov. 2011; 262 – 265.
2. Sun Z, et al. On-road Vehicles Detection: A Review. *IEEE Trans. On Pattern Analysis and Machine Intelligence* 2006; 28- 5: 649-711.
3. Dollar P, et al. Pedestrian Detection a Benchmark. *Proc. Of IEEE conference on Computer Vision and Pattern Recognition* Jun 2009; 304-311.
4. Rasolzadeh B, et al. Response Bining: Improved Weak Classifiers for Boosting. *IEEE Intelligent Vehicles Symposium* 2006; 344-349.
5. Viola P, et al. Detecting Pedestrians Using Patterns of Motion and Appearance. *IEEE International Conference on Computer Vision* Oct. 2003; 2: 734-741.
6. Zheng W, Liang L. Fast Car Detection Using Image Strip Features. *IEEE Conference on Computer Vision and Pattern Recognition* Jun. 2009; 2703-2710.
7. Sivaraman S, Trivedi M. Active Learning Based Robust Monocular Vehicle Detection for On-road Safety Systems. *IEEE Intelligent Vehicles Symposium* 2009; 399-404.

8. Konstantin P. Statistics Related to Drowsy Driver Crashes 7 Jul. 2010; www.americanindian.net.
9. Coetzer R, Hancke G. Driver fatigue detection: A survey. *IEEE AFRICON Conference* September 2009.
10. Lal S, Craig A, Boord P, Kirkup L, and Nguyen H. Development of an algorithm for an eeg-based driver fatigue countermeasure. *Journal of Safety Research* Feb. 2003; 1-34: 321–328.
11. Seong K, Haet L , Jung K, Jae B, Suk B, Suk K. ECG, EOG detection from helmet based system. *6th International Special Topic Conference on Information Technology Applications in Biomedicine ITAB* 2007; 191–193.
12. Boyraz P, Hansen J. Active accident avoidance case study: integrating drowsiness monitoring system with lateral control and speed regulation in passenger vehicles. *IEEE International Conference on Vehicular Electronics and Safety ICVES 2008*; 293–298
13. Sayed R, Eskandarian A. Unobtrusive drowsiness detection by neural network learning of driver steering. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* Jun 2001; 215-9: 969–975.
14. Eskandarian A and Mortazavi A. Evaluation of a smart algorithm for commercial vehicle driver drowsiness detection. *Intelligent Vehicles Symposium 2007 IEEE* Jun 2007; 553–559.
15. Breuer J. Attention assist: Don't fall asleep!. *Daimler, Tech. Rep* November 2008.
16. Wierwille W, Ellsworth L, Wreggit S, Fairbanks R, and Kim C. Research on vehicle-based driver status/performance monitoring: development, validation and refinement of algorithms for detection of driver drowsiness. *National highway traffic safety administration* 1994; 808- 247.
17. Dinges D, Mallis M, and Powell J. Evaluation of techniques for ocular measurement as an index of fatigue and the basis for alertness management. Department of transport safety April 1998; 808-762.
18. SmartEye (2010) Antisleep 2.0. Internet published whitepaper, [accessed November 2010].
19. Seeing Machines. Driver state sensor. *user manual 2.0*; 2007.
20. Tabrizi PR, Zoroofi RA. Drowsiness detection based on brightness and numeral features of eye image. *Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing IIH-MSP'09* 2009; 1310– 1313
21. Zhang Z, Zhang J S. Driver fatigue detection based intelligent vehicle control. *The 18th IEEE International Conference on Pattern Recognition* 2006.
22. Devi MS, Bajaj PR. Driver fatigue detection based on eye tracking. *First International Conference on Emerging Trends in Engineering and Technology* 2008; 649–652
23. Hong T, Qin H, Sun Q. An improved real time eye state identification system in driver drowsiness detection. *IEEE International Conference on Control and Automation* May 2007; 0: 1449–1453
24. Coetzer R C, Hancke G P. Eye detection for a real-time vehicle driver fatigue monitoring system. *2011 IEEE Intelligent Vehicles Symposium (IV)* ; 5-9 June 2011; 66 – 71
25. Ibrahim LF, Abulkhair M, AlShomrani AD, AL-Garni M, AL-Mutiry A, AL-Gamdi F, Kalenen R. Using Haar Classifiers to Detect Driver Fatigue and Provide Alerts. *Multimedia tools and applications*, Springer 2014; 71-3: 1857-1877. DOI: 10.1007/s11042-012-1308-5.
26. Wang X, Wang Z, Sun J, Zhang H. The correlation template matching algorithm based TD filter and ESO filter. *International Conference on Machine Learning and Cybernetics, Guangzhou, China*; 2005; 9:5361–5365
27. Adam Kerin. *Introducing the Snapdragon 810 and 808 Processors: The Ultimate Connected Computing Experience*. *www.qualcomm.com/media.* [Online] Apr. 2014.
28. Qualcomm Technologies. Snapdragon SDK for Android. *www.developer.qualcomm.com/mobile-development.* [Online]
29. Google. Mobile Courses, Android Development. *www.developers.google.com.* [Online] Sep. 2012.
30. D. Switkin, Senior Software Engineer, Google Inc. Android Application Development. *www.moss.csc.ncsu.edu.* [Online]
31. Eclipse Org. Eclipse. *www.help.eclipse.org.* [Online]
32. Android Developer. The Android SDK. *www.developer.android.com.* [Online]
33. Android NDK. *www.Developer.android.com.* [Online]
34. Wambler, scott. Agile Modeling. *www.agileModeling.com.* [Online]
35. The Complete Mining Fatigue Monitoring system, *www.ifatigue.com/iFatigue%20Fleet%20Monitoring%20System.pdf.* [Online]
36. "Accuracy." Business Dictionary. *www.businessdictionary.com.* [Online] Feb. 2011.