i-ETC: ISEL Academic Journal of Electronics,
Telecommunications and Computers
CETC2011 Issue, Vol. 2, n. 1 (2013) ID-9

MOBILEWHEEL
A mobile driving station

Vitor Cunha
Faculdade de Engenharia da Universidade do Porto (FEUP), Porto, Portugal
pro10003@fe.up.pt

Carlos Campos
Faculdade de Engenharia da Universidade do Porto (FEUP), Porto, Portugal
pro10005@fe.up.pt

Keywords: Human-Computer Interaction, Driving Simulator, Mobile Sensing.

Abstract: Current mobile devices (e.g., smartphones) are equipped with several sensors that allow different forms of user interaction. These devices also offer several connectivity options and a growing computing power which supports its use in new Human Computer Interaction (HCI) scenarios. This paper presents the *mobileWheel*, a system that exploits the capabilities of current mobile devices as a means of interaction with a real-time graphical driving simulation running on a desktop computer. The main aim of this work is to study the possibility of deploying several simplified driving stations to carry out driving simulation trials with scientific purposes. The developed application on the mobile device performs data acquisition from various sensors (focusing on the 3D accelerometer) and also provides different types of feedback to the user. This system represents a ubiquitous, simple and affordable alternative approach to the traditional control of virtual vehicles in driving simulators and could also be applied in other similar architectures. To evaluate and validate this approach several tests were conducted with volunteer users. In these experimental tests different driving operation modes were evaluated: combinations of throttle and gearbox controls. Based on the results gathered, we can conclude that the best combination of driving interface, the one that allows better user control over the driving task, is the use of the accelerometer to control the steering wheel coupled with the use of automatic transmission.

1 INTRODUCTION

Scientific studies related to driving simulation must conform to specific requirements so they don't undermine the conducted experimental work. Traditional driving stations, such as the one used in the DriS [1] driving simulator (Fig. 1), require an immersive environment with the presence of an actual vehicle properly equipped with sensors so the participant can control the simulation and to allow data collection on the experimental simulation to pos-processing. In the simulated road environment, in addition to the vehicle driven by the participant, there may be other agents, including other vehicles. The aim of described work in this paper is to study the practicability of a simplified driving station by studying different control interfaces and also determine the most suited for implementing multiple driving stations that are equally interactive. This approach allows that other vehicles involved in the simulation could also be controlled by actual participants, thus improving the system scalability. In the previously described context, the increasing number and variety of sensors available on current mobile devices (e.g., smartphones, PDAs, tablets) makes them attractive to use in several interaction contexts. In order to exploit the capabilities

Figure 1: DriS driving simulator.

of these devices and provide an alternative interaction in the particular situation of driving simulations, this article presents the system *mobileWheel*. The main elements are two software applications, one running on a PC and designed to provide a real-time graphical simulation of the virtual environment and the other one in a smartphone that acts as the controller and dashboard of the driven vehicle. One of the sensors with the highest potential to control the virtual vehicle is the multiaxial accelerometer, which is used in a similar way in current game consoles and by Vajk et al. in [2]. In the proposed system, the smartphone application acquires data from this sensor, processes it, and then sends it to the application on the PC via the wireless network infrastructure. Additionally, several commands and settings are input through the touchscreen. The driving simulation application logic then uses a dynamic vehicle model that, based on the data received from the mobile device, updates the graphical simulation. Within this iteration, the vehicle's dashboard displayed on the smartphone's screen is also updated with the new computed values. In addition to visual feedback, sounds and vibration are used to signal events in the simulation (e.g., collision).

The remainder of this paper is structured as follows: Section 2 presents and discusses previous works that use the accelerometer of mobile devices. Section 3 describes in detail the *mobileWheel* system. Section 4 presents the system's evaluation process and results discussion. Finally, in Section 5 the conclusions and proposals for possible future developments are presented.

2 RELATED WORK

Several already published works demonstrate the potential of the embedded acceleration sensor present in current mobile devices. Some examples include the use of this type of sensor as a means to recognize gestures to control software applications (e.g., [3]), integrate continuous sensing approaches (e.g., [4]), among many others. Lane's et al. [5] survey discusses the major possibilities offered by the variety of sensors present in most current mobile devices.

One approach that uses the data from the accelerometer as a way of controlling a game that runs on the device and that does not use its keyboard is presented by Gilbertson et al. [6]. When compared with the traditional control through the keyboard buttons, this method allowed users to achieve better scores. Vajk et al. [2] used mobile phones as controllers in a multiplayer game projected in a large screen. The authors stress that the success of this approach was due to the fact that the users didn't need to look at the device's screen, resembling a games console controller. On the other hand, Wang and Ganjineh [7] used an iPhone application as a way to control a real instrumented vehicle. In this case the accelerometer is only used to steer the vehicle while the throttle is controlled through onscreen buttons. The iPhone (and other platforms) is also used as the mobile interface to control the lowcost multicopter Parrot AR.Drone [8], in this case the device's accelerometer is used together with the touchscreen.

The proposed *mobileWheel* system allows the use of the data gathered from the 3D accelerometer, to control the direction and throttle of a virtual vehicle, while using the PC and smartphone screens to display graphical information from the simulation to the user.

3 THE MOBILEWHEEL SYSTEM

To achieve the proposed features, Fig. 2 presents the *mobileWheel* adopted client-server system architecture. The applications for the PC and Smartphone are the main blocks of this approach, which interact through the network infrastructure. A detailed description of each block is presented in the following sections.

3.1 Server Application

The real-time driving simulation application was developed using mainly the OpenSceneGraph (OSG) API [9]. This API consists of a set of libraries used for the development of 3D applications: `osg` (core), `osgUtil` , `osgDB` (reading and writing 3D database). Additionally it also provides a set of libraries for the manipulation and visualization of 3D environments, specifically the `osgGA` (GUI abstraction library) and the `osgViewer` (Viewer library) libraries. The `osgGA` library contains a set of classes and methods to help the easy creation of handlers to control the simulation. In the scope of this server application the `DriveManipulatorPDA` handler was developed to control the simulation, which was derived from the class `osgGA::MatrixManipulator` (OSG v2.8.2).

Linked to the handler was developed a dynamic model of a vehicle, where the engine and gearbox parameters are simulated. Two types of gearbox were implemented: manual and automatic. The selection between the different models in the PC server application is made by the operation settings of the application installed in the smartphone. The engine speed can vary between a minimum of 900 rpm and a maximum of 7000 rpm. The automatic transmission model

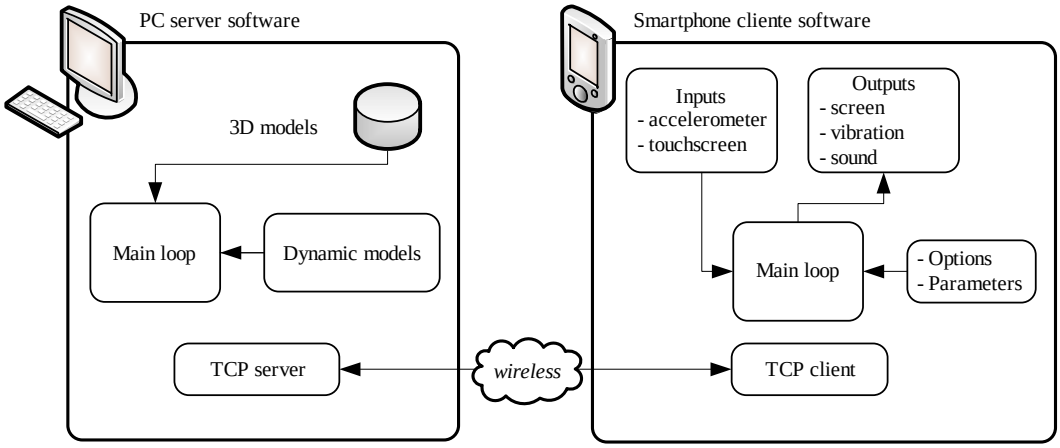

Figure 2: *mobileWheel* system architecture.

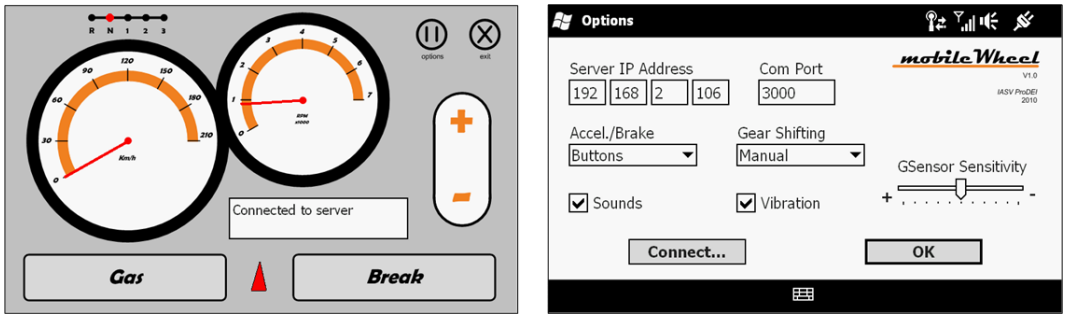

Figure 3: Driving (left) and settings (right) screens on the smartphone's application.

has three modes: Drive, Neutral and Reverse. In Neutral mode the vehicle is disengaged, where a change in the engine rotation does not produce any variation in the vehicle's speed. In Drive mode there is a linear variation between the engine speed and vehicle speed ranging from 0 km/h (900 rpm) to 200 km/h (7000 rpm). The Reverse mode implies a negative linear variation between the engine rotation and the speed of the vehicle corresponding to 0 km/h at 900 rpm up to -50 km/h at 7000 rpm.

With the manual gearbox model, in addition to the neutral mode, 4 gears (Reverse + 1^{st}, 2^{nd} and 3^{rd}) can be used. In the Reverse gear, the relationship between engine rotation and vehicle speed is the same as in the automatic gearbox Reverse mode. In 1^{st} gear the vehicle can vary from 0 km/h to 50 km/h (7000 rpm). With the 2^{nd} gear the vehicle speed goes up to 110 km/h and in 3^{rd} gear reaches the maximum speed of 200 km/h.

The communication between the PC and smartphone applications is done through a TCP/IP socket connection. At startup the PC application creates an independent thread for the TCP server, which performs all data reading and writing operations with the smartphone client application. This approach also allows the use of several mobile client controllers for more complex driving simulation scenarios.

3.2 Client Application

The smartphone client application was developed in C# (.NET CF 3.5) for the Windows Mobile platform. Given the absence of an abstraction layer for the accelerometer (and other sensors) in the Windows Mobile 6.5 operating system, the WMUS API [10] was used. This API allows that the necessary data can be retrieved from the accelerometer in the mobile device used, an HTC HD2.

Fig. 3 illustrates the graphical interface that consists of two main forms. The settings screen lets the user set the various parameters and options of the application: communication settings for connecting to the PC (server); throttle control, through on-screen buttons or the accelerometer; manual or automatic gearbox; smoothing applied to the accelerometer data; use of sounds and vibration. All settings are stored in a local XML file to avoid losing the settings between runs.

The two options for gearbox type and throttle control result in four possible combinations to control the

	Steering Wheel	Throttle	Gearbox
Data type	unsigned char	unsigned char	unsigned char
Data range	0 - 180	0, 1, 2, 3	0, 1, 2

Table 1: Client to Server data structure.

	Speed	RPM	Current Gear	Collision	Sound
Data type	unsigned char	unsigned char	unsigned char	unsigned char	unsigned char
Data range	0 - 255	0 - 255	0, 1, 2, 3, 4	0, 1	0, 1, 2

Table 2: Server to Client data structure.

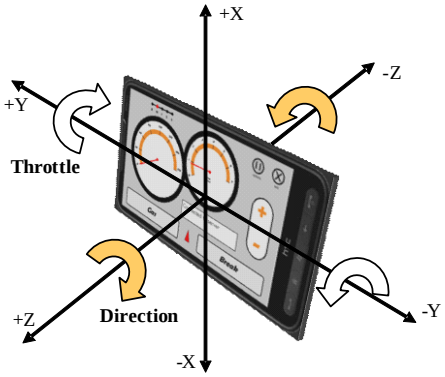

Figure 4: HTC HD2 accelerometer axes.

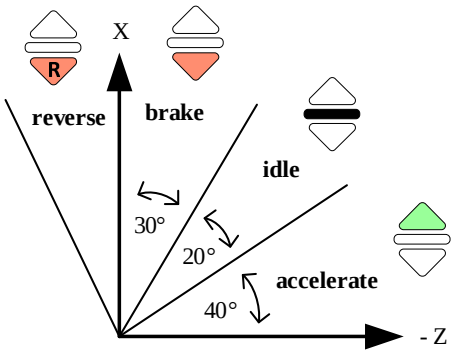

Figure 5: Throttle control action assigned to each angular range. The featured symbols are shown in the speedometer (white bottom area of the dial) to inform the user of the currently active action.

virtual vehicle. Common to all control modes is the use of the accelerometer to control the direction of the vehicle by rotating the device around the Z axis (Fig. 4). The throttle (acceleration) control based on this sensor is performed by rotating the device around the Y axis. In this case angular intervals are used (Fig. 5), where which one is related to a throttle action: accelerate, idle, brake and reverse (to gear Reverse when in automatic gearbox). The rotation of the device is converted into angular values using inverse trigonometric functions that use the values obtained from the accelerometer regarding the XZ (throttle) and XY (direction) axes. Regarding the main driving screen, this form shows the vehicle's dashboard for visual feedback (e.g., speed) and also capture commands through the touchscreen to send to the server. When the options for manual gearbox and/or throttle control through onscreen buttons are selected, graphical elements in the GUI are activated to capture those commands.

In addition to visual feedback through the graphical interface, the *mobileWheel* system includes the use of vibration and sounds. Both mechanisms are triggered when the respective command is received from the simulation on the (server) PC. A set of predefined sound files is stored in the smartphone so the simulation application only signals which should be used. Two different situations were implemented: collision (sound and vibration) and speeding over a predefined limit (sound).

3.3 Communications Protocol

The data exchange between the PC server application and the smartphone client application uses a purpose developed protocol. Data is transmitted via a TCP/IP socket connection at a rate of 10 Hz imposed by the mobile device. Table 1 shows the data structure sent to the driving simulation application on the PC. The client application sends three variables to the server: steering wheel angle, throttle command (0 - Reverse, 1 - Brake, 2 - Idle 3 - Accelerate), gearbox (0 - automatic gearbox selected, 1 - manual gearbox selected, 2 - shift one gear up, 3 - shift one gear down).

Table 2 shows the data structure sent in the opposite direction. The software application on the PC sends a structure with five variables in response to the data sent by the smartphone, in this case: speed for the speedometer (0 - 210 km/h), RPMs (0 - 7000 rpm), current gear to display in the dashboard (0 - Reverse, 1 - Neutral, 2 - Drive/1st gear, 3 - 2nd gear, 4 - 3rd gear), collision (on/off), sound file to play (0 - none, 1 - collision, 2 - over speed limit).

Figure 6: Volunteer candidate performing the driving task.

Figure 7: View of the 3D virtual road environment.

4 EVALUATION

In order to assess the HCI capabilities of the system, several tests oriented to the task of driving were performed. The analysis of the results obtained from these tests will allow determining which driving interface is best suited for controlling a simulated vehicle. The main driving task, performed by the volunteer candidates, consisted in driving in a virtual 3D environment until given the indication to stop. Candidates were given the instruction: "Drive normally as in a two-way street, without taking any turns at intersections or junctions, until given the order to stop.". The candidates, before performing the experimental task, drove freely in another virtual environment to become familiar with the system controls. During the driving task several experimental data related to the candidate's behavior and performance were collected. Each candidate performed the driving task using the four different control combinations (modes) in random order: throttle through the accelerometer with automatic gearbox; throttle through the accelerometer with manual gearbox; throttle through onscreen buttons with automatic gearbox and throttle through onscreen buttons with manual gearbox. The parameters collected during the experiments were:

- Time, measured in seconds since the beginning of the task until the order to stop was given;
- Diverted attention from the road to the dashboard displayed in the smartphone, which quantifies the number of times the candidate devalues the main driving task to operate the smartphone's screen;
- Misalignments, measuring the level of misalignment relative to the expected driving path (0 - None, 1 - Mild, 2 - Severe, 3 - Very severe);
- Evaluation of the number of times the user goes into the wrong (incoming traffic) side of the road (0 - Never, 1 - Occasionally, 2 - Often, 3 - Always);
- Off-road/Collision, number of times the candidate crashed or gone totally off-road.

Fig. 6 shows the lab environment where the experiments were carried out, while Fig. 7 pictures the 3D virtual road environment used. The sample of volunteer candidates was characterized by considering the following parameters:

- Driver's license;
- Usual driver;
- Usual computer user;
- Computer games player;
- Games console user;
- Car racing games player.

The data collected shows that all candidates have license and use a computer in a daily basis but only 50% of the candidates play driving/racing games. Fig. 8 sums up the volunteer users profile.

The data analysis methodology was based on applying weights to the different parameters gathered for each interaction mode and candidate. The scores shown in the chart of Fig 9 result from the following applied weights: Time = 1; Deviated attention from the road = 1; Misalignment = 10; Wrong lane driving = 10; Off-road/Collision = 60. With this methodology the interaction mode with the best result is the one with the highest numerical value. As illustrated in Fig. 9, the interaction mode with the best result is the one using automatic gearbox and the accelerometer to control the throttle (and direction) of the vehicle. In a first stage the tests were performed with ten candidates. After that, in order to sustain the preliminary obtained results a series of additional tests was performed for a total of 21 participants. These additional tests support the earlier results and are included in the previously presented analysis.

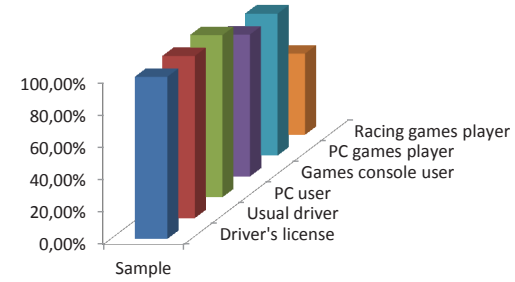

Figure 8: Sample profile.

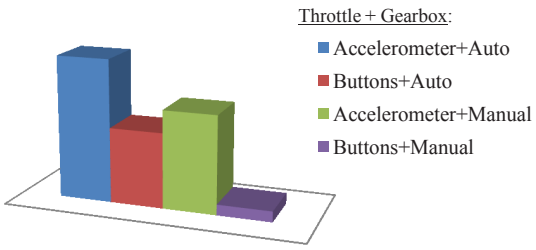

Figure 9: Total scores by control mode.

5 CONCLUSIONS AND FUTURE WORK

The presented work demonstrates the potential of interaction offered by current mobile devices, including the use of the 3D accelerometer for control. The results obtained point in the same direction of the ones reported in [2]. The sample shows that the results are consistent and provide a valid data analysis. Data was collected based on the direct observation of the driving task, and each analyst focused solely on collecting data regarding one parameter. By analyzing only the variable Time, it can be observed that the average time required by the candidates to perform the main task is lower with the configuration accelerometer/manual gearbox. This situation can be attributed to the fact that the candidates after placing the vehicle in motion, do not often use the gearbox. In all experiments, the data exchanged between the client and server applications was stored in disk files, allowing further analysis. To conclude, the main contribution of this work was the evaluation of the presented control approach in the particular scenario of real-time driving simulation. From the several interfaces evaluated to control a virtual vehicle, the one that produced the best results according to the metrics used is the combination that uses the 3D accelerometer to control the steering and speed of the vehicle (automatic gearbox). Based on the presented results it can be stated that this approach effectively enables the use of simplified driving stations in the context of driving simulators with scientific purposes.

In the future, is intended to develop the support for several simultaneous users and also to demonstrate the applicability of the *mobileWheel* system as a mobile driving station alternative in the DriS driving simulator.

REFERENCES

[1] J.M. Leit ao, A. Coelho, and F.N. Ferreira. Dris - a virtual driving simulator. In *Second International Seminar on the Human Factors in Road Traffic*, Braga, Portugal, 1997.

[2] T. Vajk, P. Coulton, W. Bamford, and R. Edwards. Using a mobile phone as a "wii-like" controller for playing games on a large public display. *International Journal of Computer Games Technology*, 2008(2):1–7, 2008.

[3] S. Strachan, R. Murray-Smith, and S. O'Modhrain. Bodyspace: inferring body pose for natural control of a music player. In *CHI '07 extended abstracts on Human factors in computing systems*, CHI EA '07, pages 2001–2006, NY, USA, 2007.

[4] H. Lu, J. Yang, Z. Liu, N.D. Lane, T. Choudhury, and A.T. Campbell. The jigsaw continuous sensing engine for mobile phone applications. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 71–84, NY, USA, 2010.

[5] N.D. Lane, E. Miluzzo, Hong Lu, D. Peebles, T. Choudhury, and A.T. Campbell. A survey of mobile phone sensing. *IEEE Communications Magazine*, 48(9):140 –150, Sept. 2010.

[6] P. Gilbertson, P. Coulton, F. Chehimi, and T. Vajk. Using "tilt" as an interface to control "no-button" 3-d mobile games. *Computers in Entertainment*, 6(3):38:1–38:13, Nov. 2008.

[7] M. Wang and T. Ganjineh. Remote controlling an autonomous car with an iphone. Technical report, Free University of Berlin, Mar. 2010.

[8] AR.Drone. http://ardrone.parrot.com.

[9] OpenSceneGraph. http://www.openscenegraph.org.

[10] Windows Mobile Unified Sensor API. http://sensorapi.codeplex.com.