Driver monitoring algorithm for Advanced Driver Assistance Systems

Aleksandra Simić, Ognjen Kocić, Milan Z. Bjelica and Milena Milošević

***Abstract* —Fast expansion of Advanced Driver Assistance Systems (ADAS) market and applications has resulted in a high demand for various accompanying algorithms. In this paper we present an implementation of Driver monitoring algorithm. Main goal of the algorithm is to automatically asses if driver is tired and in that case, raise a proper alert. It is widely used as a standard component of rest recommendation systems. Our approach is based on combination of computer vision algorithms for face detection and eyes detection. Additionally, we have tested our implementation in controlled environment on a real ADAS platform board.**

***Keywords* — Advanced Driver Assistance Systems, driver monitoring, eyelid detection, face detection**

I. Introduction

MAIN goal of Advance Driver Assistance System (ADAS) is to provide safer environment for driver and to contribute to traffic safety in general. ADAS is constantly providing new solutions for non-trivial real world problems like automated freeway driving, which is addressed by lane detection algorithm [1], pedestrian detection [2], road signs recognition [3], automated parking [4], driver fatigue detection [5] and many more. Driver monitoring has a special place among these algorithms, as it is addressing the driver's fatigue problem, which is responsible for serious number of road accidents.

This paper provides detailed description of an algorithm design and implementation for the aforementioned driver monitoring. Driver's face and eyes position are used for detecting if driver is paying attention. Implemented algorithm is based on a Viola-Jones object detection framework (2001) [6]. The algorithm is suitable because it can be implemented in a real-time fashion and was specially motivated by frontal face detection. It will not found a face if the driver is turned to the side. Moreover, after the face was found, eye center detection algorithm is implemented using common face proportions (to locate the eyes region) and weighted matrix sum (to detect the iris).

Aleksandra Simić is with the Faculty of Mathematics, University of Belgrade, Studentski Trg 16, 11000 Belgrade, Serbia (phone: 381-65-3835662; e-mail: asimic@matf.bg.ac.rs).

Ognjen Kocić is with the Faculty of Mathematics, University of Belgrade, Studentski Trg 16, 11000 Belgrade, Serbia (phone: 381-65-2235055; e-mail: ognjen@matf.bg.ac.rs).

Dr. Milan Bjelica is with the Department of Computer Engineering and Communications, Faculty of Technical Sciences, 21000 Novi Sad, Serbia (email: milan.bjelica@rt-rk.com)

Milena Miloševic is with RT-RK Institute for Computer Based Systems, Novi Sad, Serbia (email: milena.milosevic@rt-rk.com)

II. Related work

Detection of driver's fatigue level represents one of the most researched topics when it comes to ADAS applications [5]. In general, there are few different approaches. First one combines readings from multiple sensors that are attached to driver. Conclusions are drawn using a fusion of collected data. For example, researchers have used heart rate measurements for this purpose [7].

Some may found usage of sensors too intrusive for real world use case. As often used alternative, driver is recorded with some camera equipment and computer vision algorithms are applied in order to detect where the driver is looking and if he/she pays attention to the road. Solution described in this paper can be classified into this group.

Additionally, there are methods that consider driving patterns as a main indicator of driver tiredness [8].

Finally, it might be important to note one major difference between these approaches. When detecting long term drowsiness it is more suitable to use sensor or pattern methods. However, if we are trying to prevent accident that might occur as a result of imminent distraction, computer vision based methods are proven to be more beneficial.

III. Algorithm implementation

The implemented algorithm can be divided into few phases. These phases and their connections are shown in Fig. 1.

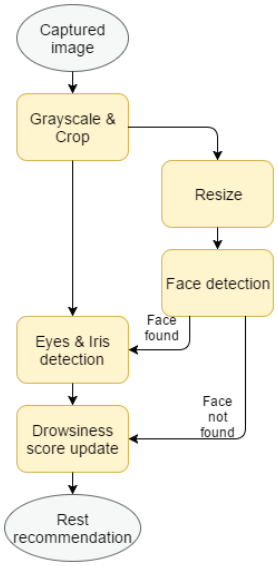


Fig. 1. High level algorithm diagram

978-1-5090-4086-5/16/$31.00 ©2016 IEEE

Phases “Grayscale & Crop” and “Resize” can be grouped into one phase – named “Input image preprocessing”. Those steps are shown separated in Fig.1. because it is important to see that resized image is input only for face detection (reasons will be given later).

A. *Input Image preprocessing*

Firstly, BGRA frame that is produced by camera is converted to grayscale. The reason for doing this is because a grayscale image is a requirement for Viola Jones algorithm for face detection (described in phase B) and it is also well suitable for iris detection.

Besides that, we took advantage of the fact that we need to detect only one face, which by assumption occupies large area in center of input frame. Since face should be centered, we cropped 35% of frame’s width, because we do not want to preprocess and later analyze parts of image where there is no chance for face or eyes to be found. As the result of these simple preprocessing steps, image that is suitable for eyes and iris detection was successfully formed.

It is already mentioned that we search for a very large face. Having that in mind, we concluded that input image can be downscaled in purpose of getting face detection algorithm that works faster and achieves the same accuracy. This cannot be applied for eyes and iris detection because iris occupies only small portion of image and working with small image would damage detection quality.

B. *Face detection*

VJ framework can be divided in two phases. First phase is a preparation phase and it consists from *Haar feature selection* and *AdaBoost* training, while second is a face detection phase. Haar features can be thought of as convenient rectangle shaped filters (shown on Fig. below).

Fig. 2. Haar features

It was observed by Viola and Jones that the grayscale image of human face has some special characteristics. For example, eye region of image is darker than the rest. Furthermore, vertical nose region is distinctively brighter. Filters shown in Fig. 2. are applied to a face image and sums of pixels in brighter regions of rectangles are subtracted from the dark ones. Given value is then compared to some threshold which is used to determine if current region is the region we are interested in. Some observations to be made are that there are 2 types of two-rectangle features, 2 types of three-rectangle features and a single type of four-rectangle features (considering symmetry in our deduction process). Smallest rectangle for first feature left to right from Fig. 2. is 1x2 pixels and because of that width must be an even number, which also stands for height of the second feature. Similarly, for third feature left to right, width must be divisible by 3. When listed limitations for described features are included and 24x24 pixels base filter (it is standard detection window size) is used, there is about 160K features which is too much for a real time application to examine. *AdaBoost* training is used to select the best features that will provide the most information about scanned part of image. As a result of the first phase, complete set of features that will be used for face detection is determined. First phase can be done only one time, whereas we can run face detection using that data many times. In general, this is a good approach since there are no performance penalties in actual face detection that are related to preparation phase of the algorithm. After features are selected, they shall be applied to the image.

In general, on a single image one can expect human faces in many sizes. For example, one face may fit in a 50x50 pixels bounding rectangle, while other might be better placed in 30x30 pixels bounding rectangle. If the standard detection window of 24x24 pixels is used, it is not difficult to see that some faces might not fit into it. Because of that, feature detection is run on an image pyramid with intuition that at some level in pyramid both faces will fit into 24x24 rectangle (of course not at the same level). Image pyramid consists of sequence of images down sized by some factor. Pseudo-code is given as follows:

```
for each image in image pyramid do
  compute integral and squared integral image
  for each position of sliding detection window do
    for each stage in cascade classifier do
      initialize score for this stage
      for each feature in stage do
        apply filter to detection rectangle region of image
        update stage score
      done
      if stage score < threshold do
        reject rectangle region
        break two loops
      done
    done
    accept rectangle of interest
  done
done
```

Fig. 3. Face detection pseudo-code

Output of this algorithm is an array of rectangles that represent founded faces. It is possible that one face is founded in more than one phase (more than one image in image pyramid). Because of that, we need to group all rectangles that have at least 65% overlapping, which was determined empirically.

Finally, even after this step, more than one face can be found (for example someone can watch the road over the driver’s shoulder). In that case, the rectangle that is closest to the image center is chosen.

Output of face detection phase is one rectangle that represents driver’s face, if the face was found or invalid rectangle otherwise.

C. *Eyes and iris detection*

First step of eyes center detection algorithm is to crop input image to already detected face rectangle. Standard biometric proportions are then applied to further reduce processing to rectangles that contain only left and right eye. These proportions are shown in Fig. 4. Blue rectangle represents face whereas green rectangles represent eye regions.

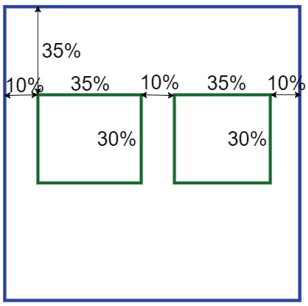


Fig. 4. Biometric proportions

These regions are input to iris detection phase. We used a fact that iris is darker then sclera around it and we developed a custom algorithm that uses weighted matrix sums. Matrix used can be thought of as some kind of convolution kernel. It is populated with floating point values that are concentrically reduced by factor k from inside out. An example is shown in Fig. 5.

1	1	1	1	1	1	1
1	1.25	1.25	1.25	1.25	1.25	1
1	1.25	1.56	1.56	1.56	1.25	1
1	1.25	1.56	1.95	1.56	1.25	1
1	1.25	1.56	1.56	1.56	1.25	1
1	1.25	1.25	1.25	1.25	1.25	1
1	1	1	1	1	1	1

Fig. 5 Weight matrix, k=1.25, dimension = 7

Matrix dimension must be odd number and it depends on eye region size. Coefficient k, used to calculate matrix values, is empirically determined value. Before applying this matrix on surroundings of each pixel, we have done one more matrix modification - normalization. This means that we have divided each element of matrix with sum of initial matrix that is not normalized (Fig. 6). Total sum of all coefficients in normalized matrix equals 1.

0.0171	0.0171	0.0171	0.0171	0.0171	0.0171	0.0171
0.0171	0.0214	0.0214	0.0214	0.0214	0.0214	0.0171
0.0171	0.0214	0.0267	0.0267	0.0267	0.0214	0.0171
0.0171	0.0214	0.0267	0.0334	0.0267	0.0214	0.0171
0.0171	0.0214	0.0267	0.0267	0.0267	0.0214	0.0171
0.0171	0.0214	0.0214	0.0214	0.0214	0.0214	0.0171
0.0171	0.0171	0.0171	0.0171	0.0171	0.0171	0.0171

Fig. 6. Normalized weight matrix (from Fig.5.)

For each considered pixel, weighted sum of its ambience is calculated. When whole image has been processed, iris should contain pixel for which the sum was minimal. Intuition behind algorithm is that we want to punish more bright pixels near center of applied filter, then pixels on the edge of filter. Moreover, these punishments should happen gradually because of fact that neighbor pixels are also very important for correctness of algorithm.

When center of eye is determined, next task is to deduce if eye is opened or closed. Again, our approach was associated with weight matrices. Let's consider grayscale image of closed eye for a moment. In the center of this image we have a strong line of dark values that correspond to eyelashes. However, when eye is opened, such line cannot be established (Fig. 7). With this distinction to guide us, we constructed another weight matrix with values vertically progressing from center of matrix to the top and bottom. It should be clear that this matrix is applied on top of eye center pixel which lies in middle (Fig. 8). Additionally, matrix width is reduced by 50% (25% on each side) in order to eliminate border noise (Fig. 8). As well as previously presented concentric matrix, this matrix was also normalized.


Fig. 7. (a) Eye close (b) Eye open

Finally, a criterion if eye is open was formed by comparing quotient of average pixel value in region of interest and weighted matrix sum against the threshold. This threshold was established after series of experiments.

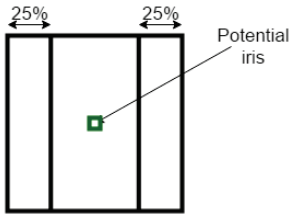


Fig. 8. Second weighted matrix position

D. *Drowsiness score calculation*

Final phase of described algorithm is calculation of drowsiness score. Drowsiness score is value in range between 0 and 10. This range is divided into three intervals:

- 0-3 low
- 3-7 medium
- 7-10 high

New score is calculated based on previous score *PrevScore* and output of face and eye detect phases for current frame.

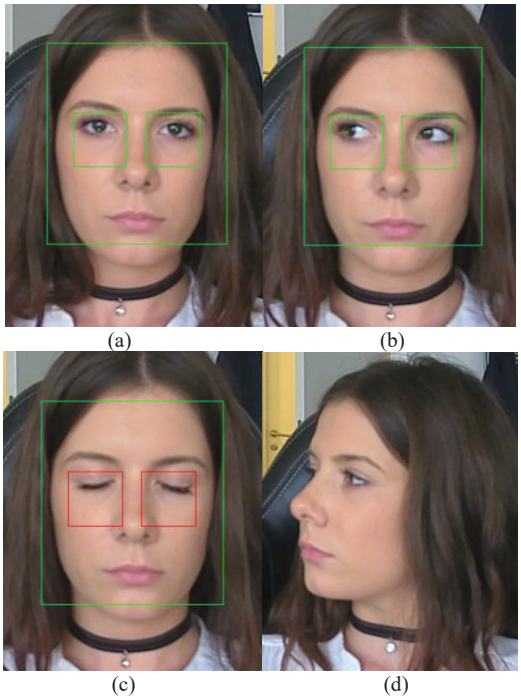
Difference between intervals is:

- If *PrevScore* is in low interval, then score twice faster decreases than it rises
- If *PrevScore* is in medium interval, then score decreases and rises at the same rate
- If *PrevScore* is in high interval, then score twice faster rises, than it decreases

When score is in low interval, we can tell that driver is not distracted at all. In case of medium and high score, driver is drowsy and system should eventually alert the driver.

IV. EVALUATION

Described algorithm gives the best results if camera and light source are placed in front of the driver. Algorithm output used for verification of detected regions is shown in Fig. 9.



(a) (b)

(c) (d)

Fig. 9. Algorithm output examples

Image (a) shows a case when a face is found, eyes are open and driver is looking ahead. In this case, drowsiness score is decreasing depending on its current value.

Image (b) shows that our algorithm also works well in case when driver does not look ahead. It is natural for a driver to look left or right in order to check side mirrors or to address their attention to some other road event. It should be clear that drowsiness score should not rise in this case.

Image (c) shows driver with closed eyes and red rectangle shows that algorithm detected that eyes are not opened. This may be a serious problem if driver is sleeping. However, it may be the case that camera took the frame when they were blinking. No matter what happened drowsiness score will rise.

Image (d) shows case when driver turns head to some side. In that case face detect algorithm will not find driver's face and that is sign that driver is not paying attention to the road. Not all rotations are sanctioned, whereas in this particular image head is completely turned right. Of course, score that measures distraction will increase.

The algorithm was tested on real ADAS platform board. It works as real-time algorithm (about 20 frames per second) with high level of accuracy. The main reason for good driver monitoring performance is minimal impact of wrongly classified frames due to high framerate and the manner of drowsiness score calculation. The algorithm was implemented only for demo purposes and detailed statistical evaluation will be part of future work.

The algorithm was not tested at night because it requires infrared cameras which were not available at the time.

V. Conclusion

In this paper we described one implementation of driver monitoring algorithm, which is in early stage of development. Implemented algorithm is not state of the art, but we have achieved some solid results. There is more than one way we can improve this algorithm:

- Using infrared cameras would give better input images – we would have images with the same illumination level, so light would not be a factor.
- Using advanced and more accurate methods for iris detection which are based on Bayesian classification of extracted features.
- Integration with data coming from the car itself, like current speed, steering pattern etc.
- Creating more sophisticated drowsiness score calculation scheme and testing it in actual vehicle.

First two ideas do not require usage of additional equipment. Consequently, working on these ideas will be our first step. Upon successful completion of these steps, further refinements can be implemented during the in car testing.

Acknowledgment

This work was partially supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia, under grant number: TR32041.

References

[1] M. Aly, "Real time Detection of Lane Markers in Urban Streets" in *IEEE Intelligent Vehicles Symposium, Eindhoven, The Netherlands, June 2008*

[2] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian Detection: An Evaluation of the State of the Art" in *IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume: 34, Issue: 4, April 2012)*

[3] S. Hossain, Z. Hyder, "Traffic Road Sign Detection and Recognition for Automotive Vehicles" in *International Journal of Computer Applications,* Volume 120 - Number 24, 2015

[4] H. Al-Absi, J. Devaraj, P. Sebastian, V. Yap, "Vision-based automated parking system" in *10th International Conference on Information Sciences Signal Processing and their Applications (ISSPA),* 2010

[5] Hang-Bong Kang, "Various Approaches for Driver and Driving Behavior Monitoring: A Review" in *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference*, 2-8 Dec. 2013.

[6] P. Viola, M. Jones, "Rapid object detection using a boosted cascade of simple features" in *Computer Vision and Pattern Recognition, 2001. CVPR 2001.*

[7] J. Vicente, P. Laguna, A. Bartra, R. Bailón, "Drowsiness detection using heart rate variability" in *Medical & Biological Engineering & Computing,* June 2016, Volume 54, Issue 6, pp 927–937

[8] J. Kim, S. Kim, H. Jung, B. Lee, E. Chung, "Driver's Drowsiness Warning System Based on Analyzing Driving Patterns and Facial Images" in *23rd International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2013