Immersive Vehicle Simulators for Prototyping, Training and Ergonomics

Marcelo Kallmann[1], Patrick Lemoine[1], Daniel Thalmann[1], Frederic Cordier[2],
Nadia Magnenat-Thalmann[2], Cecilia Ruspa[3], and Silvia Quattrocolo[3]

[1]*Swiss Federal Institute of Technology (EPFL), Virtual Reality Lab (VRlab)*
Lausanne Switzerland CH-1015
{marcelo.kallmann, patrick.lemoine, daniel.thalmann}@epfl.ch

[2]*University of Geneva, MIRALab*
General Dufour 24, Geneve, Switzerland CH-1211
{cordier, thalmann}@miralab.unige.ch

[3]*Centro Ricerche Fiat (CRF), Human Factors - Physical Ergonomics*
Strada Torino, 50 - 10043 Orbassano TO - Italy
{cecilia.ruspa, silvia.quattrocolo}@crf.it

Abstract

In most vehicle simulators, complete physical mock-ups equipped with steering wheel, gearshift and pedals are required. These devices are essential in trying to simulate real conditions, but as a drawback the system becomes expensive, huge (non mobile), and limited to reflect changes on the vehicle type, dimensions, or interior design.

We have implemented different configurations for an immersive vehicle simulator, ranging from the use of a real mock-up equipped with force-feedback steering wheel, gearshift and pedals, to fully virtual control metaphors, which are based only on trackers and datagloves. We propose such fully virtual control metaphors as an alternative to minimize the use of physical devices in simulators.

keywords. virtual reality, direct interaction, real time graphics, vehicle simulators.

1. Introduction and Related Work

It is a common practice to develop vehicle simulators based on physical mock-ups. They might be equipped with hydraulic platforms [1], or based on a real car placed in front of large (some times cylindrical) projection screens [2,3,4]. The primary application of vehicle simulators is driving training and behavior analysis [5], but they have also been used for virtual prototyping issues.

Today most car manufacturers use vehicle simulators as part of product conception. Car simulators allow engineers to test the car before it is built, evaluate ergonomic aspects, interior design and even road behavior. However, for virtual prototyping issues [6,7], simulators based on physical reproduction of cabins require substantial time and cost to be manufactured. Therefore, they cannot be reconstructed each time to reflect a new part model which is updated on the CAD design. This difficulty mostly appears when performing ergonomics and visibility evaluations according to changes in the car interior.

Immersive virtual reality provides a natural alternative. Virtual prototypes can replace physical mockups for the analysis of design aspects like: layout and packaging efficiency; visibility of instruments, controls and mirrors; reachability and accessibility; clearances and collisions; human performance; aesthetics and appeal; etc. The goal is to immerse a person in the virtual car interior, able to study the design and interact with the virtual car.

Following predictions stated since the beginning of research on Virtual Reality [8], the ultimate vehicle simulator would be fully virtual, with no physical parts, and the user would experience not only realistic graphics and sound generation, but also full realistic force-feedback. Actual technology is still far to reach these requirements, and compromise solutions have to be taken.

We present in this paper a simulator system built for both training and ergonomics-related tasks, and which can be used in mainly two different configurations:

- based on a physical mock-up of the vehicle, equipped with force-feedback primary controls.
- based on a fully virtual control metaphor, allowing to interact with the vehicle only through the use of motion trackers and datagloves.

We analyze these configurations for the use on several different simulated scenarios, and identify possible tasks for which fully virtual controls are suited to be used.

2. Simulator Requirements

Our simulator software needs to handle several scenarios, which were specified by the psychologists and ergonomics partners of the project.

Visibility tasks. To analyze visibility changes (dashboard instruments, exterior environment, mirrors, etc), according to different vehicle designs.

Reaching tasks. To analyze the good positioning of primary controls, while the virtual vehicle is stopped.

Interaction Tasks. To analyze the ergonomics during interaction with the primary controls while the vehicle is static (or runs slowly). For instance, one of the interaction tasks consists of instructing the user to simply change gears one by one.

Driving Tasks. Here the user is supposed to drive the vehicle and to perform several simple driving tasks. Each driving task has few traffic decisions to be taken. They have been designed to analyze the user driving behavior while using primary controls and the facility to control the vehicle. Examples are: stop and go according to traffic lights, entering a narrow garage, reversing, parking between two cars, etc.

Training Tasks. For these tasks, the user drives the vehicle inside an animated environment with some traffic. Here the user performs some decision-making according to the traffic situation. Some example of training tasks are: to turn left with incoming traffic, to drive on a highway with other vehicles, to pass a roundabout, to turn right with bicyclists or pedestrians crossing, etc.

The simulator is also required to handle mirrors and to control force-feedback primary controls, which are mounted on a mock-up. For instance, the motorized steering-wheel behavior changes according to the speed of the vehicle, and the pedals and the gearshift behavior are fully defined by software [9,10].

Concerning the visualization, we use both Head Mounted Displays (HMD), and shutter glasses. In this last case, stereo images are projected in a large screen placed in front of the mock-up. A 6 degrees of freedom (DOF) tracker is placed in the user's head, in order to capture the viewing position and direction. When using the HMD, we have a fully virtual scenario, and an avatar (a virtual representation of the user's body) is displayed in the scene. Arms and hand fingers are animated using two additional 6 DOF trackers and datagloves. These trackers are placed on the user's hands and an inverse kinematics algorithm animates the elbow and shoulder joints according to the hands position.

Currently we use Intersense trackers, which are precise enough, although they cannot handle emitter-sensor occlusion. Magnetic trackers cannot be used because several motors and metallic parts are used in the mock-up.

3. Software Architecture

We have developed a system able to simulate all specified tasks. The system is configurable according to scripts written in Python [11]. For each task, two scripts must be given:

(a) An initialization script that configures the system parameters, loading all scenario components.

(b) A run time script, which controls the behavior of all animated objects in the scene during the simulation.

The system is distributed in two high-end PC computers (figure 1). The control computer reads the selected task script and interprets them, sending messages to update the scene graphics, which is maintained by the visualization computer.
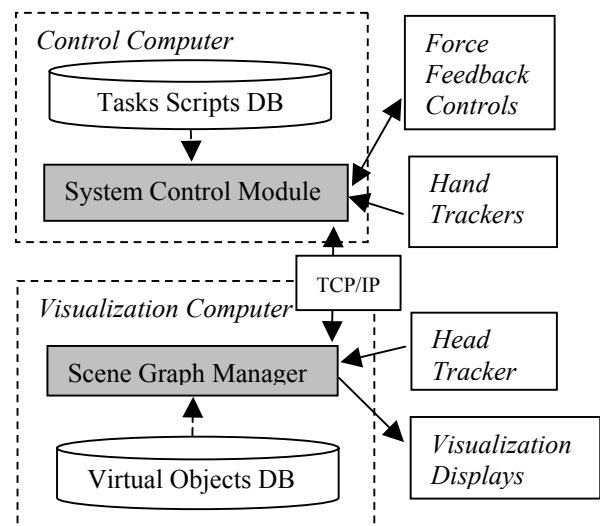
Figure 1. Simplified diagram of the system architecture: main nodes and connections.

The System Control Module in the control computer is responsible to:

• Start the application with a graphical interface permitting to select the desired script scenario to run. Scripts are loaded from a local database containing all modeled scenarios and are interpreted in run time.

• Read the position of the primary controls (steering wheel, pedals, and gearshift) and update the vehicle behavior accordingly.

• Read the values of the hand trackers. These values are used to animate the avatar (in case a HMD is being used). Also, when the primary controls are not available, the fully virtual control metaphor based on the hands position (to be presented later in the paper) will input to the system the modification applied to the position of the steering wheel and gearshift.

• Output the correct forces to the primary controls according to the vehicle state. For instance, the steering

wheel gets heavier according to the current speed of the vehicle.

• At each frame, and according to the specified frame rate, all needed updates to perform in the scene graph are sent to the visualization computer.

The Scene Graph Manager module in the visualization computer is responsible to:

• Read the messages sent by the control computer. During initialization, objects are required to be loaded from a local database. During run time, messages are mostly related to update the position of each animated object in the scenario.

• Generate stereo images for the selected display. Both HMDs and shutter glasses can be specified.

• Read the values of the head tracker. The user's head orientation is not only used to place the virtual camera, but also to select the activation of mirrors.

In order to save computation time, we have done the simplification that only one mirror is activated at a time, i.e., only the mirror closer to where the user is looking is activated. Note that the generation of each mirror image requires a full rendering of the scene from the mirrors point of view using the OpenGL stencil buffer [12] to clip the scene to the mirrors boundary. Moreover, as we display stereo images, each mirror image needs to be generated twice. In the end, in stereo mode and with one mirror activated, the scene needs to be rendered four times per frame.

4. Simulation Examples

Figure 2 presents snapshots of some simulated tasks, in order to give a better picture of the different issues taken into account by the simulator.

The top row of figure 2 exemplifies two training tasks. The city street with animated traffic and traffic lights is used to train engaging cross roads. In the highway scenario the user needs to overtake cars in order to keep an instructed constant speed.

The middle row of figure 2 illustrates one scenario used for the parallel-parking task and the garage-entering task (the garage is on the right side of the house).

The bottom row of figure 2 exemplifies one visibility test where a cylinder is placed at the same position as the left frontal pillar of the vehicle. By changing the size of the cylinder, it is possible to simulate how far the size of pillars affects the visibility of the driver.

5. System Configurations

In this section the possible configurations of the simulator system are presented, classified according to the type of interaction with the user. Note that for all configurations, different visualization modes can be used:

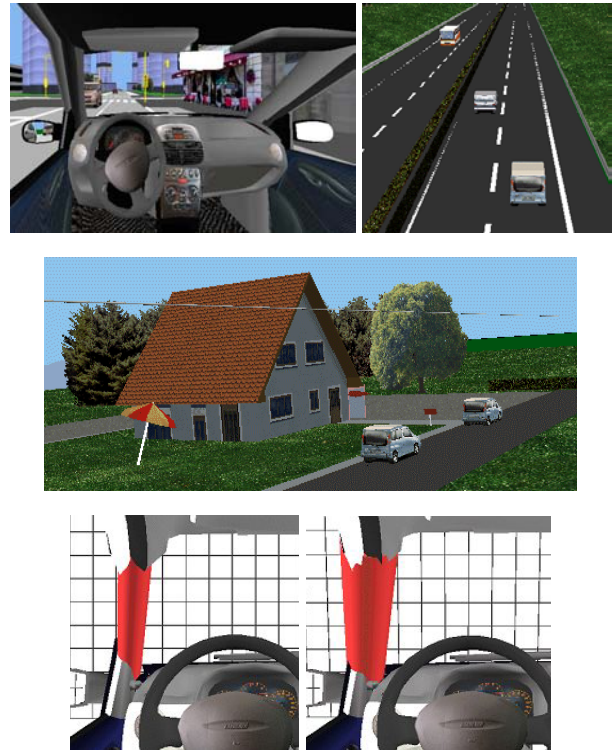using HMDs, shutter glasses with large projected screens or even a simple monitor.

Figure 2. Snapshots of some simulated tasks.

Mockup Configuration. The standard configuration for the simulator is the mockup configuration (figure 3). It is certainly the best for performing most specified tasks. However there are some drawbacks regarding the cost, and the difficulty to move it or to update it. For instance, during the development of the project, which has different partners located in different countries (Italy, Germany and Switzerland) we soon needed to develop alternatives to test and implement the simulator software without the mockup, as only one mockup was constructed (in Italy).

When HMD visualization is used, the user is completely immersed in the virtual environment, seeing its animated avatar representation, but can still interact with the real mockup. Real objects provide force feedback while their virtual counterparts provide visual feedback.

Therefore, matching real objects to their virtual models is an important task to be carried out. As an initialization procedure, we define a transformation matrix that is used to compute the position of the trackers in the virtual environment. This consists of computing the relative position of the real devices (seat, gearshift and steering wheel) from the motion capture system. This matrix is defined at once when all the devices (seat, steering wheel, gearshift and the motion capture device) are installed. The procedure requires defining three feature points in the real

and the virtual environment. They are located on objects for which high precision is required: one in the gearshift, one in the steering wheel and the last one on the seat. These positions define a coordinate system for the real world and for the virtual world.

Figure 3. The mock-up configuration using shutter glasses (left) or HMD (right).

Desktop Configuration. The first alternative developed was simply done by replacing the primary controls by standard desktop components such as joysticks or game like steering wheels. The desktop configuration is sufficient to test the animated scenarios, but it doesn't permit to interact with the primary controls corresponding to the virtual car model being simulated.

Virtual Configuration. A fully virtual configuration was created, immersing the user in the virtual environment. In this configuration we use two hand trackers to get the position of the user's hand. The first use of these positions is to give a visual feedback of the location of the users hands in the virtual environment. The user can visualize his/her hands representation in several ways: with an animated avatar, with only two flying hands, or simply with two spheres representing the tracker location. For the animation of hands, data gloves are used. A fully animated avatar gives a more realistic feedback, however for using the system many times consecutively, a point representation is a more practical solution because it avoids wearing and connecting two devices (the data gloves). Figure 4 shows the devices used by this configuration. Note that data gloves use are optional, permitting to capture or not fingers movement (in figure 4, only one dataglove is being used).

In case the avatar is displayed, two techniques are used in order to fit its animation to the user. Firstly, measurements of the user are taken and applied to the avatar. These measurements concern mainly the upper body: length of the forearms and upper arms, width of the shoulders and length of the neck and the trunk. The lower body is not animated and therefore does not need to be matched to the user size. Secondly, a method has been developed to ensure the matching of the position of avatar hands with those of the user, by using inverse kinematics to reconstruct joint angles. The usual approach would be to apply these calculated joint angles on the avatar

skeleton from the root to the hands. In such approach, the animation of end joints such as the hands lacks of precision due to the accumulation of angle errors. Our approach consists of accumulating the joint angles in the reverse order, i.e. from the hands to the root. Therefore, two skeletons are used, one for the left and one for the right arms. After applying the joint angles on one of the skeletons, the skeleton is moved in a way that the position of the hand fits to the position of the corresponding tracker. This ensures that the hands of the virtual human and the user have exactly the same position.

Tracked hands positions are also used to interact with the primary controls, permitting not only to approach reaching tasks, but also interaction tasks. For this purpose, a simple interaction metaphor was created. First, interaction regions (see figure 6) are associated to the interactive primary controls. In our case a torus approximates the steering wheel, and a sphere approximates the top of the gearshift. As these primitives can be defined by implicit equations, it becomes easy to compute containment and distance queries from a given point in space. During runtime, while a hand position moves inside an interaction region, the associated primary control moves together with the hand. This simple mechanism allows the user to manipulate the controls and actually drive the car, following the principle that interactive objects keep their own interaction information [13]. However, this interaction metaphor is not easy enough for performing driving tasks as it requires a lot of attention during manipulation. This main drawback made us to develop a smarter interface.

Smart Virtual Configuration. In this configuration, we simplified the required motor skills to interact with the steering wheel and gearshift. First, two regions are specified close to the gearshift to indicate places to activate a gear change to a lower gear or to a higher gear. For instance, we have specified that if the users hand gets closer to the gearshift, but between the gearshift and the panel, a higher gear is activated. If the hand gets into a region between the gearshift and the driver, a lower gear is activated.

Once these regions are determined, we specify that the users hands stay attached to the steering wheel most of the time. The steering rotation angle is simply deduced from the relative position of the hands, taking into account only the vertical coordinates. But when the right hand position gets closer to the gearshift and enters into one interaction region, a gear is changed, and if the avatar is displayed, its arm is automatically animated to change the position of the gearshift. We can say that, in this case, the avatar gets some autonomy, as it understands the user interaction intention and performs the interaction in the virtual world. For the moment, our systems run only with the described proximity criteria, but more sophisticated

methods to also take into account recognition of hands gestures could be integrated [14].

This smarter interaction metaphor allows the user to concentrate on the animated scenario, without the need to pay attention to the virtual controls. An analysis of the configurations is done in the next section.

Figure 4 illustrates the avatar being controlled by the user wearing VR devices. Figure 5 illustrates the case when simple spheres are used to represent hand positions. These visual spheres provide sufficient information to let the user interact with the steering wheel and gearshift in the Virtual Configuration modes. Figure 6 illustrates the notion of interaction regions.

Figure 4. The virtual configuration.

Figure 5. Spheres may also represent hand positions. Whenever a control gets attached to the user hand, the corresponding sphere changes its color.

Figure 6. Main interaction regions used: a torus approximates the steering wheel shape (left), and a sphere approximates the topmost part of the gearshift (right).

6. Analysis

The main drawbacks of the mockup configuration are related to the mockup itself: (a) its high cost, (b) its difficulty to be updated in order to reflect changes in virtual prototypes, and (c) its cumbersome size and weight which makes mobility to be a problem. These main drawbacks motivated us to develop the virtual control metaphors shown in the previous section. Table 1 presents how each configuration fits in relation to these issues.

Table 1. Configurations Characteristics

	Characteristics		
Configs.	Cost	Mobility	Prototyping use
Mockup	high	difficult	difficult
Desktop	low	very easy	not possible
Virtual	medium	easy	possible
Smart Virtual	medium	easy	possible

In summary, to overcome the limitations of the mockup configuration, the desktop configuration gives the lowest cost solution but cannot be used for virtual prototyping tasks. The two virtual configurations give a compromise solution in relation to the cost, and might also be used for virtual prototyping issues, as it becomes very easy to replace virtual objects and interact with them in a fully virtual environment.

However the virtual configurations, as well the desktop one, are not applicable to simulate all types of tasks.

The mockup configuration might be applicable for all the tasks. Even if some visibility tasks might not be well performed when physical parts of the mockup occlude a region of interest during the visibility test.

Such a drawback does not occur if HMD visualization is used. Note that all configurations can be equipped with a HMD visualization resource, so that visibility tasks can be performed equally in all configurations. We have also performed some tests in a CAVE environment, with the cooperation of the Fraunhofer Institute.

It is not possible to perform Reaching, Interaction and Driving tasks with the desktop configuration, as these tasks require the driver to be able to interact with the primary controls corresponding to the virtual vehicle being simulated.

The virtual configuration can be used for reaching tasks. But as there is no force-feedback in the virtual configurations, it is not possible to analyze ergonomics issues related to the grasping of controls. Only ergonomics aspects related to reaching distances might be performed. The smart virtual configuration is not suited to

reaching tasks simply because (in its standard form) interactions are triggered each time the user approaches the virtual primary control.

To train the decision-making and vehicle control when facing traffic situations, it is possible to use the desktop configuration and the smart virtual configuration. The virtual configuration is not applicable for training, because the interaction metaphor requires too much attention from the driver, and this is not acceptable as the driver must pay full attention on the traffic and on the external environment. Moreover, arms fatigue quickly occurs as the user does not have a real steering wheel to support his/her arms.

In order to make use of the advantages of both the virtual configuration and the smart virtual configuration, the simulator system can be easily defined to change between these two modes in run time, according to desired interaction mode.

The analysis presented here is based on the experience of the authors during development and test of the system and as well demonstration to several visitors.

7. Conclusions and Future Work

In this paper, two fully virtual interaction metaphors for a vehicle simulator have been presented. One requires the user to interact with virtual primary controls as if they were real, and the other simplifies the interaction and proposes easier ways to interact with the primary controls.

The applicability of these virtual interaction configurations is complementary, and together they can be applied to several simulation scenarios. We believe that these two virtual configurations might constitute an interesting solution, in particular for Virtual Prototyping uses.

As future work, we intend to test each presented configuration with several test users, in order to determine if our analysis (concerning the suitability of each system) is confirmed.

8. Acknowledgements

9. References

[1] J. S. Freeman, G. Watson, Y. E. Papelis, T. C. Lin, A. Tayyab, R. A. Romano, and J. G. Kuhl, "The Iowa Driving Simulator: An Implementation and Application Overview", SAE World Congress, 1995.

[2] Bailey, A.C, Jamson, A.H., Parkes A.M. and Wright, S. (1999). Recent and Future Development of the Leeds Driving Simulator. Presented at DSC99, Paris. July 1999. See also http://mistral.leeds.ac. uk/.

[3] University of Central Florida Driving Simulator. Center for Advanced Transportation Systems Simulation, University of Central Florida (http://catss.engr.ucf.edu/html/Simulator.htm).

[4] Wrap-around Driving Simulator. Human Factors Research Laboratory, University of Minnesota (http://education.umn.edu/ kls/research/hfrl/facilities.htm).

[5] S. Espié, F. Saad, B. Schnetzler et al, "Microscopic Traffic Simulation and Driver Behaviour Modelling : the ARCHISIM Project", VTI Konferenz, In Proceedings of the Strategic Highway Research Program (SHRP) and Traffic Safety on Two continents, Lille, France, September 26-28, 1994.

[6] K.-P. Beier, "Virtual Reality in Automotive Design and Manufacturing", Proceedings of Convergence'94, International Congress on Transportation Electronics, SAE - Society of Automotive Engineers, Dearborn, Michigan, October 1994.

[7] M. R. Niesen, and G. R. Luecke, "Virtual Dynamic Prototyping for Operator Interface Design", 8th IEEE International Workshop on Robot and Human Communication, Pisa, Italy 1999.

[8] I.E. Sutherland, "The ultimate display", In Proceedings of the IFIP Congress, volume 2, 506-508, NY, May, 1965.

[9] A. Frisoli, C. A. Avizzano, and M. Bergamasco, "Simulation of a manual gearshift with a 2 DOF force-feedback joystick", ICRA - International Conference on Robotics and Automation Seoul, Korea, 2001

[10] Frisoli A, Avizzano CA, Bergamasco M, Data S, Santi C, "Dynamic modeling of primary commands for a car simulator", IEEE 2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM '01) 8-12 July 2001 Como, Italy.

[11] M. Lutz, "Programming in Python", Sebastapol, O'Reilly, 1996.

[12] M. Woo, J. Neider, T. Davis, and D. Shreiner, "OpenGL(R) Programming Guide", Addison Wesley, 0-201-60458-2, 1999.

[13] M. Kallmann, and D. Thalmann, "Direct 3D Interaction with Smart Objects", Proc. of ACM VRST, 1999, London.

[14] H. Nishino, K. Utsumiya, D. Kuraoka, K. Korida, "Interactive Two-Handed Gesture Interface in 3D Virtual Environments", ACM Symposium on Virtual Reality Software and Technology - VRST, 1-8, 1997.