

# COST ORIENTED VR-SIMULATION ENVIRONMENT FOR COMPUTER AIDED CONTROL DESIGN

Dirk Wollherr   Martin Buss

*Control Systems Group*  
*Faculty of Electrical Engineering and Computer Science*  
*Technical University Berlin*  
<http://www.rs.tu-berlin.de>

**Abstract:** In this paper the use of inexpensive standard hardware and software is proposed in place of high-cost commercial solutions to set up graphical VR environments and simulations with a human in the control loop. For this purpose a generic simulation environment for implementation of control simulation and evaluation experiments using VR with haptic feedback has been developed. As example setups a car simulator with a human in the control loop and an inverted pendulum as an experiment for student laboratories are presented.

**Keywords:** haptic control design, haptic interface, real-time simulation

## 1. INTRODUCTION

An important issue of interest in today's research are telepresence and teleembodiment. The goal is to immerse the user in the illusion to be present at a distant place (real or virtual) and not just to interact with a computer. In this context one usually has to generate a virtual world where the user can act. In the past a lot of very expensive, dedicated hardware has been necessary to implement VR and real-time dynamic simulators. With the increasing power of personal computers and powerful graphic hardware for the mass-market it is possible to use standard hardware to set up VR in combination with real-time embedded control systems.

In this paper a framework based on standard hardware components to rapidly implement VR simulations and hardware-in-the-loop simulators is presented. The idea is to generate simulation code directly using the MATLAB Realtime Workshop in connection with a self-programmed Linux Realtime Target. This allows to rapidly prototype a simulation environment, to generate and compile the simulation code. Such a system offers new possibilities for setting up experiments at moderate cost both for scientific use or as an experimentation environment for students. The framework

presented here has been used to implement a low cost car simulator with haptic and graphic feedback to the driver who is part of the control loop.

Road vehicles with a high centre of gravity can easily tilt due to extreme steering input by the driver, even at speeds as low as 30 km/h (van Zanten *et al.*, 1995). Therefore it is desirable to assist the driver electronically in order to avoid rollovers of the vehicle. The developed car simulator is used to test a rollover avoidance controller (RAC) not with computer-generated input data but with a real driver's input, which does not follow predetermined trajectories. Furthermore the effect of new forms of haptic feedback on the driver can be evaluated. Thinking of future vehicles which are equipped with a drive-by-wire system, i.e. there is no mechanical connection between the tires and the steering input device (steering wheel), it is no longer necessary that the driver feels the forces exerted on the front tires. This opens up new ways of communicating additional information on the vehicle status to the driver, here a force proportional to the roll angle is used. Haptic feedback is provided in this implementation by a force feedback paddle, first developed at the TU München (Baier, 1997).

In another example requiring short development cycles the usability of our framework for setting up a laboratory experiment for graduate students is demonstrated. Students obtain the possibility to design a control algorithm and rapidly generate realtime executable code to compare simulation results with hardware in the loop experiment data. The system combines real-time vision sensing with an embedded controller, again implemented using standard hardware.

The problem of developing a general framework for simulations and experiments is currently being followed at several research institutions worldwide: e.g. at the Swiss Federal Institute of Technology Lausanne (EPFL) a group has successfully set up realtime control experiments for distance learning (Salzmann *et al.*, 2000) using a realtime kernel for MacOS developed at the EPFL. The interface to the user and the access to data acquisition hardware is provided by LABVIEW. Another promising approach is being followed at the FernUniversität Hagen (Jochheim and Röhrig, 1999; Röhrig and Jochheim, 1999), where the use of commercial control design software is abandoned. Instead a set of applets has been written working as a user interface.

The here presented approach aims at rapid development and implementation of new concepts in computer-aided control engineering. This adopts the efforts being made in industry to reduce the time-to-market to research. The goal is not to provide a tool for developing end-user applications, but to allow quick and inexpensive implementation of case studies that probe the feasibility of new concepts and help to preview and discuss problems arising in the practical realization. Our approach combines the flexibility and versatility of open software with its full accessibility of existing source code together with advanced control design tools such as MATLAB.

For the organization of this paper: In Section 2 the principle system architecture of the here proposed controller environment is presented. This includes both, the hardware requirements and the software architecture. As example applications a car simulation for rollover studies and the setup of a laboratory experiment are presented in Section 3. Section 4 concludes the paper.

## 2. SIMULATOR SYSTEM ARCHITECTURE

A VR simulation generally consists of two parts: the numerical simulation of the dynamical system model and an interface to the human user. From the different aspects of implementing an user interface, only visual and haptic feedback are considered in this paper, though stimulation of other human modalities such as the auditive sense might increase the impression of reality.

The basic idea about the proposed concept for designing simulation environments is to split up the software into different tasks like "haptic interaction", "visual feedback" or "simulation". These tasks are distributed over as many different computers as necessary to provide sufficient computational power. The use of several industry standard PCs is economically justifiable as they are generally cheaper than dedicated VR workstations or real-time simulators. These computers are linked over a LAN and exchange the necessary data, like system states and coordinates, through TCP/UDP sockets or CORBA.

Distributing the tasks on different independent computers can also have practical advantages for use in the laboratory. A team cooperating on the same project can develop the tasks separately. As soon as all tasks work reliably the efforts can easily be linked. Furthermore, hardware, such as framegrabber systems or haptic feedback devices, which are used in several projects need not be moved and reconfigured on a new computer. The same hardware configuration can be used, only the data is sent to another computer. Since the hardware requirements highly depend on the tasks to be performed, a general rule cannot be given. The following gives an idea of a configuration which works for our applications.

### 2.1 Hardware

The key components of the simulation prototyping environment are standard personal computers linked through a local area network (LAN). Especially the dynamical system real-time computation often requires extensive computational power. As the binary compiled version of the simulation often runs significantly faster than the code interpreted by MATLAB, it is difficult to judge from MATLAB simulation times on the hardware requirements for realtime PCs. For presentation of 3D virtual environments at adequate frame rates a video adapter with 3D acceleration is required.

As the commercially available low cost haptic feedback devices neither have open programming interfaces nor provide sufficient power and accuracy, the device adopted from (Baier, 1997) is used, see Fig. 1. The operator can determine a control input by moving a lever attached to the axis of a motor into the desired position which is measured by a pulse encoder. If the motor exerts a moment on the lever, it gets bent. Strain gauges attached at the lower part of the lever are used to measure this flexion and estimate the moment resulting from the interaction between the human and the paddle.

All these data are processed by a Sensoray 626 multi-function I/O board providing ADC, DAC and quadrature decoders, see Fig. 2. One DAC channel is used to control the motor through a PWM-amplifier, while an ADC measures the output of the strain gauges. The

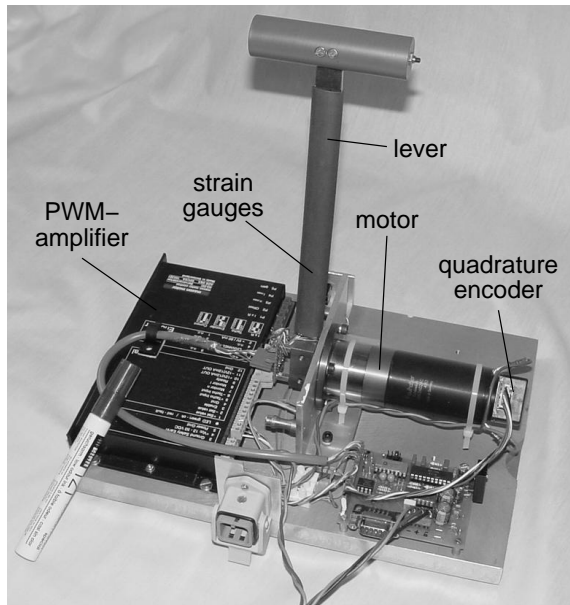


Fig. 1. Force feedback paddle.

quadrature decoder determines the actual position and velocity of the lever. With this hardware architecture force, position or impedance control can be realized.

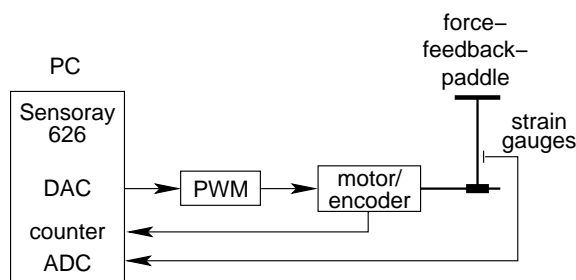


Fig. 2. Scheme for the control of the ff-paddle.

The example of the force feedback paddle demonstrates how new devices needed for experiments can be built using a (small) number of standard hardware components like multifunction I/O-boards, PWM-amplifiers and motors with quadrature encoders. Devices assembled from parts of this "building kit" have the virtue for the researcher to be known in detail and to be adaptable exactly to the individual needs.

## 2.2 Software

As an operating system for the computers Realtime (RT) Linux was chosen. Besides its economic advantage – RT Linux is distributed free of charge – this system is technically competitive. While Linux itself already is capable of handling soft real-time requirements, its real-time extension RT Linux (<http://www.rtlinux.org>) meets hard timing constraints, i.e. it is possible to start processes at a fixed scheduled time. A RT Linux periodic task runs within 35 microseconds of its scheduled time on x86

hardware, the delay between the detection of an interrupt and the launching of the corresponding program is less than 15 microseconds (Barabanov, 1997).

Another reason for choosing RT Linux is the availability of an enormous amount of software in its source code. The researcher can build upon this existing, well tested and efficient code and adapt it to the individual needs. The time needed for development is shortened significantly.

During the planning process of the software architecture possible package delays due to congested LANs must be taken care of. Therefore it must be assured that control loops cannot become unstable with long delays or loss of single packets. A careful choice of the form of the data can speed up the communication process: for data-transmission absolute values like world coordinates are recommended, instead of incremental changes with respect to the previous data. This way the loss of single packets during transmission often is acceptable, hence faster UDP sockets can be used. In contrast to the TCP protocol, the UDP protocol does not check for correct delivery of packages which reduces the communication overhead of the protocol and hence speeds up data transmission.

Coding a problem by hand being very time consuming and susceptible to errors a new target for MATLAB's RT Workshop has been developed. This target allows generating and compiling of standalone real-time code for RT Linux from a Simulink model. The control loop to be implemented is composed of ordinary Simulink blocksets, hence the migration from designing a controller in offline mode to evaluating it in an experiment is subject to substituting the system model by hardware in the loop which can also be accessed through Simulink blocksets. Another benefit of this procedure is the clear structure compared to a large number of manually linked files of source code. Consequently maintenance and modification of the model become easier and less time consuming. MATLAB was chosen as a platform as it is widely used in our group. Of course any other modern simulation tool, capable of producing standalone code, could have been chosen instead. Possible candidates are MATRIX<sub>X</sub> or MODELICA.

The visual output of the virtual environment is realized on a separate computer also receiving all necessary coordinates and system states from the simulation task through TCP- or UDP- socket connections. The graphics task does not run as a real-time application as the soft real-time capabilities of Linux are sufficient to ensure time consistency between the simulation and the feedback to the driver.

In the following, two example applications are presented demonstrating the implementation of practical problems.

### 3. APPLICATIONS

#### 3.1 Rollover Avoidance

The framework described above has been used to implement a car driving simulator used for evaluation of a rollover avoidance controller (RAC). Following the idea of (Ackermann and Odenthal, 1999) a driver assistance system for vehicles with high centre of gravity was suggested in (Wollherr *et al.*, 2001) and (Mareczek *et al.*, 2001) which limits the range of the steering angle accessible for the driver. The RAC determines how close the vehicle is to tilting from the measured state  $\mathbf{x}$  of the vehicle and the steering angle  $\delta^w$  adjusted on the front wheels by evaluating the rollover coefficient

$$R(\mathbf{x}, \delta^w) = \frac{F_{z,R} - F_{z,L}}{F_{z,R} + F_{z,L}}.$$

While the wheels on both sides of the vehicle have contact to the ground,  $|R| < 1$ ,  $|R| > 1$  means the car is tilting. If the driver choses a steering angle which makes the vehicle tilt, i.e.  $|R|$  becomes close to 1, the RAC determines all steering angles which prevent from rollovers. From all admissible angles the one closest to the angle commanded by the driver is adjusted at the front wheels. For further information on the controller please refer to the above mentioned literature.

#### Realization

In this setup the tasks of the simulator have been split into three parts (Fig. 3): *i*) simulation of the dynamical system model; *ii*) control of the haptic interface; *iii*) 3D graphical visualization. Each of these tasks runs on a seperate computer, although it would also be feasible to handle the haptic interface and the graphical output on a single CPU. The PCs are all equipped with AMD Athlon 900 processors and 256 MB RAM. They are linked through a 100 Mbit switched ethernet LAN which causes communication delays of about 140 – 150  $\mu$ secs.

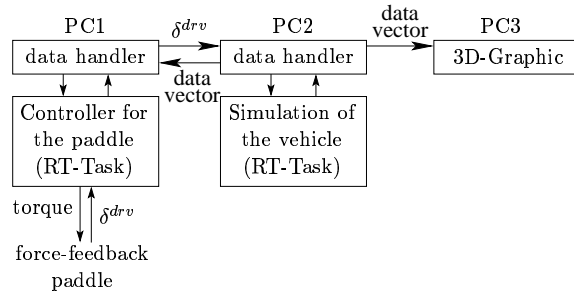


Fig. 3. Communication structure of the simulation software.

The interface to the user is split into two aspects: the visual feedback and a haptic feedback through the steering device. The virtual environment has been programmed in C using the Maverik library, a toolkit for VR application programmers using OpenGL/Mesa for

lowlevel rendering. Fig. 4 shows an abstract driving situation with the driver looking through the vehicle's windscreen and the virtual steering wheel in front of him. Pylons on alternating sides of the road represent obstacles which the driver is supposed to avoid. The inclination of the horizon corresponds to the tilting angle of the vehicle. There are two markings on the steering wheel. The light grey marking in Fig. 4 denotes the steering angle  $\delta^{drv}$  chosen by the driver, the dark grey one represents the invariance angle  $\delta^{inv}$ . A bar is located on the lower edge of the screen, whose length proportionally depends on the actual rollover coefficient  $R$ .

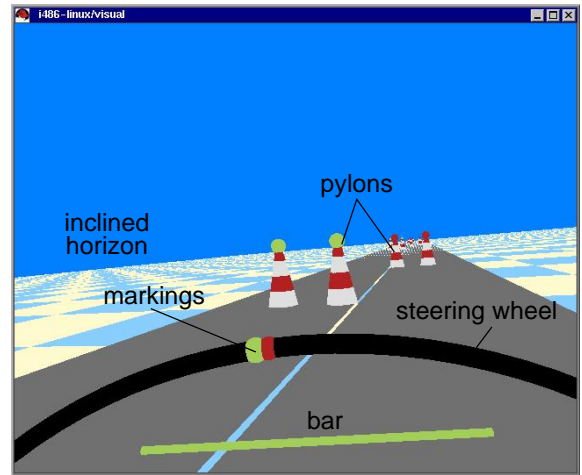


Fig. 4. 3D simulation environment: street with pylons and virtual steering wheel

Through the force feedback paddle the driver gets additional information about the rollover coefficient. Experiments showed that this information helps the driver to better understand the complex dynamics of the vehicle. In these experiments the driver was procured a momentum

$$M_L = (e^{-5(R+1)} - e^{-5(-R+1)})M_L^{max},$$

with  $M_L^{max}$  being the maximum admissible momentum.

#### Experimental Results

The vehicle model used in the experiments is the so-called nonlinear one-track model as described in (Riekert and Schunck, 1940) extended by an additional mass following (Segel, 1956-1957). Forces exerted on the tires are modelled by the HSRI tire model (Mitschke, 1990). A detailed description of this model can be found in (Wollherr *et al.*, 2001).

The system parameters have been chosen to describe a 14.3 t lorry with its centre of gravity at a hight 2.28 m above the ground and a track width of 0.93 m. From this data one can see that the vehicle is unrealistically instable and tilts easily. The choice of parameters was subject to demonstrating the capabilities of the controller.

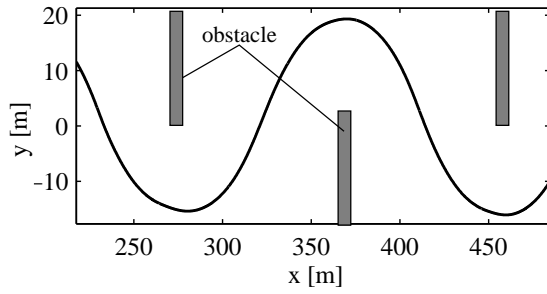


Fig. 5. driven trajectory

Fig. 5 shows a slalom manoeuvre steered by a human driver trying to avoid three obstacles. The corresponding steering angle  $\delta^{drv}$  of the driver is shown in the lower plot of Fig. 6 as a solid line. As soon as the rollover coefficient  $R$  in the upper graph of Fig. 6 depasses the threshold  $|R| = 0.9$  (dotted lines), the RAC becomes active and commands the invariance angle  $\delta^{inv}$ , marked as a dashed line in the lower plot. Due to delays because of the steering dynamics  $|R|$  depasses the threshold 0.9, but one can see that the threshold has been chosen sufficiently high to refrain the car from tilting. Note that a vehicle not equipped with a RAC would have tilted as soon as  $R$  reaches the value  $\pm 1$ .

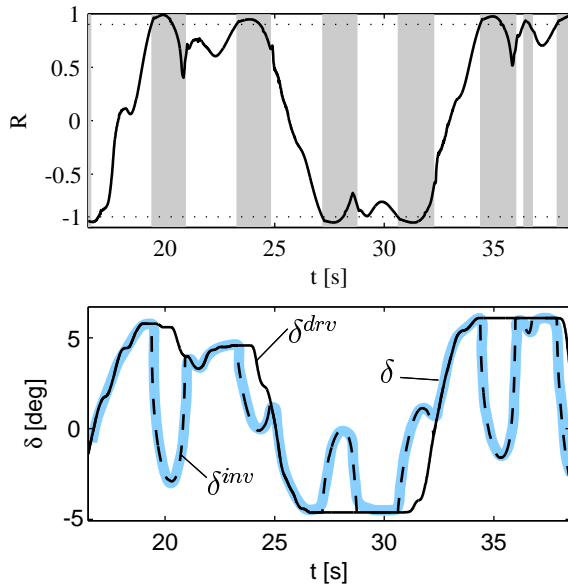


Fig. 6. simulation results

### 3.2 Inverted Pendulum

Another application of the presented framework is the control of an inverted pendulum. This setup is used as an experiment for students in an advanced course laboratory offering the possibility to test and compare various control strategies. This problem – on first sight quite different to the one described before – has similar requirements on the underlying software: due to the limited time the students have for the laboratory,

an environment allowing short development cycles is needed. The proposed approach provides an efficient framework for such laboratory experiments.

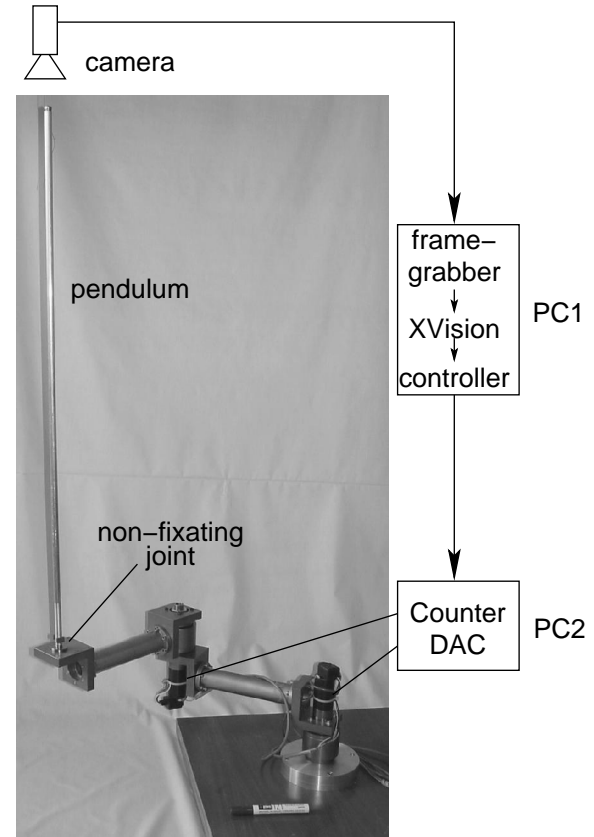


Fig. 7. SCARA robot with an inverted pendulum

In this experiment the SCARA robot in Fig. 7 is to ballance an inverted pendulum in the upright position. Pulse encoders at the two driven joints of the robot allow to determine the position of the actuator of the robot. The orientation and the inclination angle of the pendulum are measured with a video camera tracking the upper end of the pendulum which is marked with an LED to achieve better tracking results.

In this setup the software has been split into two tasks running on different PCs: the controller and the visual tracking system. The software XVision (<http://www.cs.jhu.edu/CIPS/xvision/>) is used for tracking applying state of the art algorithms. It was modified to directly send the tracking coordinates to the controller through a socket connection. The controlling computer is equipped with a Servo-To-Go motor control board featuring DAC, ADC and pulse encoders on a single board. It is used to access the motors at the joints of the SCARA robot.

Students can design controllers using MATLAB and test them in offline simulations using a model of the SCARA robot to be controlled. When offline simulations of the controller work, realtime code is directly generated and the controller can be evaluated with hardware in the loop. This is very useful to demon-

strate the effects of potentially not modelled system properties such as friction and stiction. Experimental results of this inverted pendulum control experiment will be presented elsewhere due to paper length constraints. Future plans include to extend the setup by a third computer providing a VR representation of the system. This computer can serve java applets on a distant computer enabling internet experiments. A further step is the implementation of augmented reality offering the possibility to disturb the experiment through the internet.

#### 4. CONCLUSIONS

In this paper a novel framework for rapid control prototyping and development of VR simulation environments is presented. It allows fast development of new simulation environments at moderate cost. Furthermore the framework is versatile and easily expandable as all programming interfaces, except MATLAB, are open in the public domain.

The example application of vehicle rollover avoidance control can be seen as a first step towards Computer Aided Control Design (CACD) with haptic feedback. During the development of the RAC this kind of CACD proved to be very helpful to evaluate the RAC. Further investigations on this subject will follow.

#### Acknowledgments

The authors would like to thank H. Baier and Prof. G. Schmidt, from the Institute of Automatic Control Engineering at TU München for their useful comments.

#### 5. REFERENCES

- Ackermann, J. and D. Odenthal (1999). Damping of Vehicle Roll Dynamics by Gain Scheduled Active Steering. In: *Proceedings of European Control Conference*. Karlsruhe, Germany.
- Baier, H. (1997). Model-Based Teleoperation of a Drilling Machine in Uncertain Communication Barriers. Internal Report, Institute of Automatic Control Engineering, Technische Universität München.
- Barabanov, M. (1997). A linux-based real-time operating system. Master's thesis. New Mexico Institute of Mining and Technology. Scorro, New Mexico.
- Jochheim, A. and C. Röhrig (1999). The virtual lab for teleoperated control of real experiments. In: *Proceedings of the Conference on Decision and Control*. Phoenix, Arizona USA. pp. 819–824.
- Mareczek, J., D. Wollherr, M. Buss and G. Schmidt (2001). Überschlagsvermeidung bei Kraftfahrzeugen durch Invarianzregelung. to appear in *at - Automatisierungstechnik*.
- Mitschke, M. (1990). *Dynamik der Kraftfahrzeuge*. Vol. C. Springer. Berlin, Germany.
- Riekert, P. and T.E. Schunck (1940). Zur Fahrmechanik des gummibereiften Kraftfahrzeugs. *Ingenieur Archiv* **11**, 210–224.
- Röhrig, C. and A. Jochheim (1999). The virtual lab for controlling real experiments via internet. In: *Proceedings of the IEEE International Symposium on Computer Aided Control System Design*. Kohala Coast, Hawaii USA. pp. 279–284.
- Salzmann, C., D. Gillet and P. Huguenin (2000). Introduction to real-time control using labview with an application to distance learning. *The International Journal of Engineering Education* **16**(3), 255 – 272.
- Segel, L. (1956-1957). Theoretical prediction and experimental substantiation of the response of the automobile to steering control. In: *Proc. IMechE*. pp. 310–330.
- van Zanten, A. T., R. Erhard and G. Pfaff (1995). VDC, The Vehicle Dynamics Control System of Bosch. In: *Advancements in ABS/TCS and Brake Technology (SP-1075)*. Detroit, Michigan. pp. 9–26.
- Wollherr, D., J. Mareczek, M. Buss and G. Schmidt (2001). Rollover avoidance for steerable vehicles by invariance control. In: *Proceedings of the European Control Conference*, to be published. Porto, Portugal.