A FRAMEWORK FOR THE EVALUATION OF AUTOMOTIVE TELEMATICS SYSTEMS

Gennaro Costagliola, Sergio Di Martino, Filomena Ferrucci
Dipartimento di Matematica e Informatica, Università degli Studi di Salerno,via S. Allende, Baronissi, Italy

Keywords: Human-Computer Interaction, Human Factors, Virtual Reality, Automotive Systems

Abstract: The evaluation of interfaces for in-car communication and information applications is an important and challenging task. Indeed, it is necessary not only to consider the user interaction with the interface but also to understand the effects of this interaction on driver-vehicle performances. As a result, there is a strong need of tools and approaches that allow researchers to effectively evaluate such interfaces while user is driving. To address the problem in the paper we propose a framework that has been specifically conceived for such evaluation. It is based on the integration of a suitable car simulator and an in-car system and allows us to get a high amount of data and carry out repeatable tests in a safe and controlled environment. Moreover, the proposed solution is not much expensive and quite simple to set-up.

1 INTRODUCTION

In-car telematics systems have achieved in the last few years very impressive enhancements in the number of provided functionality. In fact, while the early systems supplied mainly some basic route calculations, currently the more advanced commercial systems (e.g.: BMW iDrive, Fiat Connect+ or GM onStar) allow drivers to exploit a plethora of services, such as web browsing, e-mail checking, phone calls, playing infotainment, and so on. For that reason they are also referred as *Intelligent Transportation Systems* (ITSs).

Despite this improvement, interaction with ITSs is somehow far to be well understood. This problem has a fundamental relevance, because in the automotive domain the user is normally busy in the demanding and mission-critical task of the driving. If the system requires too much attention due to a bad design of the interface, the user can be distracted from his/her main activity, with potentially fatal consequences. Many studies conducted on this argument show that distraction is the most prevalent cause of crash, accounting till 56% in the USA (Wang, 1996). Thus, currently there is a profound concern that these statistics will inflate as the potential for mental distraction increases with the growing diffusion of ITSs (Burns 2001, Tijerina 2001). Then, because safety is paramount, many institutions have identified as a short term priority the research on Human-Machine Interaction for the vehicular domain. In particular, safety evaluation of ITSs, specifically in the context of driver distraction, is an open and demanding research field.

Such evaluation could be carried out by exploiting a car simulator in a safe and controlled environment, in order to get a high amount of data and carry out repeatable tests. Currently there are available many car simulators, coming from both the market and the academia (an interesting list is provided by Inrets, 2004). Usually these products are conceived for very complex purposes, such as driver training, ergonomics evaluations, rapid prototyping, and road behavior analysis. For that reason, they require advanced computational resources to handle the huge amount of numerical data resulting from the simulation of several complex phenomena. Consequently, such existing solutions are typically very expensive (both in terms of required hardware and software) and/or necessitate very advanced skills for their set-up.

However, at the best of our knowledge, none of these products has been specifically conceived for the evaluation of ITS user interfaces and their impact on driver distraction. Thus, there is the need for simpler and cheaper solutions, expressly suited to address this problem. To this aim, we designed and implemented a solution, which turns out to be much more economical and easy to set-up than others, and at the same time it allows us to perform effective automotive user interfaces assessments.

Costagliola G., Di Martino S. and Ferrucci F. (2005).
A FRAMEWORK FOR THE EVALUATION OF AUTOMOTIVE TELEMATICS SYSTEMS.
In *Proceedings of the Seventh International Conference on Enterprise Information Systems*, pages 3-11
DOI: 10.5220/0002532500030011

The proposal is composed of three main modules: a driving simulator, a telematics system simulator, and some facilities suited to support running tests and to enhance driver sense of presence in the virtual scene. As for the car simulator, we successfully adopted the *Racer* free simulation engine, which offers a wide set of features useful for our purposes. Instead the ITS was implemented by our team exploiting some automotive rapid prototyping tools. Both the components can execute on traditional hardware in order to limit the costs. Finally, the further facilities are standard electronic equipment, such as some video-cameras and a SVGA projector.

In this paper we report on the experience we gained from that project, in order to allow other research centers to easily set-up similar solutions.

The remainder of the paper is structured as follows. In section 2 we will introduce the main aspects to consider when dealing with driver distraction, as well as the approaches used to evaluate it. In section 3 we will propose a generic cost-effective framework for ITS user interface evaluations, while in section 4 we will describe how we implemented such framework, detailing the adopted technical solutions. In section 5 we will focus on an interesting feature we developed, i.e. the interaction between the driving simulator and the navigation assistance software. Finally, a discussion on final remarks and future work will conclude the paper.

2 DRIVER DISTRACTION

The recent enhancements in ubiquitous computing and telecommunication systems have generated a strong momentum of convergence between these technologies and find in the automotive telematics a very interesting field of application. Indeed, current commercial ITSs are becoming even more some kind of traditional PC, able to connect to the WWW, check mail, play MP3 or DVD. Unfortunately, such growth in the number of services offered has not been paired by equivalent improvements in the usability of these systems. Indeed, this is an issue quite recent and somehow far to be well-established. Moreover, it is widely recognized that the specific questions inducted by the vehicular domain do not allow designers to transpose HCI techniques, approaches, and interaction metaphors established for traditional desktop environments (e.g. Marcus 2004). The main difference is that when designing desktop applications, designers can make the assumption that the user's attention will be mainly focused on the interaction with the system. On the contrary, when dealing with the automotive domain designers cannot rely on a significant user attention, because the interaction with an information system is only one task among the several actions achieved at the same time by the user. In particular, the user performs the main task of driving, and concurrently (s)he can also do a set of *secondary tasks*, involving interactions with entertainment systems, climate controllers, navigation aids, etc…

It is widely recognized that the use of an ITS requires driver's visual and cognitive resources (Gellatly, 1997). If these demands are "excessive" then his/her performance on the primary task of driving may be degraded. If this co-occurs with other external unexpected events, a crash or a near miss may result. Many efforts have been devoted in the literature for understanding mechanisms behind driver distraction inducted by ITSs. However, there is still much research to carry out about the interaction with these systems: the current situation recalls what happened in the '70, with the proliferation of many different attempts to design the "definitive" Graphical User Interface (GUI). Similarly, nowadays we can find on the market dozens of different approaches, devices, and metaphors for vehicular systems.

As a result, there is a strong need of tools and approaches allowing researchers for an effective evaluation of these interaction proposals.

2.1 Evaluation of Driver-Vehicle Performances

From all the issues exposed above, it is clear that the evaluation of an ITS User Interface (UI) is something far from the evaluation of traditional UIs. Indeed, it is necessary not only to consider the user interaction with the interface but also to understand the effects of this interaction on driver-vehicle performances.

2.1.1 Methods

Static evaluations of vehicular systems carried out when user is totally focusing on the system, provide not much information about the effectiveness of the UI. Instead, it is necessary to set up a meaningful test-bed where the user is mainly focused on the primary task of driving and concurrently interacts with the system. Moreover, such test-bed should allow researchers to evaluate driver performances by taking into account some useful indicators. To set up such kinds of test-bed, usually the following two approaches are adopted: (I) the interaction with an ITS is analyzed while the user is driving a real car on a track closed to the traffic, or (II) the driving is simulated in a laboratory. Each of the two approaches presents some advantages and drawbacks. The former one is probably more realistic, because the user drives a “real” car, but it requires the availability of a closed track and a car equipped with specific instrumentation able both to capture information such as travel speed and lane position and to video record the road scene and driver eye glance (e.g. Tijerina, 1998). However, the major drawback of this approach derives form the difficulty of exactly reconstructing a complex scenario (involving asynchronous events) to replicate the experiment, which is essential to effectively assess the UI.

On the contrary, driving a car simulator has the substantial advantage that tests are accomplished in a safe and controlled environment, where the risk of personal injury and property damage is eliminated. Moreover, it is more comfortable for researchers, which can get a higher amount of data and carry out more repeatable tests, by presenting to different users the same scenario.On the other hand, the use of car simulators is effective to evaluate many different and complex aspects concerning with the automotive research. As a matter of fact, several universities, companies and research centers, such as the UMTRI, the NADS and the Iowa University, have realized sophisticated laboratories equipped with car simulators. These systems are usually intended as “complete” driving simulators, able to simulate a high variety of physical phenomenon ranging from the kinematics effects inducted by different suspension geometry, to very complex traffic scenarios. Nevertheless, these laboratories usually cost hundreds of thousands of dollars and are very difficult to set-up. As an example, the outstanding simulation facilities installed at UMTRI have a total cost of over than $ 250.000 (Green, 2003).

2.1.2 Metrics and parameters

In order to assess an ITS UI it is important to quantify the safety degree of the considered ITS. Nevertheless, safety cannot be directly measured (probably except in retrospect) (Tijerina, 2001). Thus, several indirect measures of safety have been proposed that are based on the evaluation of driver distraction inducted by the system (e.g. CAMP, 2000). Summarizing, it is possible to say that distraction can be both visual and cognitive (looked-but-did-not-see). This leads towards to two main drawbacks: degraded vehicle control and degraded object/event detection (Brown, 1994). Usually, the former situation arises when the driver’s eye glances away from the road scene (without taking into account factors such as driver fatigue) resulting in problems in lane-keeping, speed maintenance, etc… The latter instead is usually due to an excessive cognitive workload (for example inducted by a cell call), and is a more insidious to evaluate, because vehicle control remains largely unaffected but detection and reactions of unexpected object and event is degraded (Tijerina, 2001).

These considerations suggest several indicators to take into account to measure driver distraction. As an example, measurement of speed maintaining performance is a good indicator for the evaluation of visual attention, but says nothing about the selective withdrawal of attention that might be inducted by an excessive cognitive workload (Tijerina, 2001). Other indicators are driver eye glance behavior, durations, and scanning patterns, lane-keeping, speed maintenance, car following performance, and driver reaction times to asynchronous events. Finally, measures of the in-vehicle task, such as task completion time, have been used or are being proposed as an index of the distraction potential of a device (Green, 1998).

3 THE PROPOSED FRAMEWORK

The purpose of our research was to implement a framework for supporting the evaluation of automotive telematics system user interfaces. The main goals of our proposal were:

- To be specifically suited for telematics assessment, i.e. don’t caring about extreme realism or other simulation aspects, such as road conditions, different engine types, kinematics of suspensions, etc...,
- To effectively support running tests, i.e. easily collect the needed data about subjects behaviors,
- To be cost-effective both in hardware and human resources, i.e. being able to execute on standard, economic hardware, without requiring complex installations or set-ups.
- To allow us to test the navigator module in the virtual environment. This implies that the driving module and the navigator have to share the same map and the information about the car position. It is worth to point out that currently usability evaluations of navigation systems are performed using real cars and not simulators (e.g.: Tijerina, 1998), because, at the best of our knowledge, currently there aren’t simulation environments offering this fundamental feature.

Such evaluation framework is intended as a composition of three main kinds of facilities, i.e. a driving simulator, a telematics system, and some instrumentation to record subject’s interactions. In the following subsections we will detail the characteristics of these components, while in section

4 we provide a more deeper description of the framework we set-up in our lab and in section 5 we will describe the link between the driving simulator and the navigator module.

3.1 The Driving Simulator

The main aim of this simulator is to propose a realistic driving environment, which should facilitate running experiments. In the meantime test subjects should receive credible feedbacks from their actions (e.g.: steering wheel shake when leaving the lane and going off-road), as well as feel a sense of presence in the virtual environment (Green, 2003). These goals may be achieved by a simulator able to:

1. Provide a realistic dynamics program governing the behaviour of the virtual vehicle.
2. Provide realistic rendering of the scenario with a frame rate of at least 30 fps.
3. Enhance the sense of presence in the virtual scenario. This can be achieved by projecting the simulated scenario onto a wide-screen that covers a significant subject's angle of view, by providing a realistic spatial audio, by using at least a 5.1 surround system, and by providing realistic force-feedbacks on the steering wheel.

To ensure the effectiveness for the evaluation, the simulator should provide further some specific features. Among these, there is the possibility to generate asynchronous external events to test driver workload. For example, other simulated cars on the track with their own behavior (e.g. braking, turning, etc…) can add much meaningfulness to the simulation.

Another fundamental aspect for supporting experiments is the *telemetry* logging, i.e. the recording of the numerical data on what the car and the driver are doing. This because by analyzing this information it is possible to better understand user's behaviors and feedbacks to specific events, recognizing potential degraded vehicle controls (i.e. problems in lane-keeping or speed maintenance) or degraded object/event detection (i.e. abnormal delay between an asynchronous event and driver response). Moreover, having an history of these data, it is possible to compare driver performance when altering external factors, such as different sensorial channels used to provide information to the user, or different layouts/organizations for graphical/vocal user interfaces. The most relevant information to store deals with the vehicle dynamics, the asynchronous events generated by the simulator (i.e. the traffic), and the user inputs. For example, basing on this set of data, it is possible to recognize degraded vehicle controls (i.e. problems in lane-keeping or speed maintenance) or degraded object/event detection (i.e. abnormal delay between an asynchronous event and driver response).

Finally, the driving simulator should provide some user-friendly tools for designing tracks and scenarios.

3.2 The ITS Simulator

About the telematics simulator, its main characteristic regards the possibility to easily define or modify the User Interface. Indeed, in order to assess different proposal, the simulator should permit to change the appearance and the behavior of the widgets composing the interface, to modify their displacement on the screen (in order to verify different layouts) and to rearrange the menu item clustering. Moreover, if the assessment regards also multi-modal aspects, the simulator should provide speech-to-text and text-to-speech technologies, and even some primitives for defining haptic feedbacks.

Finally, this simulator should also facilitate running experiments, i.e. it should collect significant data and measurements about both the asynchronous events generated by the telematics system (i.e. route guidance indication or an incoming call), and the subject behaviors.

About non-functional requirement, the simulator should execute on traditional hardware. If the system is developed together with an automotive OEM, the use of standard embedded technologies can add great value, permitting the porting of some modules on automotive hardware.

3.3 Other facilities

To complete the framework, there is the needing of some other instrumentation, useful to perform comprehensive data collection about subject's action and distraction, such as eye-tracking. This can be accomplished with a set of standard video-cameras, placed in hidden spots. The minimal configuration, as suggested by (Green, 2003), consists of three cameras, one recording the subject's face, one the vehicle interior, and one the forward scene. The first shot reflects anxiety and difficulties with a task, showing in the meantime the eye's glance and where subjects are looking. The second one shows control use and may be analyzed to determine task times and the number of errors, while the third one is useful to show the primary source of demand.

4 AN IMPLEMENTATION OF THE FRAMEWORK

In the last three years, there was a strong collaboration between our faculty and the HMI department of one of the most important European automotive manufacturers. As a result, we defined some novel ITSs interfaces. Thus we needed some facilities to assess these proposals, having however strong economical constraints about the resources we could dedicate to this aim.

By conducting deep evaluations on open-source driving simulators, and by developing some specific applications, we were able to set-up a test-bed facility matching the requirements exposed in the previous section. In particular, we implemented the above depicted framework by integrating two different modules: a free driving simulator, *Racer* (Van Gaal, 2000), and a prototype of next generation telematics system we developed.

About the major features provided by the framework, it offers an extensive data logging of driver inputs and vehicle motion, the audio/video recording of user actions, and the possibility to define arbitrary tracks, with basic traffic characteristics. Moreover, a distinguish feature of our proposal is the possibility to tightly connect the simulator and the telematics system, by sharing the same track/map, as will be detailed in section 5. This allows us to conduct extensive and effective assessment on the navigator module.

4.1 The basic architecture

The proposed system is mainly based on two software modules, running on two different PCs. The resulting architecture, shown in Figure 1, is composed of:

- A graphical workstation, suited to run the simulation engine. In our lab we adopted an HP Ewo W6000, based on an Intel Xeon 2,8 Ghz, 512 Mb of RAM, an nVidia Quadro4 video card and a Creative Labs Audigy audio card; the operating system is Windows XP Pro, SP2.
- A tablet PC, suited to execute the telematics system simulator. In our lab we adopted an HP tc1100, 512 Mb RAM, running Windows XP Tablet edition.
- A force-feedback wheel, with rudders. We selected the Logitech Formula Force GP, which seemed us a very good compromise between price and offered features;
- A 5.1 audio system. We selected the Creative Labs MegaWorks THX 5.1;
- A SVGA projector;
- At least 3 cameras, to record respectively driver eye glance, interaction with the "dashboard" and the whole simulation scenario.

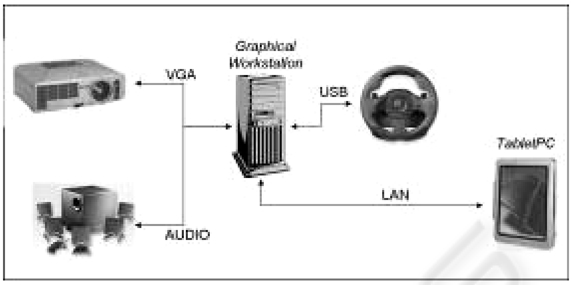

Figure 1: The architecture of the framework

4.2 The Simulation Engine

Currently they are available a lot of car simulation engines suitable for HMI evaluation purposes, ranging from big and expensive commercial solutions, such as *GlobalSim HyperDrive* (GlobalSim), to small, free and/or open source projects, such as *Torcs*. Usually the formers are mainly focused on simulating with the highest detail the physics of a vehicle, but they usually have high costs and require the set-up of lots of parameters to start the simulation. On the other hands, the latter very often are focused on providing fun more than accuracy in simulation, being intended as videogames. After a wide-ranging evaluation, we selected the engine named *Racer*, a free, open-source car simulation project, because we experienced that with some particular adjustments to the configuration files, it was able to accomplish all the fundamental tasks required for the HMI evaluation. Indeed, among the main advantages of this engine, it provides satisfactory physics by using 6 DOF models and motion formulae from SAE, it is very flexible because almost all simulation parameters are customizable through ASCII files, there is a good documentation about the file formats, it supports force-feedback devices, it provides high-quality OpenGL rendering (as visible in Figure 2), the tracks and the scenes can be created with relative simplicity through many free user-friendly editors, and last but not least, it is totally free for non-commercial use.

Finally, *Racer* allows for a basilar simulation of traffic conditions, exploiting the features related to the AI. In particular, the simulation engine allowed us to program different vehicles to follow specific routes and behaviors on the track.

Figure 2: A Racer screenshot

4.3 The telematics system

In 2003, the Department of Mathematics and Informatics of the University of Salerno and the Fiat research centre “Elasis” started an EU granted project aimed to realize a prototype of next-generation telematics systems. The main goals of such a prototype were to define an architectural model for the development of future ITSs, to evaluate the risks inducted by novel technologies such as wireless protocols, Bluetooth profiles, etc…, to evaluate the risks, the costs and the benefits of novel services, such as tele-aid, remote diagnostics, etc…, and to conduct usability tests on novel multimodal interfaces, encompassing vocal, video and tactile interaction.

Figure 3: Screenshot of the ITS user interface

The system has to provide a wide set of features, such as GPS Navigator, Entertainment section (Tuner, CD, MP3- Wma, DVD, DivX), Phone Cell (calls, SMS), “*@ module*” (WWW, e-mail), and “*Innovative Services*” (remote diagnostics, accident prevention, tele-aid, etc…). Moreover, it has to exploit a wide set of Bluetooth protocol profiles, such as SAP, headset, Sync, FTP, etc, in order to interact with the typical tomorrow’s Personal Area Network devices. The prototype was implemented using C#, for Microsoft *.NET* platform.

The graphical and haptic user interface developed for the prototype are described in (Costagliola, 2004c), and shown in Figure 3, while the vocal user interface was presented in (Costagliola, 2004b).

For this prototype we defined a specific architecture, characterized by a sharp division between logics and interface. This allowed us to successfully employ the system, together with the simulator, to assess the distraction inducted by different interface layouts and innovative services, as well as the effectiveness of multi-sensorial interactions. Indeed, thanks to a meta-UI generator based on XML, the prototype consents to modify with minimal efforts the widgets composing the UI, their disposition on the screen and the menu clustering.

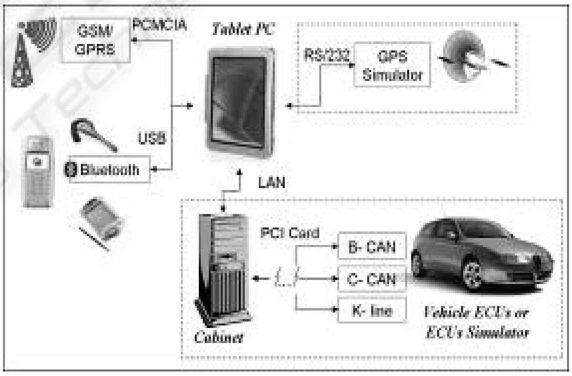

Figure 4: The architecture of the developed prototype

Further implementation details of the system, as well as a description of the innovative and flexible design pattern we defined for the development can be found in (Costagliola, 2004a). The resulting architecture is depicted in Figure 4. Some modules have been modified from the original project, in order to best fit simulation needs. In particular, the modules surrounded by dashed lines have been replaced by some signals generated by the simulation engine.

Table 1: Logged data
Racer log file generated at 05-11-2004 12:15:03

time	Steer	force_feed	throttle	brake	vx	vy	vz	x	y	z
9895	-8507501	0.000340	0.000000	0.398000	0.001296	-0.000058	0.003972	-57678322	0.516207	245962250
9900	-8507501	0.000443	0.000000	0.398000	0.001786	-0.000051	0.004131	-57678318	0.516207	245962265
9905	-8507501	0.000525	0.000000	0.398000	0.002164	-0.000044	0.004230	-57678314	0.516207	245962280
9910	-8507501	0.000598	0.000000	0.398000	0.002429	-0.000038	0.004320	-57678310	0.516207	245962296
9915	-8507501	0.000662	0.000000	0.398000	0.002599	-0.000033	0.004403	-57678307	0.516207	245962311

4.4 Logging of information

In our system, to record the simulator data, we took advantage of the logging feature provided by Racer. Indeed, simply setting the parameter *log.enable* to 1 in the *debug.ini* file, it is possible to activate the registration of the simulation data. Moreover, changing other parameters in the *log* section of the debug.ini file it is possible to indicate what information to store and the sample frequency. In our simulator, then, the data coming from the simulator are stored in a log ASCII file at a frequency of 20 Hz. An example of such data is shown in Table 1: every 50 milliseconds, we keep track of the Steering position, the force-feedback provided on the wheel, the value of the throttle and brake, the vehicle speed on the 3 axis, and its position on the track. As for the telematics systems, we store all those information, together with the time they took place, in a space separated ASCII file. This allow for an easy data analysis with tools like Microsoft Excel or SPSS.

4.5 Total costs

Table 2: Costs for setting-up the simulator

Item	*Cost*
Graphical Workstation	4,500€
Tablet PC	1,400€
SVGA Projector	750€
Logitech Wheel w/ Force Feedback	60€
5.1 Audio System	100€
#3 Digital Cameras	1,500€
Navtools SDK	5,000€
Total	**13,310€**

In our opinion, one of the main advantages of our proposal is in the trade-off between costs and offered functionality. In Table 2 there are summarized the costs we sustained for the start-up of the simulator. As one can see, following our approach, with less than 15.000 € it is possible to set-up a test-bed for automotive HMI evaluation, which is a very significant reduction if compared with the hundred of thousands dollars usually required for other driving simulators.

5 INTEGRATION BETWEEN THE TWO SIMULATORS

A distinguishing feature offered by our implementation of the framework is the integration between the simulation engine and the navigator module on the telematics systems. This means that the road driven by the user on the car simulator is shared as a map on the ITS. This permits to exploit many standard navigation features, such as Map Display and Route Guidance, namely the process of generating and then providing to subjects turn-by-turn graphical/vocal directions for a calculated route. Such integration is a very powerful instrument, because it allows us to perform many significant route guidance experiments. As an example, we can evaluate the best modality for providing route guidance to the user (vocal, iconic, etc…), or the most appropriate vocabulary to support the way-finding, as well as assess the cognitive work inducted by these different modalities. At best of our knowledge, there are no simulators offering such characteristic.
In the following we will describe how we have implemented such integration by illustrating how the driving simulator and the ITS share the same cartographical information. Moreover, we will describe how the navigator is aware of the actions made by the subject in the driving simulator, in order to update in real time the position of the car shown on the navigator map, and to undertake the necessary Route Guidance actions.

5.1 Sharing the cartography

The navigator we implemented is based on *NAVTEQ* technology. This is one of the two standard global cartographical databases adopted for automotive systems (the other one is *TeleAtlas*). Navteq provides a useful SDK to create navigation system applications and adopts the open format *SDAL* (SDAL, 1999) for the navigation map database. The SDK comes with some maps, such as the European one.

On the other hand, *Racer* adopts its own graphical file format to represent the tracks, named *DOF1* and based on the SGI IFF file format. DOF1 exploits OpenGL XYZ coordinate system and contains all the information about the scene graph of the model. In particular, it holds data about the geometry objects composing the track, i.e. information about the vertices and the normals, together with many other data, such as the texture used to render the surfaces.

Taking into account the adopted file formats, there are two ways to share the cartography between these two simulators. The first one is to create a SDAL map starting from a DOF1, while the second one is the reverse approach.

Some initial trials we conducted to create a SDAL file starting from a *Racer* track gave us bad results, mainly because SDAL file format is very complex. Indeed, it is principally focused on optimization because it is conceived for automotive systems, which usually have restricted hardware resources. As a matter of fact, it makes use of specific and different data structures for the various features, such as Map Display (optimized for pan and zoom), Route Calculation (organized to facilitate rapid route calculation), and Route Guidance (organized as *manoeuvre parcels* containing additional information needed for direction generation and route guidance). An overview of the data structures is provided in (SDAL, 1999). Moreover, this initial approach compelled us to give up some Navteq features, such as the estimated travel times for a given segment.

Instead creating a DOF file from an existing part of a map resulted more practical, giving us much better results. Our work then consisted in implementing a kind of translator able to create the appropriate DOF1 file starting from a small area of a SDAL map. In particular, such translator generates the geometry primitives starting from the *parcels* that are the basic units of I/O used in the SDAL format. Because Navtools SDK provides a wide set of functions to access SDAL information, the main difficulty was to correctly estimate a shared scale factor for the two files, i.e. given a *NavTech Unit*, (equal to 1/100,000 of a degree) used to store latitude/longitude in the SDAL database, to understand the corresponding value in the DOF1 file.

5.2 Updating the localization

To address the second issue concerning with updating in real-time the position of the car on the map we have let the simulator to export information about the car movements and the navigator to accept such information as if it comes from GPS sensor. In particular, it was required to get information about coordinates, speed, and heading of the car. As described in 4.4, *Racer* outputs this information in its log file. Thus, we implemented a daemon working in background on the graphical workstation, listening to the changes in the log file and, after some elaboration, sending the necessary data on a TCP socket shared with the TabletPC. Here, Navtools Vehicle Positioning System provides some procedures to access an I/O object (the socket) to compute the vehicle location. Such location can obviously be used for all necessary navigation features, such as Map Display, Route Calculation, and Route Guidance.

6 CONCLUSIONS AND FUTURE WORK

Safety on the roads is one of the main goals for everyone involved in the automotive field. The advent of ITSs can distract user from the main task of driving the car, with potentially fatal effects. Nevertheless, it has been estimated that these systems will become commonplace in the last few years. Thus, it is a short term priority to investigate solutions to enhance usability of ITSs and then limit driver distraction. Currently many research institutes across the world are involved in the definition of novel User Interfaces for automotive systems, but the evaluation of such interfaces is a challenging and expensive task. Indeed it requires non-trivial resources, intended both as a private track, cars and instrumentation, or as very specialized car simulators in labs.

In this paper we propose a solution for the evaluation of user interfaces in the automotive domain by using a simulator. Currently there are available off-the-shelf many commercial car simulators, but they are usually more oriented to represent with the highest realism all the aspects of the kinematics of a vehicle. As a result, they are very expensive (tens or hundreds of thousands of dollars) other than being relatively difficult to set-up.

Our proposal, instead, is intended to provide the features suited to evaluate the vehicle-driver's performance when using a telematics system, thus not focusing on extremely detailed simulation aspects. The consequence is that the complete simulator can be set-up with less than 15,000 $. In particular, the proposed solution is composed by a graphical workstation, running the very interesting free car simulator, *Racer*, connected via LAN to a TabletPC running an our own developed telematics system, encompassing a wide set of next-generation services, such as Bluetooth, integrated Cell Phone, vocal user interface, etc…Both the PCs are able to provide logging features of the user actions, in order to record and then analyze driver's behaviours and performances during the interaction with the ITS. Obviously this is the most relevant aspect of the simulator because it allows researches to evaluate the distraction inducted by each specific feature/User Interface of the telematics system.

Moreover, another distinguishing characteristic of our proposal is the possibility to tightly connect the simulator and the ITS, by letting them share the same track/map. This allows us to conduct very effective and detailed investigations about the interactions (and the related effects on distraction) between driver and navigator.

Finally, about future work, we are planning to exploit a new feature of *Racer*, i.e. the *multiview*, which enables to use multiple computers to render a widescreen view, as well as we are developing a tool for the rapid development of automotive user interfaces starting from visual specifications, in order to evaluate the distraction inducted by different control layouts or menu item clustering.

REFERENCES

Brown, I. D., 1994. Driver fatigue. *Human Factors*, 36 (2), 298-314

Burns, C. Lansdown, C., 2001. *E-Distraction: The Challenges for Safe and Usable Internet Services in Vehicles*

CAMP, 2000. Crash Avoidance Metrics Partnership (CAMP), *Proposed Driver Workload Metrics and Methods Project*

Costagliola, G. et al., 2004a. Towards an Architectural Design Pattern for Automotive Telematics Systems. In *ICSE 2004 workshop on Software Engineering for Automotive Systems*. IEE Press.

Costagliola, G. et al., 2004b. An Innovative Vocal Interface for Automotive Information Systems. In *ICEIS'04, 6th International Conference on Enterprise Information Systems*. ICEIS Press.

Costagliola, G. et al., 2004c. Handy: a new Interaction Device for Vehicular Information Systems. *Mobile Human-Computer Interaction - Mobile HCI 2004*, Lecture Notes in Computer Science, Vol. 3160

Inrets, 2004. List of principal driving simulators, available at http://www.inrets.fr/ur/sara/Pg_simus_e.html

Gellatly A., 1997. *The use of speech recognition technology in automotive applications*. PhD thesis, Virginia Polytechnic Institute and State University.

Globalsim Hyperdrive Simulator. Info available at www.globalsim.com

Green, P., 1999. SAE J2364 – *Navigation and route guidance function accessibility while driving*. Warrendale, PA: Society of automotive engineers.

Green P. et al., 2003, Audio-Visual System Design Recommendations from Experience with the UMTRI Driving Simulator, in *DSC North America 2003 Proceedings*, Dearborn, Michigan.

Marcus A., 2004. *Vehicle User Interfaces: the next revolution*, Interactions, 1.

SDAL File Format, 1999. Specification available at www.sdalformat.com

Tijerina L., Parmer, E., Goodman, M., 1998. Driver workload assessment of route guidance system destination entry while driving: a test track study. In *Proceedings of the 5th ITS World Congress*, Seoul, Korea.

Tijerina L., 2001. *Issues in the Evaluation of Driver Distraction Associated with In-Vehicle Information and Telecommunications Systems*

Torcs, The Open Racing Car Simulator, available at http://torcs.sourceforge.net/

Van Gaal, 2000. Racer Simulator Engine, available at www.racer.nl

Wang, J. S., et al., 1996. The role of driver inattention in crashes; new statistics from the 1995 Crashworthiness Data System. *Proceedings of 40th Annual meeting of the Association for the Advancement of Automotive Medicine*, Vancouver.