Multimodal Focus Attention and Stress Detection and feedback in an Augmented Driver Simulator

Alexandre Benoit, Laurent Bonnaud, Alice Caplier, Phillipe Ngo, Lionel Lawson, Daniela G. Trevisan, Vjekoslav Levacic, Céline Mancas, Guillaume Chanel
Laboratoire des Images et des Signaux, INPG Grenoble, France
www.similar.cc

Abstract. This paper presents a driver simulator, which takes into account information about the user's state of mind (level of attention, fatigue state, stress state). The user's state of mind analysis is based on video data and biological signals. Facial movements such as eyes blinking, yawning, head rotations... are detected on video data: they are used in order to evaluate the fatigue and attention level of the driver. The user's electrocardiogram and galvanic skin response are recorded and analyzed in order to evaluate the stress level of the driver. A driver simulator software is modified so that the system is able to appropriately react to these critical situations of fatigue and stress: some audio and visual messages are sent to the driver, wheel vibrations are generated and the driver is supposed to react to the alert messages. A multi threaded system is proposed to support multi messages sent by different modalities. Strategies for data fusion and fission are also provided.

1. Introduction

The main goal of this project is to use multimodal signal and video processing to provide an augmented user's interface for driving. In ths aper, we are focusing on passive modalities. The term augmented here can be understood as an attentive interface supporting the user interaction. So far at most basic level, the system should contain at least five components: (1) sensors for determining the user's state of mind; (2) modules for features or data extraction; (3) a fusion process to evaluate incoming sensor information; (4) an adaptive user interface based on the results of step 3 and (5) an underlying computational architecture to integrate these components.
In this paper, we address the following issues:

Please use the following format when citing this chapter:
Benoit, Alexandre, Bonnaud, Laurent, Caplier, Alice, Ngo, Phillipe, Lawson, Lionel, Trevisan, Daniela, Levacic, Vjekoslav, Mancas, Celine, Chanel, Guillaume, 2006, in IFIP International Federation for Information Processing, Volume 204, Artificial Intelligence Applications and Innovations, eds. Maglogiannis, I., Karpouzis, K., Bramer, M., (Boston: Springer), pp. 337–344

- Which driver simulator to use?
- How to characterize a user's state of fatigue or stress?
- Which biological signals to take into account?
- What kind of alarms to send to the user?
- How to integrate all these pieces – data fusion and fission mechanism?
- Which software architecture is more appropriate to support such kind of integration?

A software architecture supporting real time processing is the first requirement of the project because the system has to be interactive. A distributed approach supporting multi threaded server can address such needs.

We are focusing on stress and fatigue detection. The detection is based on video information and/or on biological information. From video data we extract relevant information to detect fatigue state while the biological signals provide data for stress detection. The following step is the definition of the alarms to provide to the user. Textual and vocal messages and force feedback are considered to alert the user.

The rest of the paper is organized as follows: section 2 present the global architecture of the demonstrator, section 2.1 describes how we detect driver's hypo-vigilance states by the analysis of video data, section 3 presents how to detect driver's stress states by the analysis of some biological signal, sections 4 describes the data fusion and fission strategies and section 5 gives details about the demonstrator implementation.

2. Conceptual architecture

The diagram of **Fig 1** presents the conceptual architecture of our attentive driver simulator. We propose a distributed approach to integrate our components. On one PC under Linux we have integrated all video data based detection and analysis as well as the fusion and fission components. Another PC under Windows is used to run the driver simulator and a third PC is used for biological signals acquisition and analysis. Communication between all the PCs is done by exchanging XML messages. For that the Dialog Controller included in the driver Simulator software should be able to receive multi messages (i.e. from biological signals station and from video based station). A multi threaded server approach is developed and included in the driver simulator.

2.1. Hypo-vigilance detection based on video data

The state of hypo-vigilance (either related to fatigue or inattention) is detected by the analysis of video data. The required sensor is a camera facing the driver. Three indices are considered as hypo-vigilance signs: yawning, head rotations and eyes closing for more than 1s.

Face detection: In this paper, we are not focusing on face localization. The face detector should be robust (no error in face localization) and should work in real time. We chose to use the free toolbox MPT [5]. This face detector extracts a square-bounding box around each face in the processed image. The MPT face detector

works nearly at 30 frames per second for pictures of size (320x200 pixels), which is not the case of other face detectors such as OpenCV [13] for example.

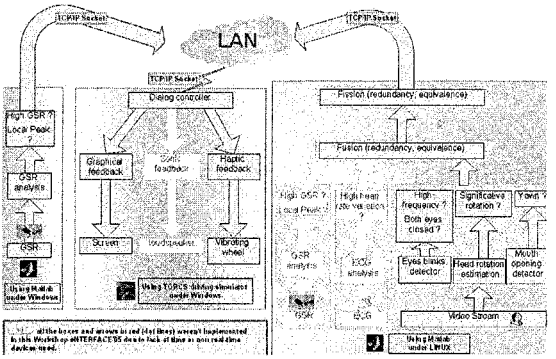

Fig 1: Overview of the system architecture

Head motion analysis: Once a bounding box around the driver's face has been detected, head motion such as head rotations, eyes closing and yawning are detected by using an algorithm working in a way close to the human visual system. In a first step, a filter coming from the modeling of the human retina is applied. This filter enhances moving contours and cancel static ones. In a second step, the FFT of the filtered image is computed in the log polar domain as a modeling of the primary visual cortex. The detail of the proposed method is described in [1].

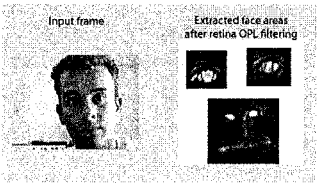

Fig 2: OPL filtering results for eyes and mouth.

As a result of retinal filterings, noise and luminance variations are attenuated and moving contours are enhanced.

The modeling of the primary visual cortex consists in a frequency analysis of the spectrum of the retina filters outputs in each region of interest of the face: global head, eyes and mouth (see in sequel for the description of eyes and mouth region of interest extraction).

In order to estimate the rigid head rotations [3], the proposed method analyses the spectrum of the retina filters output in the log polar domain. It detects head motion events and is able to extract its orientation.

For the detection of yawning or eyes closing, three identical processes are done independently [4]. On each region of interest (each eye and the mouth), a spectrum analysis is also carried out: we are looking for vertical motion related to eyes closing or to yawning.

Eyes and mouth detection: The mouth can be easily extracted in the lower half of the detected bounding box of the face.

Concerning the eyes, the spectrum analysis in the region of interest is accurate only if each eye is correctly localized. Indeed around the eyes, several vertical or horizontal contours can generate false detection (hair boundary for example). The MPT toolbox proposes an eye detector but it requires too much computing time (frame rate of 22 fps) so that it has been discarded. We use another solution: eye region is supposed to be the area in which there is the most energized contours. Assuming that the eyes are localized in the 2 upper quarters of the detected face, we use the retina output. The retina output gives the contours in these areas and due to the fact that the eye region (containing iris and eyelid) is the only area in which there are horizontal and vertical contours, the eye detection can be achieved easily. We use two oriented low pass filters: one horizontal low pass filter and a vertical low pass filter and we multiply their response. The maximum answer is obtained in the area with the most horizontal and vertical contours: the eye regions. The eye area detection is performed at 30 frames per second.

Hypo-vigilance alarms generation: We detect eyes closing and then generate an alarm.We detect mouth yawning: when a yawn occurs, the mouth is wide open. This generates a very high-energy increase on the log-polar spectrum that can be easily extracted. The global head motion events are detected with the global head spectrum analysis. We only extract the fact that a head motion has occurred.

Fusion strategy: After the video analysis, Boolean information about yawning or not, about eyes closing or not and about head moving or not are available. A simple fusion strategy based on the three index is proposed:

```
if head motion is detected
  send an alarm to the user
  hypo-vigilance value=100
else
  if both eyes are closed during 1s
    send an alarm to the user
    hypo-vigilance value = 50
  if the driver is yawning
    send an alarm to the user
    hypo-vigilance value = 50+hypovigilence value
end
```

The variable hypo-vigilance associated to each index is set to 50 or 100. The highest the value, the highest the hypo-vigilance.

Note that in this very simple fusion strategy, information about head motion kind of rotation is not taken into account. A more sophisticated fusion strategy has been tested and is described in section 4.

3. Stress detection based on Biological signals analysis

The application consists of the following three main parts: a) initialization part, b) haptic loop and c) visual loop. The initialization part establishes connection with the hardware devices, reads the scene, initializes the collision detection algorithm and starts the haptic and visual loops. The haptic loop updates the scene using data from the devices, checks for collisions between the hand and scene objects, sets the new

position of the hand and objects, triggers feedback forces and enables sound playback. The visual loop reads the current position of scene objects and renders the scene.

Biological signals are used in order to detect stress situation. ECG (Electrocardiogram) and GSR (Galvanic Skin Response) are announced by literature as very promising to detect driver stress in real situations [11, 12]. In a stressful time, the GSR signal and the heart rate signal (extracted from the ECG) are supposed to increase. Two different experiments have been considered; they aim at detecting either driver stress over a long time period or punctual driver stress.

The main drawback of the data acquisition system is that for the moment, on line analysis is not possible. For that reason, the study on biological signal for stress detection has not yet been implemented in the current demonstrator.

4 Fusion strategy

In this section, we describe and test a data fusion based on Bayesian Network. It is used for the purpose of hypo-vigilance detection but it also represents a global fusion method for the integration of additional information in the detection process. Note that this fusion process is not integrated in the final demonstrator for the moment due to the lack of significant data.

Human fatigue generation is a very complicated process. First, fatigue is not observable and it can only be inferred from the available information. In fact, fatigue can be regarded as the result of many contextual variables such as working environments, health and sleep history. Also, it is the cause of many symptoms, e.g. the visual cues, such as irregular eyelid movements, yawning and frequent head tilts. Second, human's visual characteristics vary significantly with age, height, health and shape of face. To effectively monitor fatigue, a system that integrates evidences from multiple sources into one representative format is needed. Naturally, a Bayesian Networks (BN) model is a good option to deal with such an issue.

A BN provides a mechanism for graphical representation of uncertain knowledge and for inferring high-level activities from the observed data. Specifically, a BN consists of nodes and arcs connected together forming a directed acyclic graph. Each node can be viewed as a domain variable that can take a set of discrete values or a continuous value. An arc represents a probabilistic dependency between the parent node and the child node.

Some contextual information such as temperature, time of day, sleep history, etc can be used to build a prior probability for the fatigue node. For that we use the parameters proposed in [7]. For the face data fusion we have considered a very preliminary version where the network evidences change when: eyes closed more than 1 sec; yawning occurs; down head motion are detected simultaneously or not. As result we got the level of fatigue, which is sent to the data fission component.

Data fission duty is to collect the data from data fusion and to generate an alert XML message that is sent to the driver simulator. Data fission function is called at the rate the driver's state detection is progressing. Generated messages are in XML format. We decided for XML because it is extendable and messages are sent only when the driver's state changes. Driver's state may be defined by a fatigue value (either coming from the Bayesian Network result or from the simple fusion process)

that is an output variable of data fusion. Table 1 and Table 2 present the fusion strategy for the simple method and for the Bayesian network based method respectively.

Data fission only creates the message if the driver's state has changed and is different than the previous one. If the user's state is the same as in previous call, data fission generates 'NOT_CHANGED' message. In that way the XML message does not need to be sent to the driver simulator after each call of the data fission function.

Table 1: Fission strategy with the simple fusion process

Fatigue range	50	50	100
Message	Open the eyes	Yawning: be careful	Stop moving the head
Shaking power	'100'	'100'	'100'

Table 2: fission strategy with the BN based method

Fatigue range	[0,33]	[33,66]	[66,100]
Message	''	'Tired'	'Asleep'
Message color	''	'Green'	'Red'
Shaking power	'0'	'0'	'100'

Once the alert message has been sent, the driver is supposed to acknowledge to the system that the message has been understood. For example, in the case of the simple fusion process, each time an alert is detected, wheel vibrations are triggered. The driver has to stop these vibrations by pushing a button. The reaction time is also recorded, this time being correlated with the hypo-vigilance or fatigue user's state.

5 Demonstrator

5.1 Overview of the global system

The developed demonstrator is made of 2 PCs: one under Windows for the driver simulator and one under Linux for hypo-vigilance states detection, 1 SONY digital camera, 1 LOGITECH force feed back wheel, 1 projection screen, 1 video-projector and 2 loudspeakers.

5.2 Driver Simulator

Around ten driver simulators have been studied. The choice of the driver simulator has to take into account some features such as: open source software, "First person view": (i.e. cockpit view with wheel) and dashboard, source code easy to modify and possible use of a vibration feedback wheel.

The chosen driver simulator is TORCS [9] because it is a well architectured GPL program with well structured source code and a well designed user interface.

This simulator is working under Linux and windows platforms. The main sources are written in C++ with the OpenGL library. The graphics quality of the simulator is correct and it has a first person view. Fig.3 presents an illustration of TORCS simulator.

We integrate an interaction from the Data Analysis Kernel to our driving Simulator.

The main work consisted in

- Allowing a Text Message to be displayed within the game graphical interface.
- Creating a multi-threaded Server within the application whose purpose is accepting different clients connexions.
- Integrating a force Feedback wheel in order to warn the user with another modality than the visual one.
- Allowing the user to make a feedback on the message displayed by stopping it.
- Parsing XML messages from the multimodal analysis of the driver. Indeed, it is possible to change the color, the string of the sent message and the feedback power.

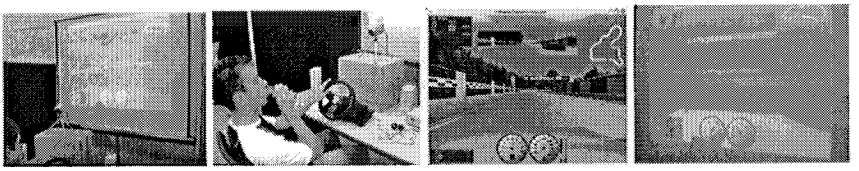

Fig 3: Global views of the demonstrator, Torcs driver simulator illustration and alert message example

5.3 Implementation of hypo-vigilance detection

Due to the fact that ECG and GSR signals cannot be processed on line with the data acquisition station we used, the detection of stress state has not been implemented in real time.

5.4 Alert message generation

The Display Changes: Three different visual or audio messages depending on the index of hypo-vigilance can be displayed on the first view of the driver simulator (see Table 1 for the considered visual messages and Fig. 3 for an example of message incrustation during the game). In order to show the message, we need to change all the graphical classes within the source code. The communication between the main program and all the libraries is created by using some global variables and also by creating new links between several libraries.

Force Feedback implementation: Force Feedback is used to make the wheel shaking when hypo-vigilance is detected. Shaking power is defined by XML message of data fission. Shaking becomes stronger and gradually reaches its maximum value. Force Feedback uses the DirectInput library, a part of the Microsoft DirectX SDK. The library is based on "The Force Feedback Direct Input Library (DIL)" made by Bryan Warren and Alex Koch. This library can be loaded at [8]. Here, this library has been altered from functions to class. In class we can set the time period that shaking needs to reach the maximum vibration time, vibration activation threshold and to modify the shaking of the wheel dynamically. We also use the class to check whether the button is pressed, and if pressed stops the wheel shaking.

Message parsing and controlling the input devices of driver simulator: After the XML message arrives through the socket connection it is parsed. We use Microsoft

XML parser to parse the message. You can download the MSXML parser from Microsoft web site. After the parsing, controller class activates the screen display message or starts the wheel shaking.

The Server Side: We choose to implement a Server side for the interaction between the multimodal devices and the user. The network protocol used is TCP/IP. We implement this socket by using threads. Those threads access global variables under mutual exclusion. We use a "GPL" library called Openthreads for this implementation.

6. Future Works and Conclusion

We have developed an augmented driver simulator based on video analysis for driver's attention controlling. First promising studies about physiological data have to be improved and integrated in the global system. This will induce the development of an appropriate data fusion method in order to control both the driver's attention level and the driver's stress.

Once the driver has been alerted, it will be necessary to perform some specific tests in order to control that driver's stress or fatigue has actually decreased.

For the moment, the global system is running almost 10 frames per second. It will be necessary to optimize video data analysis algorithms in order to speed up the frame rate.

References

1. Benoit A., Caplier A. –Motion Estimator Inspired from Biological Model for Head Motion Interpretation – WIAMIS05, Montreux, Suisse, avril 2005.
2. Torralba A. B., Herault J. "An efficient neuromorphic analog network for motion estimation." IEEE Transactions on Circuits and Systems-I: Special Issue on Bio-Inspired Processors and CNNs for Vision. Vol 46, No. 2, February 1999.
3. Benoit A. , Caplier A. "Head nods analysis : interpretation of non verbal communication gestures " IEEE ICIP, Genova, Italy, 2005.
4. Benoit A. , Caplier A. "Hypovigilence Analysis: Open or Closed Eye or Mouth ? Blinking or Yawning Frequency ?" IEEE AVSS, Como, Italy, 2005.
5. Machine Perception Toolbox (MPT) http://mplab.ucsd.edu/grants/project1/free-software/MPTWebSite/API/.
6. Bayes Net Toolbox for MatLab http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html
7. Qiang Ji, Zhiwei Zhu and Peilin Lan, Real-Time Nonintrusive Monitoring and Prediction of Driver Fatigue, IEEE Transactions on Vehicular Technology, Vol. 53, No. 4, July, p1052-1068, 2004.
8. Force Feedback Direct Input Library http://courses.washington.edu/css450/Fall2003/web_contents/direct_input_lib/DirectInput.html
9. TORCS Driver Simulator: http://torcs.sourceforge.net/
10. Similar Network of Excellence: www.similar.cc
11. S. K. Lal, A. Craig, "Driver fatigue: electroencephalography and psychological assessment", Psychophysiology, 39, 3, May 2002.
12. J.Healey, J.Seger, R.Picard, "Quantifying driver stress: developing a system for collecting and processing bio-metric signals in natural situation", MIT technical report n°483.
13. OpenCV : www.intel.com/technology/computing/opencv