A framework for context-aware driver status assessment systems

by

Céline Craye

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2013

Abstract

The automotive industry is actively supporting research and innovation to meet manufacturers' requirements related to safety issues, performance and environment. The Green ITS project is among the efforts in that regard.

Safety is a major customer and manufacturer concern. Therefore, much effort have been directed to developing cutting-edge technologies able to assess driver status in term of alertness and suitability. In that regard, we aim to create with this thesis a framework for a context-aware driver status assessment system. Context-aware means that the machine uses background information about the driver and environmental conditions to better ascertain and understand driver status. The system also relies on multiple sensors, mainly video and audio. Using context and multi-sensor data, we need to perform multi-modal analysis and data fusion in order to infer as much knowledge as possible about the driver. Last, the project is to be continued by other students, so the system should be modular and well-documented.

With this in mind, a driving simulator integrating multiple sensors was built. This simulator is a starting point for experimentation related to driver status assessment, and a prototype of software for real-time driver status assessment is integrated to the platform.

To make the system context-aware, we designed a driver identification module based on audio-visual data fusion. Thus, at the beginning of driving sessions, the users are identified and background knowledge about them is loaded to better understand and analyze their behavior.

A driver status assessment system was then constructed based on two different modules. The first one is for driver fatigue detection, based on an infrared camera. Fatigue is inferred via percentage of eye closure, which is the best indicator of fatigue for vision systems. The second one is a driver distraction recognition system, based on a Kinect sensor. Using body, head, and facial expressions, a fusion strategy is employed to deduce the type of distraction a driver is subject to. Of course, fatigue and distraction are only a fraction of all possible drivers' states, but these two aspects have been studied here primarily because of their dramatic impact on traffic safety.

Through experimental results, we show that our system is efficient for driver identification and driver inattention detection tasks. Nevertheless, it is also very modular and could be further complemented by driver status analysis, context or additional sensor acquisition.

Acknowledgments

First, I would like to express my deepest gratitude to Dr. Fakhri Karray who was my supervisor during my master's program. He provided me help and guidance throughout my studies. I also thank him for trusting me and helping me join an industrial partner.

I would also like to thank Professors Michailovich, Poupart, and Kamel for their precious help and advice both during and after their courses. The knowledge and skills I gained during my master's would not have been the same without them.

Of course, I would like to thank my committee members for their comments and reviews.

Many thanks to all my friends from PAMI and University of Waterloo, who have made my life happier during these years. A special thanks to Abdullah Rashwan who contributed and helped me greatly in my project.

Thanks to all my Vestec co-workers, especially Kacem Abida. My time at Vestec was extremely valuable and a very pleasant professional experience.

I also need to acknowledge Drs Friedel and Rosenberg, who made possible my enrollment in the master program at the University of Waterloo. I would not be here without their support.

Last but not least, I need to thank my family and my boyfriend Simon Ferré for their wisdom and their intellectual and emotional support throughout my studies.

Dedication

I dedicate this master's thesis to my parents, my brother, my sister, and Simon.

Table of Contents

List of Tables

List of Figures

List of Acronyms

AdaBoost	ADAptive BOOSTing	xii, xiv, 52, 55, 72, 75, 77, 78, 86, 88, 89, 93
ADAS	Advanced driver assistance system	12
AU	Animation Units	73, 84
CMOS	Complementary metal oxide semi-conductor	29
ECG	ElectroCardioGram	18, 22
EDA	Electrodermal Activity	18
EEG	ElectroEncephaloGram	18, 21, 22
EMG	ElectroMyoGram	18
EOG	Electro OcculoGram	18
GMM	Gaussian Mixture Model	10, 39, 42, 52
GPS	Global positioning system	6, 12, 13, 23, 94
Green ITS	Green Intelligent Transportation System	4
GUI	Graphical User Interface	31

HMM	Hidden Markov Model	xii, 85, 86, 88, 89
IR	InfraRed	18, 33
KPCA	Kernel Principal Component Analysis	22
LED	Light-Emitting Diode	19, 29
LIDAR	LIght Detection And Ranging	12
MFCC	Mel-Frequency Cepstral Coefficients	38, 39, 46, 52
MLP	Multi-Layer Perceptron	38
MMSE	Minimum Mean Square Error	47
NHTSA	National Highway Traffic Safety Administration	5, 16, 54
OEM	Original Equipment Manufacturer	4
PCA	Principal Component Analysis	41, 42, 55, 64
PERCLOS	PERcentage of eye CLOSure	20–22, 32, 33, 54, 55, 65, 66, 70, 92
PPG	PhotoPlethysmogram	22
RADAR	RAdio Detection And Ranging	12, 95
RBF	Radial Basis Function	61, 65
RGBD	Red Green Blue Depth	73
SDK	Software Development Kit	19, 30, 33, 52, 73, 79, 82

SNR	Signal to Noise Ratio	44, 50
SVM	Support Vector Machine	22, 38, 45, 61, 65, 66, 70, 82

Chapter 1

Introduction

1.1 Motivations

The automotive industry is actively supporting research and innovation to meet manufacturers's requirements related to energy optimization, safety issues, performance, and environmental protection. Cutting-edge technologies are designed to cope with intelligent transportation systems in safety. To enable intra- and inter-vehicle communications, critical information, such as driver status (in term of alertness and suitability), road obstacles, or lane departure, has to be assessed. This master's thesis is part of the Green ITS project, related to the aforementioned issues.

Safety is a major customer concern and has dramatically penetrated every aspect of automobile development. Introduction of sensor applications and innovations in microelectronics will drive demand for automotive sensors, and it is estimated that the North American light vehicle OEM market will increase 11.7 percent annually to $15.8 billion in 2014. As a result, the primary objective of the Green Intelligent Transportation System (Green ITS) project is to develop an efficient sensing and fusion system to support real-time decision making for in-car environment and next generation transportation vehicle systems.

Despite all efforts to make cars a safe product, the main source of car accidents remains the driver. It is estimated that 80% to 90% of fatal and injury accidents are driver-related [3]. The main causes include speeding, alcohol or drugs, distraction (such as phoning and texting), and drowsiness. In particular, fatigue and distraction have a strong impact on driver alertness. It was shown, for example, that driving when drowsy increases crash risk up to six times compared to normal driving [4]. Even worse, the crash risk when writing an SMS is twenty-three times worse than driving with no distraction [5]. The National Highway Traffic Safety Administration (NHTSA) estimates that over 30% of car crashes are inattention-related, including distracted, "lost in thought", sleepy, or fatigued drivers. It is therefore of major importance to develop intelligent car systems able to assess, predict, and avoid dangerous behavior by the driver.

Audio-visual interaction in intelligent vehicle systems has been an active research area and is becoming a common feature in today's smart cars. Driver's assistance system based on audio-visual sensors help drivers avoid or mitigate an accident by sensing the nature and significance of the danger. These intelligent sensor interfaces can alert, warn, and/or actively assist the driver. For example, sensors mounted above the steering wheel can track the driver's pupils and facial orientation to detect head movement, eye direction, and blinking patterns. Intelligent interfaces based on voice recognition commands are designed to facilitate some of the tasks that the driver has to perform, and hence reduce driver distraction or inattention, which are major factors that lead to road accidents. Some common speech-driven applications in the automotive industry include: hands-free mobile communication, changing the radio station, controlling some of the car's setting, or finding directions by providing verbal commands to navigation systems. There is ongoing competition between car companies to integrate new technologies and features into their future models for enhancement to driver safety and communication capabilities. With the increasing demand for more intelligent car interfaces in next generation intelligent vehicles, new potential applications utilizing automatic recognition of driver distraction and fatigue in the automotive environment have been of increasing interest.

Visual behavior is one of the most powerful, natural, and immediate means for detecting human emotional, cognitive, physical, or psychological states. It is an interesting and challenging problem that has an impact on several fields, such as human-computer interaction, data-driven animation, etc. Efficient facial representation is a vital aspect of automated driver state recognition, as face orientation, movements, and expressions are the most discriminative features for this task. In addition, speech signals are one of the most natural and commonly-used methods which can provide vital information about the state of human stress or fatigue. Incorporating the emotional content from speech will lead to a development of a more natural man-machine interaction. Driver performance is also a good indicator of driver condition. When fatigued or stressed, the driving style is very different, and analysis of sensor signals such as the steering wheel or pedals can be extremely helpful in determining the driver's state. Last, physiological sensors have demonstrated impressive capacity to determine the driver's stress, inattention or fatigue. They are, however, uncomfortable over long periods because of contact with the driver. Therefore, we will limit the use of physiological sensors in our work. Sensor fusion is an important area of intelligent vehicle and automated highway research. Multiple sensors inevitably offer redundant and complementary sources of information, which could be utilized to improve the overall robustness of the sensing systems.

The term 'context' refers to information that can be used to characterize the situation of an object. Context can be the object's identity, auxiliary information, past activity, or intent. In context-aware systems, data is processed in a manner that complements information acquired using conventional cameras and supports the current needs of the system. Hence, the system does not attend to all stimuli, but rather selectively chooses the information required for recognition. This can dramatically shorten the time needed to extract useful knowledge from information sources. In our proposed project context can refer to temporal (exact time), spatial (location from Bluetooth-connected GPS receivers), lighting, noise level, and social information (the people with whom the target interacts). By combining hard sensor information, such as where you are and the conditions around you, with soft sensors, such as your social network, complex algorithms can compare what the camera sees to a database of face images and movements to detect what the driver wants or needs. Furthermore, introducing affective intelligence is a relatively recent approach that will have a great impact on understanding the driver's behavior and condition in next generation cars, for enhanced driver safety.

Therefore, we propose a multi-modal-based detection and tracking system that incorporates the use of facial and body features, voice, heart rate, steering wheel signal, and contextual information about the driver and the road, for detection of driver condition and for enhanced communication. This will allow us to make use of the complementary information from all modalities to improve the performance and robustness of the system.

1.2 Challenges and objectives

We aim to create an experimental testbed environment able to assess driver status based on different sensor technologies and contextual information. This involves machine learning techniques and data fusion, in order to turn sensor data and context knowledge into high-level information about driver alertness and attention. The project aims at studying challenges associated with multi-modal data acquisition inside the car environment and developing efficient recognition solutions for intelligent car interface systems. This involves many challenges, which we attempt to tackle in the proposed work.

1.2.1 Challenges

First, the noise surrounding the driver inside the car can severely degrade the speech recognition. It comes from several sources, such as the vehicle's motor, radio, fan, or even from other people. The noise level as well as the nature of its sources makes the car noise signal highly non-stationary. Therefore, efficient speech denoising without distorting the signal is a critical issue.

Camera sensors mounted above the steering wheel are exposed to significant illumination variations. Shadows, reflections during daytime, glare and low light at night, or just weather condition changes can severely alter the face aspect, and is likely to reduce system performance.

Most of the existing work has focused on recognition of facial and audible expressions in a constrained environment. For example, the face is near-frontal, and high-resolution sensors. Emotional audio recognition is performed using pre-defined sequences of controlled emotions. Inside a car, the driver is non-cooperative, meaning that he does not behave to help the system. Therefore, face movements, high pose variation and non-cooperative talking are aspects that will limit system performance compared to well-constrained environments.

Feature representation is a challenging task at many levels. It first requires accurate and reliable data, involving pre-processing noise reduction, representation transformation, and so on. It should be driver-independent, context-independent and stable in time. The aforementioned issues regarding noise and light variation are only some examples of factors affecting accuracy. Moreover, the type of feature selected to ascertain the driver's state is also critical and should be carefully examined.

Last, the fusion of multiple sensors data is a complex task. Extracting meaningful information from each sensor, and making it consistent with that of other sensors is a challenging problem. Moreover, determining the importance of each signal and combining it accordingly, requires good knowledge about both the signals and fusions strategies.

1.2.2 Objectives

The project is subdivided into eight tasks to progressively and efficiently design the testbed/platform:

1. Literature review in the field of intelligent human-machine interactions in intelligent vehicles. A comprehensive review of the different modules is to be conducted to become familiar with the subject and existing approaches.
2. Identification and selection of sensors and data acquisition systems to collect relevant measurements and data in the in-vehicle environment.
3. (a) Algorithm development for head localization and face detection for driver identification, and (b) natural voice command recognition for driver verification and enhanced communication. The main objective is investigating the effect of noisy conditions on the performance of each system independently, and proposing potential solutions to the challenges associated with each modality.
4. Development of systems for detection of driver condition. The main purpose of this task is to develop a driver behavior monitoring system to investigate the influence of different states on driving performance.
5. Design and development of a flexible multi-modal architecture to integrate the developed solutions into a common framework.
6. Investigate efficient ways to fuse different sensor and context information for enhanced driver identification in uncontrolled conditions, and design of efficient man-machine interaction for safe vehicle operation.
7. Generate potential internal/external warning signals for preventive action.
8. Carry out simulations to evaluate the solutions developed by the other tasks. Further optimization and integration of selected existing simulation facilities, to support laboratory evaluation, will be conducted.

Figure 1.1 is a summary overview of the different modules and tasks.

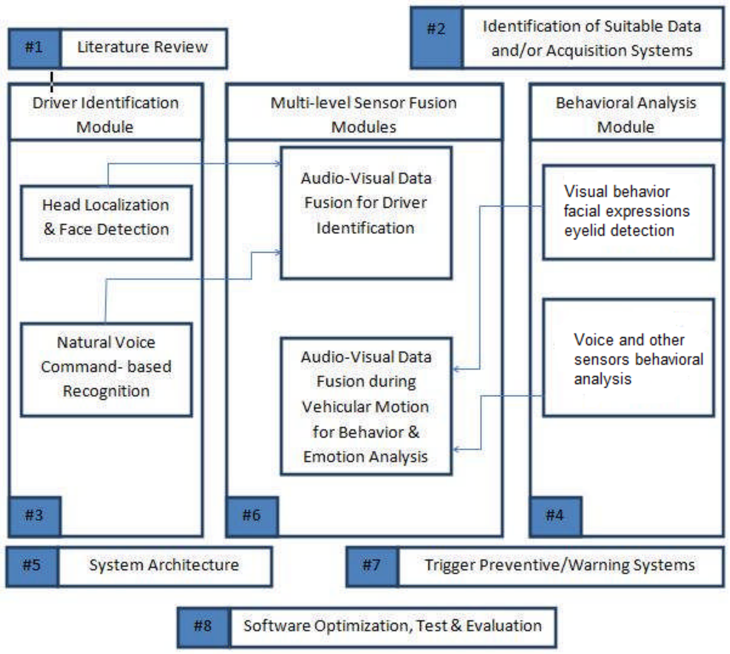

#1
Literature Review
#2
Identification of Suitable Data and/or Acquisition Systems
Driver Identification Module
Multi-level Sensor Fusion Modules
Behavioral Analysis Module
Head Localization & Face Detection
Audio-Visual Data Fusion for Driver Identification
Visual behavior facial expressions eyelid detection
Natural Voice Command- based Recognition
Audio-Visual Data Fusion during Vehicular Motion for Behavior & Emotion Analysis
Voice and other sensors behavioral analysis
#3
#6
#4
#5
System Architecture
#7
Trigger Preventive/Warning Systems
#8
Software Optimization, Test & Evaluation

Figure 1.1: Overview of the project

1.3 Project overview

Regarding the original objectives, Tasks 1, 2, and 3 were fully completed. Other tasks are still in progress. For example, the current work focuses on the visual behavior of Task 4 and does not take into account other type of sensor entries. A strategy for the multi-level sensor fusion has been designed, but was not integrated into the platform. The system architecture has been designed and is now integrated in the software, but it probably needs refinement for future research. The accident-prevention warning system strategy is, for now, a simple beeping system. It could probably be improved later on. Last, optimization, tests, and evaluations need continuous updates. Below is a description of what was achieved in the project in the scope of the master's degree.

1.3.1 Literature review

This chapter is a literature review in the field of man-machine interaction systems for intelligent vehicles. First the state of the art of systems for navigation and road monitoring, voice command, driver identification, and driver status assessment is described. Then, the architecture of man-machine interaction systems is explained in detail. As most of our work is on driver status assessment and driver inattention, we illustrate our ideas with publications and examples related to that specific field. More precisely, we define different types of data acquisition systems, popular features extracted from the original data, analysis and fusion, and the type of decisions and outputs provided by such systems. Last, we provide a summary table of publications directly related to our work.

1.3.2 Framework design for experiments and assessment of the proposed approach

After reviewing the possible prototypes to collect relevant measures and data in the in-car environment, we present our solution. We have built a driving simulator system associated with various sensors. The driving simulator uses a realistic driving software, City Car Driving, and several sensors, such as an infrared camera, a microphone array, a Kinect, and a heart rate monitor. Most of the signal can be acquired in real time and interfaced with a dedicated software package. Last, we describe the experiments that have been carried out on the simulator to determine driver fatigue and inattention of eight volunteers.

1.3.3 Driver identification

In this chapter, we tackle the problem of audio-visual data fusion for driver identification. Person identification inside an actual car involves a number of difficulties compared to the common indoor studies found in the literature. We have used the AVICAR database, recorded inside a car, to tackle the challenges associated with this environment. We have used Gradientfaces for face recognition, GMM for voice recognition, and different types of fusion modules to find the optimal one. The fusion based on a nearest neighbor classifier has shown the best accuracy. We obtain classification results close to 100% using data fusion, compared to 15% to 70% using single experts only. We also integrated this algorithm into our driving simulator and we validated its robustness based on audio and video sequences recorded on our platform.

1.3.4 Driver fatigue detection

This chapter presents a driver fatigue detection strategy based on a single infrared camera. We first developed a new approach for driver eye tracking and blinking, based on an improved version of a particle filter. We used two different state transition models and two different observation model distributions to adapt the tracking depending on the situation. This approach is robust to significant face rotations or partial occlusion. Then, based on the estimated eye position, we developed an eye blink detector and compute the percentage of eye closure, found to be the best feature in vision systems for fatigue detection. Evaluation was done on five different people, executing a challenging sequence of movements and blinking patterns. Results show that our method is robust to face rotation, partial occlusion, and illumination variation.

1.3.5 Driver distraction level assessment

Based on Kinect sensor and computer vision tools, we have built an efficient module for detecting driver distraction, and recognizing the type of distraction. Based on color and depth map data from the Kinect, our system is composed of four sub-modules analyzing eye behavior, arm position, facial expressions, and facial orientation. Each module produces relevant information for assessing driver distraction. They are merged together later on using two different classification strategies: AdaBoost classifier and Hidden Markov Models. Evaluation is done on our experimental testbed. Qualitative and quantitative results show strong and accurate detection and recognition capacity (85% accuracy for the type of distraction and 90% for distraction detection).

This chapter is an introduction to the work proposed in this thesis. We first define our motivations: automotive industry demand for intelligent systems keeps increasing, mostly to improve vehicle safety. On the other hand, audio-visual-based man-machine interactions are becoming commonplace in intelligent vehicles, and can greatly help in determining driver status in term of alertness and concentration. Context and other physiological sensors can also reveal information useful in that task. Therefore, we aim to design a contextual and multi-modal-based driver status identification system. We then describe our objectives subdivided into eight tasks, and the main challenges associated therewith. Last, we give an overview of the thesis by giving a short description of each chapter.

Chapter 2

Literature review

2.1 Overview of man-machine interaction systems in intelligent vehicles

There are a number of man-machine interaction systems available in a car. A steering wheel, for example, would be the most basic, serving the most important task, directing the vehicle. In recent intelligent vehicles though, most man-machine interaction systems are either for entertainment or safety. In this section, we provide examples of commercial products and research on safety-related man-machine interaction devices. Among them, the so-called advanced driver assistance systems (ADAS) are comprised of a large variety of device. We propose to review some of them.

2.1.1 Navigation and road monitoring systems

Global positioning systems (GPS) are probably one of the most popular ADAS systems in today's cars. The technology is now mature and many car manufacturers integrate them into their models. Other driver assistance systems related with the physical road include, for example, blind spot detection, collision avoidance, and lane departure or lane change assistance. Each of these systems are currently commercialized and used in high-end vehicles. However, road monitoring (including pedestrians, obstacles, road signs or other vehicles) is still an open problem and a very active field. Based on RADARs, LIDARs, vision systems, or some combination thereof, many prototypes and studies are being conducted to improve the reliability and efficiency of the technology. This issue is outside the scope of this work, but a few surveys exist in the field [6], [7], [8].

2.1.2 Voice command systems

Car technology is constantly being improved and comes with more and more features, some of which can distract the driver. Therefore, reducing the manual and visual distraction related to those devices should improve their alertness. In that regard, voice command seems to be a very promising field. Even if most voice recognition systems available today have a small dictionary of words and commands, the market should evolve quickly and will offer products with larger dictionariess and wider fields of application. Currently, the driver is able to use voice command mainly for GPS and navigation systems. It should evolve later on to Siri-like applications, allowing the driver to use voice command for hotels, restaurants, fuel, directions, weather, or road conditions.

Another important issue voice recognizer developers are struggling with is noise. In a noisy environment, the voice signal is strongly distorted, thus affecting the word recognition rate and the reliability of the system [9], [10]. Speech recognition in the noisy environment is an entire field of research in its own right, and is outside the scope of this work.

Examples of commercial voice command systems for vehicles include Ford Sync [11], Lexus voice command [12], Chrysler UConnect [13] or Buick Intellilink [14].

2.1.3 Driver identification systems

Another type of man-machine interaction system in a car is driver identification. This type of system is not directly related to driver assistance or safety, but can be used by intelligent systems to know more about the driver and background. Popular person identification methods are based on the face, voice, iris or fingerprints. Numbers of commercial products currently exist for person identification in indoor environments. FaceLab [15], for example is an eye-tracking tool, but also has a face recognition tool. Safran Morpho [16], global leader in multi-biometrics technologies offers numerous of access control terminals based on face, voice, or fingerprint identification.

Of particular interest to us are identification systems using a combination of these features, and especially audio and video, as for example [17], [18], [19], [20], or [21]. These multi-modal person identification systems are as yet used for indoor applications only, and, to our knowledge, no such product exists for car applications.

Driver identification inside a vehicle can be performed using various approaches. A few groups have proposed driver identification systems based on behavior signals from the pedals and steering wheel [22], [23], [24], but the most popular approach so far seems to be fingerprint-based. For example, Bayometric [25] has created a web-based registration and identification system based on fingerprint recognition. The system is currently used for taxi companies. The CarGuard system [26], from the Canadian Biometric Human Identification Center, is a fingerprint identification system to prevent vehicle and freight theft. Using a simple card ID system, the driver ID system by FleetVision [27] is a simple approach for driver identification and a complement to car keys. In the future, Volkswagen plans to integrate a biometric driver identification system based on an infrared camera [28]. The technology is currently not mature enough for a reliable commercial product.

2.1.4 Driver status assessment systems

As most accidents are driver-related, it is of great interest to monitor the driver themselves and assess their alertness and their ability to take the right decision in case of imminent danger. There are many types of driver status that can negatively impact the driver's behavior on the road. Among these are frustration, anger, sadness, stress, fatigue, distraction, intoxication, and so on. Therefore, different types of systems have been designed to assess the particular status of the driver.

Driver emotions and frustration

Emotions and mood can deeply influence driver performance on the road. Therefore, it is of great interest that intelligent cars should be able to access driver emotional state and respond accordingly. In that regard, a few groups have worked on driver emotion analysis, either based on speech [1], [29], [30], facial features [31], [32], audio-visual combination strategies [33], or multiple physiological sensors [34]. Emotions of interest in a vehicle are usually classified into eight different states, namely: anger, frustration, grief, sadness, boredom, happiness, contentment and surprise. They can be positioned on a two-dimensional axis based on valence and arousal (see figure 2.1).

Among these emotions, driver rage and driver frustration are probably the most dangerous ones on the road. According to the frustration-aggression hypothesis [35], frustration is defined as an interference with goal-directed behavior, and it generates an aggressive inclination to the object or person perceived as its cause. In the context of driving, a driver's goal is to go from one point to another with a minimum of interruptions. Thus,

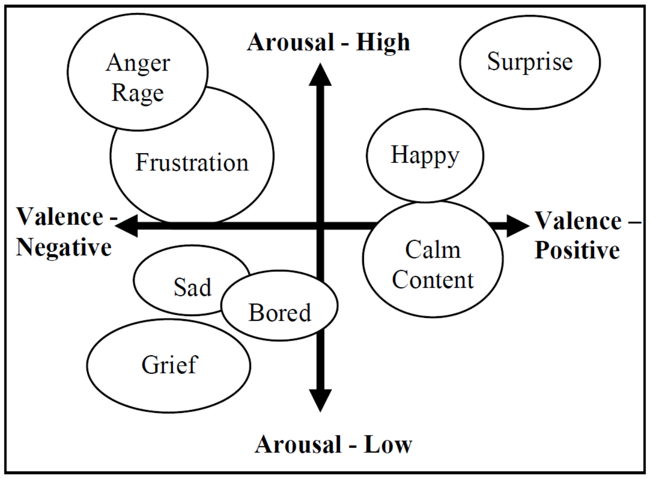


Figure 2.1: Representation of emotions along valence and arousal axes (Picture from [1])

traffic, red lights, pedestrians, or dangerous surrounding drivers are all potential reasons for driver rage and diminished performance. In that regard, a few studies [36], [37] have focused on that particular emotion.

Once emotional state is assessed, the intelligent vehicle can adapt its environment to improve the driving experience. For example, changing the play-list to more relaxing or more enthusiastic songs, or modifying warning displays could positively affect the driver's emotions. However, to our knowledge, no commercial products associated with this technology have been released yet.

Driver fatigue and distraction

Driver fatigue and distraction have been intensively studied recently as they are responsible for a huge proportion of road accidents. They differ from emotions in the sense that it is the driver's decision to avoid them. A system can detect if the driver is tired or not, but the decision of taking a nap is on the driver solely. Both fatigue and distraction are part of driver inattention. They should however be differentiated.

Inattention is the most general term. It was defined in [38] and seems to have an accepted general definition as follows:

"Driver inattention represents diminished attention to activities that are critical for safe driving in the absence of a competing activity".

Differences between inattention, distraction, and fatigue have been extensively discussed in [38], [39], or [40]. In most cases, when considering inattention of a driver, only distraction and fatigue are taken into account. Emotions, though, can also be considered as cognitive distraction.

Fatigue can be defined as a combination of symptoms and a subjective feeling of drowsiness. No general definition has been accepted so far, but the European Transport Safety Council has suggested that fatigue "concerns the inability or disinclination to continue an activity, generally because the activity has been going on for too long". The causes for fatigue can be physical, physiological, or psychological [40].

The NHTSA [41] has defined distracted driving as "an activity that could divert a person's attention away from the primary task of driving". Unlike fatigue, distraction is much more tightly bounded in time and does not affect the driver on a long-term basis. It is commonly classified into three categories, namely

- Manual distraction: The driver takes their hands off the wheel. This includes, for example, text messaging, eating, using a navigation system, or adjusting the radio.
- Visual distraction: The driver takes their eyes off the road, for example, reading or watching a video.
- Cognitive distraction: The driver's mind is off driving. This can happen when talking to other passengers, texting, or simply thinking.

The literature in the field of driver inattention is extensive. More details are provided later on in this chapter, and table 2.1 is a summary of interesting publications directly related to our work. Recent literature reviews in this field have also been published and provide a good overview of the state of the art [42], [43], [44], [45], [46].

Recently, commercial products, such as Eye Alert [47], Delphi's Driver State Monitor (DSM) [48], Toyota and Lexus' Driver Attention Monitor [49], or SMI's InSight system [50] have been conceived to detect driver fatigue and inattention using cameras inside a vehicle. Other products such as Volvo's Driver Alert Control system [51], Ford's Driver Alert [52], and Volkswagen's Fatigue Detection system [53], rely on road monitoring and steering wheel movements to detect fatigue.

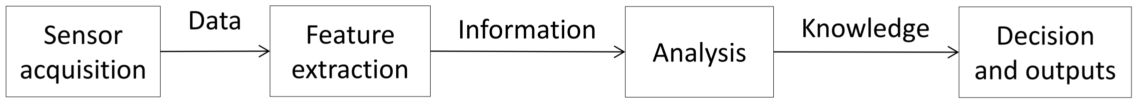


Figure 2.2: General architecture of man-machine interaction systems

2.2 Man-machine interaction system architecture

Any man-machine interaction system is characterized by inputs, processing and outputs. Sophisticated man-machine interaction systems usually involve a hardware part, for input acquisition and output display/audio, and a software part, including data processing and analysis. More specifically, such a man-machine interaction system can be divided into four parts as shown in figure 2.2. First, the system needs raw input data from sensor acquisition. Then, using signal processing and data cleaning, we turn this raw data into meaningful information. This stage is called feature extraction. Next, we analyze this information, either by performing decision rules, statistics, machine learning, or fusion strategies. In this way, we turn this information into knowledge. This knowledge will make possible any decisions by the system and interactions with humans.

This architecture is general for any man-machine interaction system, but we focus our explanations in this section on the particular case of driver status systems, which is the heart of our work.

2.2.1 Sensor acquisition

Below is a list of sensors we can find inside a car and their utility for driver status assessment.

Physiological sensors

Physiological sensors are able to capture driver physiological data such as heart rate, brain activity, hand moisture, or breathing. These types of sensors are excellent indicators of stress, fatigue, or any emotion. For driver emotion and inattention detection, Healey *et al.* [34] have used electrocardiogram (ECG), electromyogram (EMG), skin conductivity (EDA), and chest cavity expansion for measuring driver stress level. Electroencephalograms (EEG) are also popular and efficient sensors for driver fatigue, as mentioned by Shiwu *et al.* [54]. Last, Damousis *et al.* [55] have used electro-occulogram (EOG) for high-precision eyelid activity, claiming that similar results could be obtained with high frame rate cameras. Nevertheless, physiological sensors are intrusive and cannot be used inside a car for commercial applications. They can be used as ground truth for studies, but they do not represent a realistic solution for driver inattention monitoring.

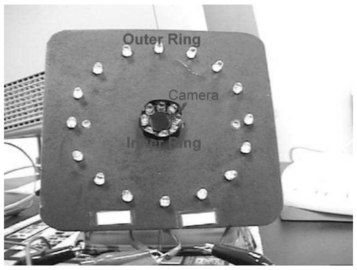
Microphones

Voice can be exploited as a powerful indicator for emotional driver state. Several groups have equipped their simulator or car with microphones to record and analyze voice [1], [29], [56]. Inside a vehicle, noise can affect the signal severely, and using a single microphone with no denoising strategies is not feasible. Denoising can be done by denoising algorithms using a single microphone, as Tawari *et al.*. did [29]. Otherwise, beamforming techniques can be applied to the signal using a microphone array. Beamforming allows noise reduction with no heavy processing, as it only relies on the spatial position of the microphones and wavelength superposition.

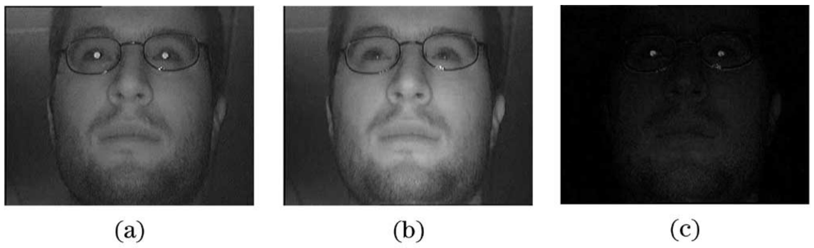
Cameras

As facial expressions are a very natural feature for driver emotions and inattention, many existing systems exploit a camera as their main (sometimes only) sensor. Either a single or a pair of cameras is used. When more than one camera is involved [57], stereo-vision can be done, providing a powerful tool for face orientation estimation and tracking. However, stereo-vision methods require calibration as well as complex and heavy computation to be efficient. Another reason for using more than one camera is to have better resolution on a particular area of the face. For example, Ji *et al.* [58] have used a pair of cameras: one monitoring the entire face, the other one focusing on the eyes.

In a vehicle, the use of an infrared (IR) camera coupled with an infrared illuminator makes a lot of sense, as it is less sensitive to outdoor illumination and works even at night. Ji [58] [59]



from [58]



from [60]

Figure 2.3: On the left, an infrared camera and illuminators device. On the right, an illustration of the bright (a) and dark (b) pupil effects, and resulting frame subtraction

and Bergasa [60] have developed an efficient infrared illumination system, insensitive to light variation and convenient for eye detection. The system consists of two rings of infrared LEDs with a camera in the middle. The two rings are turned on alternately to light the driver. The inner ring makes the pupils of the driver very bright (similar to the red eye effect with a color camera), whereas the outer ring leaves them dark. Inner and outer ring activations are synchronized with the frame acquisition. The image resulting from the subtraction of two successive frames is illumination-independent and characterized by a sharp contrast in the pupil area (see figure 2.3).

Recently, Li *et al.* [61] have used a Kinect and its associated SDK for driver fatigue recognition. The advantage of Kinect is that it provides a depth map associated with a color image. This makes face and body detection and tracking much easier.

2.2.2 Feature extraction

The feature extraction is extremely important as it provides higher-level information to the system for further analysis. Depending on the sensor, feature extraction is either extremely straightforward, or requires a lot of processing and complex algorithms. For example, physiological sensor data are typically one-dimensional signals, and tend to repeat in time. Spectral analysis is therefore the most appropriate tool for feature extraction. On the contrary, video-cameras provide two-dimensional signals, varying in time. Feature extraction in this case requires complex computer vision tools for eyes, head, and body detection and analysis. Let us characterize the behavior of fatigued or distracted drivers before explaining the type of feature we can extract from the different sensors.

Behavior of inattentive drivers

When tired, drivers try to resist sleep, and show very specific symptoms such as

- frequent yawning
- nodding
- slow and infrequent head motion
- increased duration and frequency of eye closure
- mental confusion
- slow reactions
- moving body and head to reduce muscular tensions
- shallow breath
- increased heart rate

The symptoms vary depending on the driver and do not appear at the same degree of fatigue, but most of them can be observed and analyzed with vision systems. In particular, the PERCLOS (percentage of eye closure) has been found to be a valid physiological measurement for driver fatigue [62]. Kircher *et al.* [63] have extensively reviewed the validity of symptoms defining drowsiness. When extremely tired or falling asleep, the driver is not able to follow the road properly. Therefore, they are subject to lane deviation, and their slow reactions make them use the pedals and steering wheel with slower movements.

Driver behavior when distracted depends on the type of distraction. Some distractions are much more demanding than others, and some can be a combination of the three categories. For example sending an SMS is considered to be one of the most distractive actions inside a vehicle, as it requires manual, visual, and cognitive attention. In general, when cognitive distraction is involved, the behavior of inspecting the forward view is modified: the field of view tends to be narrowed, and use of the mirrors is reduced. According to Rantanen *et al.* [64], the visual field can shrink up to 13.6% because of a cognitive distraction. It is commonly accepted that the eye glance pattern [65] and more precisely eyes-off-road glance duration and head-off-road glance time are good measures of distraction. In term of driver behavior on the road, Raynney *et al.* [66] suggested that distraction involved an important lack of vehicle control, such as drifting to the wrong side of the road, or unexpected speed changes.

Physiological features

Spectral analysis of heart rate variability shows that the 0-1Hz range is a sensitive indicator of driver fatigue [67], [68]. Moreover, electroencephalograph (EEG) has been found to be a valid, objective, and accurate measure of driver inattention using δ, θ, α, β, and σ brain wave activities [54], [69]. When strong cognitive distraction is imposed on the driver, the heart rate increases by up to 8 bpm, which can be a good indicator of driver distraction [70]. For distraction, the EEG also contains information about cognitive demand and driver's engagement in the task [71].

Driver performance-based features

Using indicators of driver performance, it is also possible to infer the level of inattention. Driver performance corresponds to how well a driver is behaving within the road environment. Have they noticed all important signs on the road, do they stay in the right lane, are they too close to other vehicles, are they driving smoothly? Feature extraction in these cases is not necessarily related to the aforementioned sensors and can involve steering wheel movements, pedal activity, or any sensor related to road monitoring. Popular features in that case includes standard deviation of the lateral position of the vehicle [72]; [73]; steering wheel and pedal speed and acceleration; steering wheel angle and heading error [72]; or headway tracking [74] (distance between the vehicle and the car in front). These features have been found to correlate well with driver inattention [73], but they are also affected by external factors such as driver experience, road type, weather, and outdoor lighting. Moreover, the measures rely on long-term statistics and the system is unable to predict immediate dangers such as micro-sleep events.

Vision-based features

Feature extraction with vision systems is a key component. The accuracy of the extracted features directly impacts the system performance. An efficient feature extraction system often relies on a good localization of body, face, and eyes. A more extensive review of computer vision tools for these tasks will be described later on.

For driver inattention, the so-called eyes-off-road glance duration and head-off-road glance time are considered as valid measures for visual distraction, and can be assessed using an embedded camera [65]. For driver fatigue, PERCLOS (percentage of eye closure) [62] is considered to be the best correlated physiological feature. Other behaviors such as yawning [59], [58], [57], nodding [60], [75], face motion [57], [61], [75], gaze estimation [60], [58], or blink detection [76], [55], are also popular features, widely used in the field. Several statistics like PERCLOS, gaze distribution, and yawning or nodding frequencies can be computed based on the aforementioned features to provide higher-level features.

2.2.3 Data analysis

Once features are extracted from a single or multiple sources, this information should be turned into an even higher level of understanding for the system to be able to make a decision. The term data analysis includes many possible approaches.

Sometimes, simple decision rules can be applied for decision making. For example one can consider that a signal over a certain range, or showing a very specific and easily detectable behaviour is enough to make a decision: PERCLOS over 50% [62], or high activity of the heart rate variation around 0.1 Hz in the spectral domain are indicators of extreme fatigue [67]. Appropriate thresholds on those measures can be enough to decide if a driver is dangerously tired.

When the signal is too complex, the decision is non-linear, or too many dimensions are involved, simple decision rules cannot be efficiently applied. Therefore, we need to use machine learning, probabilistic, or regression techniques. For example, drowsiness has been successfully assessed using EEG features and SVM [77] or KPCA [78]. Chua *et al.* [79] have used a multi-linear regression for ECG and PPG-based fatigue detection.

Sometimes, it is necessary to combine features of different natures. We talk in that case of multi-modal data fusion. The features can come from the same sensor (typically a camera, see table 2.1), or from different sources. The most popular fusion techniques are: fuzzy logic; Bayesian networks and dynamic Bayesian networks; machine learning techniques such as neural networks; or different types of heuristics and decision rules. In the case of multiple sources, approaches, such as in Daza *et al.* [72], have used both driver (PERCLOS, nodding, etc.) and driving (lane variation, steering wheel movements, etc.) data to assess fatigue via artificial neural networks. Fletcher *et al.* [80] have successfully merged driver gaze and road information to detect if a driver was missing any road sign. Their approach was based on correlation and regression techniques to associate gaze and road information. Last, Li *et al.* [61] have used all of computer vision, steering wheel signal, and pulse oximeter to infer driver fatigue. In this case too, fusion was done through an artificial neural network.

2.2.4 Decision-output

Last, the decision and output are made available to the driver. The output can be very different depending on the type of system and its primary goal. A GPS system displays on a screen the map and itinerary, possibly with audio output to give the driver real time indications. A voice command system will respond according to the driver's order, for example by turning on the radio, or increasing the temperature.

In the case of a driver status assessment system, an important difference between emotions and inattention has to be noted. For emotion recognition, the system should try to attenuate negative ones and make the driving more pleasant. This can be done by changing the type of music a driver is listening to, or by modifying the temperature inside the vehicle, for example. For driver inattention, the driver is the only one able to improve the situation, and the best the system can do is share that information. The most straightforward strategy is to emit a warning or a beep to make the driver aware of their dangerous behavior. If tired, the system can suggest a nap or a break to the driver. If inattentive, it can ask the driver to be more careful, and, if possible disable the source of the distraction. Another possibility is to share the message with other vehicles, asking other drivers to be careful and aware of the danger.

We summarize in table 2.1 the main approaches used for fatigue and inattention detection using computer vision, as directly related to our work.

This chapter is a literature review in the field of man machine interaction systems for intelligent vehicles. First the state of the art is described for navigation and road monitoring systems, voice-based command systems, driver identification systems, and driver status assessment systems. Then, the architecture of man machine interaction systems is explained in details. As most of our work is on driver status assessment and driver inattention, we illustrate our ideas with publications and examples related to this specific field. More precisely, we define different types of data acquisition systems, popular features extracted from the original data, methods of analysis and fusion, and the types of decisions and outputs provided by the systems. Last, we provide a summary table of publications directly related to our work.

Publication	**Sensors**	**Features**	**Fusion**
Bergassa et al. [60]	1 IR camera, bright/dark pupil alternation	PERCLOS Eye closure duration blink frequency nodding face position fixed gaze	Fuzzy logic
Smith et al. [76]	1 color camera	eye blinking eye rotation gaze tracking	Finite state automaton
D'Orazio et al. [81]	1 color camera	eye blinking PERCLOS	Gaussian mixture model
Damousis et al. [55]	Electro-occulogram (no camera)	Blink duration blink interval blink frequency lid closure duration lid opening duration	Fuzzy logic
Ji et al. [59], [58], [2]	2 IR camera, bright/dark pupil alternation	Yawning frequency AECS PERCLOS gaze distribution fixation saccade ratio head tilt frequency	Bayesian network [58] and dynamic Bayesian network [59]
Senaratne et al. [75]	1 color camera	PERCLOS slouching position adjustment nodding	Fuzzy logic
Li et al. [61]	Kinect	Head position head orientation eyebrow position blink frequency blink duration	Neural networks
Bergassa et al. [57]	3 visible cameras	Head pose PERCLOS yawning	none
Zhao et al. [82]	1 visible camera	Yawning Nodding Eye closure	Neural networks
Yuging et al. [83]	1 visible camera	PERCLOS Face rotation	none
Li et al. [84]	2 visible cameras	Yawning	none
Fan et al. [85]	1 visible camera	Eye blinking Yawning	Bayesian network

Table 2.1: Computer vision-based inattention detection systems

Chapter 3

Framework Design for Experiments and Assessment of the Proposed Approach

This chapter presents the solution adopted for our framework design. We have designed and created a driving simulator, and mounted on it different types of sensor for recording. A software interface was implemented to allow easy and centralized data acquisition. Last, experiments have been carried out on the simulator for driver fatigue and inattention analysis.

3.1 Motivations and specifications

As part of the framework of a multi-modal context-aware man-machine system, we need a reliable platform for carrying out experiments and running demos. Most of the time, teams choose to evaluate their systems either in real cars ([57], [60], [80], [81], [1]) or driving simulators ([61], [86], [87], [58], [59]). We have chosen the driving simulator strategy as much more affordable and convenient for the development and the evaluation. Driver simulators also provide a safe environment to put the driver in very dangerous situations.

Specifications for our driving simulator and experimental testbed are as follows:

- low-cost

- realistic driving experience
- multi-sensor acquisition capacity
- real time data processing capacity
- recording and storage capacity for off-line data analysis.

Among studies using a driving simulator, the description of the simulator itself was rarely extensive as it was not the purpose of the study itself. A few papers, though, were a bit more explicit about their experimental platform [87], [88]. A literature review on existing driving simulators used for research was written by the Institute for Transport Studies of the University of Leeds [89] although it is a bit out of date now.

The price of a driving simulator can run from a few thousand dollars to a few millions. The main pricing difference is the capacity of providing a realistic driving experience to the driver. The most expensive ones such as the Mercedes-Benz High-Tech Driving Simulator, consist of a real car surrounded by a 360^o giant screen, in a dynamic platform to simulate inertial forces and vibrations of the road. Other high quality simulators used for academic research are the one at the CEIT research center [90], or the VTI simulator in Sweden [91]. They both include numerous sensors for collecting data about driver activity.

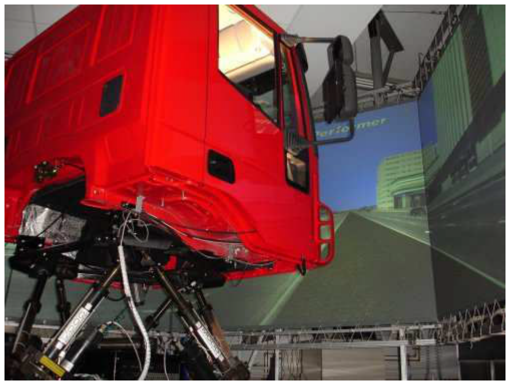
For a much lower budget, companies like STsoftware [92] or Stisimdrive [93] offer all-in-one driving simulator packages, including both equipment and software. These are mainly used for driving schools, and sometimes for research [87], but they are unfortunately outside our budget (around \$30 000). Some papers [88] propose solutions to build a low cost-driving simulator, while providing realistic sensations for the driver. The price of such a setup is also too expensive for us (around \$16 000), but it provides some good references and ideas to build our own setup. Last, video game amateurs have built their own driving simulator for personal purposes, and present possible low-cost solutions. Figure 3.1 is an overview of a few existing simulators.

3.2 Hardware and equipment

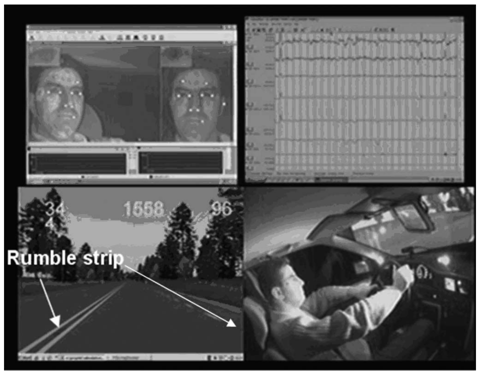
Figure 3.2 shows a final version of the driving simulator. The detailed product list and prices can be found in appendix A. Below is a description of the system components.

Mercedes-Benz Unveils High-Tech Driving Simulator

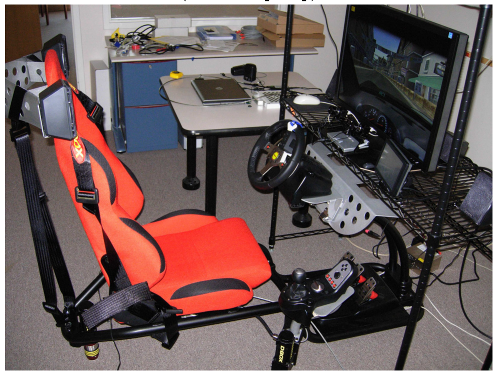
StiSimDrive simulator from the Cobvis-D project [87]



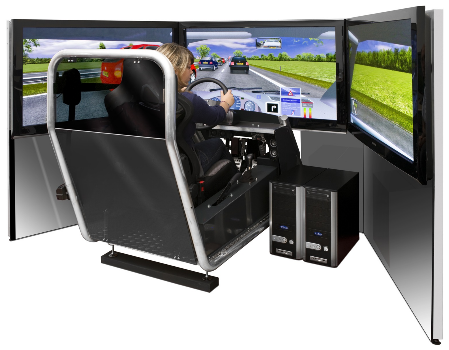
CEIT's naturalistic simulator (from [72])



VTIs driving simulator during the SENSATION IP Pilot 2.5 [55]



A low cost simulator (from [88])



STsoftware Jentig50 simulator [92]

Figure 3.1: A few existing driving simulators

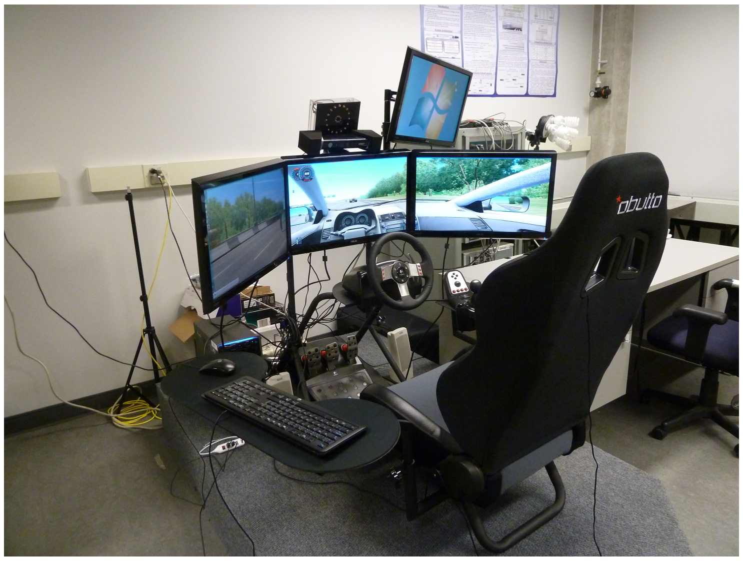


Figure 3.2: Driving simulator

3.2.1 Driving simulator

For the driving simulator itself, including structure, display system, and driving software, the following equipment was ordered:

Cockpit

This is basically a seat with a steering wheel/pedal/shifter mount and a monitor stand. We have chosen the Obutto oZone cockpit, which provides good equipment for an affordable price.

Driver input system

This includes steering wheel, pedal, and shifter, compatible with the driving simulation software. We have chosen the Logitech G27 racing wheel which is a standard piece of good-quality equipment, compatible with most of the driving software.

Driving simulation software

There are many packages, at many prices. Affordable driving software that is not also racing software are somewhat rare, though. We have chosen City Car Driving V1.2, which is very affordable while being realistic for city driving. A large panel of maps and many options are available, such as traffic density, weather, and time of day, vehicle type, and so on.

Triple monitor display

For this, we need an additional option for the cockpit: the triple monitor mount, and three monitors. The ASUS VH242H LCD Monitor is an affordable one with a good screen quality. Moreover, to stretch the display to three screens, we need an additional graphic expansion. We have chosen the Matrox-TripleHead2Go as it is officially compatible with the software. Last, the original graphic card was not powerful enough to support high resolution with a triple monitor display. We have ordered an additional graphic card to solve this problem.

Loud speakers

A set of speakers for the software audio output. We already had some available in the lab.

3.2.2 Sensors

Microphone array

We have chosen the Andrea Electronics Microphone DA-350 which is a high-quality microphone for hand-free car application.

Near-infrared camera

Based on the bright/dark pupil illumination system proposed by Ji *et al.*, two cameras were ordered and built by the Hong Kong Polytechnic University. The cameras are low cost CMOS sensors with an original resolution of 640 × 480 and a frame rate of around twenty-five frames per second. The LED rings emit light at 850mn (near infrared spectrum). The cameras are working properly, but the alternation between inner and outer rings is too slow to provide an acceptable frame rate and clean frame subtraction. Therefore, we use the cameras in inner ring mode only (bright pupil effect) to make eye localization easier.

Kinect

A Kinect for Windows was ordered before doing experimentation. The reason for this choice is that the infrared camera was not working as expected. Moreover, we found that the recent release of the Kinect SDK provided a very efficient face tracker, able to assess in real time facial orientation and expressions. This feature is extremely helpful for further analysis. Moreover, the Kinect is equipped with a microphone array.

Heart rate monitor

At the same time as the Kinect, we found that a heart rate monitor could be useful in our experiments. The CMS-50E OLED Fingertip Pulse Oximeter is an affordable heart rate monitor and oximeter. The device is simply clipped to a finger and is quite light and unobtrusive. Car manufacturers now plan to integrate a heart rate sensor directly into the steering wheels of future smart cars, making the monitoring non-intrusive. The heart rate and oxygen rate can be transferred to the computer in real-time.

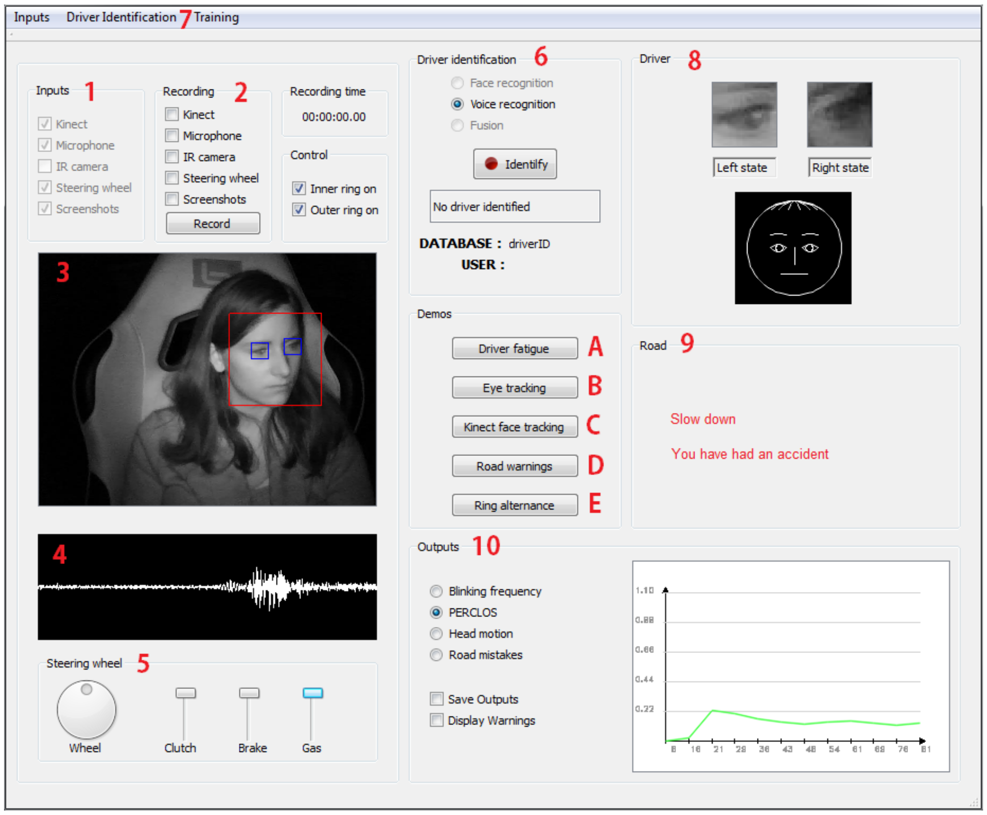
3.2.3 Other

Computer desktop

The desktop runs the driving software and the data acquisition software at the same time.

Additional monitor

During driving sessions, the driving software is displayed on the three monitors. We needed an additional one to get access to our own software. The fourth monitor is mounted on top of the others and a pivoting arm allows visualization of the software by the experimenter only, to avoid driver distraction.



Recording time: 00:00:00.00

No driver identified

DATABASE : driverID
USER :

Slow down

You have had an accident

Figure 3.3: Software interface

Lighting setup

In order to simulate several lighting conditions for the tests, a set of projectors was ordered. The InspironPhoto 2400W kit is a photography package with several spotlights and softeners, providing a wide range of lighting conditions.

3.3 Software

Figure 3.3 is a screenshot of the GUI. The software was written in C++ using Qt for the GUI and audio acquisition. Video acquisition and computer vision processing is done using OpenCV library. The steering wheel and pedal signal are collected using the Microsoft Dinput library. The main functionalities are described below.

3.3.1 Inputs and recording

The left side of the interface is related to inputs and recording. The list of inputs accessible by the interface are listed in 1. If the checkbox is checked, the input is actually plugged in. This includes:

- infrared camera
- Kinect
- Microphone
- Steering wheel and pedal signals

Input states can be viewed on frames 3 (IR camera and Kinect), 4 (Audio), and 5 (Steering wheel and pedals).

The list of inputs to record can be checked and unchecked in 2. When clicking on Record button, each stream is recorded in an appropriate file whose name is the timestamp of the starting time (same for each stream). Data from the Kinect is recorded in two separate AVI files (for depth and color). The steering wheel signal is synchronized with the camera frames, or is acquired every 10 ms if no camera is recorded. The signal is saved in a text file with a timestamp for each sample. If the driving software is on, it is possible to record the road. To avoid too much writing on the disk at the same time, the road is recorded using screenshots taken every second and stored in a specific folder.

3.3.2 Real-time functionalities

Functionnalities are in the middle column of the interface. This includes a driver identification module and a few demos.

The driver identification system described in chapter 4 is available in the interface. On frame 6, the user can choose the type of recognition (consistent with available inputs), and press Identify when ready. Menu 7 also has a driver identification section to add a new user, add more data, or build a model for the database.

The fatigue detection algorithm described in chapter 5 was also added to the interface (button A). Eye tracking and eye blinking results are displayed in frame 8, and real-time statistics such as PERCLOS or blinking frequency can be displayed in frame 10. The user can save these statistics in a text file by checking Save outputs. The checkbox Display

Warnings enables beep and popup display if PERCLOS is above 50%. Eye tracking demo (button B) is the same as fatigue detection, except that eye blinking detection and statistics computation is disabled.

Kinect face tracking (button C) uses the Kinect SDK to track the face. Eyes are displayed in frame 8, and face orientation and facial expressions are represented by the face below the eyes. Head pitch, yaw and roll values are available in frame 10.

If City Car Driving is open, driver performance can be accessed by starting the Road warinings demo (Button D). Indeed, whenever the driver makes a mistake on the road, City Car Driving displays a warning on the screen. Based on the screenshots and image processing, we can recognize these warnings in real-time and integrate them into the interface. They are displayed in frame 9.

Last the Ring alternance demo (Button E) shows how the IR camera is alternating between the inner and the outer rings.

3.4 Data collection

In order to create a context-aware human-machine interface, we need some data to work on. Therefore, we have prepared a series of tests to put the different drivers in distracted and fatigued conditions. More details on the experiments can be found in appendix B.

3.4.1 Experimental platform

A total of eight drivers were asked to participate in the study. This number seems to be reasonable compared to similar studies (The number of drivers for experiments is usually between eight and ten). Drivers were either men (six of them) or women, from different countries (Egypt, Iran, France and Singapore), either wearing glasses (four of them) or not, with ages varying from twenty-four to forty years old. They were all experienced drivers, but using their car at varying frequencies. The study was reviewed and received ethics clearance through the Office of Research Ethics at the University of Waterloo.

Every aforementioned sensor was used during the study, except the microphone array, as Kinect already had one. The IR camera was focused on the driver's head, while the Kinect recorded the entire upper body. Screenshots of the road were taken every second. Data recording was done directly on the desktop computer, except for Kinect data (video and audio), which was recorded on a separate laptop using Kinect Studio. Heart rate monitoring was uploaded to the computer after each session.

3.4.2 Procedure

Each driver was first introduced to the car driving simulator and asked to drive for a few minutes to become familiar with the simulator. They were then explained the general context and instructions. During the driving sessions, instructions were displayed on the screen.

The drivers were recorded during four fifteen-minute sessions. Two sessions were in the morning, when the driver was awake and alert, and two sessions at the end of a full business day, when the driver was tired. Each session was either on a highway with low traffic, or in the city with higher traffic. During each session, the driver had to follow a well-defined procedure involving several tasks putting them into distracted situations. Half of the sequence was normal driving, the other half was distracted driving. The different tasks the driver had to accomplish were:

- making a phone call (the phone conversation was done through a headset, where a computer voice asked the driver to repeat predefined sentences)
- drinking
- sending an SMS
- adjusting the radio
- looking at a map (with no need to search or localize anything)
- looking at an outdoor object (the driver was asked to look at a panel and write down on his phone the digits written on the panel)

These actions are of course a small subset of all possible distractions in a car, but we did our best to make it representative of the most common and general distractions.

The night sessions were slightly different from the morning ones. Drivers were asked to simulate extreme drowsiness by doing the following actions:

- closing their eyes for a few seconds

- yawning
- nodding

and the last session was a little bit longer to allow ten minutes of driving without any distraction.

In total, twenty-eight driving sessions (out of thirty-two) were suitable for analysis. Figure 3.4 shows sample frames from the driving sessions.

After reviewing the possible prototypes to collect relevant measures and data in car environment, we present our solution. We have built a driving simulator system associated to various sensors. The driving simulator uses a realistic driving software, City Car Driving, and several sensors, such as an infrared camera, a microphone array, a Kinect, and a heart rate monitor. Most of the signal can be acquired in real time and interfaced with a dedicated software. Last, we describe the experiments that have been carried out on the simulator to determine driver fatigue and inattention of eight volunteers.

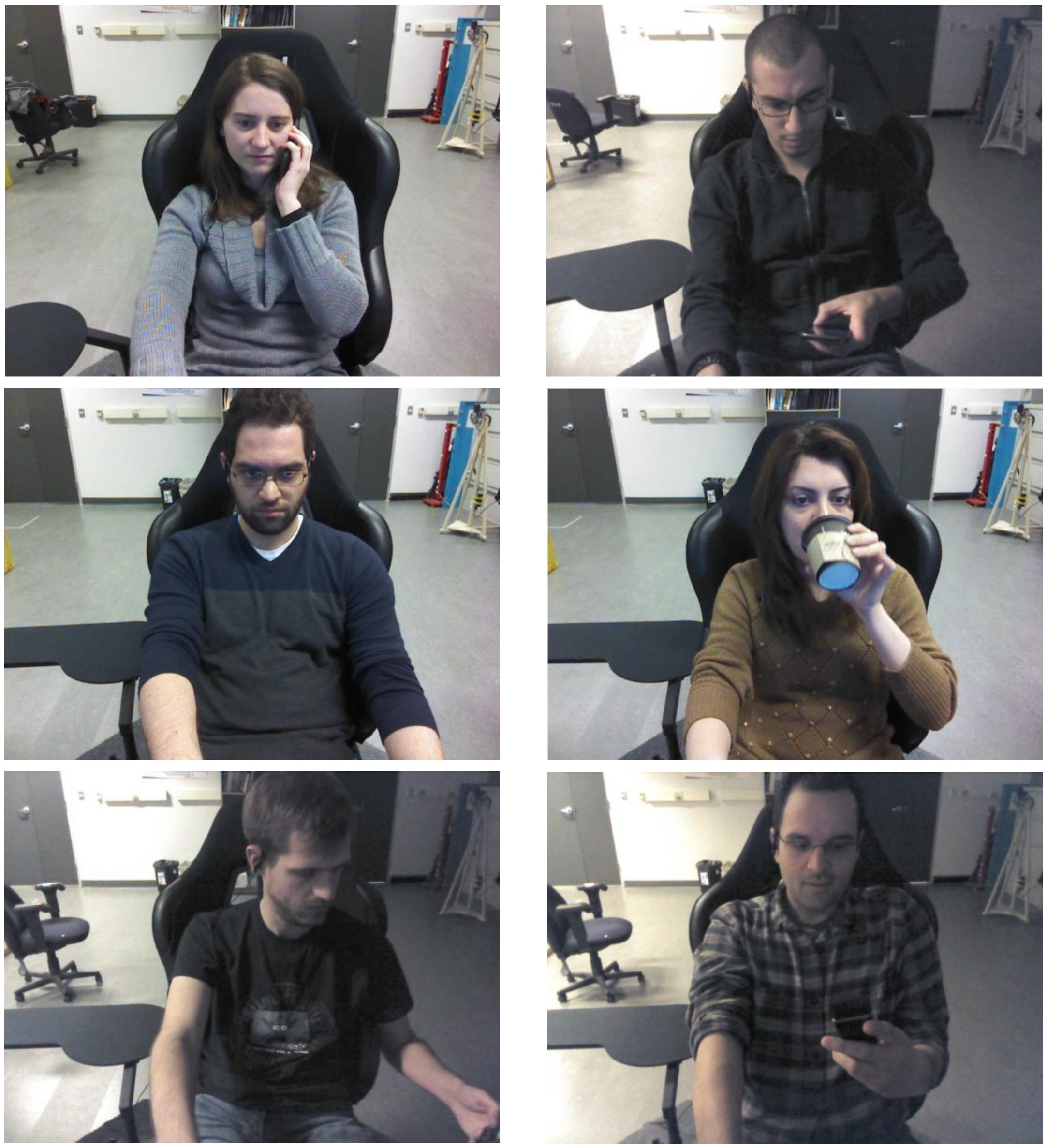


Figure 3.4: Examples of driver actions

Chapter 4

Driver Identification

In this chapter, we describe a driver identification system based on audio and video data fusion. Our method was tested on the AVICAR database [94], which consists of audio-visual sequences of subjects inside a vehicle. Then, the algorithm was implemented and adapted to our driving simulator. The results showed that the system is very efficient for our specific application.

4.1 Motivations

In context-aware man-machine interaction, the term context refers to information that can be used to characterize the situation of an object (in our case, the driver). Contextual information can be object-independent or object-dependent. In the case of a driver inside a vehicle, the type and vehicle model, the weather, the time of day, or the type of road are driver-independent. However, background knowledge about the driver, such as age, sex, nationality, profession, or sleep disorders, is another type of context able to improve assessment of the driver's state. In that regard, it is of great interest to start a driving session by identifying the driver, so as to use background knowledge associated with this particular subject.

Biometrics data fusion for subject identification has been widely studied in the past few years, for a number of applications and contexts [95], [96]. The idea of using several experts and combining them to increase the accuracy of the recognition is quite natural, and early work on that field even predates audio-visual research. Audio-visual data fusion for person identification is still an active topic, and several papers examining it have been published. For example, Wasniowski [97] made a short but interesting description of data fusion principles for biometric verification, suggesting that Support Vector Machine (SVM) was an efficient classifier for the fusion stage. Fox *et al.* [17] have proposed a robust person identification system using face, mouth, and speech experts. Fusion of these three is done by a weighted sum rule that does not require any training, and adapts to the reliability of each expert given the test conditions. Ben-Yacoub *et al.* [18] tested audio-visual person identification using several classifiers (Bayesian network, decision tree, SVM, MLP, etc.). They concluded that the most efficient classifier, in their case, was the Bayesian network. Munoz *et al.* [19] used a modular neural network with fuzzy integration to perform data fusion of face, fingerprints and voice. Using SVM, Wang et al [20] achieved robust biometric identification fusing irises, face, and fingerprints. Based on expert weighting according to reliability, Jiang *et al.* [21] built an efficient audio-visual human recognizer for human machine interactions. However, there are very few papers which consider the outdoor identification. To our knowledge, only Vajaria *et al.* [98] have considered this possibility, and nobody has published anything entirely dedicated to driver identification based on audio-visual data fusion.

4.2 Voice and face identification

We chose to restrict our driver identification to face and voice recognition only, performing data fusion to increase the recognition rate. These two experts have the advantage of being complementary [33], and can be efficiently combined. In this section, we describe the methods we used for face and voice recognition.

4.2.1 Voice recognition

Voice recognition is a mature field, and standard acoustic methods are quite efficient for this task [99]. It is actually a much easier task than automatic speech recognition, even where the recognition rate can be significantly affected by noise. Several classifiers can be used to train the system, such as Gaussian Mixture Models or Hidden Markov Models. Most of the time we used the well-known MFCC [100] features as an input of our classifier.

First, we selected a speech segment of an arbitrary length for each speaker and we extracted the MFCC features for this segment. The signal was divided into frames of 32 ms with a Hamming window and a 50% overlap between frames. For each frame, we computed a twelve-dimensional MFCC vector. For each speaker, we ended up with a set of MFCC features proportional to the length of the segment.

Second, we trained a Gaussian Mixture Model (or GMM) for each speaker [101]. The GMM classifier aims to approximate the probability distribution of the training set with a linear combination of Gaussians. To do this, we used for each speaker the set of MFCC features corresponding to the training segment. We set the number of Gaussians to 50, and we computed a first estimate of the mean and variance of each Gaussian with k-means clustering. We then used ten iterations of the expectation maximization algorithm to obtain the final GMM parameters. We ended up with a number of GMM equal to the number of speakers.

The recognition was done as follows: given the testing speech signal of a particular subject A_i, we compute the MFCC features for this signal. These features correspond to the observations O_i. We would like to know the probability $Pr(\lambda_j|O_i)$, where λ_j is the class corresponding to the subject A_j. Theoretically, we should have

$$Argmax_j(Pr(\lambda_j|O_i)) = i$$

which simply means that the most likely subject, given an observation, is the subject who has produced the observed signal. We can determine this using the Bayes rule:

$$Pr(\lambda_j|O_i) = \frac{Pr(\lambda_j)Pr(O_i|\lambda_j)}{Pr(O_j)}$$

In practice, $Pr(\lambda_j)$ is set to be equiprobable for each class, $Pr(O_i|\lambda_j)$ is simply found using the GMM of the class λ_j computed in the training phase, and $Pr(O_j)$ does not depend on the class we are looking for and can be omitted.

If we just wanted to perform voice recognition without any fusion, the decision would be done simply by selecting the class corresponding to the highest probability. However, we keep the probability for each class as our voice expert, to be further transmitted to the fusion module. Figure 4.1 shows the architecture of the voice recognition module.

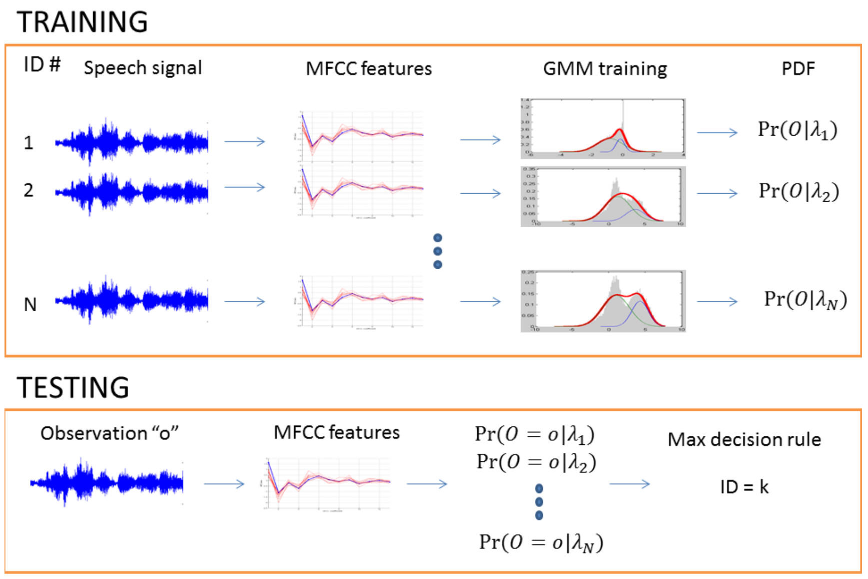


Figure 4.1: General architecture of the voice recognition module

4.2.2 Face recognition

Face recognition has received significant attention over the past two decades. The recognition methods can be categorized into two classes. One is based on template matching and is local, and the other is based on dimensionality reduction and uses the entire face. We will here focus on the dimensionality reduction class, and use the method called Gradientfaces [102]. We chose this method for various reasons: first, it is said to be illumination-invariant, which is a critical point for our application. Second, it does not require perfect alignment to identify the face, which is one of the main weaknesses of the template matching approaches. Last, it is pretty quick to compute, compared to other approaches solving recursive and complex differential equations to remove the luminance.

First, we extracted some frames from the video records for each subject. We isolated the head using a Haar cascade classifier trained for face detection, which is available in the OpenCV library. The head area was isolated and resized to a 64 by 64 pixel square. We only considered the grayscale of the image for now. Once this was done, we applied the preprocessing step of the Gradientfaces algorithm. The idea was to compute the Gradientface G of an image I. This corresponds to the orientation of the gradient of an image along the

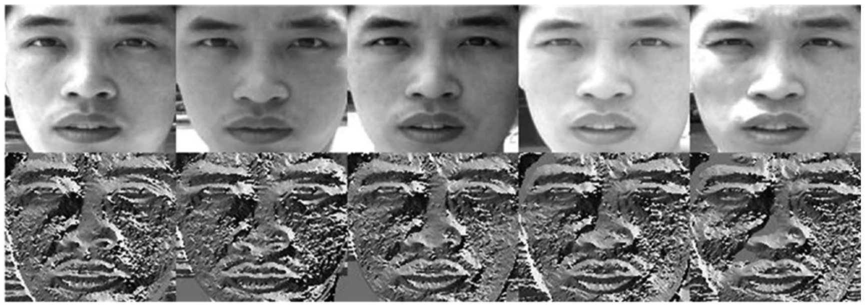


Figure 4.2: Example of original images in different illumination conditions, and their associated Gradientfaces.

two axes x and y. Mathematically, G is defined as follows:

$$G = Arctan(\frac{\partial I/\partial y}{\partial I/\partial x})$$

The image was first smoothed by a Gaussian kernel, then we created two derivative images by applying derivative kernels (in x and y directions) to the smoothed image. Last, the Gradientface was obtained using the two derivative images. This method is relatively simple and quick to apply. Moreover, it is shown to outperform some more complex preprocessing methods for illumination-invariant face recognition. Figure 4.2 shows an example of Gradientfaces images.

The next steps strongly rely on the Principal Component Analysis (PCA) applied to face images [103]. Now, each image is seen as a vector of N dimensions, where N is the number of pixels. The dimensionality of this space is too high to be used for classification, therefore, we need to use a dimensionality reduction tool. If we consider that the set of all possible faces correspond to a very small subset of all the images of dimension N, then it is possible to reduce the dimensionality by finding an appropriate basis to express this subset. The PCA is able to find such a subspace. It actually finds the eigenvectors of the covariance matrix of an input set. The eigenvectors whose eigenvalues are the highest correspond to the more relevant one. Taking the first k eigenvectors (ranked with respect to their eigenvalues) usually allows correct estimation of the input set.
In practice, we transformed all the 2D images into a 1D vector and put them side by side so as to form an $N \times M$ matrix, where N is the number of pixels for each image and M is the total number of images. We then computed the covariance of this matrix and apply

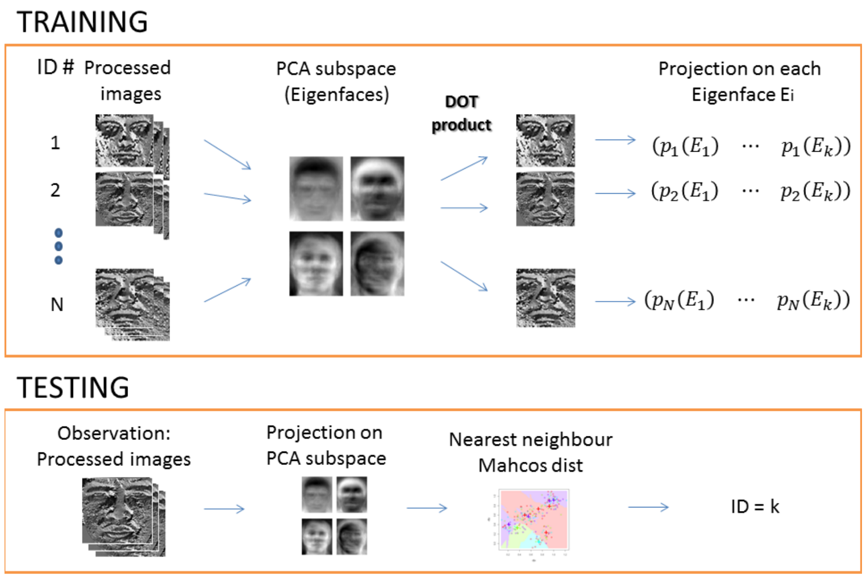

TRAINING
ID #
Processed images
1
2
N
PCA subspace (Eigenfaces)
DOT product
Projection on each Eigenface Ei
$(p_1(E_1) \cdots p_1(E_k))$
$(p_2(E_1) \cdots p_2(E_k))$
$(p_N(E_1) \cdots p_N(E_k))$
TESTING
Observation: Processed images
Projection on PCA subspace
Nearest neighbour Mahcos dist
ID = k

Figure 4.3: General architecture of the face recognition module

the PCA. In our case, we used half of the eigenvectors (or eigenfaces) found this way. We then projected each image on each eigenface in order to express each image in the new basis. It is important to note that we use the PCA only once for all the images (unlike the GMM for audio). The testing was done by expressing the image we want to classify in the basis formed by the eigenvectors, and simply finding the nearest neighbor in the training set in term of cosine distance. For our experiments, we used several test images for each subject. In this case, we computed the mean vector of all the images of the same subject. Figure 4.3 shows the global architecture of the face recognition module.

4.3 Fusion

4.3.1 Overview

By definition, data fusion is the process of integration of multiple data and knowledge representing the same real-world object into a consistent, accurate, and useful representation.

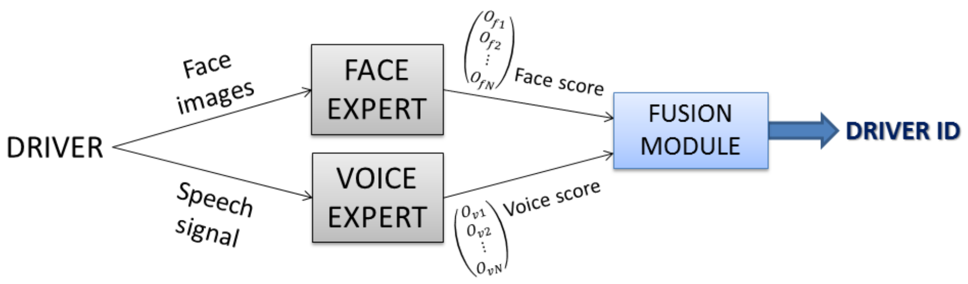


Figure 4.4: General architecture of a fusion module

In our case, the information extracted by voice and audio are complementary, and can be efficiently combined to improve the recognition rate. Fusion can be performed at different stages of the classification. Two popular approaches are feature fusion and decision fusion [104].

Feature fusion consists in putting together features from the different sources and considering them as a unique vector for classification. In practice, this approach may not be appropriate for the following reasons: first, the number of features for each source may not be the same, so we would not attribute the same weight to each expert. Second, too large a number of features may lead to the curse of dimensionality for some classifiers, thus leading to poor performance in the classification. Feature fusion is not very common in audio-visual fusion, given that the natures of the two sources are too different to be efficiently merged.

Decision fusion considers the two modalities as a new feature. Face and voice recognizers are used as experts, rather than classifiers. This means that instead of providing an output as a class, it provides an output as a confidence score. In our case, the confidence score for the voice recognition will be the log likelihood for each observation to belong to a class. For the face, it is the cosine distance of the testing and training subspaces, determined by principal component analysis. The overall scheme of a fusion module is presented in Figure 4.4.

4.3.2 Tested approaches

Normalization

Before applying any of the rules, the first step is to normalize the confidence scores from each expert to make them more consistent. Indeed, the scores for audio and video correspond to two different spaces and cannot be compared as is. Normalization methods can be the min-max rule, the Z-norm, or the tanh transformation. Normalization also exists in probabilistic inference, and consists of making all the probabilities sum up to one for a factor.

Basic decision rules

Among basic decision rules, the sum, product, and max rules are the most common. The sum and the product rule consist of summing (or multiplying) the N elements of each expert together (N is the number of outputs from each expert) and taking the class corresponding to the maximum value.

A heuristic, the weighted sum rule

As was explained earlier, decision fusion has the advantage of providing ways to assess the reliability of the source. Once this is known, it is possible to increase the accuracy of the recognition of the sum rule by assigning a particular weight to each expert. The first approach we will test is a heuristic approach to improving the results of the basic decision rules described earlier. We will consider the weighted sum rule defined as:

$$l(O_s, O_f|\lambda_i) = \alpha_s l(O_s|\lambda_i)_{norm} + \alpha_f l(O_f|\lambda_i)_{norm}$$

with α_s being a weight assigned to the speech scores, and α_f being a weight for the face scores. To find the αs, we need to determine a measure of confidence in each of the experts. Some studies [105] have relied on measures based on the signal itself, for example, the SNR ratio for speech, and image quality for face. In practice, it is not easy to take these metrics into account. Indeed, an image can be high quality, but misalignment or strong facial expressions can highly corrupt the recognition. In addition, using different metrics for each expert may be difficult to normalize and merge. Some attempts have been made to learn the optimal parameters [106], but we will here focus on a much simpler approach.
Our goal is to find a way to express the reliability of each source. Based on the work of Fox *et al.* [17], we will use the following metric for each expert:

$$\xi_m = \frac{l(O_m|\lambda_{n_1})_{norm} - l(O_m|\lambda_{n_2})_{norm}}{mean(l(O_m)_{norm})}$$

where O_m represents a speech or a face observation, and λ_{n_1} (resp λ_{n_2}) represents the class with the highest score (resp the second highest). Based on this score, it is now possible to find a mapping between the alphas and the ξ_m. Fox *et al.* have used an iterative procedure to optimize the α_m, but it seemed to be inappropriate in our case, since the distribution of the scores of face and speech experts were very different, even once normalized. Our best results were achieved simply by taking the α_m as a normalized version of the ξ_m using the min-max rule.

Learning approaches

The last approach we have tried is to use a classifier that requires training. We choose first to use an SVM, as some papers on face or audio-visual identification [97] have claimed to obtain excellent results with this kind of classifier.

We constituted a training set by running the face and voice recognition modules (in classification mode) for each subject twenty consecutive times. Each time, we selected different frames and speech sequences from our database. For a given observation, the output of the face expert (as well as the voice expert) was a vector of twenty output scores (one value for each subject in the training set). Therefore, the training set was a set of 400 vector (20 different subjects $\times$ 20 consecutive times.) each of them containing 40 features (20 subjects $\times$ 2 experts). We then train a twenty-classes SVM, one class for each subject.

4.4 Experimental results

4.4.1 Database and conditions

Most of the literature on audio-visual data fusion uses the XM2VTS audio-visual database [107]. This database consists of almost 300 subjects recorded during four sessions, each of them separated by one month. Each subject is recorded by a microphone and a camera. The subjects has to run through a predefined scenario including talking, moving their face and changing their expressions. This database has the advantage to be a reference database to compare the different audio-visual data fusion algorithms, but it is not appropriate in our case for the following reasons:

- the subjects are filmed in front of a dark screen, so that face detection is straightforward;
- illumination is static and optimal;
- the audio is clean. This avoids many difficulties as MFCC features are extremely noise-sensitive.

For our application, the audio-visual recognition must be efficient inside a vehicle. Therefore, we evaluated our method using two different database. Most of the experiments and evaluation have been conducted on the AVICAR database, therefore the evaluation mostly analyzes results from AVICAR samples, and numerical results are provided for our own database for information. Figure 4.5 shows sample frames from the two databases.

The first of these is the AVICAR database [94]. The AVICAR database consists of one hundred subjects recorded inside a car, uttering letters, numbers, and sentences. The car was equipped with four cameras and eight microphones. They have been recorded in five different driving conditions: The car stationary (IDL), the car at 35 mph with the windows closed (35U), then 35 mph with the windows open (35D), and last, 55 mph with the windows closed and open (55U and 55D). Each driving session was typically six minutes long. For our tests, we only selected twenty subjects, which is more than the usual number of regular drivers of a given vehicle. For each subject, we select 20s of audio, and ten frames of video for training and testing. Of course, we use different driving sessions for training and testing.

The second database was recorded in our driving simulator, either during the data collection phase described in Section 3.4, or during other recording tests. Video was obtained with the infrared camera, and audio was either from the microphone array with no background noise, or the Kinect microphone with car background noise. We have a total of ten different subjects, recorded under different illumination conditions. Three subjects were recorded three months before the data collection, and again during data collection.

4.4.2 Results without data fusion

Before starting evaluation of data fusion, we must have an idea of the performance of each expert.

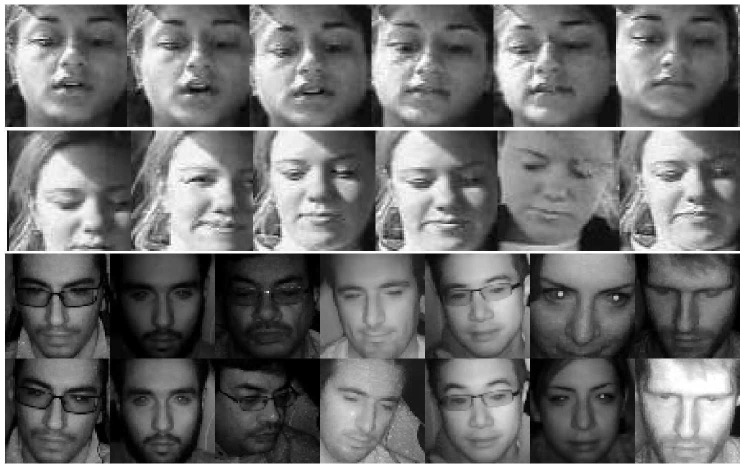


Figure 4.5: Sample data from the AVICAR database (first two rows) and our own database (last two rows)

For Audio, the results are given in Table 4.1. As was explained in the previous section, we used, for training and for each subject a 10 s sequence corresponding to the TIMIT [108] sentences in the IDL condition.

Our first experiment was to test the improvements with a denoising algorithm. We uses for thit the MMSE denoiser filter [109]. The results suggest that this filter can increase the recognition rate, especially when the noise level is not too high. We then changed the length of the testing data to find what the minimum required length for good performances is. In this experiment, we would expect that the more we increase the number of the testing data, the better the results. This supposition is true for very short sequences (a few seconds), but the classification does not change a lot for samples over 10 s. In some cases, the accuracy is even worse, because the speaker sometimes makes long pauses in their speech, thus giving noise a more important weight. In practice, we don't want an identification to be more than a few seconds in length.

For the video, we selected among the four cameras the most frontal face point of view. The training was done with the sequence IDL, corresponding to the phase where the vehicle does not move. The frames we chose for testing and training were randomly selected in the sequences.

Table 4.1: Results of audio recognition on AVICAR

Noise condition	**With and without denoising**		**length of test samples**			
	Without	With	1s	10s	20s	30s
IDL	100%	100%	60%	100%	100%	100%
35U	25%	60%	40%	60%	60%	65%
35D	25%	35%	25%	35%	30%	35%
55U	15%	40%	35%	40%	45%	40%
55D	10%	15%	10%	15%	10%	10%
Average	35%	50%	34%	50%	49%	50%

Table 4.2: Results of face recognition on AVICAR

Noise condition	**# frames for training**					**# frames for testing**				
	1	5	10	20	30	1	5	10	20	30
IDL	50%	95%	95%	95%	95%	70%	95%	95%	95%	95%
35D	20%	60%	70%	60%	65%	65%	60%	70%	65%	70%
55U	30%	55%	65%	65%	65%	60%	60%	65%	70%	70%
55D	40%	40%	55%	65%	65%	50%	65%	55%	65%	60%
Average	35	62.5	71.25	71.25	72.5	61.25	70	71.25	73.75	73.75

The first experiment consisted of varying the number of frames for training. We set the number of testing frames to ten. The recognition rate is presented in table 4.2. As can be seen, the results for only one frame are much below results with a higher number of frames. Between five and thirty frames, there was only a slight improvement. In the second set of experiments, we set the number of frames for training to ten, and we varied the number of frames for testing. The conclusions were pretty much the same as for the first experiment. In practice, we want the identification to be done in a few seconds, so even thirty frames is still acceptable.
It is interesting to note that the face recognition rate does not drastically change depending on the sequence we choose. This suggests that face recognition is pretty context-independent, and is likely to increase the performance of the system, even in very noisy situations. The gap between IDL and the other sequences can be explained by two factors: first, as the car does not move, there is no vibration of the road, and the image is less blurred. Second, when the windows were open, some people were wearing a hat or a hood, making the recognition a little bit more challenging.

Table 4.3: Results of the fusion stage with basic classifiers for AVICAR and our database

Dataset	Face only	Voice only	Product rule	Sum rule	Sum rule (Fox *et al.*)	Our sum rule
IDL	95%	100%	100%	100%	100%	100%
35U	65%	60%	70%	70%	80%	85%
35D	70%	35%	75%	75%	75%	80%
55U	65%	40%	65%	65%	70%	75%
55D	55%	15%	55%	60%	60%	50%
Average	70%	50%	73%	74%	77%	78%
Our database	96.6%	93.3%	100%	100%	100%	100%

Last, experts were only evaluated on our own database, using ten frames and 10s of speech for training and testing. We performed 3-fold cross validation selecting different sequences each time for training and testing. The resutls were 96.6% recognition for face only, and 93.3% for voice only, which is already very satisfying. In the next sections, any fusion strategy has lead to 100% accuracy. Therefore, we add these results into the tables, but we will not comment on them.

4.4.3 Results with data fusion

We intend to prove in this section the efficiency of data fusion for different classifiers. In each case, we use ten frames for training and ten frames for testing, and a 10s speech signal for training and testing. We start our fusion by normalizing all the signals according to the min-max rule.

Basic decision rules and weighted sum rule

First, we apply simple decision rules for data fusion. Then, to increase the performance, we apply the weighted sum rule, taking into account how reliable the source is. We present the results of the rules described in Section 4.3.2, including the method used by Fox *et al.*, as well as our own. Our results are presented in Table 4.3.

The product and sum rule slightly improve the results compared to the best of the two experts. Even with a simple decision rule, the benefits of data fusion can be noticed. The sum rule seems to slightly outperform the product rule, especially in the case of high SNR (35D and 55D). This is because the sum rule is less sensitive to outliers, which are more likely in very noisy conditions.
In addition, the weighted sum rule proposed by Fox *et al.* outperforms the regular sum rule by up to 10%. Once again, adding an indicator of confidence of the source into the data can efficiently improve the data fusion. However, the method described by Fox *et al.* had two drawbacks: first, Fox *et al.* used a triple-expert fusion, so making a mistake in the weighting of one expert had less influence than in the two-expert case. Moreover, metrics of reliability in our case provided very different results for speech and face. The reliability scores were much higher for face than for speech, even with the normalized scores. This is probably due to the different probability distribution of these two experts. Consequently, the optimization proposed by Fox *et al.* gave much more weight to the face expert than the speech expert. Changing the weighting decision rule allowed us to give more importance to the speech expert, thus improving the results, with the exception of only the 55D dataset. In that case, the confidence scores of the speech were not reliable at all, thus leading to poor classification.

Learning approaches

We now want to evaluate the learning approach described in section 4.3.2. We generated one hundred training samples for each of our experiments. The training sets were selected as described in Section 4.3.2. For each situation, we trained two different sets: in the first case, we used faces and voice segments from all different situations (IDL, 35U, 35D, 55U, 55D), including the one we wanted to test. In the second case, we used all the situations but the one we wanted to test. Table 4.4 shows the results. We evaluates our method with an SVM classifier and a nearest neighbor classifier.

The SVM results revealed a huge gap between the training including the current situation and the training without it. One possible explanation is that we used a relatively small number of training data for each class, which might not be enough to build a reliable model for any noise situation. If the training set contain the current situation, a testing vector has a good probability to be quite close to one of the training vectors. This then has a good chance to be correctly classified. If the training set does not contains the current situation, there is no guarantee that the classification will work.

Table 4.4: Results of the fusion stage with learning approaches on AVICAR and our database

Dataset	**Without fusion**		**SVM**		**Nearest neighbor**	
	Face only	Voice only	Same noise	Other noise	Same noise	Other noise
IDL	95%	100%	90%	5%	100%	52%
35U	65%	60%	99%	43%	100%	99%
35D	70%	35%	99%	37%	100%	95%
55U	65%	40%	93%	48%	100%	99%
55D	55%	15%	95%	31%	100%	94%
Our database	96.6%	93.3%	n/a	100%	n/a	100%

The nearest neighbor provides excellent results, even when the current situation is not included in the training set. The only situation where the results are not satisfying is the IDL situation. This can be explained by the fact that there is hardly any noise in the speech signal. The training is done with the IDL situation. Consequently, if the test observation comes from IDL too, the confidence scores will be much higher than scores for noisy conditions. Thus, the output scores in noisy conditions will be very far from those in the IDL scenario.

4.5 Conclusion and possible improvements

In conclusion, we have presented in this chapter an algorithm designed for audio-visual based driver identification. We have tested our method on two different databases. The results have shown that data fusion is able to increase the performance of the system compared to each expert taken separately. For the AVICAR database, basic decision rules improve the classification by between 5 to 10%. More complex fusion strategies, such as the weighted sum rule, can increase the performance by between 10 to 20%. The advantage of the weighted sum rule is that it provides results based on a simple criterion, which can be computed online without any training. Last, decisions based on learning can increase the results even more. Our learning-based classifiers reached up to 100% recognition, even when the quality of the signal was poor. For our own database, each expert score alone is already pretty high. This is because we used a smaller number of subjects, our simulator is in a light-controlled environment, and there is very little noise on the audio data. Any fusion approach has led to 100% accuracy. Consequently, we have managed to design a very efficient, robust, and reliable algorithm to perform audio-visual data fusion.

Possible improvements could be suggested. First, the face recognition module is very dependent on the good detection and cropping of the face. Some subjects were moving a lot during the video sequence, and the rough tracking of the face led to bad cropping. Improving face detection and face alignment is very likely to improve the face recognition as well. In that regard, the Kinect SDK provides a powerful face tracker, able to determine face orientation. This could be useful for accurate alignment. Besides this, the 3D face model provided by the Kinect could lead to more efficient face recognition approaches based on 3D modeling.

For speech recognition, we currently use MFCC features, which are very sensitive to noise. To increase accuracy, we can either look for a better speech denoising algorithm, or look for other features, less sensitive to noise. Moreover, the GMM classification is far from being the most efficient in practice. A more sophisticated approach involves Hidden Markov Model, or even conditional random fields.

The fusion module has shown excellent results with a simple nearest neighbor classifier, but it might be interesting to look at ensemble learning techniques if we plan to use driver identification in a more complex situation. We could, for example, use an AdaBoost classifier. Indeed, AdaBoost is able to find the relevant features and to weight them accordingly in order to increase the classification performance.

In this chapter, we have tackled the problem of audio-visual data fusion for driver identification. Person identification inside a car involves a number of difficulties compared to the common indoor studies found in the literature. We have used the AVICAR database, recorded inside a car, to tackle the challenges associated with this environment. We have used Gradientfaces for face recognition, GMM for voice recognition, and different types of fusion modules to find the optimal one. The fusion based on a nearest neighbor classifier has shown the best accuracy. We have obtained classification results close to 100% using data fusion, compared to 15% to 70% using single experts only. We also have integrated this algorithm into our driving simulator, and validated its robustness based on audio and *video sequences recorded on our platform.*

Chapter 5

Driver Fatigue Detection

In this chapter, we propose a simple fatigue detection framework based on an infrared camera. The main feature for detecting fatigue using computer vision is the computation of the percentage of eye closure (PERCLOS) within a period of time. For that, we need to develop an efficient gaze tracker and an eyelid activity detector. When eye closure is accurate enough, we can compute PERCLOS and emit a warning when it is too high. Most of the work in this research was on the eye-tracking strategy.

This chapter is organized as follows. In Section 5.1, we explain our motivations for doing fatigue detection, and provide a review of the literature. In section 5.2, we highlight some basics about particle filters, and we outline our pupil detection method and our particle filter design. In section 5.3, we explain how to obtain eyelid activity and compute PERCLOS for fatigue detection. Experimental results are presented in section 5.4. Last, section 5.5 gives concluding remarks and possible improvements.

5.1 Motivations

A lot of research in human machine interaction systems in intelligent vehicles have been on detecting driver fatigue, either at an early stage, or during extreme drowsiness. The reason for this is that driver fatigue is one of the main causes of vehicle accidents on the road. The NHTSA estimates that about 20% of road accidents are fatigue-related [110]. Even more alarming, 60% of heavy trucks fatal accidents are due to the driver falling asleep while driving. Statistically speaking, Klauser *et al.* [4] have shown that driving when drowsy can increase crash risk up to six times as compared to normal driving. It is therefore of major interest to study driver fatigue, and to design an efficient methodology able to detect drowsiness.

Among strategies described in Chapter 2 to detect fatigue, we want to restrict ourselves to computer vision-based approaches. We want to use our infrared camera as a first step and progressively integrate additional or other types of sensors. In the literature, dozens of features have been used to detect fatigue using computer vision. The most popular are PERCLOS. (Percentage of Eye Closure), yawning frequency, nodding, fixed face, and fixed gaze. However, only PERCLOS has been found to be a valid physiological sign of driver fatigue [62], and is often said to be the more reliable feature. Therefore, we restrict this study to PERCLOS only. We could of course add more fatigue-related features later on.

PERCLOS can be implemented in three steps. First the driver's eyes need to be efficiently detected and tracked. Then, we need to determine for each frame whether eyes are open or closed. Last, we need to compute the percentage of time when the eyes are more than 80% closed based on the eyelid position, during a time window of just a few minutes. Besides this, computing PERCLOS in a car environment is a challenging task, as the eyes' appearance can change significantly depending on the driver, their face position, and the outdoor lighting conditions.

Driver eye tracking is a very active field. Efficient commercial products, such as FaceLAB [15] or Smarteye [111], already exist for this task, although they often need expensive devices, significant calibration process, and can show good accuracy only in constrained and controlled environments. Most eye tracking algorithms are either based on pupil/iris detection, or eye template matching. For example, D'Orazio *et al.* [81] have presented an iris detection method using improved Hough circles. This approach works well for frontal faces, but the round shape constraint makes the method sensitive to face rotation. Valenti and Gevers [112] have proposed a pupil localization and tracking based on isophote curvature. Although very efficient, this method requires high resolution images. On the other hand, template matching approaches rely on a training set. For example, Wu and Triverdi's algorithm [113] was based on two cooperative particle filters, using tensor PCA to detect both open and closed eyes. Kim *et al.* [114] have used AdaBoost and multi-layer perceptron to successively detect eyes, eye tails, and iris. Choi et al [115] have used a scanning approach using AdaBoost and grouping to detect the eyes' positions. Other methods use several facial features to predict the expected eye location: Senaratne *et al.* [75] have based their algorithm on Landmark Model Matching. Smith *et al.* [76] first detected the mouth corners to estimate the position of the eyes. A different strategy consists of using near infrared imaging for pupil detection. For example, Zhu and Ji [58], have based their pupil detection methods on the alternation of the dark and bright pupil effect. They also used a Kalman filter for tracking, later improved by a combination of meanshift and Kalman filtering [116]. Based on dark and bright pupil effect alternation, Bergasa *et al.* [60] have later used an improved Kalman filter with an adaptive search window. These methods are robust to illumination variation, but remain sensitive to facial rotation. In addition, they require a high frame rate and a sophisticated device to alternate between bright and dark pupil frames.

5.2 Eye tracking

In this section, we present an efficient real-time eye tracking system for driver monitoring based on an improved particle filter. Our method has the advantage of working with a low-cost camera device with a low frame rate. In addition, the use of multiple distributions in our particle filter allows us to be more flexible, depending on the situation, and to overcome facial rotation or rapid shape motion, which are limitations present in most current approaches. We also use a single particle filter to track both eyes at the same time. This has the advantage of running a single algorithm instead of two in parallel, and to cooperatively use the location information of the two eyes to improve and reinforce the predicted location of the eyes in future frames.
The overall algorithm is presented in Figure 5.1. In this section, we first give theoretical background on particle filters, then we explain in details each step of the algorithm.

5.2.1 Theoretical background

Particle filter, or condensation algorithm, was presented by Isard and Blake [117] for tracking curves in clutter. Today, it is used in many tracking applications as an alternative to the Kalman filter. The main idea of the algorithm is to estimate the state of an object at time t, given all the past states and observations.
More precisely, given a target state sequence $X_{0:t}$, representing all states between time 0 and t, and an observation $Z_{0:t}$ (both are stochastic processes), we want to estimate the probability of the state x_t at time t given all the observations:

$$p(x_t|Z_{0:t}, X_{0:t-1}) \tag{5.1}$$

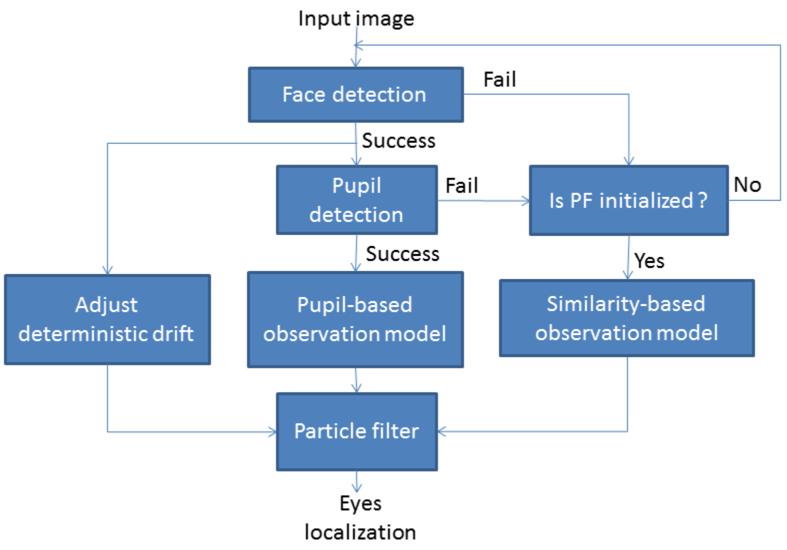


Figure 5.1: Particle filter algorithm

To make this estimation possible, we need to make different assumptions in order to simplify the equation. The first one is to consider that the states' dynamics can be approximated by a Markov process. We obtain the following simplification:

$$p(x_t|Z_{0:t-1}, X_{0:t-1}) = p(x_t|x_{t-1}) \tag{5.2}$$

The second one is to consider that the observations are independent from each other given the states $X_{0:t}$. This assumption gives us the following equation:

$$p(z_t|Z_{0:t-1}, X_t) = p(z_t|x_t) \tag{5.3}$$

Now, given these simplifications and using Bayes' rule, we obtain the resulting equation:

$$p(x_t|Z_{0:t}) \propto p(z_t|x_t) \int_{x_{t-1}} p(x_t|x_{t-1}) p(x_{t-1}|Z_{0:t-1}) dx_{t-1} \tag{5.4}$$

This equation tells us that the probability we are looking for is proportional to a first term $p(z_t|x_t)$, called the observation model, multiplied by a cumulative factor depending on the previous probabilities, and a second term $p(x_t|x_{t-1})$, called the state transition model. In practice, the state transition model will be estimated as the sum of a deterministic component (called later the deterministic drift), and a random component (or noise). The observation model is rarely associated with a close form; therefore, approximations must be made to estimate it. The Kalman filter considers this distribution as to be Gaussian, which is enough in many situations. On the other hands, the particle filter uses sampling techniques to obtain an approximated distribution without any prior assumption.

More precisely, particle filters based on sampling importance resampling uses N particles $s_t^{(i)}$ associated with a weight $\omega_t^{(i)}$ to approximate the expectation of the target state given the observations.

$$E[x_t|Z_{0:t}] = \int x_t p(x_t|Z_{0:t})dt = \sum_{i=1}^{N} \omega_t^{(i)} s_t^{(i)} \tag{5.5}$$

$\omega_t^{(i)}$s are the weights, initialized to $1/N$ and iteratively updated using the following rule:

$$\omega_t^{(i)} = \omega_t^{(i-1)} \frac{p(z_t|s_t^{(i)})p(s_t^{(i)}|s_{t-1}^{(i)})}{\pi(s_t^{(i)}|s_{t-1}^{(i)}, Z_{0:t})} \tag{5.6}$$

Where π is the importance distribution. In our case, and in most of the situations, we approximate π by the state transition, thus leading to the simplified equation of the weights:

$$\omega_t^{(i)} = \omega_t^{(i-1)} p(z_t|s_t^{(i)}) \tag{5.7}$$

The particle filter is then composed of four steps:

1. Determine the predicted position of the N particles based on the distribution π
2. Determine the probability of the observation given each particle
3. Estimate the weights $\omega_t^{(i)}$ and normalize
4. Take the estimated target state as the expectation (given in equation 5.5)

5.2.2 Application for eye tracking

Particle states

The states in the particle filter correspond to random variables describing the object to be tracked. In computer vision, the states are typically coordinates, sizes, or velocities. For eye tracking, the most common states are pupil center coordinates, pupil radius, and sometimes velocity of the pupil. Moreover, they use a different particle (or Kalman) filter for each eye, independent of one another. In our case, we use a four-state particle filter consisting of the coordinates (x and y) of the left and right eyes:

$$\mathbf{x} = (x_{lx}, x_{ly}, x_{rx}, x_{ry}) \tag{5.8}$$

We do not consider the velocity as a state, but store the location of the previous states, and use that to predict the position of the next frame, as will be explained in Section 5.2.4. This method is equivalent to using the velocity as a state parameter. Moreover, our method is applied in the case where the distance between the subject and the camera is almost constant; therefore, we do not need the pupil size information, although this feature could be easily added to our algorithm.

We chose to track the two eyes at the same time to take advantage of their position with respect to one another. Indeed, the location and motion of the two eyes are highly correlated as they belong to the same solid object, the head. Finding the location of each eye given the other one at the same time offers us a better prediction possibility.

Multi state transition model

The particle filter uses a state transition model $p(x_t|x_{t-1})$ to predict the location of the eyes in the next frame. This state transition model can be seen as the addition of a deterministic drift depending on the previous state and a random noise:

$$\mathbf{x_t} = f(x_{t-1}) + g(\mu) \tag{5.9}$$

Where μ is a Gaussian, zero-mean random variable. Without any knowledge, a simple assumption is to consider that the position at frame t is the same as that at frame $t-1$ plus a little additional noise. This is true as long as the object does not move too rapidly.

This assumption can be further improved by determining trajectory parameters given a training sequence. A common method to determine these parameters is to represent the state position as an auto-regressive model of order 1:

$$\mathbf{x_t} = A \times x_{t-1} + B \times \mu \tag{5.10}$$

Where A and B are matrices to determine. We use this approach, but we add a second transition model to make our algorithm more robust. This second one is based on the face location, and is able to handle those situations where the face is moving very rapidly. Details and discussions about these methods are provided in Section 5.2.4.

Multi-observation model

The observation model $p(z_t|x_t)$ is a distribution based on a measurement, or observation. In computer vision, this measurement is typically an estimation of the location of the object and can have very different forms. For example, the observation in [58] is the position of a set of pupil candidates, if any. The corresponding observation model looks like this:

$$p(z_t|x_t) \propto \sum exp(\|x_t - z_t\|^2) \tag{5.11}$$

In [113], the observation is a measure of similarity between an area of the image at a given location, and a subspace constructed with a training sequence. Mathematically, the observation model has the following form:

$$p(z_t|x_t) \propto D(x_t, S(z_t))) \tag{5.12}$$

where D is an arbitrary distance, and $S(z_t)$ is a subspace constructed with offline observations.

The advantage of the particle filter compared to a simple raw observation is the capacity to find a trade-off between the confidence in the observation and the position prediction given the past frames. In the case where the observation is missing, incomplete, or false (which occurs in cases of blur, occlusion, or glint), the particle filter can estimate for a few frames the location of the object until it reappears. However, this estimation often deviates after a few frames.
Most methods rely on a single observation model, making their algorithm efficient only

in specific situations. We have increased the robustness of our particle filter by switching between two distributions depending on the situation. If the pupil is detected, we use a Gaussian-based distribution, which relies strongly on the pupil position. If the pupil is not detected, we use an appearance-based distribution, which is less accurate but much more robust. This method allows us to find the location of the eyes in cases of facial rotation, bad illumination conditions, or when the face detection algorithm has failed.

5.2.3 Face and pupils detection

We use a face detection algorithm based on boosted cascades using Haar wavelets features. This method has been proposed by Viola and Jones [118], and Lienhart and Maydt [119], and is fully implemented in the Open Source Computer Vision Library (OpenCV). Using frontal and profile face training sets provided by OpenCV, the face is successfully detected in almost every frame.

As described in Section 3.2.2, our camera has been designed such that infrared diodes are mounted all around and close enough to the sensor to create the bright pupil effect. Thus, the pupil looks extremely bright, and has a strong contrast with the iris. Mathematically, this corresponds to a significant local maximum of the image. Bright pupil effect is commonly used in near-infrared imaging, because it makes pupil detection easier, and requires less complex devices than approaches based on dark and bright pupil alternation [58]. Moreover, apart from the pupil detection step, the algorithm can be applied to visible images too.
Pupil detection is done as follows in the detected face area. First, local maxima are detected in the image. A small subset is selected for iris candidates based on their sharpness and intensity variation. Then, to further reduce the number of candidates, we use a two-class SVM classifier (eyes and non-eyes). The SVM is trained with around 500 patches of eyes and non-eyes. We use an RBF kernel with $\gamma = 0.1$. For each candidate, we create a patch centered on the candidate location, and reject it if SVM classifies it as non-eye. The SVM has around 90% accuracy, thus filtering out most wrong candidates. In most cases, SVM filtering provides two correct candidates, but if more than two candidates are found, we determine the best pair of candidates based on their distance, the angle they form with the horizontal axis and their position in the face location. If a pair of candidates is found, it corresponds almost every time to the right pupil locations.

5.2.4 Particle filter design

Our particle filter is composed of a hundred particles of four states describing the eyes:

$$\mathbf{x} = (x_{lx}, x_{ly}, x_{rx}, x_{ry}) \tag{5.13}$$

which correspond to the left eye (x_{lx} and x_{ly}) and right eye (x_{rx} and x_{ry}) coordinates. We initialize the particle filter whenever a pair of pupils has been detected. The tracking stops if a face is not detected in more than ten successive frames, and restarts as soon as pupils are detected again. We chose to track the two eyes at the same time to take advantage of their position with respect to each other. As they both belong to the same solid object (the face), their position and trajectory are highly correlated. We discuss how to use the joint information of the eyes' location in Section 5.2.5.

5.2.5 Multi-state transition model: $p(x_t|x_{t-1})$

The state transition model is used to predict the location of the eyes in the next frame. This state transition model can be seen as the addition of a deterministic drift from the previous state (i.e., the most probable location of the object) and a random noise (uncertainty about this location). Based on this decomposition, we model the state transition in two different ways.

State transition model based on an auto-regressive model

This commonly used approach approximates the state transition as a first order auto-regressive model:

$$\mathbf{X}_t = \bar{\mathbf{X}} + A(\mathbf{X}_{t-1} - \bar{\mathbf{X}}) + B\mathbf{N_t} \tag{5.14}$$

where $\mathbf{X}_t = (\mathbf{x}_t, \mathbf{x}_{t-1})^T$, $\bar{\mathbf{X}}$ is the mean value of $\mathbf{X}_t$, and $\mathbf{N_t}$ is a Gaussian unit random noise vector. A defines the deterministic drift, and B controls the noise parameters. In practice, A is estimated using the multidimensional Yule-Walker equation [120] or the method described in [121], given a training sequence. B is estimated as the covariance of the deterministic prediction error (residual).
The choice of four states instead of two for the particle filter is crucial here, because the coordinates of the two eyes are highly correlated. A and B will contain the correlation information and make the prediction more accurate. When determining the new position

of the particles, the auto-regressive model will make the new position more realistic, and avoid divergence or convergence of the eyes.

State transition model based on face velocity

The eyes' displacement is highly correlated with face motion, especially in case of translation motion; therefore, the face position gives us an idea of the eyes' position according to the following rule:

$$\mathbf{X}_t = \mathbf{X}_{t-1} + \mathbf{V}_t + B\mathbf{N_t} \tag{5.15}$$

where $\mathbf{V}_t = (\mathbf{v}_t, \mathbf{v}_t, \mathbf{v}_{t-1}, \mathbf{v}_{t-1})^T$, $\mathbf{v}_t$ is the difference between the face center at time t and at time $t-1$ (i.e., velocity).

The two state-transition models are complementary. While the first one is robust to strong face rotation, the second one will handle the case of rapid translation movements. As we do not know which model is more appropriate in future frames, we randomly assign each particle one of the two state-transition models. A particle with an inappropriate transition model is likely to be assigned a low weight. Then, resampling is done at each frame according to the particle weights, and will change the particle states according to higher-weighted particles.

5.2.6 Multi-observation models: $p(z_t|x_t)$

The observation model is a distribution based on a measurement, or observation. For eye tracking, this measurement can be an estimation of pupil location (as in [58]), or a template matching-based score (as in [113]). In order to provide a trade-off between accuracy and robustness, we use two different observation models and switch from one to the other depending on the situation.

Pupil-based observation

If a pupil has been detected, we consider a density probability function giving a lot of weight to this observation. The closer the particle to the observation, the higher the probability:

$$p(\mathbf{z}|\mathbf{x}) = \frac{1}{\alpha}(1 + \frac{1}{\sqrt{2\pi\sigma^2}}\sum_{i\in l,r} exp(-\frac{\|x_i - z_i\|^2}{2\sigma^2})) \tag{5.16}$$

with z_l and z_r the left and right pupil coordinates observed. α is a normalization factor we do not need to consider here, and σ is the variance of the error between observation and true position. It was set to 2 in our implementation, although it has little influence on the performance.

Similarity-based observation

If this pupil has not been detected, we use an observation model based on similarity. For this, we construct a database of around 2000 eye patches. We pre-process them using normalization techniques, and we use PCA to reduce the dimensionality, and obtain a subspace S of 20 dimensions. Now, for each particle, we crop the area around the particle to obtain a patch for each eye. Let us call, for example, $\mathbf{V}(\mathbf{z_l}, \mathbf{x_l})$ the vector corresponding to the left eye patch. After pre-processing, we use distance-from-feature space (DFFS) and distance-in-feature-space (DIFS) to approximate the likelihood density function of a Gaussian distribution, as explained by Moghaddam and Pentl [122].

$$p(\mathbf{z}|\mathbf{x}) = DFFS(\mathbf{V}(\mathbf{z_l}, \mathbf{x_l}), S) + DIFS(\mathbf{V}(\mathbf{z_l}, \mathbf{x_l}), S) \tag{5.17}$$

where DFFS represents the reconstruction error of the point in the subspace:

$$DFFS(\mathbf{V}(\mathbf{z_l}, \mathbf{x_l}), S) = \|\mathbf{V} - Proj(\mathbf{V}, S)\| \tag{5.18}$$

and DIFS is the Mahalanobis distance between the projected point and the origin of the subspace.

$$DIFS(V(x, z), S) = \sqrt{(\mathbf{V} - \mu)^T \Sigma^{-1} (\mathbf{V} - \mu)}^2 \tag{5.19}$$

where μ is the mean value of the subspace and Σ is its covariance.

5.3 Application to driver fatigue detection

5.3.1 Eye closure detection

Several eye closure strategies have been adopted in the literature for eye closure detection. The simplest approach considers color variation between successive frames [123], [124], or template matching [125]. These methods are efficient under the condition that eye tracking is extremely stable and reliable, and that the subjects are not moving their face too much.

More sophisticated methods involve optical flow analysis of the eye region [126], [127]. Although more robust than the simple methods described before, this approach requires a high frame rate and remains sensitive to rapid face motion. It is therefore not applicable for us. The last common strategy is to use classifiers based on eye templates, and decide frame by frame whether the eye is open or closed [128], [129]. We chose to take this approach.

Once an eye is detected based on the particle filter response, we take the two patches associated with the particle of highest probability. We pre-process the patches using min-max normalization, and collect a set of 1000 eye templates either open or closed. We label them manually. When the eyes were half-closed, we decided their class based on the definition of the PERCLOS described in the next section. In other words, if more than 80% of the iris is visible, the eye is considered open.

We then train an SVM classifier. The best results were obtained using an RBF kernel with $\sigma = 0.3$. Note that our dataset was balanced, i.e., there were equally as many open- and closed-eye samples. Moreover, to increase the accuracy, we have trained an SVM dedicated to left eyes by horizontally flipping right eyes before evaluating them.

5.3.2 PERCLOS computation

For PERCLOS calculation, an eye is defined as closed if the eyelid surface is covering more than 80% of the pupil. As shown in figure 5.2, PERCLOS would formally be computed by considering frames between t_2 and t_3 as closed. Our camera resolution neither has the necessary resolution nor a high-enough frame rate to accurately detect that limit. Therefore, we rely on the labeling we have done during SVM training, and consider that statistics over a long time window attenuate misclassification, and will eventually converge to the true PERCLOS value.

To improve eye closure recognition, a few strategies are adopted. First, the decision of whether the eye is closed or not is obtained by summing up the output of the two eyes. In the case where one of the eyes is not clearly visible and misclassified, summing up the two scores improves the number of correct decisions. Moreover, we systematically tested each eye template with the SVM classifier defined in Section 5.2.3, to make sure we are testing an eye. If the SVM considers the template a non-eye, we disable the output of the SVM blinking for this eye and rely solely on the other eye. If both eyes are detected as non-eyes,

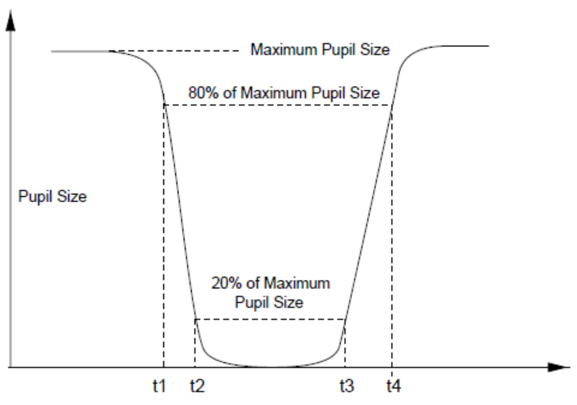


Figure 5.2: Eye closure duration: Time between t_2 and t_3 (source: [2])

we simply discard the SVM output for this frame.

PERCLOS is often computed within a few minutes. As our sequences are relatively short, we consider a two-minutes sliding window, and iteratively recompute the PERCLOS value for each frame.

5.3.3 Integration

The eye tracking and PERCLOS calculation algorithm were written in C++ and integrated into the interface. When fatigue detection is on, PERCLOS is re-evaluated for each frame. If the percentage reaches a certain threshold (0.5 in our case), an audio alarm is triggered and a pop-up window is displayed. Drivers are therefore aware of their high state of fatigue and is encouraged to take a nap.

5.4 Experimental results

5.4.1 Data collection

The evaluation was done using the driving simulator, but using a different dataset than the one described in Section 3.4. The main purpose was to evaluate the efficiency of our eye tracker. Therefore, we decided to carry out our experiments so as to have a wide panel of conditions and commonly-used benchmarks, thus demonstrating the good performance of

Table 5.1: Numerical results for eye tracking

	% time in each mode				**% accuracy for each step**					
Subject	face detected	pupil based	similar. based	not active	Step 1	Step 2	Step 3	Step 4	Step 5	**Overall**
1	97.3	61.69	38.04	0	100	95.08	98.55	100	96.53	**97.35**
2	100	24.09	75.92	0	97.21	97.54	97.86	92.97	94.97	**95.74**
3	93.97	9.20	80.77	4.56	100	78.51	83.27	100	100	**93.22**
4	100	46.52	53.48	0	97.09	89.66	97.15	97.39	91.12	**94.53**
5	83.41	43.66	41.27	12.36	100	81.76	92.91	100	66.71	**85.42**
driver	99.21	88.98	7.76	0.77	n/a	n/a	n/a	n/a	n/a	**99.34**

our approach. Five subjects (called 1 to 5 in Table 6.4) of different genders and nationalities were recorded during a session of around two minutes in length. They were asked to follow a sequence of movements as given below:

1. Move the eyes without moving the head
2. Turn the head left, right, up, and down, slowly then rapidly
3. Move the head doing translation movements, slowly then rapidly
4. Blink at different frequencies without moving the head
5. Blink while moving the head.

In addition, a seven-minutes video sequence was recorded with one of the subjects driving in the simulator (called 'driver' in tables). This allowed us to verify that our algorithm was working properly in driving conditions.

5.4.2 Eye tracking results

Figure 5.3 provides some examples of video frames, providing qualitative results. The algorithm remains efficient even in extreme situations where the eye is hardly or no longer visible.

Table 6.4 presents numerical results for each subject. First, we give for each video sequence details about the algorithm itself. As can be seen, face detection is extremely accurate

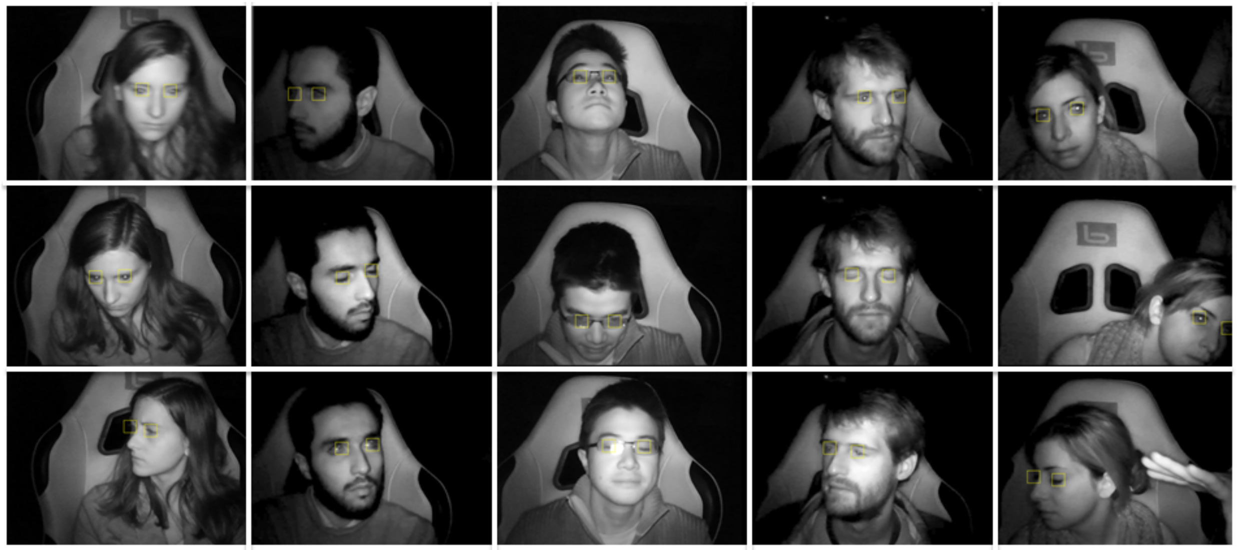


Figure 5.3: Example of tracking results for different drivers and different pose and illumination conditions.

and the particle filter is rarely inactive. Depending on the subject, either pupil-based or similarity-based mode is more used. This is because pupil detection is very accurate, but sometimes has too many constraints to detect both eyes at the same time.

When looking at the accuracy results for each step described in Section 5.4.1, one can notice that detection is much better for steps 1,3, and 4. This is because in these cases, the subjects were asked not to turn their head, making the tracking more pupil-based than similarity-based. Pupil-based observation requires relatively good visibility of the two pupils. It fails in cases of significant face rotation, blinking, partial occlusion, or reflection off glasses. In thit case, similarity-based observation is used, but these situations are extremely challenging. Therefore, similarity-based observation may successfully track the eyes in extreme situations for a few frames, but is very likely to drift away if the subject does not comes back to a normal situation.

The percentage of frames where eyes are correctly detected (called overall accuracy) is very high, except for subject 5. This is because subject 5 was moving and turning his head much more significantly than others. His face was sometimes partially out of the frame of the camera, and face was not always correctly detected (see % face detected column). Subject 3 was wearing glasses. In this case, reflection off the glasses sometimes leads to false detection of the pupil, making the tracking more difficult or inaccurate. Moreover, the pupils appear less bright when wearing glasses, so pupil detection is much more likely to

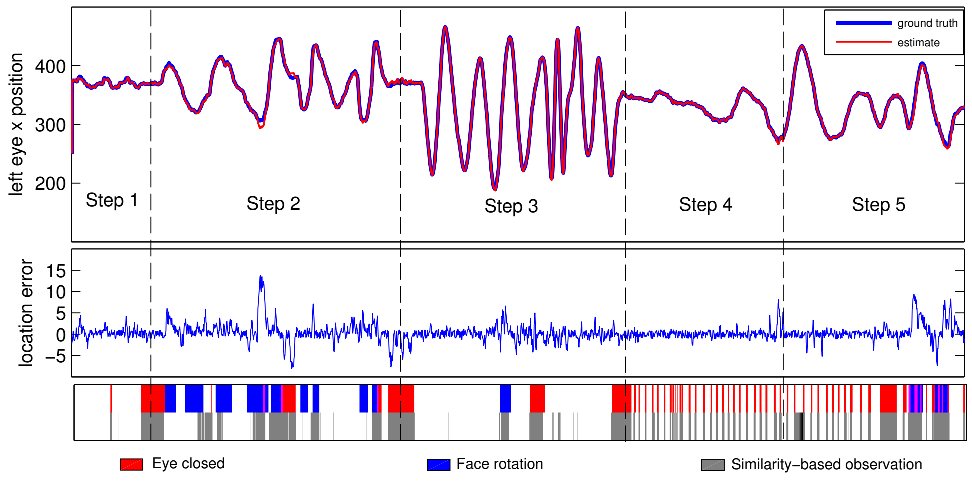


Figure 5.4: Detailed performance for subject 1. X-axis is time.

fail. This explains the small percentage of time of pupil-based observation for this subject. Nevertheless, the overall accuracy obtained suggests that the algorithm works fine despite glasses. Last, the excellent performance during the driving session is encouraging, as our algorithm is to be used during driving conditions.

In figure 5.4, we provide some details about the video sequence for subject 1. The five steps are represented. At the top, the left eye x coordinate is represented as a function of time. Particle filter and ground truth are extremely close, showing the efficiency of our method. In the middle, distance between ground truth and particle filter position is represented. The error is most of the time less than five pixels. Below is a representation of 'challenging' frames (closed eyes in red, and strong face rotation in blue), and frames where the similarity-based observation model was activated (white means pupil-based observation). A strong correlation can be deduced between the similarity-based mode and challenging situations, mainly because the pupil is hardly or not detectable in these cases.

5.5 Conclusion and possible improvements

We have designed a fatigue detection system based on computer vision. We have based our fatigue measure on the percentage of eye closure, which is currently the most reliable feature for that task using a vision system. For that, we have first developed an algorithm for tracking eyes based on particle filtering, which is robust to face rotation, eye occlusion, and illumination variation. Results show that the algorithm is able to handle those difficulties in most cases. Moreover, in the particular case of a driving session, the eye tracking has shown very good results. We then have based our blink detector on an SVM classifier, trained with open- and closed-eyes templates. Although simple, this approach has shown good performance, and allows accurate computation of the PERCLOS.

The method could be improved at several levels: pupil detection is still far from perfect, and could be improved for a better detection in more situations. Moreover, the pupil detection is based on the bright pupil effect, but a more general detection strategy could open doors to color imaging, making the eye tracker usable with any camera. Another limitation of the system was found in the case where the driver was wearing glasses. Indeed, the performance was shown to be reduced in that case, especially because of false detection of the pupil due to reflections from the glasses. Further investigation could be done to limit as much as possible this situation. In addition, the blink detection is done on a frame by frame basis, and no temporal refinement is done. We believe that a good improvement in blink detection could be achieved, if misdetections were filtered out by temporal considerations.

In a future work, testing the system on truly tired drivers would be interesting. It could first confirm the efficiency of the method, and second, show how reliable PERCLOS is for this task. Last, eye tracking is often a starting point to many facial-based features. Other statistics such as blink duration or blink frequency could be easily added to the system and fused to improve fatigue detection. Later, gaze tracking and fixed gaze detection could be integrated, and further still, features involving mouth or face orientation would provide a rich and reliable fatigue detection system.

This chapter has presented a driver fatigue detection strategy based on a single infrared camera. We have first developed a new approach for driver eye tracking and blinking, based on an improved version of a particle filter. We have used two different state transition models' and two different observation models' distributions to adapt the tracking depending on the situation. This approach is robust to significant face rotations and partial occlusion. Then, based on the estimated eye position, we have developed an eye blink detector and computed the percentage of eye closure, found to be the best feature in vision systems for fatigue detection. Evaluation was done on five different people executing a challenging sequence of movements and blinking patterns. Results show that our method is robust to face rotation, partial occlusion, and illumination variation.

Chapter 6

Driver Distraction Level Assessment

This chapter presents an efficient driver distraction detector and recognizer based on Kinect data. Most of the effort was in extracting good features from different parts of the face and the body. More precisely, we extracted arm position, face orientation, facial expressions and eye behavior. We then combine all the extracted information to recognize the type of distraction of a driver. The result of this work is a module that could be integrated with the interface, and used later on to monitor dangerous driving.

This chapter is organized as follows: Section 6.1 explains our motivations and the contributions of this work. Section 6.2 describes the feature extraction principles based on Kinect color and depth video streams. Section 6.3 explains how to combine all the information to infer driver distraction. Experimental results are given in section 6.4, and section 6.5 closes the chapter with concluding remarks and possible improvements.

6.1 Motivations and contributions

6.1.1 Motivations

Alarming statistics about distracted driving can be found on the official US government website about distracted driving [41]. In 2010, 18% of injury crashes were distraction-affected. 3331 people were killed in 2011 in crashes involving distracted drivers, and distraction is responsible for 11% of fatal crashes of drivers under the age of twenty. Those statistics are even more worrying as the number of possible distractions within a car keeps increasing. The boom of new technologies has brought drivers dozens of new reasons to get distracted.

The influence of distraction on drivers performance has been widely studied [130], [131], [132], [133], [5], and interesting facts have been deduced: cell phone use represents 18% of distracted fatal driver accidents in America. Indeed, cell phone conversations create an important cognitive distraction, thus reducing the brain activity related with driving by 37% (which may be worse than drinking alcohol). Handsfree cell phones have not been found particularly safer than hand-held use. More importantly, text messaging requires visual, manual, and cognitive attention at the same time, making it the most dangerous distraction. It was found that text messaging takes the driver's eyes off the road for 4.6 seconds, which is sufficient to drive the length of a football field completely blind. The crash risk when text messaging is twenty-three times worse than driving with no distraction.

All these facts suggest that drivers should be aware of the risk, but it is also the car manufacturer's responsibility to offer intelligent assistance tools to avoid driver distraction, and to limit crash risks. This issue is still an open problem, as the variety of actions, the differences between drivers and outdoor conditions make this task extremely challenging.

In that regard, it is of great interest to design a system able to detect driver distraction and emit a warning to keep the driver aware of any dangerous behavior. Nevertheless, it is also interesting to detect what type of distraction the driver is subject to. Indeed, determining the type of driver distraction provides higher level information than just a "level" of distraction. It could be used for number of applications related to intelligent transportation systems. For inter-vehicle communication, providing to the other drivers a sense of which unsafe action the driver is taking can be more valuable than telling them how unsafe it is

6.1.2 Contributions

Our approach aims at determining first whether a driver is distracted or not, and, if distracted, recognizing the type of distraction. Based on computer vision techniques, we propose four different modules for feature extraction, focusing on arm position, face orientation, facial expression, and eye behavior. We propose two strategies to combine the output information from each module: an AdaBoost classifier with temporal smoothing, and a Hidden Markov Model.

Among state of the art techniques, most focus on driver inattention, including features for driver fatigue and distraction in general. To our knowledge, no serious study has been done on distraction only, such as trying to detect the type of action the driver is accomplishing.

Our system is based on a Kinect sensor. Originally conceived for entertainment and video game applications, it has quickly received a lot of attention in the computer vision community. Indeed, not only was it the first low-cost depth sensor for general public, but it also came with a very polished SDK, giving developers a large field of possibilities. For example, the SDK provides a quite efficient skeletal tracking algorithm and tools for gesture recognition. In our case, the RGBD (RGB-depth) data is very helpful for driver segmentation, as well as for face detection and tracking. To our knowledge, only Li *et al.* [61] have published work making use of the Kinect for car safety applications.

As most existing systems rely solely on the driver's face behavior, we also use gestures to help us is our inference task. Unlike traditional approaches, our sensor is placed in such a way that the driver's upper body is visible. Thus, we can extract driver arm position and motion. This feature will be of major help in determining driver distraction.

6.2 Feature extraction

This section explains how to extract knowledge about each body and face components. We divide our task into four independent modules. More precisely, we aim to obtain (1) arm position estimation, (2) face orientation, (3) facial features - called animation units (AUs) - such as mouth shape and eyebrow raising, and (4) gaze estimation and eye closure. Each module uses depth data, color data, or both. They are fused later on via different strategies to determine the type of distraction. Figure 6.1 presents the general architecture of the method. The next sections describe in detail the realization of each module.

6.2.1 Arm position

A lot of effort has been given by Microsoft to providing an efficient skeleton tracking to help developers in gesture recognition, both for the entire and the half body (in seated

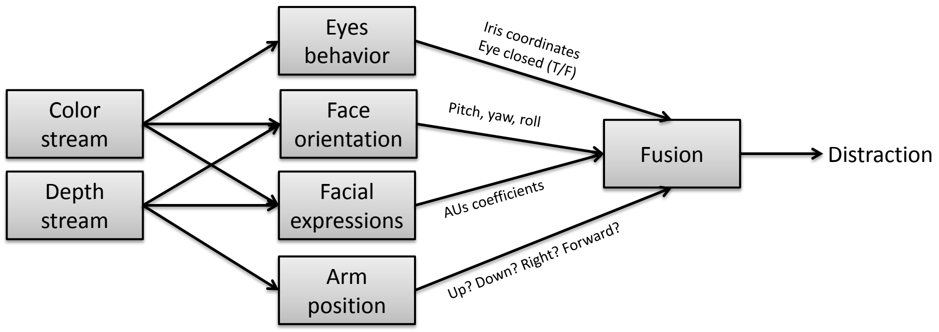


Figure 6.1: Overall software architecture and modules division

mode) [134]. However, the algorithm has been tested on our recorded sequences, and has shown poor results, even in seated mode. There are several possible reasons for this.

First the back of our simulator seat was high and close to the driver's body, making segmentation quite challenging. Moreover, skeleton detection in seated mode relies on body movements, which is not appropriate in the case of driving. Last, we have noticed acceptable tracking performance for lateral movements of the arms, but not for frontal movements. Most of the actions we asked the driver to perform were more frontal than lateral, explaining the poor accuracy in our experiments.

Other methods and source code is available for arm tracking or pose estimation. For example, Watson and Gobeille [135] have provided open source code for arm tracking for Puppet Parade. Zhu and Fujimura [136] have built a detector for upper body parts, while Shotton *et al.* [137] have designed a very efficient approach for pose recognition based on part estimations. None of these approach was satisfying for us, either because they required a too important training set, or because we obtained poor performance on our sequences. For this reason, we decided to build our own arm detection system and add on top of it a bit of machine learning for arm position recognition.

We have created a simple yet efficient feature extractor based on the depth map data. Figure 6.2 shows the main steps of our method.

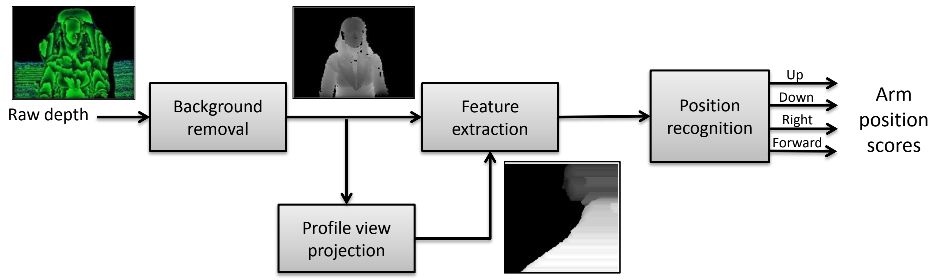


Figure 6.2: Main blocks of the arm position module. The input is the raw depth data. After processing, we extract features and use an AdaBoost classifier to output four scores reflecting the arm position.

Background removal

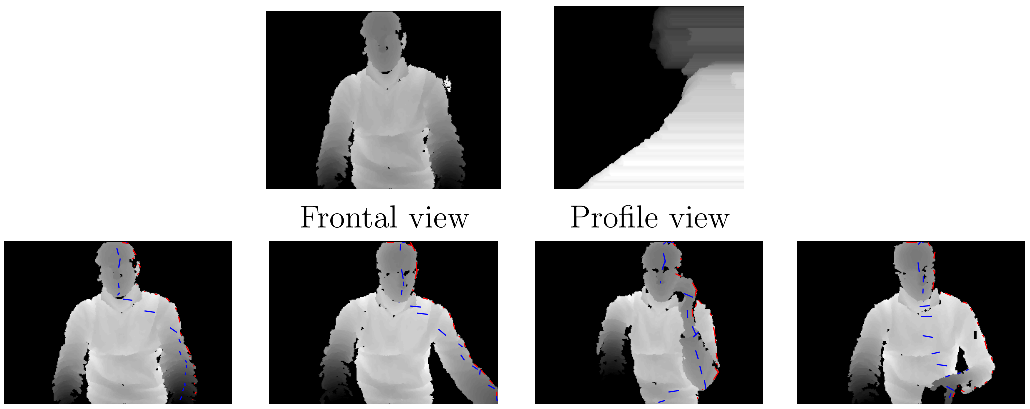
First of all, we need to segment foreground and background. Kinect depth data already contains 3 bits of player data, encoding the location of up to eight persons. This information is quite helpful for us, but not accurate enough, as the driving seat is too close to the driver to be efficiently removed. For this reason, we use a segmentation approach based on background removal. Short video sequences of the seat without the driver were recorded, and a depth map of the seat only was obtained. Then, for a video sequence with the driver, we localize the seat by surface fitting: the seat location is approximately the same as the one with the background only. We therefore apply small horizontal, vertical, and scale translations to the depth map of the background only, and compare it to the depth map with driver. The best seat position is determined as the one which minimizes the sum of squared distances between the two seats' positions. Because the seat is not moving much from one sequence to another, this method is quite efficient. Once the optimal position is obtained, background removal is done by image subtraction and thresholding. We then combine Kinect segmentation with our background removal with a logical AND on the two binary-segmented images, and we remove potential small artifacts by keeping only the biggest blob.

Feature extraction

Based on the segmented depth map, we now need to find features for discriminating arm position. Most of the time, the Kinect records drivers with a frontal view, and their right arm is therefore on the right side of his body. Based on this assumption, we extract features based on foreground contours. We first apply to the binary foreground image the marching squares algorithm, which is the 2D version of the marching cubes [138]. This provides us an ordered list of contour pixels (in this list, each pixel is preceded and followed by the previous and next pixels of the contour). Then, starting from the highest point of the contour curve (corresponding to the top of the head), we remove the left section of the contour (as only the right arm is of interest to us). The remaining contour is the key component of our arm feature extraction method.

We then cut the remaining "half" contour into twenty successive segments, composed of the same number of pixels. Using the depth map, each pixel of the contour is associated with a 3D point, such that each segment of the contour corresponds to a 3D point cloud. For each point cloud, we perform principal component analysis, and keep the eigenvector of the main principal component. This eigenvector is 3-dimensional and corresponds to the point cloud's main orientation. It is scale-invariant and therefore driver-invariant. Collecting the point cloud's main orientation for each segment, we obtain a 20×3 feature vector.

Using only the right contour from the frontal view is not enough, as arm and especially forearm might not always be detected. For example, if the user is sending an SMS, the forearm is likely to be in front of the body rather than on the right. In this case, the forearm is not represented by the contour. To overcome this situation, we apply the aforementioned technique to what we call the profile view: each pixel of the depth map can be associated with a point in 3D world coordinates. Suppose that the $\vec{X}$ and $\vec{Y}$ coordinates are the image pixel coordinates, and $\vec{Z}$ coordinate is the (depth) value of the pixel. Thus, the depth map corresponding to the frontal view is the projection of the 3D world points onto the $(\vec{X}, \vec{Y})$ plan. The profile view would then be the projection on the $(\vec{Y}, \vec{Z})$ plan. Figure 6.3 is an example of a segmented depth map and associated profile view. Note that the profile view is obtained by projecting only the right side pixels of the driver. Once again, we apply contour detection and feature extraction as described above. We now have 120 features. On figure 6.3, examples of features extracted for different poses are shown. Using the profile view makes possible the detection of the forearm even when the arm is in front of the body.


Frontal view Profile view

(a) Forward (b) Right (c) Up (d) Down

Figure 6.3: First row: an example of frontal and associated profile views. Second row: Examples of arm positions and associated features. Red dots are projections of point clouds' local orientations from the frontal view and blue dots from the profile view. As can be seen, profile view features are particularly useful in the case of up and down positions.

Position recognition:

Among the 120 extracted features, some are discriminative, some are useless, and some might even be contradictory. Selecting or weighting the features is therefore necessary for good classification. AdaBoost [139] is a very appropriate tool in this regard. Briefly, AdaBoost (for Adaptive Boosting) is able to turn a set of T weak classifiers into a stronger one, by linearly combining each of them in an optimal way.

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)) \tag{6.1}$$

At each iteration (T in total), a weak classifier h_t is selected from among a family of classifiers, namely, the one minimizing the weighted error rate. The weights and α_t are updated based on the minimum error rate. The weak classifiers we use are decision trees, which, by definition, are trained to select the most representative features maximizing the information gain.

The output of our classifier is the estimated position among four possible states: arm up, arm down, arm right, and arm forward. AdaBoost only solves two-class problems, so we use a 1-vs-all approach to make the classifier cope with four classes. More precisely, we train

Table 6.1: Accuracy of AdaBoost for arm position recognition

Subject	Arm position accuracy
1	97.87
2	89.53
3	86.01
4	91.75
Average	**91.29**

Table 6.2: Normalized confusion matrix

	F	R	U	D
forward	96.88	1.97	0.17	0.99
right	25.80	65.66	2.19	6.35
up	0.64	0.49	98.63	0.24
down	6.87	15.48	5.56	72.09

four sub-classifiers, each of them specialized for one class (i.e., using as positive examples some samples for a specific class, and as negative examples, some samples from any other class). The output of the classifier is the class with highest value among the four sub classifiers. For practical reasons, the output of the arm detection module is the score for each position, rather than the discrete AdaBoost decision.

Numerical results:

Tables 6.1 and 6.2 show an average normalized confusion matrix and a classification accuracies for a few drivers. For each test, the classifier was trained using sessions of some drivers, and tested with a session of a driver that was not used in the training set. Results suggest that the classifier is extremely accurate. The main misclassifications are between arm forward and arm right. This is because the frontier between the two was not always clear. Taking output scores rather than output class should limit the confusion between the two classes when doing data fusion.

6.2.2 Eyes behavior

Eye behavior is a very important feature for driver distraction, including both eye gaze and eye blinking. Statistics about gaze position in time can tell a lot about driver behavior and attention level. Therefore, this module aims to detect the iris of the driver and deduce gaze position. It also detects whether the eye is open or closed. Figure 6.4 shows a summary of the module.

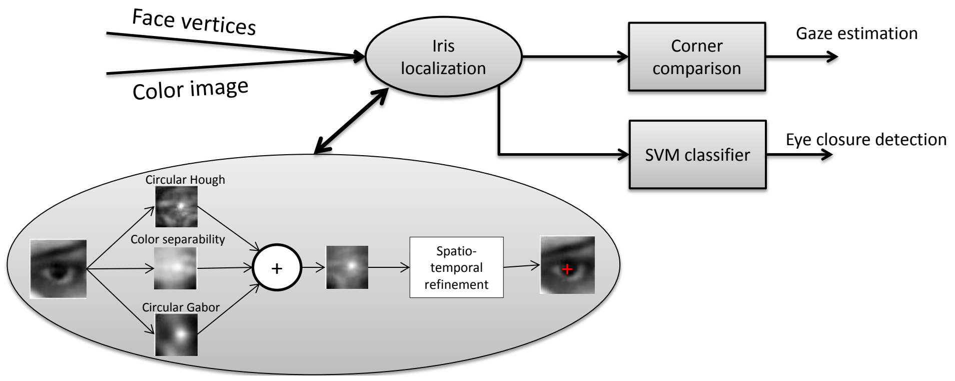


Figure 6.4: Main steps of the eye behavior module

Iris localization:

Iris detection first relies on finding the eye position. This is done using Kinect SDK face tracking, further described in Section 6.2.3. Once eye location is determined, we use a robust iris detection method based on cost function maximization and spatio-temporal considerations. For this module, we only rely on color stream: we detect the eye location (the face tracker provides us the eye corners) and create a squared patch with the same width as the eye corner distance for our processing. To make sure that our parameters are scale-invariant, we resize this square to a 60×60 patch. Our cost function is based on the response of three different filters described below.

The first one is the well-known circular Hough transform that has been widely used in iris detection problems, for example in [140] or [141]. We choose a low edge detection threshold to make sure that the iris contour is detected in any situation, and we use a varying radius from six to nine pixels. Thus, we make sure that the iris will always be responsive to the filter. Despite parameter tuning, using this transformation alone was not accurate enough, because of the poor quality of the image and illumination conditions.

The second filter was successfully used in Zhang *et al.* [142], and relies on the circular Gabor filter. It provides significant impulse response to round objects, which is appropriate for iris detection. For that we convolve our eye template with the following Gabor kernel:

$$G(x, y) = g(x, y)exp(2i\pi F\sqrt{x^2 + y^2}) \qquad (6.2)$$

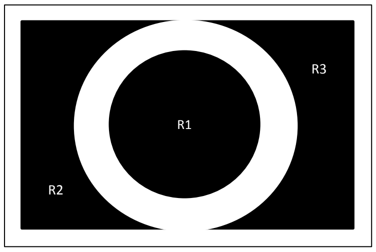


Figure 6.5: Masks R_1, R_2 and R_3 for separability measure. Dark areas represent pixel value 1 and white areas represent pixel value 0

where F is the radial frequency, set to 0.0884 in our case and g is the Gaussian envelope, defined as:

$$g(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} exp(-\frac{x^2+y^2}{2\sigma^2}) \tag{6.3}$$

with σ the variance, set to 4.5 in our system.

The third filter is inspired by Kawaguchi *et al.* [141]. It relies on the high intensity difference between the iris and its immediate neighborhood. For this, we convolve the eye template with the masks R_1, R_2, R_3 represented in Figure 6.5. The radius of mask R_1 is 6, and for masks R_2 and R_3 is 15. We obtain three transformed eye templates, called C_1, C_2, and C_3. We combine them to obtain our separability response using the following formula:

$$S(x,y) = \frac{C_2(x,y) - C_1(x,y)}{C_1(x,y)} + \frac{C_3(x,y) - C_1(x,y)}{C_1(x,y)} \tag{6.4}$$

Last, we normalize our three filter responses, sum them up, and take the maximum value as our iris center estimate (called P_{sum}).

The detection is now quite accurate, but a few mistakes can be avoided and corrected using spatio-temporal information. We therefore use a simple spatio temporal consistency checking to improve detection performance. For each template, we look at the predicted position of each four filter (including the sum of the three filters) and the previous estimated locations. We take our decision by applying the following rules:

- among all filter responses, at least one of them is predicting the correct location

- if at least three of them agree on the location, we consider the detection as reliable, otherwise, it is unreliable
- if both iris locations are reliable, we update the eyes orientation and eyes distance, useful for the refinement
- if only one iris location is reliable, we predict the other eye position based on recorded eyes orientation and distance
  - if the prediction is consistent with one of the filter prediction, we use that location and consider it reliable
  - if not, we take P_{sum} as iris location and consider it unreliable
- if no iris location is reliable, we look at all possible pairs of positions and take the closest to the former position, and consider it unreliable.

Gaze estimation

Now that the iris is detected, we generate a feature set based on gaze estimation. Gaze can be estimated from a 3D model of the face determining the 3D-world orientation, and the iris location given the eyeballs' sizes and positions [143]. We could use such an approach, but we need only a rough estimation of where the driver is looking, and statistical measures of the eye gaze distributions. For that reason, the features we extract are simply the relative position of the iris to the eyes' corners: from the face tracing, we extract the eye corner positions, and we generate the 4 dimensional feature vector as follows:

$$\begin{pmatrix} x_l \\ y_l \\ x_r \\ y_r \end{pmatrix} = \begin{pmatrix} \frac{X_l - C_l^{(l)}}{C_l^{(l)} - C_l^{(r)}} \\ \\ \frac{X_r - C_r^{(l)}}{C_r^{(l)} - C_r^{(r)}} \end{pmatrix} \tag{6.5}$$

with X_l,X_r the left (right) iris position and C_i^j $(i, j \in left, right)$ the j corner of the i eye.

Eye closure detection

We use an additional feature which determines whether the driver's irises are visible (eyes open) or not (eyes closed). This will be helpful when taking into account the amount of time the driver is not looking at the road and the potential danger this represents. A simple yet efficient approach for this is to construct a database of open and closed eyes, and to apply template matching or classification techniques [128], [129], [123]. We use a simple classification approach, as we did in Section 5.3. When pupil position is estimated, we create a small iris template, centered at the iris location. We normalize the grayscale to make it more illumination-insensitive. We create a subset of around 2000 eyes templates, and we manually label each of them as whether it corresponds to an open eye with visible pupil, or a closed eye (or non-visible pupil). Using this dataset, we train an SVM classifier using an RBF kernel with $\sigma = 13$. Last, we use the SVM for each session, and we add to the output of the module the SVM score (not the output label) for each eye and each frame.

Results

Figure 6.6 shows examples of pupil detection to qualitatively illustrate the performance of our detector. Table 6.3 illustrates the performance of the eye behavior module for a few sessions. For each session, correct corner detection is the percentage of frames where corners were detected accurately enough. Results show high accuracy and good consistency for each session. Correct iris detection is the percentage of frames where the iris is correctly localized, among the frames where eyes are open and corner detection is acceptable. We obtain an excellent accuracy, except for a session where luminosity was poor and reflection from the glasses made the detection very challenging. Last, SVM performance was evaluated for each session by training an SVM with other users than the one in the session. Because the open and closed classes were unbalanced, we provides results in term of sensitivity and specificity. Performance varies depending on the session, but results suggest a high reliability for the classifier, and a good trade-off between sensitivity and specificity.

6.2.3 Head pose and facial expression

As explained before, the Kinect SDK provides very useful features and cutting-edge algorithms to help developers. The face tracking algorithm [144] is one of them, and reveals itself to be particularly useful for our work. It uses cooperative fusion of the depth map and the color image to first estimate the root of the head, and then provides a robust and accurate face model. It relies on the active appearance model [145] extended to 3D coordinates. The raw output of the face tracking is a set of 100 vertices and 121 triangles forming a mesh of the face. The algorithm runs in our case at 15 fps and is robust to

Table 6.3: Corner, iris and blink detection performance for a few drivers

Subject	Correct corner detection	Correct iris detection	Blink detection sensitivity	Blink detection specificity
1	97.83	99.28	98.76	97.67
2	97.46	98.84	93.26	95.14
3	97.03	93.05	98.78	85.99
4	99.34	99.51	93.47	98.3
Average	**97.92**	**97.67**	**96.06**	**94.28**

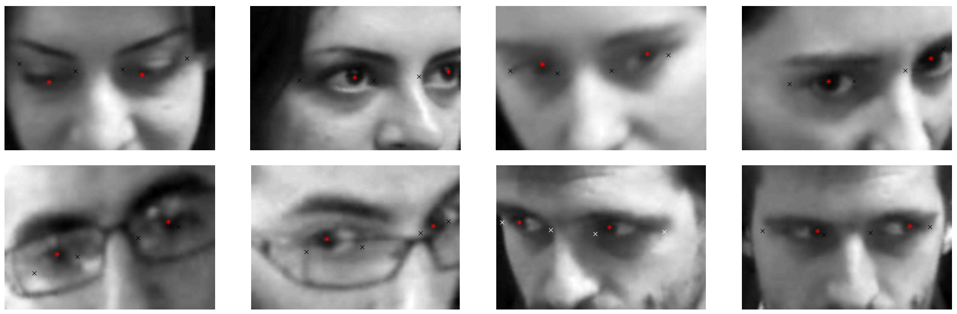


Figure 6.6: Examples of iris detection involving various poses, expressions, and image qualities.

illumination change. The face was tracked in most situations, but failed in case of rapid and significant face rotations, or when occluded by an object (typically when drinking). A few seconds were also required most of the time to correctly initialize the face model and fit it to the face. In the case where face tracking was not providing any output, the eye behavior module was deactivated as no eye location could be found.

Upper level information is also available from the face tracking, making the output much more meaningful and useful for our tasks. We use that high level information as the output of two of our modules.

Face orientation

Based on 3D vertices coordinates, face tracking can provide head orientation angles and head center 3D position. For our work, we extract only the head orientation, namely the pitch, roll, and yaw angles, which are values between -180 and 180 degrees. The position depended too much on the driver's height and did not help in the classification task.

Facial expression

Face tracking also provides six animation units (AUs) based on the definition of the Candide3 model [146]. AUs are expressed as coefficients and represent how strongly distorted features of the face are. We extract only mouth-related AUs: upper lip raiser (AU10), jaw lowerer (AU26/27), lip stretcher (AU20), and lip corner depressor (AU13/15). Other AUs are eyebrow-related and did not help in the recognition task. More detailed information about AUs provided by the face tracker can be found on the Microsoft website [144]. The output of the module is a set of 6 AUs extracted from the face model (if any).

Figure 6.7 shows sample images and associated face tracking results. As can be seen, tracking is efficient in a number of situations.

6.3 Fusion

In this section, we explain how to merge the aforementioned module outputs, and deduce the type of actions taken by the driver using two strategies.

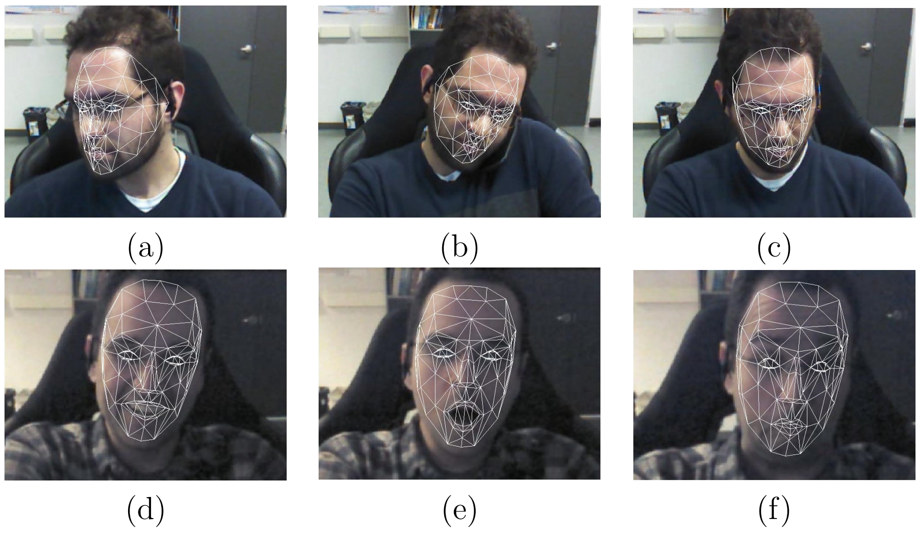


Figure 6.7: Examples of face tracking under different face poses (a),(b),(c) and facial expressions (d), (e), (f)

The field of gesture recognition has received a lot of attention in the past few years. Similar to speech recognition tasks, gesture recognition can be successfully achieved using Hidden Markov models [147], conditional random fields [148], or recursive types of classifiers, making the classification time-dependent [149]. HMM have been successfully used for control gesture recognition inside a car [150]. Our goal is slightly different and somehow more difficult in our case, as the driver is not cooperative when taking the action. Plus, there is no single way to accomplish an action which might be unexpectedly interrupted because of road constraints. Even for a human being, isolating and recognizing actions inside a car is not always obvious.

6.3.1 Dataset:

Among the common tasks it is possible to accomplish in a car, a few of them were selected for our experiments and gathered into five main classes: phone call, text message, object distraction, drinking, and normal driving. More about the conditions of the experiments has been described in Section 3.4. From a classification point of view, our dataset is the concatenation of the outputs of all of the modules described in section 6.2 (i.e., $4 + 6 + 3 + 4 = 17$ features), for all frames (we keep chronological order for temporal considerations). The video sequences were manually labeled for training and evaluation.

6.3.2 Feature extraction and temporal smoothing

Classification could be achieved using each frame independently giving raw output, but a much better performance was found using temporal considerations. For this, we apply a median filter for each feature, using a hundred-sample sliding window. This corresponds to a window length of approximately six seconds, which means that the beginning of an action is predicted with a delay of three seconds. We also computed the standard deviation within the hundred-sample window, providing information about the temporal variation of each feature. The standard deviation information has shown better performance and stability compared to regular speed and acceleration features (δ and δ^2 features). As a result, we have thirty-four features per sample.

6.3.3 AdaBoost classifier

A first approach for classification is to use a time-independent classifier, and add temporal refinement to increase frame by-frame-accuracy. For practical reasons, we use an AdaBoost classifier again using a 1-vs-all approach. Each class is trained using a simple real AdaBoost initialized with a decision tree of depth four and 300 iterations. We classify each sample independently, and add temporal refinement: based on the hundred-sample window, we changed the classified output into the mode (most frequent output) of the subset. This way isolated misclassifications were removed.

6.3.4 Hidden Markov Model

A more complex strategy is to use Hidden Markov Models, similar to gesture or speech recognition. For this, we use the HMM Toolkit (HTK) [100]. We train a different Markov model for each class, and use the Viterbi algorithm to decide which state each sample belongs to. We try several configurations, varying the number of hidden states (from five to twenty) and the type of data (raw or smoothed, with or without δ and δ^2 features).

6.4 Experimental results

6.4.1 Data collection

We use the dataset collected and described in Section 3.4. We evaluate the system based on six out of eight drivers. The last two drivers will be evaluated in a future work. As a reminder, drivers were asked to accomplish a given list of actions during each driving session, namely:

- making a phone call
- drinking
- sending an SMS
- adjusting the radio
- looking at a map (with no need to search or localize anything)
- looking at an outdoor object (the driver was asked to look at a panel and write down on the phone the digits written on the panel)

For now, we can forget about driver fatigue and fatigue signs such as yawning or nodding. In addition, we decided to label our dataset according to five classes, by combining together some actions:

- giving a phone call
- drinking
- sending an SMS (including the outdoor distraction task)
- looking at an object inside the vehicle (map + radio)
- normal driving

This choice is justified by the fact that some actions looked quite similar even to a human eye, and they had a similar distraction impact on the driver. For example, looking at a map and adjusting the radio required little cognitive attention, and the driver was looking at the road most of the time while taking this action. Similarly, the action consisting of looking at an outdoor object was in fact very similar to sending an SMS, as most of the time and effort was spent on writing down the digits on the phone, rather than looking at them.

6.4.2 Action recognition performance

We evaluate the action recognition using AdaBoost and HMM classifiers. For each driver, we evaluate the distraction recognition capacity by training a classifier using all driver sessions except the driver to be evaluated, and testing using all sessions involving this driver. For AdaBoost, the best accuracy was obtained using a Real AdaBoost algorithm with initialization based on a decision tree of depth four with 300 iterations. For HMM, the optimal parameters were a ten-state automaton with a single Gaussian mixture for modeling each node, and using the smoothed data described in Section 6.3.2 rather than the original raw data. No significant improvement was found when using δ and δ^2 features.

Table 6.4 presents the overall accuracy of the AdaBoost and HMM classifiers for each driver. We provide the accuracy for the five classes and the accuracy for distraction detection only: in this case, we have merged all the classes involving distraction into a single class, and compared it with the normal driving class. Average accuracies are quite close between AdaBoost and HMM (85.05% and 84.78%), but the results for each driver may vary. More specifically, HMM outperforms AdaBoost for most of the drivers, but for a few drivers, HMM performs significantly worse than AdaBoost. This may be due to instabilities related to the high dimensionality of the features and the number of states compared to the size of our training set. Increasing the number of drivers might solve this issue. Also, one might be surprised that HMM does not perform as superbly as it does for the gesture recognition task. Again, this is because the driver is not cooperative in this case, and there is no single way to accomplish a distractive task. For example, when drinking, a driver can put the container down between each swallow, or just keep it in the hand. When phoning, the driver can place the phone between head and shoulder, or keep it in hand. Moreover, actions can be interrupted suddenly, because the driver needs both hands on the wheel to turn, or greater focus on the road to avoid accidents. All these constraints make the actions less similar in time, and significantly limits the HMM performance. Using a bigger dataset might eventually improve the classification results. As regards distraction-only accuracy (89.84% and 89.64%), results suggest that our system can successfully detect whether a driver is actually distracted or not.

Another test we did was to train the AdaBoost classifier with each module separately and evaluate their inference capacity (see Table 6.5). We found that features related to arm position were by far the most discriminative, probably because normal driving position was easy to detect, and represented almost 40% of a driving session. Other features were also useful, but provided efficient recognition only for specific actions. For example, we

Table 6.4: Accuracies of AdaBoost and HMM classifiers for distraction recognition (5 classes) and distraction detection (2 classes)

	AdaBoost		**HMM**	
Subject	Distraction recognition	Distraction detection	Distraction recognition	Distraction detection
1	89.36	91.44	84.00	92.31
2	86.02	90.05	87.41	91.16
3	87.38	90.95	90.19	96.41
4	85.48	94.72	81.85	84.65
5	81.14	85.05	83.68	83.82
6	80.94	87.7	81.52	89.53
Average	**85.05**	**89.84**	**84.78**	**89.64**

Table 6.5: Accuracies per features

Subject	Arm	Orientation	Expression	Eyes	All
1	76.49	55.9	54.31	65.7	89.36
2	70.04	56.35	55.45	61.72	86.02
3	85.99	66.8	56.48	38.6	87.38
4	69.80	63.63	55.09	59.95	85.48
5	74.55	66.6	56.30	59.06	81.14
6	81.01	77.87	64.25	63.85	80.94

found that face orientation was very discriminative for drinking and text messaging, facial expression helped a lot in differentiating phone call and drinking, and eye behavior was efficient for text messaging and normal driving. AdaBoost was an appropriate choice of classifier, as decision trees as weak classifiers were doing feature selection for each type of action.

The overall accuracy is a good indicator of system performance, but it does not say how each class is correctly detected. Table 6.6 provides a few classification metrics for each class, based on the average of each driver performance. For each class, extremelly high accuracy is due to an unbalanced testing set. Indeed, a single action (except normal driving) represents between 10% and 20% of the entire sequence. Therefore, recall and precision measures are better indicators of the classification capacity. From the table, it is clear that phone call and normal driving were quite successfully detected. Drinking was a little behind, mainly because the action was sometimes very fast (the driver just swallowed

Table 6.6: Classification measures for each class

Subject	Phone call	Text message	Drinking	Object distraction	Normal driving
Accuracy	95.54	96.24	95.55	92.79	89.98
Sensitivity /recall	80.56	73.63	86.14	24.78	96.00
Specificity	97.90	98.65	96.19	98.91	87.38
Precision	87.51	89.22	51.45	68.28	85.48
f measure	81.71	72.04	60.79	26.30	81.14
g-means	87.98	78.32	90.91	40.16	80.94

for a few seconds and put the drink down) and attenuated by temporal smoothing. The worst performance was for object distraction, probably because this action required neither huge visual nor cognitive attention. In that regard, the action was pretty similar to normal driving and therefore hard to detect, even for a human being.

In order to get more insight about those results, Figure 6.8 displays the frame-by-frame classification for a given sequence. Ground truth is the blue lines, estimated class is the red one. In this example, phone call and text message are accurately detected, drinking comes with a few false positives, and object distraction is often considered normal driving. We have added a few frames to provide a better visualization of why detection was successful or not. Correct detections are represented with green frames, whereas false detections are in red. The drinking false positives are often due to strong arm movements (when changing gear during a phone call, for example). We believe that a better temporal analysis could eliminate this type of false positive detection. For object distraction misclassification, the sample frames show that the driver does not look extremely distracted, and it can be hard to say whether the driver is actually adjusting the radio or just keeping a hand on the gear lever. Fortunately object distraction is the less demanding distraction and therefore the less dangerous, making the misclassification in that case less critical.

6.5 Conclusion and possible improvements

We have successfully built an inattention detection and recognition system based on a Kinect sensor, computer vision, and machine learning. Based on four modules extracting data from arm position, face orientation, facial expression, and eye behaviour, we have constructed two types of classifiers to perform inattentive action recognition. Based on

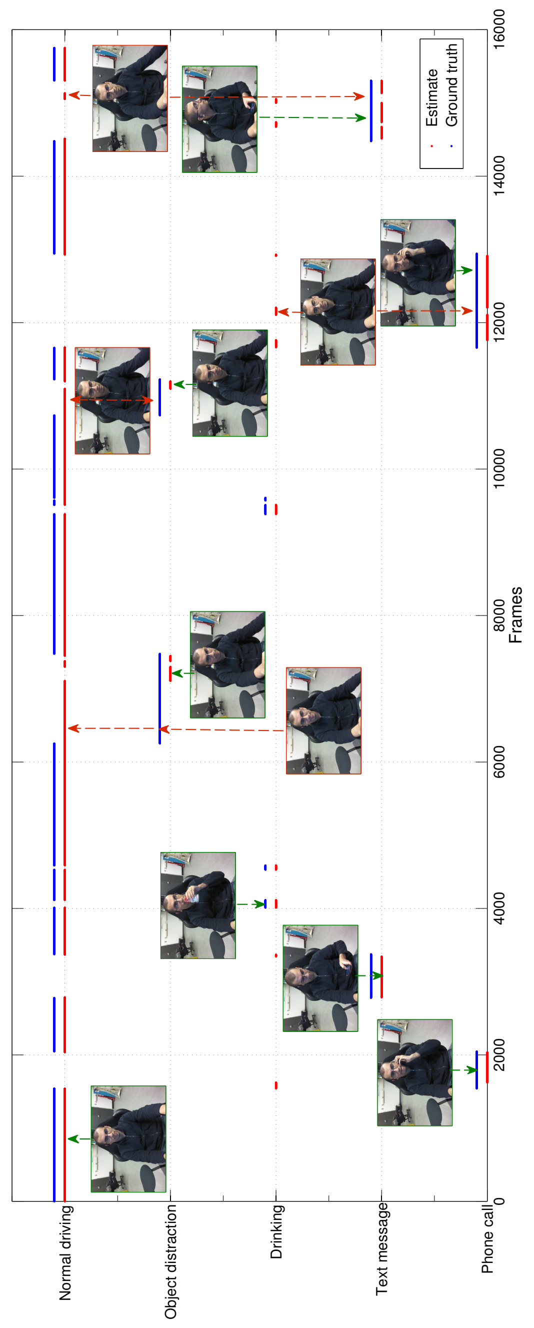


Figure 6.8: Results of action recognition for a given sequence. Ground truth (red) and estimated actions (blue) are displayed for each frame (x axis).

data collected with a driving simulator, we were able to evaluate our work, and results showed that our method is accurate and might be used in a real car.

Compared to existing approaches aiming to detect inattention or provide a level of inattention, our system outputs higher-level information, more suitable for context-aware human-machine interaction. Not only can it be used for immediate driver safety, but also for long term statistics about driver habits, or for inter-vehicle communications systems. Moreover, the different modules we have constructed are extremely flexible, and could be used for other type of statistical computation and inference.

Kinect is still a new product, and it is sometimes criticized as not being efficient in an outdoor or night environment. In this work, we did not want to restrict ourselves to sensosr that are currently used and that are found to be efficient. Indeed, there are many alternatives to obtaining a depth map (such as time of flight cameras) and color cameras could be replaced by infrared to be efficient even at night.

Possible improvements are as follows: first, we believe that action recognition could be improved using more temporal information. For example, drinking and phone calls are sometimes mixed up and alternating in successive frames, whereas it is very unlikely in practice that a driver is doing both at the same time. Such mistakes could be avoided. Next, the modules we have designed could allow fatigue detection using PERCLOS, nodding, and yawning frequencies, for example. Therefore, a few more analyse could lead to an efficient fatigue detection system.
Moreover, background and environment information have not been exploited yet and could prove to be very useful in assessing the level of risk on the road. We might use a Bayesian network, and later on a dynamic Bayesian network to integrate additional sources related to the driver (age, driving experience, fatigue) and the environment (time of day, road type, outside traffic, vehicle speed) in the inference process. Moreover, during the recorded sessions, we have also used additional sensors such as a microphone, a heart rate monitor, or steering wheel signals. We will work on integrating these signals into the system. Last, we also plan to record more driving sessions, involving additional actions, such as interactions and chatting with other passengers.

Based on Kinect sensor and computer vision tools, we have built an efficient module for detecting driver distraction and recognizing the type of distraction. Based on color and depth map data from the Kinect, our system is composed of four sub-modules analyzing *eyes behavior, arm position, facial expression and face orientation. Each module produces relevant information for assessing driver distraction. They are merged together later on using two different classification strategies: an* AdaBoost *classifier, and Hidden Markov Models. Evaluation was performed on our experimental testbed. Qualitative and quantitative results show strong and accurate detection and recognition capacity (85% accuracy for the type of distraction, and 90% for distraction detection).*

Chapter 7

Conclusion

7.1 Contributions

In this thesis, a framework for context-aware driver status assessment systems was designed. The main contributions are listed below.

7.1.1 Experimental framework design

Equipment for building a driving simulator was acquired and set up. Different sensors were added to the system to enable multi-modal acquisition, analysis, and fusion. More precisely, we have a Kinect, two infrared cameras, a microphone array, and a heart rate monitor. The software interface associated with the system enables simultaneous acquisition of all the sensors, as well as steering wheel/pedal signals and road monitoring. Experiments were also carried out based on this system. Eight drivers were recorded in distracted and fatigued conditions for further analysis of their behavior.
In conclusion, this experimental testbed is a reliable starting point for future work. Further, other sensors and types of data collection could easily extend the present work thanks to the flexibility of the interface.

7.1.2 Driver identification framework

An important aspect of context-aware systems is the accessibility of the context. Inside a vehicle, context can be obtained for outdoor conditions via GPS positioning, heat and

humidity sensors, or RADARs, but background knowledge about the driver is also important to better understand behavior. Therefore, person identification is a key component of our framework.
Based on audio-visual data fusion, a driver identification system was developed. The algorithms were tested first on a challenging database, then they were implemented and integrated into our interface. Results on a test set of ten drivers show the excellent accuracy and reliability of the system.

7.1.3 Eye tracking and fatigue detection system

A possible example of driver state assessment is the detection of fatigue. Driver drowsiness detection is an important field of investigation for car safety. Warning the driver about being too tired and encouraging the taking of a nap could potentially avoid numerous road accidents.
Based on computer vision and infrared imaging, we have designed first an efficient eye tracker, which is robust to face rotation. Then, we have used this eye tracking system for fatigue detection by computation of the percentage of eye closure. The algorithm was integrated with the software interface, and showed good results for a simple fatigue detection approach.

7.1.4 Driver distraction detection and recognition

Another road safety concern is driver distraction. The second example of driver status assessment system based on our data collection is then on detection of driver distraction. Our system uses Kinect color and depth streams to extract discriminative features, and uses machine learning strategies to recognize the type of distraction a driver is subject to (if any). The system shows good accuracy and could be integrated with the interface later on.

7.2 Discussion and Future work

The framework designed in this thesis opens doors to a number of new experiments, and several research directions worth exploring. Below are a few suggestions of possible future work.

7.2.1 A better use of the sensors

One of the strengths of our system is its capacity for acquiring multi-modal information. Except for driver identification, our work was largely focused on computer vision approaches. Although camera-based systems are popular and efficient, integrating additional sensors could reveal extremely useful information, and improve the performance of our system. In particular, audio, heart rate, and steering wheel signals are other good indicators of driver inattention. They should be integrated in a future work.

Moreover, the Kinect has not been exploited in this work at its greatest capacity. For example, it could be used in future work for driver recognition based on 3D models, or fatigue detection. The night vision mode provided by the Kinect could also be used, and completely replace our infrared cameras.

7.2.2 Context integration and risk inference

Our driver fatigue and driver distraction detection systems are illustrations of possible man-machine interactions in a vehicle, but to that point, no context has been taken into account. As mentioned earlier in this document, context can refer either to the environment, or to driver background information. In addition, detecting driver inattention is not an end in itself and could be used to infer crash risks for example.

An efficient strategy for integrating all this knowledge is therefore required. One of the most straightforward possibilities is the use of a Bayesian Network. Based on national statistics surveys or previous studies, a probability can be associated with each piece of contextual information to help in the risk assessment. Figure 7.1 presents a possible Bayesian network for context-aware multi-modal inference.

A longer-term perspective is the integration of the time factor. A Bayesian Network is a static system, and cannot integrate temporal consideration. On the other hand, fatigue is a long-term evolving state, changing slowly as time passes. Therefore, more complex fusion and inference strategies could be used, such as dynamic Bayesian networks.

7.2.3 A more extensive study

In the scope of this thesis, we have restricted our investigations due to temporal constraints. However, further experiments and analysis could be carried out.

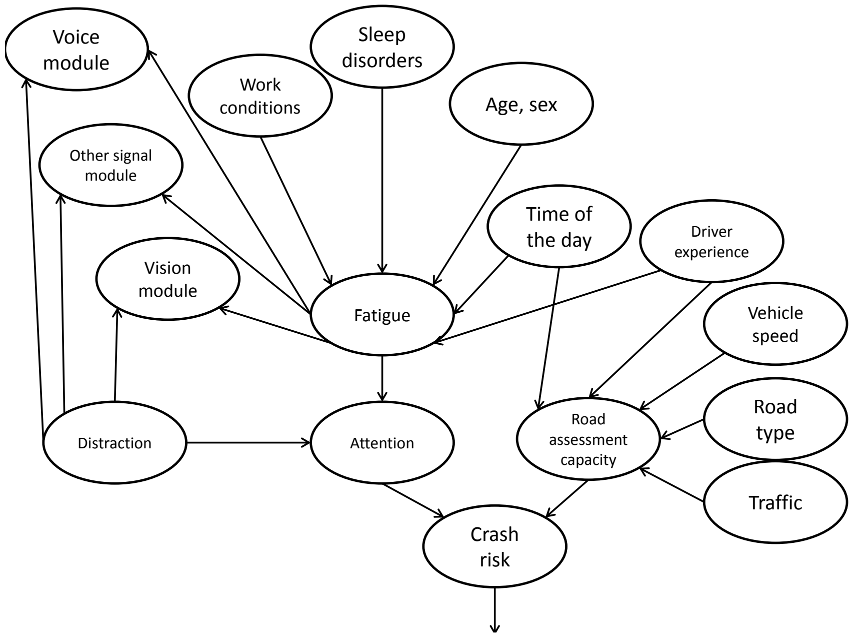


Figure 7.1: Proposed Bayesian network for context integration and crash risk inference

We chose to focus on fatigue and inattention, but there are several other possible reasons affecting driver alertness and performance. Detecting whether a driver is anxious, stressed, nervous, angry, or intoxicated, could be additional features that would be part of an efficient driver state assessment system.

In addition, our experiments were carried out on a small subset of drivers and driving conditions, making the dataset not extensive. Further experiments could, for example, integrate more distractive actions (such as passenger interaction) or true fatigue. A greater number of subjects would also be required to obtain reliable statistics and a wider panel of situations.

7.2.4 Long-term perspectives

Longer-term perspectives, that are outside the scope of the project could also be investigated.

First, using a driving simulator is a good starting point as data acquisition and testing is much easier. However, real life outdoor conditions cannot be efficiently simulated. For example, synthetic noise generated by the car driving software is not extremely realistic, and the acoustics of the laboratory room are quite different from a car environment. Illumination can be simulated either by the software or by an artificial lighting system, however, they cannot reproduce sun-light or the complexity of an outdoor environment. Last, most subjects enjoyed the driving experience, but found that braking, acceleration, and vibration sensations were somehow missing. Using a real car for experimentation is therefore the only way to obtain realistic and reliable driving experiences.

Second, this project was focused on driver behavior and contextual information. No analysis or correlation has been done on interactions between driver condition and driving performance on the road. Driving performance would involve road lane, road sign, or other-vehicle detection, and performance would be evaluated on the driver's capacity to correctly respond to the environment (such as avoiding lane deviation, slowing down when a road sign indicates to do so, or keeping a good distance from other vehicles). Such an analysis would require additional sensors and a lot of processing effort, but it could eventually serve efficiently the goal of driver state assessment.

Last, other projects could extend our work, and use driver state information for safety. In this work, we have proposed the most straightforward strategy, of simply emitting a warning if a driver is showing signs of fatigue or inattention. However, this information could be used in inter-vehicle communications. Instead of just warning the driver, sending the information to the surrounding cars would make drivers all around aware of the potential danger, thus reducing the risk of accident.

APPENDICES

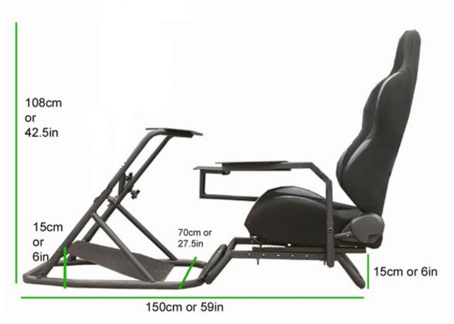
Appendix A

Hardware list detail

Products detail

1 – Cockpit

Product	**Obutto oZone cockpit** (489$)
Description	- Seat - Steering wheel/pedal/shifter mount - Monitor stand
Additional option	Triple monitor mount(115$)
Total price **(taxes and shipping fee included)**	**594.00 $**
Website: http://www.getgadget.ca/purchase.html	



2 - Steering wheel, pedals and shifter

Product	**Logitech G27 racing wheel**
Description	- Steering wheel - Pedals - Shifter
Total price **(taxes and shipping fee included)**	**282.49 $**
Website: http://www.amazon.ca/Logitech-941-000045-G27-Racing-Wheel/dp/B001NT9TK4	

3 – Driving simulation software

Product	**City Car Driving V1.2**
Description	- In this car driving simulator a special stress has been laid on the variety of road situations and realistic car driving. - Triple monitor display available
Total price **(taxes and shipping fee included)**	**25.00 $**
Website: http://citycardriving.com/buy/citycardriving	

4 – Monitors

Product	**ASUS VH242H LCD Monitor** (213.55$ each)
Description	- 23.6-inch full 1080p HD widescreen - HDMI/DVI/VGA - VGA + power supply cable included
Quantity	3
Total price **(taxes and shipping fee included)**	**640.65 $**
Website: http://www.amazon.ca/ASUS-VH242H-23-6-Inch-Widescreen-Monitor/dp/B001LYPIIS/ref=sr_1_1?s=electronics&ie=UTF8&qid=1330543615&sr=1-1	

5 – Three monitor graphic expansion

Product	**Matrox-TripleHead2Go**
Description	- Run three independent monitors from your notebook or desktop computer even if that system only supports a single monitor output - Required for City Car driving (3)
Total price **(taxes and shipping fee included)**	**373.98$**
Website: http://www.amazon.ca/Matrox-TripleHead2Go-Three-Monitor-Graphics-Expansion/dp/B000RMQZ96	

6 – Lighting kit

Product	**InspironPhoto 2400W**
Description	- Two pieces 4 Light Bank Selectable Light Fixture including power cords - 12 x45 Watt 5500K Light Bulbs - Three fully adjustable light stands - - Three 20"x28" softboxes - Heavy duty carrying case
Total price **(taxes and shipping fee included)**	**219.00$**
Website: http://www.amazon.ca/InspironPhoto-Photography-Lighting-Digital-Chromakey/dp/B005NMTI8K/ref=sr_1_8?ie=UTF8&qid=1333049874&sr=8-8	

7 – Microphone array

Product	**Andrea Electronics Microphone DA-350**
Description	- linear multi-element array microphone - Noise reduction / cancellation included - USB audio adapter
Total price **(taxes and shipping fee included)**	**364.00$**
Website: http://store.mp3car.com/Andrea_Electronics_Microphone_DA_350_p/com-017.htm	

8 – Camera

Product	**Near Infrared Camera**
Description	
Total price **(taxes and shipping fee included)**	**~300.00$**
Website:	

Three additional products were purchased after the initial order:

9 – Kinect for windows

Product	**Kinect for Windows**
Description	
Total price **(taxes and shipping fee included)**	**$264.16**
Website: http://www.amazon.ca/X360-Kinect-sensor-for-Windows/dp/B006UIS53K/ref=sr_1_15?ie=UTF8#&qid=1328883933#&sr=8-15	

10 – Heart rate monitor

Product	**CMS50E**
Description	
Total price **(taxes and shipping fee included)**	**$79.00**
http://www.amazon.ca/Finger-pulse-oximeter-CMS-50E-accessories/dp/B0035WFT2E	

11 – Graphic card

Product	**Zotac GeForce GTS 450**
Description	
Total price **(taxes and shipping fee included)**	**$150.00**

Appendix B

Driving session details

Driving simulator experiments

Céline Craye and Abdullah Rashwan

Goals

Our goal is to carry out experiments on several drivers to assess their level of inattention and fatigue. The subjects will be asked to drive the simulator in situations of inattention or fatigue. Several sensors will be used to obtain as many information as possible about the driver and its driving. Data fusion will be used later on to combine all the information and estimate if the driver is inattentive or tired.

General information

- 6 to 8 subjects
- 2 driving sessions of 30 minutes
  - 1 in the morning (driver is not tired)
  - 1 late in the evening after a working day (driver is tired)
- For each session, two driving conditions
  - City driving condition with high traffic
  - Highway driving with low traffic
- 5 minutes of free driving before the experiment to make the driver comfortable with driving when starting the experiment.
- Instructions will be provided to the driver while driving through a headset

Driving session

The tasks are repeated twice for each session, once for each road condition. Drowsiness acting is done only during the night sessions. Below are the different tasks

5 tasks for inattention

1- Phone call
2- Text message
3- Drinking
4- Map research/ Road distraction
5- Adjusting the radio

3 tasks for drowsiness (only during the evening session)

6- Eyes closed for a few seconds
7- Yawning
8- Nodding

Below is a timeline of a typical driving session during the day.

Time (min)	0:00-3:00	3:00-5:00	5:00-6:00	6:00-7:00	7:00-8:00	8:00-10:00	10:00-12:00	12:00-13:00	13:00-14:00	14:00-15:00
Action	■	1	▒	2	▒	3	▒	4	5	▒
Time (min)	15:00-18:00	18:00-20:00	20:00-21:00	21:00-22:00	22:00-23:00	23:00-25:00	25:00-27:00	27:00-28:00	28:00-29:00	29:00-30:00
Action	■	3	▒	5	▒	1	▒	2	4	▒

And a typical session during the night

Time (min)	0:00-3:00	3:00-4:00	4:00-4:30	4:30-5:30	5:30-6:00	6:00-8:00	8:00-9:00	9:00-12:00	12:00-12:30	12:30-14:30	14:30-15:00
Action	■	2	▒	4	▒	3	▒	5	▒	1	▒
Time (min)	15:00-18:00	18:00-19:00	19:00-19:30	19:30-20:30	20:30-21:00	21:00-23:00	23:00-23:30	23:30-33:30			
Action	■	2	▒	3	▒	5	1	▒			

The night session includes 10 minutes of driving on the highway without any distraction in order to look at the driver behaviour in drowsy conditions (tired + no distraction)

▒	Normal driving (including tasks 6, 7, and 8 for the evening session).
■	Loading the map and normal driving

Task description

First Phone call

Hostess: Pepi's Pizza. How can I help you?
Subject: Hi. I'd like to order a pizza please.
Hostess: Okay. I'll have to transfer your call to our take-out department. One moment please.
Recorded Message: Thank you for calling Pepi's Pizza. All of our operators are busy. Please hold for the next available person.
Take-out Clerk: Thank you for waiting. Is this for take-out or delivery?
Subject: Delivery please.
Take-out Clerk: Can I have your name and address please?
Subject: My name is... My address is ....
Take-out Clerk: Thank you. Is that an apartment or a house?
Subject: It's an apartment. Number ....
Take-out Clerk: Okay. And what would you like to order today?
Subject: I'd like a large pepperoni pizza with extra cheese.
Take-out Clerk: Ok so large pizza isn't it ?
Subject: Yes

Take-out Clerk: With pepperoni and extra cheese
Subject: That's right.
Take-out Clerk: Ok fine. Is there anything else?
Subject: No, that would be all
Take-out Clerk: Anything to drink with that?
Subject: Nothing, thanks
Take-out Clerk: Alright, it's 15 $. How would you like to pay?
Subject: Do you accept credit card?
Take-out Clerk: Credit card? Sure. You pizza should arrive in about thirty minutes. Is that ok?
Subject: Absolutely, thank you very much.
Take-out Clerk: You're welcome. Thanks for calling. Bye.

Second Phone Call

Subject: Hi, How are you?

Friend: I am good, I was trying to call you all the day, where were you?

Subject: I have been with my sister at Toronto the whole day, why?

Friend: There is a movie night event at the University, do you want to come?

Subject: Sure, I have to cancel first my meeting with Adam.

Friend: Okay, we will be waiting for you.

Subject: Great! See you then.

Friend: See you, Bye.

Subject: Bye.

First Text message

"Hey, what about eating some pizzas tonight? I just ordered one, I hope you don't mind"

Second Text message

"I will be there in 30 mins"

Third Text Message

"Hi Adam, can we meet tomorrow? Something came up"

Drinking

Buy some coffee at Tim Hortons before the test

Map research

On a map of Kitchener/Waterloo, ask to locate some streets

Adjusting the radio

Bring a real radio or just a computer and ask to select a tune.

Sensor list

- A Kinect. Recording camera stream, depth map stream and audio stream
- 1 or 2 IR camera on bright (+ dark) pupil effect
- Steering wheel + pedal feedback
- Pulse oximeter for blood pressure and heart rate
- Screenshots of the road

References

[1] Christian Martyn Jones and Ing-Marie Jonsson. Automatic recognition of affective cues in the speech of car drivers to allow appropriate responses. In *Proceedings of the 17th Australia conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future*, OZCHI '05, pages 1–10. Computer-Human Interaction Special Interest Group (CHISIG) of Australia, 2005.

[2] Ji Qiang and Yang Xiaojie. Real-Time Eye, Gaze, and Face Pose Tracking for Monitoring Driver Vigilance. *Real-Time Imaging*, 8(5):357–377, 2002.

[3] Neville A Stanton and Paul M Salmon. Human error taxonomies applied to driving: A generic driver error taxonomy and its implications for intelligent transport systems. *Safety Science*, 47(2):227–237, 2009.

[4] Sheila G Klauer, Thomas A Dingus, Vicki L Neale, JD Sudweeks, and DJ Ramsey. The impact of driver inattention on near-crash/crash risk: An analysis using the 100-car naturalistic driving study data. Technical report, 2006.

[5] Rebecca L Olson, Richard J Hanowski, Jeffrey S Hickman, and Joseph L Bocanegra. Driver distraction in commercial vehicle operations. Technical report, 2009.

[6] David Geronimo, Antonio M Lopez, Angel Domingo Sappa, and Thorsten Graf. Survey of pedestrian detection for advanced driver assistance systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(7):1239–1258, 2010.

[7] Andreas Laika and Walter Stechele. A review of different object recognition methods for the application in driver assistance systems. In *Image Analysis for Multimedia Interactive Services, 2007. WIAMIS'07. Eighth International Workshop on*, pages 10–10. IEEE, 2007.

[8] Zehang Sun, George Bebis, and Ronald Miller. On-road vehicle detection: A review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(5):694–711, 2006.

[9] Philip Lockwood and Jérôme Boudy. Experiments with a nonlinear spectral subtractor (nss), hidden markov models and the projection, for robust speech recognition in cars. *Speech Communication*, 11(2):215–228, 1992.

[10] Philipos C Loizou. *Speech enhancement: theory and practice.* CRC press, 2013.

[11] Ford. Ford sync — sync support, phone compatibility, updates, manuals, voice guides, instructions and more. http://www.ford.com/technology/sync/, 2007.

[12] Lexus. Secret voice commands. https://secure.drivers.lexus.com/lexusdrivers/magazine/articles/Vehicle-Insider/Lexus-Quick-Tip-Voice-Commands, 2013.

[13] Chrysler. Chrysler uconnect. http://www.driveuconnect.com/, 2013.

[14] Buick Intellilink. Buick intellilink infotainment system. www.buick.com/experience/intellilink-infotainment-system.html, 2012.

[15] Seeingmachines. Facelab : A face and eye tracking system. http://www.seeingmachines.com/product/facelab/in-vehicle-eyetracking/, 2013.

[16] Safran Morpho. Acces control terminals. http://www.morpho.com/identification/secure-biometric-access/access-control-terminals/, 2013.

[17] Niall A Fox, Ralph Gross, Jeffrey F Cohn, and Richard B Reilly. Robust biometric person identification using automatic classifier fusion of speech, mouth, and face experts. *Multimedia, IEEE Transactions on*, 9(4):701–714, 2007.

[18] Souheil Ben-Yacoub, Yousri Abdeljaoued, and Eddy Mayoraz. Fusion of face and speech data for person identity verification. *Neural Networks, IEEE Transactions on*, 10(5):1065–1074, 1999.

[19] Ricardo Muñoz, Oscar Castillo, and Patricia Melin. Face, fingerprint and voice recognition with modular neural networks and fuzzy integration. In *Bio-inspired Hybrid Intelligent Systems for Image Analysis and Pattern Recognition*, pages 69–79. Springer, 2009.

[20] Fenghua Wang and Jiuqiang Han. Robust multimodal biometric authentication integrating iris, face and palmprint. *Information Technology And Control.–Kaunas: Technologija*, 37(4):326–332, 2008.

[21] Richard M Jiang, Abdul H Sadka, and Danny Crookes. Multimodal biometric human recognition for perceptual human–computer interaction. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(6):676–681, 2010.

[22] Chiyomi Miyajima, Yoshihiro Nishiwaki, Koji Ozawa, Toshihiro Wakita, Katsunobu Itou, Kazuya Takeda, and Fumitada Itakura. Driver modeling based on driving behavior and its evaluation in driver identification. *Proceedings of the IEEE*, 95(2):427–437, 2007.

[23] Jian-Da Wu and Siou-Huan Ye. Driver identification using finger-vein patterns with radon transform and neural network. *Expert Systems with Applications*, 36(3):5793–5799, 2009.

[24] Toshihiro Wakita, Koji Ozawa, Chiyomi Miyajima, Kei Igarashi, ITOU Katunobu, Kazuya Takeda, and Fumitada Itakura. Driver identification using driving behavior signals. *IEICE TRANSACTIONS on Information and Systems*, 89(3):1188–1194, 2006.

[25] Bayometric inc. Driver identification system. http://www.bayometric.com/success-stories/driver-identification-system.htm, 2013.

[26] BioHIC. Carguard, biometric vehicle security system. http://biohic.com/biometric-vehicle-security-systems.html, 2011.

[27] FleetVision. Driverid system. http://www.fleetvision.co.uk/driverid.html, 2012.

[28] Volkswagen. Volkswagen group biometric driver identification. http://www.volkswagenag.com/content/vwcorp/content/en/innovation/Biometric.html, 2013.

[29] Ashish Tawari and Mohan Trivedi. Speech based emotion classification framework for driver assistance system. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pages 174–178. IEEE, 2010.

[30] Björn Schuller, Gerhard Rigoll, and Manfred Lang. Speech emotion recognition combining acoustic features and linguistic information in a hybrid support vector

machine-belief network architecture. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 1, pages I–577. IEEE, 2004.

[31] Maja Pantic and Ioannis Patras. Dynamics of facial expression: recognition of facial actions and their temporal segments from face profile image sequences. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 36(2):433–449, 2006.

[32] Yongmian Zhang and Qiang Ji. Active and dynamic information fusion for facial expression understanding from image sequences. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(5):699–714, 2005.

[33] Liyanage C De Silva, Tsutomu Miyasato, and Ryohei Nakatsu. Facial emotion recognition using multi-modal information. In *Information, Communications and Signal Processing, 1997. ICICS., Proceedings of 1997 International Conference on*, volume 1, pages 397–401. IEEE, 1997.

[34] Jennifer A Healey and Rosalind W Picard. Detecting stress during real-world driving tasks using physiological sensors. *Intelligent Transportation Systems, IEEE Transactions on*, 6(2):156–166, 2005.

[35] John Dollard, Neal E Miller, Leonard W Doob, Orval Hobart Mowrer, and Robert R Sears. Frustration and aggression. 1939.

[36] Lucas Malta, Chiyomi Miyajima, Norihide Kitaoka, and Kazuya Takeda. Analysis of real-world driver's frustration. *Intelligent Transportation Systems, IEEE Transactions on*, 12(1):109–118, 2011.

[37] Tara E Galovski, Loretta S Malta, and Edward B Blanchard. *Road rage: Assessment and treatment of the angry, aggressive driver.* American Psychological Association, 2006.

[38] John D Lee, Kristie L Young, and Michael A Regan. Defining driver distraction. 2009.

[39] Michael A Regan, Charlene Hallett, and Craig P Gordon. Driver distraction and driver inattention: Definition, relationship and taxonomy. *Accident Analysis & Prevention*, 43(5):1771–1781, 2011.

[40] Saroj KL Lal and Ashley Craig. A critical review of the psychophysiology of driver fatigue. *Biological psychology*, 55(3):173–194, 2001.

[41] NHTSA. Distraction.gov, official us government website for distracted driving. http://http://www.distraction.gov, 2013.

[42] Ann Williamson and Tim Chamberlain. Review of on-road driver fatigue monitoring devices. 2005.

[43] Kristie Young, Michael Regan, and M Hammer. Driver distraction: A review of the literature. *Distracted driving. Sydney, NSW: Australasian College of Road Safety*, pages 379–405, 2007.

[44] Qiong Wang, Jingyu Yang, Mingwu Ren, and Yujie Zheng. Driver fatigue detection: a survey. In *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, volume 2, pages 8587–8591. IEEE, 2006.

[45] Qiong Wang, Huan Wang, Chunxia Zhao, and Jingyu Yang. Driver fatigue detection technology in active safety systems. In *Remote Sensing, Environment and Transportation Engineering (RSETE), 2011 International Conference on*, pages 3097–3100. IEEE, 2011.

[46] Yanchao Dong, Zhencheng Hu, Keiichi Uchimura, and Nobuki Murayama. Driver inattention monitoring system for intelligent vehicles: A review. *Intelligent Transportation Systems, IEEE Transactions on*, 12(2):596–614, 2011.

[47] EyeAlert. Distracted driving and fatigue sentinels. http://www.eyealert.com/, 2012.

[48] N. Edenborough, R. Hammoud, A. Harbach, A. Ingold, B. Kisacanin, P. Malawey, T. Newman, G. Scharenbroch, S. Skiver, M. Smith, A. Wilhelm, G. Witt, E. Yoder, and H. Zhang. Driver state monitor from delphi. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 1206–1207 vol. 2, 2005.

[49] Lexus. Driver monitoring system. http://www.lexus.eu/car-models/ls/ls-600h/index.tmex, 2006.

[50] InSight. Sensomotoric instruments gmbh. http://www.smivision.com/en/gaze-and-eye-tracking-systems/services/smi-eye-tracking-roadshow.html, 2011.

[51] Media Volvo Car. Driver alert control. http://www.media.volvocars.com, 2007.

[52] Ford Driver Alert. Fords wake-up call for europes sleepy drivers. http://media.ford.com, 2011.

[53] Volkswagen. Driver alert, driver assistance, experience. http://www.volkswagen.co.uk/new/passat-vii/explore/experience/driver-assistance/driver-alert, 2011.

[54] Li Shiwu, Wang Linhong, Yang Zhifa, Ji Bingkui, Qiao Feiyan, and Yang Zhongkai. An active driver fatigue identification technique using multiple physiological features. In *Mechatronic Science, Electric Engineering and Computer (MEC), 2011 International Conference on*, pages 733–737. IEEE, 2011.

[55] Ioannis G Damousis and Dimitrios Tzovaras. Fuzzy fusion of eyelid activity indicators for hypovigilance-related accident prediction. *Intelligent Transportation Systems, IEEE Transactions on*, 9(3):491–500, 2008.

[56] Marko Lugger and Bin Yang. Psychological motivated multi-stage emotion classification exploiting voice quality features. *Speech Recognition, In-Tech*, 2008.

[57] Luis M Bergasa, Jose M Buenaposada, Jesus Nuevo, Pedro Jimenez, and Luis Baumela. Analysing driver's attention level using computer vision. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 1149–1154. IEEE, 2008.

[58] Q. Ji, Z. Zhu, and P. Lan. Real-Time Nonintrusive Monitoring and Prediction of Driver Fatigue. *IEEE Transactions on Vehicular Technology*, 53(4):1052–1068, 2004.

[59] Qiang Ji, Peilin Lan, and Carl Looney. A probabilistic framework for modeling and real-time monitoring human fatigue. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 36(5):862–875, 2006.

[60] L.M. Bergasa, J. Nuevo, M.A. Sotelo, R. Barea, and M.E. Lopez. Real-time system for monitoring driver vigilance. *Intelligent Transportation Systems, IEEE Transactions on*, 7(1):63 –77, 2006.

[61] Li Li, Klaudius Werber, Carlos F Calvillo, Khac Dong Dinh, Ander Guarde, and Andreas König. Multi-sensor soft-computing system for driver drowsiness detection. *Online conference on soft computing in industrial applications*, pages 1–10, 2012.

[62] David F Dinges and Richard Grace. Perclos: A valid psychophysiological measure of alertness as assessed by psychomotor vigilance. *Federal Highway Administration. Office of motor carriers, Tech. Rep. MCRT-98-006*, 1998.

[63] Albert Kircher, Marcus Uddman, and Jesper Sandin. *Vehicle control and drowsiness.* Swedish National Road and Transport Research Institute, 2002.

[64] Esa M Rantanen and Joseph H Goldberg. The effect of mental workload on the visual field size and shape. *Ergonomics*, 42(6):816–834, 1999.

[65] L Angell, JL Auflick, PA Austria, DS Kochhar, L Tijerina, WJ Biever, T Diptiman, JR Hogsett Jr, and SM Kiger. Driver workload metrics project. *National Highway Traffic Safety Administration*, 2006.

[66] Thomas A Ranney. Driver distraction: A review of the current state-of-knowledge. Technical report, 2008.

[67] Niels Egelund. Spectral analysis of heart rate variability as an indicator of driver fatigue. *Ergonomics*, 25(7):663–672, 1982.

[68] M Patel, SKL Lal, D Kavanagh, and P Rossiter. Applying neural network analysis on heart rate variability data to assess driver fatigue. *Expert Systems with Applications*, 38(6):7235–7242, 2011.

[69] Saroj KL Lal and Ashley Craig. Driver fatigue: electroencephalography and psychological assessment. *Psychophysiology*, 39(3):313–321, 2002.

[70] Masahiro Miyaji, Mikio Danno, Haruki Kawanaka, and Koji Oguri. Driver? cognitive distraction detection using adaboost on pattern recognition basis. In *Vehicular Electronics and Safety, 2008. ICVES 2008. IEEE International Conference on*, pages 51–56. IEEE, 2008.

[71] Chris Berka, Daniel J Levendowski, Michelle N Lumicao, Alan Yau, Gene Davis, Vladimir T Zivkovic, Richard E Olmstead, Patrice D Tremoulet, and Patrick L Craven. Eeg correlates of task engagement and mental workload in vigilance, learning, and memory tasks. *Aviation, space, and environmental medicine*, 78(Supplement 1):B231–B244, 2007.

[72] IG Daza, N Hernandez, LM Bergasa, I Parra, JJ Yebes, M Gavilan, R Quintero, DF Llorca, and MA Sotelo. Drowsiness monitoring based on driver and driving data fusion. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 1199–1204. IEEE, 2011.

[73] Tom Pilutti and A Galip Ulsoy. Identification of driver state for lane-keeping tasks: experimental results. In *American Control Conference, 1997. Proceedings of the 1997*, volume 5, pages 3370–3374. IEEE, 1997.

[74] Hardeep Singh, JS Bhatia, and J Kaur. Eye tracking based driver fatigue monitoring and warning system. In *Power Electronics (IICPE), 2010 India International Conference on*, pages 1–6. IEEE, 2011.

[75] Rajinda Senaratne, David Hardy, Bill Vanderaa, and Saman Halgamuge. Driver fatigue detection by fusing multiple cues. In Derong Liu, Shumin Fei, Zengguang Hou, Huaguang Zhang, and Changyin Sun, editors, *Advances in Neural Networks ISNN 2007*, volume 4492 of *Lecture Notes in Computer Science*, pages 801–809. Springer Berlin Heidelberg, 2007.

[76] Paul Smith, Mubarak Shah, and Niels da Vitoria Lobo. Determining driver visual attention with one camera. *Intelligent Transportation Systems, IEEE Transactions on*, 4(4):205–218, 2003.

[77] Mervyn VM Yeo, Xiaoping Li, Kaiquan Shen, and Einar PV Wilder-Smith. Can svm be used for automatic eeg detection of drowsiness during car driving? *Safety Science*, 47(1):115–124, 2009.

[78] Jianping Liu, Chong Zhang, and Chongxun Zheng. Eeg-based estimation of mental fatigue by using kpca–hmm and complexity parameters. *Biomedical Signal Processing and Control*, 5(2):124–130, 2010.

[79] Chern-Pin Chua, Gary McDarby, and Conor Heneghan. Combined electrocardiogram and photoplethysmogram measurements as an indicator of objective sleepiness. *Physiological measurement*, 29(8):857, 2008.

[80] Luke Fletcher, Gareth Loy, Nick Barnes, and Alexander Zelinsky. Correlating driver gaze with the road scene for driver assistance systems. *Robotics and Autonomous Systems*, 52(1):71–84, 2005.

[81] T. D'Orazio, M. Leo, C. Guaragnella, and a. Distante. A visual approach for driver inattention detection. *Pattern Recognition*, 40(8):2341–2355, 2007.

[82] Chihang Zhao, Jie Lian, Jie He, Jing Shen, Tiantian Zhu, and Hongjuan Zhang. Recognition of driver's fatigue expressions by gabor wavelet transform and multilayer perceptron classifier. In *Transportation, Mechanical, and Electrical Engineering (TMEE), 2011 International Conference on*, pages 617–620. IEEE, 2011.

[83] Jiang Yuying, Wu Yazhen, and Xu Haitao. A surveillance method for driver's fatigue and distraction based on machine vision. In *Transportation, Mechanical, and*

Electrical Engineering (TMEE), 2011 International Conference on, pages 727–730. IEEE, 2011.

[84] Lingling Li, Yangzhou Chen, and Zhenlong Li. Yawning detection for monitoring driver fatigue based on two cameras. In *Intelligent Transportation Systems, 2009. ITSC'09. 12th International IEEE Conference on*, pages 1–6. IEEE, 2009.

[85] Xiao Fan, Baocai Yin, and Yanfeng Sun. Nonintrusive driver fatigue detection. In *Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on*, pages 905–910. IEEE, 2008.

[86] Jennifer A. Healey and Rosalind W. Picard. Detecting stress during real-world driving tasks using physiological sensors. *IEEE Transactions on Intelligent Transportation Systems*, 6:156–166, 2005.

[87] Steven Beauchemin, Parisa Darvish Zadeh Varcheie, Langis Gagnon, Denis Laurendeau, Martin Lavallière, Thierry Moszkowicz, Florent Prel, and Normand Teasdale. Cobvis-d: a computer vision system for describing the cephalo-ocular behavior of drivers in a driving simulator. In *Image Analysis and Recognition*, pages 604–615. Springer, 2009.

[88] Garrett Weinberg and Bret Harsham. Developing a low-cost driving simulator for the evaluation of in-vehicle technologies. In *Proceedings of the 1st International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, AutomotiveUI '09, pages 51–54. ACM, 2009.

[89] Eva Blana. A survey of driving research simulators around the world. 1996.

[90] simulation Lander and training solutions. Centre for studies and technical research of guipuzcoa (ceit). http://www.landersimulation.com/eng/about-lander/foundation/ceit/, 1982.

[91] VTI. Driving simulation. http://www.vti.se/en/vti-offers/driving-simulation/, 2013.

[92] ST software Simulation System. Driving simulator products. http://www.stsoftware.nl/, 2012.

[93] Stisim Drive. Driving simulator products. http://www.stisimdrive.com/our-solutions/raampd-solutions.html, 2012.

[94] Bowon Lee, Mark Hasegawa-Johnson, Camille Goudeseune, Suketu Kamdar, Sarah Borys, Ming Liu, and Thomas S Huang. Avicar: audio-visual speech corpus in a car environment. In *INTERSPEECH*, 2004.

[95] Marcos Faundez-Zanuy. Data fusion in biometrics. *Aerospace and Electronic Systems Magazine, IEEE*, 20(1):34–38, 2005.

[96] Anil K. Jain, Arun Ross, and Salil Prabhakar. An introduction to biometric recognition. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):4–20, 2004.

[97] Richard A Wasniowski. Using data fusion for biometric verification. *World Academy of Science, Engineering and Technology*, 5, 2005.

[98] H Vajaria, T Islam, P Mohanty, S Sarkar, R Sankar, and R Kasturi. Evaluation and analysis of a face and voice outdoor multi-biometric system. *Pattern recognition letters*, 28(12):1572–1580, 2007.

[99] Douglas A Reynolds and Richard C Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *Speech and Audio Processing, IEEE Transactions on*, 3(1):72–83, 1995.

[100] Steve Young, Gunnar Evermann, Dan Kershaw, Gareth Moore, Julian Odell, Dave Ollason, Valtcho Valtchev, and Phil Woodland. The htk book. *Cambridge University Engineering Department*, 3:175, 2002.

[101] Jean-Michel Marin, Kerrie Mengersen, and Christian P Robert. Bayesian modelling and inference on mixtures of distributions. *Handbook of statistics*, 25:459–507, 2005.

[102] Taiping Zhang, Yuan Yan Tang, Bin Fang, Zhaowei Shang, and Xiaoyu Liu. Face recognition under varying illumination using gradientfaces. *Image Processing, IEEE Transactions on*, 18(11):2599–2606, 2009.

[103] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

[104] David L Hall and James Llinas. An introduction to multisensor data fusion. *Proceedings of the IEEE*, 85(1):6–23, 1997.

[105] Ara V Nefian, Luhong Liang, Xiaobo Pi, Xiaoxing Liu, and Kevin Murphy. Dynamic bayesian networks for audio-visual speech recognition. *EURASIP Journal on Advances in Signal Processing*, 2002(11):1274–1288, 1900.

[106] Martin Heckmann, Frédéric Berthommier, and Kristian Kroschel. Noise adaptive stream weighting in audio-visual speech recognition. *EURASIP Journal on Applied Signal Processing*, 2002(1):1260–1273, 2002.

[107] Kieron Messer, Jiri Matas, Josef Kittler, Juergen Luettin, and Gilbert Maitre. Xm2vtsdb: The extended m2vts database. In *Second international conference on audio and video-based biometric person authentication*, volume 964, pages 965–966. Citeseer, 1999.

[108] John S Garofolo, Lori F Lamel, William M Fisher, Jonathon G Fiscus, and David S Pallett. Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1. *NASA STI/Recon Technical Report N*, 93:27403, 1993.

[109] Mike Brookes et al. Voicebox: Speech processing toolbox for matlab. *Software, available [Mar. 2011] from www. ee. ic. ac. uk/hp/staff/dmb/voicebox/voicebox. html*, 1997.

[110] KP Strohl, SL Merritt, J Blatt, AI Pack, F Council, and S Rogus. Drowsy driving and automobile crashes. nccdr/nhtsa expert panel on driver fatigue and sleepiness. *Retrieved July*, 15:2008, 2004.

[111] Smart eye: Revolutionary eye tracking technology. http://www.smarteye.se/, 1999.

[112] R. Valenti and T. Gevers. Accurate eye center location and tracking using isophote curvature. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1 –8, 2008.

[113] Junwen Wu and Mohan M Trivedi. Simultaneous Eye Tracking and Blink Detection with Interactive Particle Filters. *EURASIP Journal on Advances in Signal Processing*, 2008(1):823695, 2008.

[114] Sangwook Kim, Byunghun Hwang, and Minho Lee. Gaze tracking based on pupil estimation using multilayer perception. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 2683 –2689, 2011.

[115] Inho Choi, Seungchul Han, and Daijin Kim. Eye detection and eye blink detection using adaboost learning and grouping. In *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*, pages 1 –4, 2011.

[116] Zhiwei Zhu and Qiang Ji. Robust real-time eye detection and tracking under variable lighting conditions and various face orientations. *Comput. Vis. Image Underst.*, 98(1):124–154, 2005.

[117] A. Blake and M. Isard. The condensation algorithm - conditional density propagation and applications to visual tracking. In *Advances in Neural Information Processing Systems*, pages 36–1. The MIT Press, 1996.

[118] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proc. CVPR*, 1:511–518, 2001.

[119] R. Lienhart and J. Maydt. An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–900 – I–903 vol.1, 2002.

[120] S.M. Kay and C.P. Carbone. Vector space solution to the multidimensional yule-walker equations. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, volume 3, pages III – 289–92 vol.3, 2003.

[121] David Reynard, Andrew Wildenberg, Andrew Blake, and John Marchant. Learning dynamics of complex motions from image sequences, 1996.

[122] Baback Moghaddam and Alex Pentl. Probabilistic visual learning for object detection. pages 786–793, 1995.

[123] K Grauman, M Betke, J Lombardi, J Gips, and GR Bradski. Communication via eye blinks and eyebrow raises: Video-based human-computer interfaces. *Universal Access in the Information Society*, 2(4):359–373, 2003.

[124] Chinnawat Devahasdin Na Ayudhya and Thitiwan Srinark. A method for real-time eye blink detection and its application.

[125] Michael Chau and Margrit Betke. Real time eye tracking and blink detection with usb cameras. Technical report, Boston University Computer Science Department, 2005.

[126] Ric Heishman and Zoran Duric. Using image flow to detect eye blinks in color videos. In *Applications of Computer Vision, 2007. WACV'07. IEEE Workshop on*, pages 52–52. IEEE, 2007.

[127] Thomas Brandt, Ralf Stemmer, and Andry Rakotonirainy. Affordable visual driver monitoring system for fatigue and monotony. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 7, pages 6451–6456. IEEE, 2004.

[128] Inho Choi, Seungchul Han, and Daijin Kim. Eye detection and eye blink detection using adaboost learning and grouping. In *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*, pages 1 –4, 2011.

[129] Kohei Arai and Ronny Mardiyanto. Comparative study on blink detection and gaze estimation methods for hci, in particular, gabor filter utilized blink detection method. In *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*, pages 441–446. IEEE, 2011.

[130] Jane C Stutts and William W Hunter. Driver inattention, driver distraction and traffic crashes. *ITE Journal*, 73(7):34–45, 2003.

[131] David L Strayer and Frank A Drews. Profiles in driver distraction: Effects of cell phone conversations on younger and older drivers. *Human factors*, 46(4):640–649, 2004.

[132] National Highway Traffic Safety Administration. Blueprint for ending distracted driving. dot hs 811 629. u.s. Technical report, Department of Transportation,Washington, DC., 2012.

[133] National Highway Traffic Safety Administration. Traffic safety facts, research note dot hs 811 737. u.s. Technical report, Department of Transportation,Washington, DC., 2013.

[134] Robert Held, Ankit Gupta, Brian Curless, and Maneesh Agrawala. 3d puppetry: a kinect-based interface for 3d animation. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 423–434. ACM, 2012.

[135] Theo Watson and Gobeille Emily. Puppet parade ?kinect arm tracker by design-io. https://github.com/ofTheo/kinectArmTracker, 2012.

[136] Youding Zhu and Kikuo Fujimura. Constrained optimization for human pose estimation from depth sequences. In *Computer Vision–ACCV 2007*, pages 408–418. Springer, 2007.

[137] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[138] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM Siggraph Computer Graphics*, volume 21, pages 163–169. ACM, 1987.

[139] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[140] Qi-Chuan Tian, Quan Pan, Yong-Mei Cheng, and Quan-Xue Gao. Fast algorithm and application of hough transform in iris segmentation. In *Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on*, volume 7, pages 3977–3980 vol.7, 2004.

[141] Tsuyoshi Kawaguchi and Mohamed Rizon. Iris detection using intensity and edge information. *Pattern Recognition*, 36(2):549–562, 2003.

[142] Yanfang Zhang, Nongliang Sun, Yang Gao, and Maoyong Cao. A new eye location method based on ring gabor filter. In *Automation and Logistics, 2008. ICAL 2008. IEEE International Conference on*, pages 301–305. IEEE, 2008.

[143] Tom Heyman, Vincent Spruyt, and Alessandro Ledda. 3D Face Tracking and Gaze Estimation Using a Monocular Camera. In *Proceedings of the 2nd International Conference on Positioning and Context-Awareness (PoCA 2011)*, page 23–28, Brussels, Belgium, 2011.

[144] Microsoft Kinect. Kinect face tracking. http://msdn.microsoft.com/en-us/library/jj130970.aspx, 2013.

[145] Gareth J Edwards, Christopher J Taylor, and Timothy F Cootes. Interpreting face images using active appearance models. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 300–305. IEEE, 1998.

[146] Jörgen Ahlberg. Candide-3-an updated parameterised face. 2001.

[147] Hyeon-Kyu Lee and Jin-Hyung Kim. An hmm-based threshold model approach for gesture recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(10):961–973, 1999.

[148] Sy Bor Wang, Ariadna Quattoni, L-P Morency, David Demirdjian, and Trevor Darrell. Hidden conditional random fields for gesture recognition. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1521–1527. IEEE, 2006.

[149] Kouichi Murakami and Hitomi Taguchi. Gesture recognition using recurrent neural networks. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, pages 237–242. ACM, 1991.

[150] Tracy Westeyn, Helene Brashear, Amin Atrash, and Thad Starner. Georgia tech gesture toolkit: supporting experiments in gesture recognition. In *Proceedings of the 5th international conference on Multimodal interfaces*, pages 85–92. ACM, 2003.

[151] Youngjae Kim, Youmin Kim, and Minsoo Hahn. Detecting driver fatigue based on the driver's response pattern and the front view environment of an automobile. In *Universal Communication, 2008. ISUC'08. Second International Symposium on*, pages 237–240. IEEE, 2008.

[152] Steven Beauchemin, Parisa D. Z. Varcheie, Langis Gagnon, Denis Laurendeau, Martin Lavallire, Thierry Moszkowicz, Florent Prel, Na B, and Gv A. Cobvis-d: A computer vision system for describing the cephalo-ocular behavior of drivers in a driving simulator.

[153] Shuo Chen and Chengjun Liu. Fast eye detection using different color spaces. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pages 521 –526, 2011.

[154] K. Fujimura. Real-time eye detection and tracking for driver observation under various light conditions. *Intelligent Vehicle Symposium, 2002. IEEE*, 2:344–351, 2002.

[155] Richard Grace and Sonya Steward. Drowsy driver monitor and warning system. In *International driving symposium on human factors in driver assessment, training and vehicle design*, volume 8, pages 201–208, 2001.

[156] John M Violanti and James R Marshall. Cellular phones and traffic accidents: an epidemiological approach. *Accident Analysis & Prevention*, 28(2):265–270, 1996.

[157] Shinjiro Kawato and Nobuji Tetsutani. Detection and tracking of eyes for gaze-camera control. *Image and Vision Computing*, 22(12):1031–1038, 2004.

[158] Abhishek Kar. Skeletal tracking using microsoft kinect. *Methodology*, 1:1–11, 2010.

[159] Haibo Li, Pertti Roivainen, and Robert Forchheimer. 3-d motion estimation in model-based facial image coding. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(6):545–555, 1993.

[160] Andrew D. Wilson and Aaron F. Bobick. Parametric hidden markov models for gesture recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(9):884–900, 1999.

[161] City car driving. City car driving - car driving simulator, car game. http://citycardriving.com/, 2013.