IJES

International Journal of Engineering Simulation

with Industrial Applications.

Contents

Volume 5
Number 2
ISSN 1468-1137

July 2004.

For online interactive simulations and
Hot Links visit our web site at:
www.intjes.co.uk
or use the supporting CD-Rom

VIRTUAL PROTOTYPING OF A DOMESTIC ROBOT FOR DESIGN AND NAVIGATION OPTIMIZATION

A. Martínez[1]*, L. Núñez*[1]*, M. Fernández*[2]*, S. Casas*[2] *and J. Landaluze*[1]
[1] *IKERLAN Research Centre. E-20500 Arrasate-Mondragon (The Basque Country). Spain*
[2] *ARTEC Group, Valencia University. E-46980 Paterna (Valencia). Spain*
E-mail: jlandaluze@ikerlan.es

Abstract

This paper describes the application developed as part of the "PROSIGRAT" research project carried out by IKERLAN and Valencia University's ARTEC group dealing with the use of Real Time simulation for services and products evaluation. The application consists of the development of a Real Time graphic simulation of a domestic mobile robot moving in a virtual environment and guided by a real physical navigation and control system, based on the HIL methodology concept. This is a clear example in which the Real Time simulation of a mechatronic system interacting with the environment leads to a reduction in the time-to-market of a new product that has been well tested and operates in a safe, reliable way. The paper shows the details of the application and the main conclusions obtained.

1. INTRODUCTION

Nowadays competitiveness is a key point for the companies. This concept involves the reduction of costs and time-to-market when launching new products or new versions of existing ones. When it comes to the development of complex mechatronic systems including control elements, very often the integration of the control subsystems happens at a late stage of the product development cycle. The tuning of the control algorithms and the operating tests are carried out on a real prototype of a new product, in a process prone to error that could even cause a failure of the prototype. That is why the use of new techniques that allow the validation of control systems from an early phase is so important. This is the main concept behind the application of the HIL (Hardware in the Loop) methodology. It consists of using realistic digital models of complex systems that interact with the real physical control systems (HW and SW) in the same way as the real complex system to be controlled. Another possible use of the same methodology consists of validating several design alternatives designs for a new product in a realistic way by looking at different possible control alternatives.

A key point when developing HIL applications is the modelling of complex mechatronic systems in such a way that the model can be simulated afterwards in Real Time, and maintaining with high fidelity the evolution of the physical variables that are essential for the control system being developed. Although there are several commercial modelling and simulation tools oriented towards Real Time simulation of multidisciplinary systems, their suitability for any type of application is not at all clear [4]. The problem becomes even more complex when the physical system to be controlled interacts a lot with the surrounding environment. In this case the Real Time simulation of this environment and its evolution is also necessary.

This paper shows an example of the use of the HIL methodology explained above in order to aid the design of a new product: a domestic robot (vacuum cleaner application) and its control and navigation module. It is obvious that this is a case in which the system (robot) interacts with the environment (room to be cleaned) and, depending on the results of this interaction, the robot control system has to decide the new movement strategy to follow.

First of all the paper explains the main objectives of the research work carried out and the basic characteristics of the domestic robot application selected as a case study. Secondly, a detailed description of the main application components is given. There follows a summary of the test and alternatives evaluation carried out with the help of the virtual prototype. Finally, the main conclusions obtained from this work are presented, together with the future lines of research arising from this work.

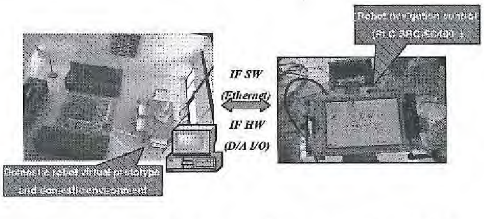

Figure 1. Robot virtual prototype for HIL

2. DOMESTIC ROBOT VIRTUAL PROTOTYPE

The main objective of the research work carried out by the Robotic Institute of Valencia University and IKERLAN as part of the PROSIGRAT project: "HIL based Virtual Prototyping and Real time Graphics Environment for Mechatronic systems" was to deal with the study and development of Real Time simulation and HIL techniques, as well as developments in the field of interactive 3D graphics. In order to assess the knowledge acquired during the project and the tools developed, as well as to validate the utility of the HIL techniques, the group planned to introduce some material objectives or demonstrators to the project. One of them was the Real Time simulation of a hydraulic cushion for a press in order to help the design and development of the control system, thereby avoiding having to use the real press [2][3].

The other demonstrator is the focus of this paper. It consists of the modelling and Real Time simulation of the components of an autonomous vacuum cleaner for domestic environments, as well as of the vacuum cleaner evolution environment, so that the navigation system could be developed and tested.

This application deals with a new product, a physical system that is still not available. So, many important decisions have to be taken regarding the best robot component design and configuration. The Real Time simulation of the robot moving in the domestic environment is also a way to aid this design.

A mobile robot is a complex system involving many multidisciplinary components: the actuators and drivers making up the locomotion system, sensors, energy supply systems, control SW and HW, etc. The analysis carried out as part of the project focused on three components: the sensors to be installed in the robot to properly detect the environment, the locomotion system that improves the movements of the robot in a domestic environment, and the control strategies to ensure the robot moves in a safe and optimal way in different domestic environments (types of furniture, floor materials, types of soils, etc.).

As for the main elements (Figure 1) into which this domestic robot virtual prototype can be split, it is worth mentioning:

- The navigation system: the real physical part of the application consisting of a HW/SW platform similar to the one that would be integrated in a real domestic robot prototype.
- The simulation of robot dynamics, the graphic simulation of the room, and the simulation of the interaction between them.

The interface between the real part of the prototype (control) and the simulated part. There follows a detailed description of these parts of the prototype.

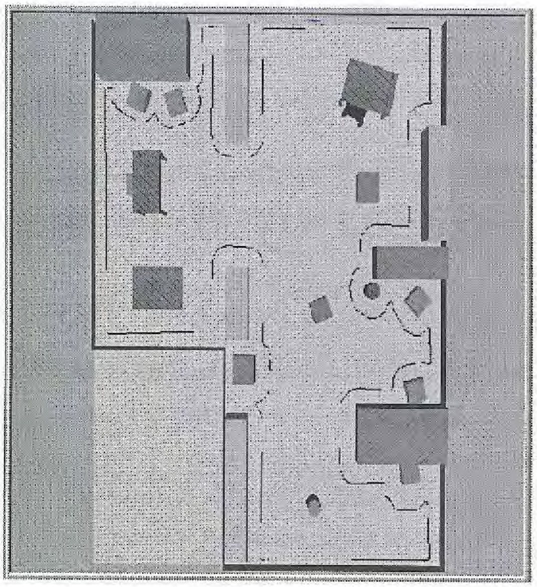

Figure 2. Wall following and obstacle avoidance

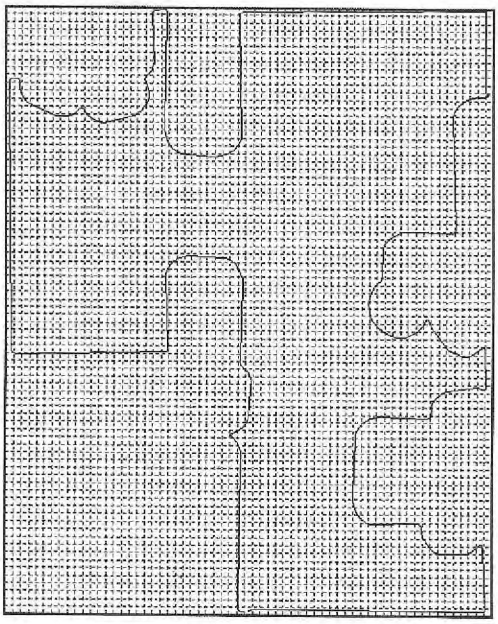

Figure 3. Area to be swept represented by a matrix

3. NAVIGATION SYSTEM

As previously explained this is the real physical part of the mobile domestic robot prototype. It consists of a group of SW modules running under the Windows CE operating system on a HW platform based on a Pentium 486 processor.

The navigation system consists of three basic stages: obtaining the contour to be swept, sweeping of inner contours and bordering of obstacles.

The first goal of the system is to obtain the outline of the robot's environment (Figure 2). For this purpose the robot uses a wall following strategy. A fuzzy control guides the robot along the walls. It takes the information provided by the ultrasonic sensors arranged all around the robot to gain a general perception of the immediate surroundings. Then, using this information and the robot's displacement direction vector, the speed and the steering angle required by the robot to follow the walls without colliding with any obstacle can be calculated [7]. It follows the walls until it reaches the starting point. This determines not only the contour of the initial navigation, but the area in which the robot must move. The contour will be limited by a rectangle whose area is as small as possible. This area is divided into cells of a prescribed size. Each cell has a status: the cell can belong to the contour, to the outside of the contour or to the inside of the contour. As a result the zone of intake will be limited to the inside of that contour.

Henceforth the behaviour of the prototype is not controlled by the fuzzy rules. At this moment the robot will use the matrix (Figure 3) obtained in the first round to calculate the points of the next path. Therefore some of the cells will change their status from "inside" to "path". By connecting the cells that are in "path" status, a path with the same shape of the initial contour is obtained, but inside the previous one. The cells whose status was "path" are converted into "swept" cells. If the room was completely empty, the robot would sweep the room in a spiral.

But the behaviour is not completely planned. The robot is not only guided by this "map", because there is the possibility of encountering an obstacle along the path. The robot goes around an obstacle in the same way it follows a wall, using the fuzzy control, until a point of the initially marked path is found. When the robot finds this point it is again controlled by the logged "map".

The process of sweeping inner contours is repeated continuously until a 3% of the area initially selected for the task left. At that moment the prototype detects the zones that have not been swept yet, called "islands". The robot only chooses the "islands" that are bigger than the size of the vacuum cleaner robot. Then the robot selects the "island" nearest to its actual position and moves ahead to it by the shortest route. Each island then becomes a "small room" to be cleaned, and the sweeping is controlled by the same rules of behaviour. Once all the islands have been swept the robot considers that the task is done.

To embed the navigation system in a Windows CE platform the fuzzy toolbox of MATLAB was first used to edit the fuzzy rules of control. Once the rules are known a model can be created in SIMULINK, including the fuzzy rules edited before in a block. Finally code can be generated with the RTW tool.

4. ROBOT DYNAMICS AND ENVIRONMENT GRAPHIC SIMULATION MODULE

As pointed out above, one of the key issues of the system presented is to provide a test environment for the HIL control strategies, which means being able to simulate the environment where the robot is working and to simulate the dynamic behaviour of the robot and its interaction with this environment, as well as the sensor information detected by robot sensors.

In order to have a better feedback and a visual first impression about control techniques, a Real Time 3D realistic simulation of the environment was required. Apart from that, another requirement for the system was the flexibility to be reconfigured to simulate different HIL control and robot systems.

4.1 Module architecture

With the requirements indicated above, a module based on three main different subsystems was designed:

- 3D Real Time graphic subsystem.
- Dynamic simulation subsystem.
- Communication subsystem.

The three subsystems work simultaneously in a synchronized way to simulate the behaviour of the robot and its environment, receiving the control information from the HIL control system and sending back the information that the HIL control system will measure from environment sensors in a real system.

The system, which has Real Time requirements, runs in a multi-threading configuration. Figure 4 shows the three main threads. Each of the threads is dedicated to running one of the three subsystems. The synchronization between subsystems is done in the application part of the graphic thread that is the base thread for this run-time environment simulation module.

The system configuration is based on the description of a correlated database that has two main components:

- Environment and robot graphic description.
- Dynamic robot and environment description.

Both descriptions are based on a hierarchical data structure represented in the form of two interrelated a-cyclic graphs. The interrelation allows the synchronized updating of visual and dynamic information, in this way maintaining the coherence between these two aspects of the system.

A brief description of the first two subsystems and their configuration now follows, whereas the communication subsystem is explained in paragraph 5.

4.2 Real Time graphic subsystem

The Real Time graphic subsystem is based on a 3D graphics library call IRIS Performer [5]. This library offers a run-time environment able to control a graphic application running at interactive frame-rates. For the system used a frame-rate of 30 frames per second was fixed, which was enough for the system requirements.

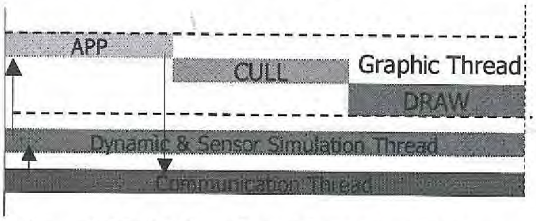

Figure 4. Multi-threading Scheme of graphic system

The basis for the 3D graphics system is the definition of a 3D graphic data structure. The data structure used was a *scene graph. This* scene graph is based on hierarchical a-cyclic graph, with different kind of nodes. These nodes allow the representation of geometry and visual properties (material, texture, lighting) of the environment elements, to configure the location and orientation of elements in the scene (Dynamic Coordinate System nodes) and also allow an optimisation of the time required to represent this scenario, which is a critical issue to maintain Real Time requirements in complex simulation scenarios. The optimisation techniques are based mainly on the level of detail management (LOD nodes) and culling of the scene parts not present in a particular frame [1].

The image generation process is based on the traversal of this *scene graph*. In fact three different traversals are performed (corresponding to the App, Cull and Draw parts of the Graphic Thread represented in Figure 4). The first traversal is used to update the graphic data with the results obtained from the dynamic simulation of the system. Using this information the different mobile parts of the robots and the scene are updated maintaining the consistency with the dynamic simulation graph. This traversal is also used for collision detection.

Once every object in the scenario has been updated the culling traversal provides the specific objects that have to be displayed in the following frame with its proper level of detail. The objects are sorted based on their material and texture in order to optimise the next traversal in the graphic thread. The last traversal is the drawing traversal, which is where the scene is actually rendered. The rendering is done using a 3D graphic pipeline schema.

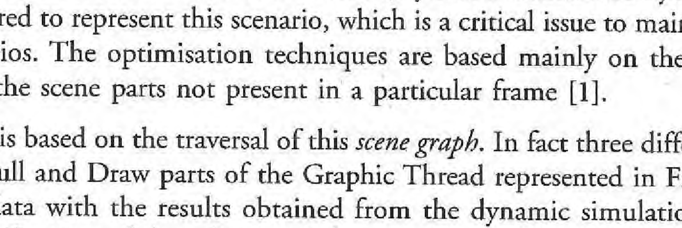

Figure 5. Example of library objects

In terms of the graphic part of the system a key issue is how to build the *scene graph* for a particular robot and its environment. To do so the system provides a graphic interface and the possibility of using a set of file formats that can be imported to represent the scenario geometries (as OBJ or VRML). It also offers a library of basic robot elements (wheels, robot bodies, etc., Figure 5) that allows the configuration of different environments in an easy way. It should be pointed out that the robot elements library not only provides the graphical scene graph to represent objects, but also includes its dynamic representation graph to simulate its behaviour.

4.3 Dynamic simulation

The aim of the dynamic simulation subsystem was to be a customisable and an easy-to-use modelling and simulation element. An element that allows different tests for physical systems to be built with no real difference between, for instance, a mobile robot and a hydraulic press, allowing at the same time a certain degree of specialization.

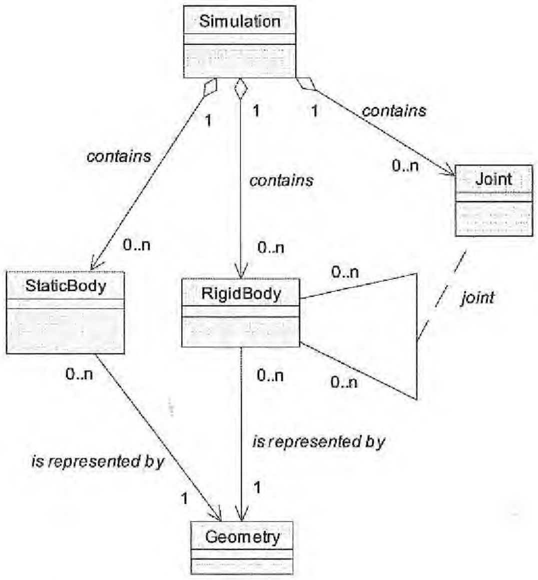

Figure 6. Simplified UML architecture

That brought Newtonian dynamics, which are fairly computable, and realistic enough to get a mechanical simulation. Based on that idea, the concepts of rigid body and joints were introduced, due to the fact that most mechanical systems can be modelled as several rigid bodies jointed together, with forces and torques acting over and among them. This can be accomplished by simulating a Newtonian dynamics system, combined with a Lagrange multiplier based constraint formulation in order to implement the joints.

At the same time, the concept of static body was introduced in order to simulate those objects that will never be moved under any circumstances. This is a way of simplifying things by letting the dynamics simulator only deal with those objects that are likely to move, and ignoring those that will never move. The representation of such a system is based on a hierarchical graph that builds a robot through the connection between these different elements. Apart from this, a correspondence with the graphic representation has to be established, which is accomplished by a set on links (represented as memory pointers) that allows navigation between graphical and mechanical data structures. In Figure 6 a representation of the relationship between these concepts is represented using UML notation.

A more specialized approach to the problem was needed. Although the main interest was in robotic simulation, more specialized robot concepts such as motors, wheels, encoders, and sensors were included as well. This flexibility allows generalization and functionality with such a combination making a powerful modelling and simulation tool for several purposes.

The dynamic simulation thread includes rigid bodies, static bodies and joints. These constraints are the key for building a system as complicated as required. In addition, this thread simulates object collisions and provides a friction model. All of these are made with the help of the dynamics ODE library (Open Dynamic Engine) [6]. This is the most CPU greedy thread, as it is intended to be executed at least 100-150 times per second.

Finally, as indicated, the system has to send feedback information to the HIL control system. This point is based on the simulation of different sensors that emulate the ones present in real robots. The sensors considered were as follows: contact sensors and ultrasound sensors. Contact sensors are simulated using collision detection algorithms between the robot body and the environment. Ultrasound sensors are simulated using ray-casting algorithms to simulate the radiation diagram of real sensors.

5. NAVIGATION-SIMULATION INTERFACE

In the description of the robot simulation environment it was indicated that a communication thread was also present in the architecture. This communication thread is in charge of sending the data from the HIL, updating the robot control and sending back the information obtained from sensors in the simulated environment.

Two interfaces are provided to implement the communication between navigation and simulation in the mobile robot simulation. The first one is a software socket communication and the second one is a hardware communication. The difference between them is that the first establishes communication between the simulation software and the navigation simulation software, meanwhile the second establishes communication between the simulation software with the navigation hardware prototype.

As it is needed, the socket communication system is designed to send all the ultrasonic sensor and encoder measurements to the navigation system (from the simulation), and to send the desired motor speeds to the simulation (from the navigation). In addition to this, there is a number of initial values to be interchanged, so an initial data communication is required.

With that fact in mind, it is not difficult to guess that four channels are provided to get and set (depending on the side considered) all the interchanging variables. These four channels are implemented as four sockets, one for each kind of communication. As the simulation acts as socket server, there are four client sockets in the navigation system: *Initialisation Data, Sensors Data, Encoders Data* and *Motors Data.*

Initialisation Data is a channel for communicating general robot data (such as mass, lengths, etc), sensor positions, motor characteristics, and encoder parameters. This channel is also used for communication of additional data to the navigation for debugging purposes.

Sensors Data is a channel used to send (from simulation to navigation) the voltage representing the distances measured by the series of ultrasonic sensors fitted to the robot. This value is used by navigation to decide the next movement strategies.

Encoders Data is, as its name implies, the channel used to send (from simulation to navigation) the values measured from the robots's motor encoders. Encoders indicate the approximate angle the wheel has rotated, indicating if the wheel is rotating regardless of the order sent to the motor (which it may be impossible to get due to physical difficulties). This is a two-way channel because navigation can send some kind of orders to the encoder, such as a reset count signal.

Finally, *Motors Data* is the channel used to send (from navigation to simulation) the voltage representing the motor speed required. This communication establishes complete feedback between navigation and simulation, expressed as a loop in which the robot world and the sensors (encoders and ultrasonic sensors) are simulated. It sends sensor values to navigation, which decides what to do, sending motors values to react to the changes in the environment.

On the other hand there is the hardware communication system, which is in fact the real way to do the HIL design process.

The communication takes place using a ZWorld BL-2100 TCP-IP board system. This board takes an Ethernet input, which is output from the simulation PC, and has several digital and analogical output signals, which are inputs in the navigation hardware control system. This makes it possible to use in the simulation the same kind of socket structure mentioned before, with the only difference that the implementation of a program in the board ROM is needed in order to extract all the sockets data and convert them into voltage signals plugged into the navigation hardware.

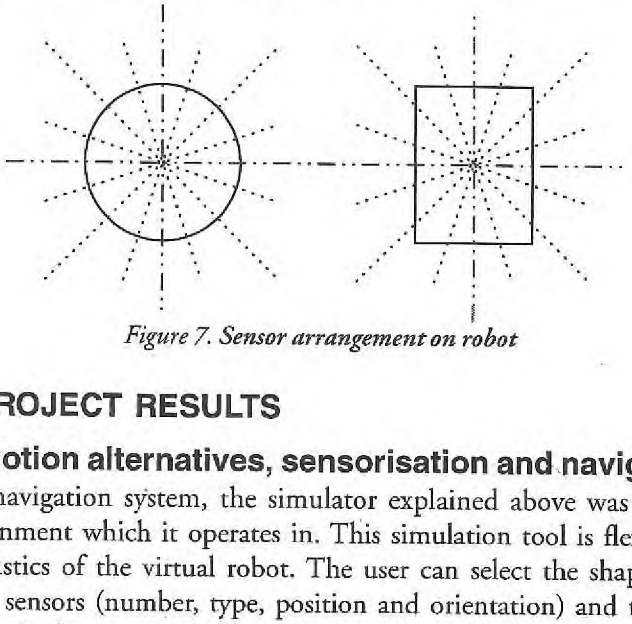

Figure 7. Sensor arrangement on robot

6. EVALUATION OF PROJECT RESULTS

6.1 Evaluation of locomotion alternatives, sensorisation and navigation strategies

To validate the implemented navigation system, the simulator explained above was used. It reproduces the virtual robot in addition to the environment which it operates in. This simulation tool is flexible enough to permit the user to select the physical characteristics of the virtual robot. The user can select the shape and initial orientation of the robot, the configuration of the sensors (number, type, position and orientation) and the locomotion system (car-like, holonomic and wheel configuration).

The simulator is a tool that allows the real navigation control to be refined and tested, applying that navigation control to the initial configuration that the user selects for the robot. There follows a summary of the conclusions, based on the test results for different robot configurations.

The results depend to a large extent on the prototype shape. It is verified that for the same distribution of sensors and comparable volumes, a circular prototype moves through the room more continuously and its trajectory is smoother than the one of a rectangular shape.

Concerning the locomotion systems, the conclusion is that the car-like model does not satisfy the needs required for the final behaviour of the robot, as the turn radii were too large. To access specific zones, it was necessary to carry out complex manoeuvres that, as well as complicating the control, increased the time needed for the execution of the task. The selected system was finally the holonomic system and the rest of conclusions given here refers to a robot with that locomotion system. A priori, it seems that due to the holonomic behaviour of the prototype, the most suitable sensor arrangement will be a symmetrical arrangement with respect to both axes (Figure 7).

The test results indicate that the prototype behaves better when its sensors are arranged in the radial direction. This was foreseeable due to the type of locomotion system chosen, as there is no privileged direction of movement. This implies that the prototype should be able to receive the same quantity of information from the environment, whichever direction it is moving in. To obtain the desired result, the sensor arrangement must be symmetrical. Whenever the symmetry of the sensor arrangement is maintained, its orientation does not necessarily have to be radial. However, the simulation demonstrated that the results are better with this radial configuration.

The minimum number of sensors that assures a correct behaviour of a 40 cm diameter prototype is twenty. Fewer sensors increase the possibility of collisions with certain types of objects.

Initially, it was considered that the only condition the robot should verify was not coming into contact with objects in the environment. So only ultrasounds sensors were analysed. A subsequent analysis introduced the possibility of gentle robot contact, and the possible use of contact sensors.

6.2 Graphic and dynamic robot prototype configuration

To show the functionalities of the system a mobile vacuum cleaner robot prototype and its working environments were set up using the system described. The holonomic vacuum cleaner robot itself was modelled with four rigid bodies (a body framework and a locomotion system consisting of one free-rotation wheel and two drive and steering wheels), two motors (one traction motor for both wheels and one steering motor for the same two wheels) with its corresponding incremental encoders, and twenty four ultrasonic sensors arranged in a circle in order to measure distances.

The robot was treated as any other dynamic object in a customisable, changeable environment, in which its behaviour should be tested. The key to this kind of simulation is the fact that objects in the environment are of the same class as the robot itself, so an environment to test any system can be built easily. This is also the main problem, because any system can be built, whether it is physically consistent or not, and there could be non-stable systems and physical aberrations. But, anyway, this is actually a feature deriving from its structure, so it should not be considered as something the system lacks, but a consequence of its flexibility.

In this vacuum cleaner application, the dynamic simulation thread now includes the locomotion system (motor and wheel simulation with encoders to measure wheel rotations) and the sensor system, in addition to the previously mentioned rigid body constrained simulation world with friction and collisions. The drawing thread remains the same, with the only exception of sensor drawing and colouring (Figure 8), which is an addition to offer a way to test the reliability of the sensors. The communication thread is based on a two-way solution. The first one is a software communication system made with asynchronous communication sockets between the simulation and a navigation simulation. The second one is a hardware board communication system that communicates the mechanical simulation with the hardware navigation prototype.

To perform the navigation a scenario consisting of a living room and the furniture in it was prepared. In order to improve the appearance, radiosity was used to light the environment (Figure 9). Apart from the visual appearance, simulations were performed for different cases of environment properties that affect the dynamics (as friction coefficient of carpet and standard floor) and for different sensors (material with different coefficient of ultrasound reflection).

To run the system a 1.5 GHz Pentium 4, equipped with a 64 Mbytes GForce 4 MX440 graphic board, was used. The system was able to run at 30 Hz. Figure 8 and Figure 9 show different aspects of the environment and robot set up for this prototype.

7. CONCLUSIONS AND FUTURE WORK

The use of Real Time simulation of physical systems as a tool to help both in the design of new systems and in the developing of control elements has been proved to be really useful. However, it must also be mentioned that the development of reduced and reliable models of complex physical systems that reproduce their real behaviour is still a hard task that is far from being automated. This task is even harder when it comes to reproducing the interaction between a machine and its environment, as in the case of the example analysed in the paper.

As far as the graphic and simulation system is concerned, it is important to point out that it performed well for the prototype testing and evaluation. The system has also shown itself to be flexible enough to be reconfigured for other kinds of robots with different architecture and locomotion systems. In this way the simulation environment constitutes an interesting research and teaching tool because it allows real evaluations of control strategies in a simulated HIL environment, without the need for building the actual robots. This practice has been the first step in actual projects, where of course the final refinement has to be carried out with real robots. However, the system can save time during the control design process.

Figure 8. Environment designed for the vacuum cleaner prototype evaluation

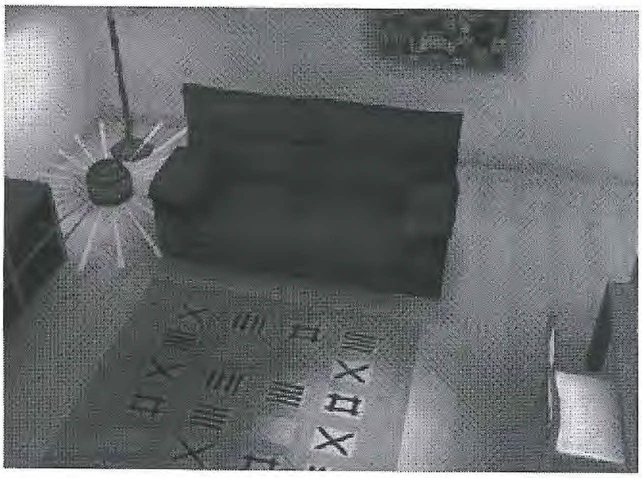

Figure 9. Active sensors in the vacuum cleaner robot

So the group working on this research project suggests that it is necessary to go on studying the establishment of a working methodology to simplify at most the generation of complex systems models, oriented towards both Real Time simulation on low cost platforms, and the generation of interaction models between the system and its environment.

The current system performs well for controlled and low reactive environments. However, the simulation of complex reactive environments, where more complex control strategies need to be evaluated, will be considered in the future. The new kind of environments will include the simulation of force feedback from the environment, deformable and elastic materials and contact textures.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of the Spanish Ministry of Science and Technology for the development of this research as part of the PROSIGRAT project (DPI2000-0283-C02-01): "HIL based Virtual Prototyping and Real Time Graphics Environment for Mechatronic systems".

REFERENCES

[1] Funkouser, T.A. Database and Display Algorithms for Interactive Visualization of Architectural Models. PhD. Thesis. 1993. University of Berkley.

[2] Landaluze J., I. Portilla, M.A. Otaduy, A. Martínez and R. Reyero. Virtual Prototyping of a Hydraulic Cushion for HIL. 2002. ESM 2002.

[3] Landaluze J., I. Portilla, A. Bidarte, A. Martínez and R. Reyero. Prototipaje Virtual de una Prensa Mecánica con Cojín Hidráulico. XIV Congreso de Máquinas-Herramientas. 2002. Donostia.

[4] Otaduy M.A., A. Martínez, A. Vidarte, J. Landaluze and R. Reyero. Object-Oriented Modelling for Real-time Simulation. 2000. ASME (American Society of Mechanical Engineering) IMECE Automated Modelling Symposium.

[5] Rohlf, J. and J. Helman. IRIS Performer: A High Performance Multiprocessing Toolkit for Real-Time 3D Graphics. 1994. SIGGRAPH Proceedigns.

[6] Smith R. "Open Dynamic Engine". 2000. http://opende.sourceforge.net/

[7] Urzelay, J., J.P. Uribe and J.M. Ezkerra. Fuzzy controller for wall following with a non-holonomous mobile robot. 1997. Fuzzy IEEE.

Contact Information

IJES

For further information, and any queries about the International Journal of Engineering Simulation (IJES), please contact:

Dr. I. A. Oraifige (Journal Co-ordinator)
School of Engineering and the Built Environment
(SEBE)
University of Wolverhampton
Wulfruna Street
Wolverhampton
WV1 1SB
UK

Telephone: (01902) 322258
Fax: (01902) 322743
International Code: (+44 1902)
E-mail: ijes@wlv.ac.uk

Chief Editors:

Professor F. R. Hall
Telephone: (+44) 1902322287

Professor M. Mihsein
Telephone: (+44) 11625770891902

UNIVERSITY OF
WOLVERHAMPTON

PRINTED BY
UNIVERSITY OF WOLVERHAMPTON PRESS