MARVIN - UNIVERSAL ROBOT-SIMULATOR

Zoltán VÁMOSSY, Viktor LANCSARICS, Tamás MELEG

Budapest Polytechnic
John von Neumann Faculty of Informatics
Institute of Applied Informatics
Budapest, 8. P.O.B. 112, Hungary H-1431
Phone: 0036 1 3689 840, Fax: 0036 1 3689 632
E-mail: vamossy@novserv.obuda.kando.hu

Abstract: The aim of this project is, the making of a PC program, which can simulate any manipulator-arm or walking-robot. The system guarantees, after the offline programming, the movement, that control of certain types of robots.

The primary aim is to provide the top freedom for the users, i.e. everything will be configured freely, starting from a virtual environment, the look of the robot, to hardware controlling. All of this can be done with an easily understandable script language.

Keywords: simulation, robot programming, mobile robots, robot arms, graphic

1. INTRODUCTION

Most of the robot simulators are applied in the industry, in the education, or in the research projects. The industrial applications, like the IGRIP (Deneb Robotics, 1993), or the COSIMIR (Festo Didactic, 1994) are usually built for different application areas, but the simulated robots consist mostly of manipulator arms. It is important to make a distinction between the PC's and the workstation applications. The simulators of the education and research area are not included in the above mentioned projects as they usually provide possibilities of examining a fix programmed robot that could either be a walking-robot or a manipulator-arm, depending on the aim of the project. Examples of such projects include the SLR-1500 manipulator simulator (Poppeova, 1998) (which is made for education purpose), or the Genghis (Rode, 1994) (which shows the walking methods of the famous six legs robot, but does not provide any other function).

Robot-simulation software has evolved with hardware. As a good example the Workspace4 (Robot Simulation Ltd., 1999), allows us to design our own robots, to plan paths, or to control different machines. Workspace4 can also simulate walking robots but only with great difficulty. Even though the presentation, the user interface, has some of the best test results, it still does not stand up to modern computer capacity.

Based on the above, it is still worth developing systems, which allow the simulation with any manipulator-arm or walking-robot. We consider of great relevance the simulations of mobile robots, because the programs that we tested do not support this simulation, and only guarantee the examination of single built-in robot. The system we are developing facilitates the work of those people who want build robots, but will not spend their time by making a simulator program. The program make it possible to demonstrate rough ideas in a short time to examine movements, in such a way that will save a lot of time and money.

2. SIMULATION

Since people are more attuned to their visual senses, most stimulation reach us by sight, increasing the requirements of the simulators that give us more detailed and better visual information. The authors tried to fill these requirements by using the Silicon Graphics' OpenGL Graphical Function-Collection, which has the serious advantage (in addition to being advanced), that it has been implemented to several platforms, and many PC's

(3D) graphic card can accelerate the execution of OpenGL functions with hardware.

The objects being viewed are 3 dimensional polygons, which must be resolved to triangles for easier storage and presentation (Füzi, 1997). (Three co-ordinates describe the triangles, and the order of these numbers determines the norm of the triangle). The 3D view of an average industrial robot can be approximated with several hundred triangles (be referred as faces), but in the case of several mobile robots the number of faces can easily be more than 10000, and all of these triangles must be filled and shaded for adequate visual effects.

Following the short description of data structures, let us see the essential modules of the system.

Modeling module: This allows the testing of each module, and the arm of the robot that the users want to simulate, and to properly specify the measurement of freedom of movement (as described in the script). There can be seen how the robot reacts, and how it will look like in several settings. The user is able to define the architecture of the robots (geometry, material, freedom-measurements) in script files with only one limitation: the connection between the arms of the robot must be tree-structured.

Animation module: The aim of this module is the simulated check of control sequences and to ensure the interactive control of hardware. The users can specify each movement in the scripts. They are able to define the individual motion characteristics of the robot, such as the default drifts, the various functions (forward, back going, rotate...etc.), even define conditions concerning execution order. The system currently permits sending of data over the parallel and serial ports.

Path planning module: This program supports the path planning functions. In later versions we would like to implement additional algorithms to the present path-planning methods and freehand path drawing. The program is able to manually manipulate (move over + rotate) objects over specific terrain, or in the course of planning it can simply leave them out. The handy and remarkably suggestive path-planning panel gives numerous parametering opportunities. The generated path can be surveyed at once in the virtual environment (terrain), showing the robot's path in the working space.

Hardware module: This part ensures the connection between the simulated and the real objects. With the later mentioned hardware driver script the developed system is able to create a real-time connection between the computer and the robot, thus giving the possibility of remote robot controlling or exploration of unknown terrain.

The computer demands of software are strongly dependent on the complexity of simulatable units, but essentially a home computer is enough for running. Currently the system runs under Windows 95/98, but it is planned versions for other platforms as well.

3. THE SCRIPT LANGUAGE

As it has been pointed out, to achieve intense freedom and configurability, all data originated from the terrain and essential for the robot are loaded from short text files, so- called scripts. The data corresponding to the main functionalities of the program are stored in separate files to increase reusability. For example, separable files (scripts) contain the data of the terrain objects and the arms, as well as the relations between the arms, the animation scenarios...etc. The user can create these files. He only has to be familiar with the grammar of the scripts, and has to have a simple text editor. This naturally is not applicable in all cases: the computation and entering the co-ordinates of a complex robot would be a tedious task. For that very reason there is a possibility to import objects, drawn in 3D Studio.

For this reason the system has a built-in Script Editor, allowing importation and editing of ASCII format scripts. For example 3D Studio Max is able to export files in this format, allowing the reutilization of already created objects, as well as processing files from other CAD systems through 3DSMax.

The above-mentioned scripts can be divided in the following seven well separateable groups: terrain, material library, geometries, connections, robot, animation, and hardware driver.

3.1 Terrain

The terrain is used for the description of the terrain surrounding the robot. The terrain consists of terrain-objects, which are described in local co-ordinate systems, but they all contain a translation and a rotation value compared to the origin and axles of a global co-ordinate system. Beside these data, a material name can be added, which defines the color, the shininess and other important features of the object.

3.2 Material Library

The material library contains the material-feature values associated to the material names. This data can be loaded from files what makes the terrain files (and as we will see, the robot-arms too) much shorter and easier to edit. Since the data of the Material Library are difficult to understand, there is a Material Editor in the program, which displays the materials and offers the optional editing. To make the visualization more realistic the users can modify very much material parameter, for example shininess, emission and transparent materials can be used also.

3.3 Geometries

Geometries contains the descriptions of the arms of the robot. It is very similar to the terrain script. The difference is, that no rotation and translation values are listed, but there is a part, what describes the connection points, where the new arm-parts can be connected.

3.4 Connections

Connections describe the connections between the elements of the robot. In this script, all parts of the robot are listed with the connections between them: what part is connected to what, and the rotation and translation values.

3.5 Robot

Robot contains the enumeration of the geometries and connections files, describing the robot. It was not mentioned at the geometries, but not all parts must be in the same file, and there can be parts in the file, what are not used in the actual configuration. The applied parts are listed in the connections script. Therefore there is a file, which contains the list of files of the needed geometries.

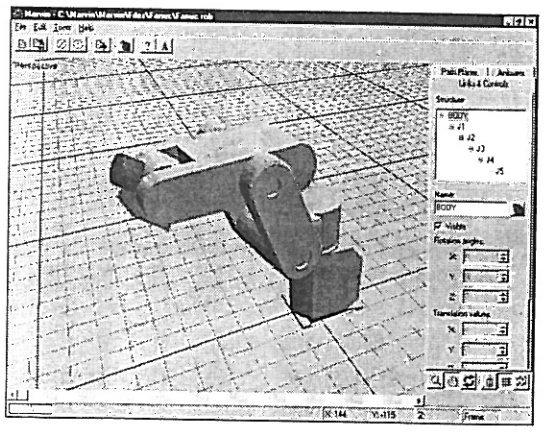

Fig. 1. Simulation of Fanuc LR Mate 100 (Fanuc Ltd.)

3.6 Animation

Animation contains the data associated to the motions of the robot. To all high-level motions (Forward, Left...) the user assigns a short scenario, what describes the positions of the arms in the key frames. The speed of the animation can be modified during the animation, and delays can be included too.

3.7 Hardware Driver

Hardware driver describes the data essential for the control of a physically existing robot. If the high level movements with an appropriate precision are described, the same movement will emerge as the user can see on the display.

In spite of the simple structure of the scripts the user can easily make a grammatical or logical error at the typing. In order to avoid the false operating, these mistakes must be recognized and corrected as soon and as accurately as possible. Before the controlling data of the robot is loaded into the memory, the system performs large number of controls, beginning with checking the number of opening and closing brackets and ending with the check of looplessness of the robot. This error checking algorithm doubles the loading time, but ensures the correct operating of the system. The present version is able to detect over 100 types of errors. The errors are displayed on the screen and saved to a log file. The location and short description of the error is also available.

4. RESULTS, CONCLUSIONS

The authors have started examining the simulating abilities of the system on a manipulator arm that is built up of 5 arm-links (Figure 1.). It consists of 2000 filled, shaded triangles. The presentation was a little bit slow on the first configuration, which was a Pentium with 150Mhz processor. We built in the wire frame imagery to eliminate this problem, and in this way it could be received adequate results. In the following step the program was tested on a 450Mhz Celeron based PC. Its higher floating-point calculation capacity allowed us to see its extraordinary capabilities: the moving was better, faster and it gave off higher frame rate. The test was continued with OpenGl compatible 3D accelerator cards. The experiences were very favorable: already a first generation 3D card has given a preferable performance, compared to the software rendering on a high capacity computer.

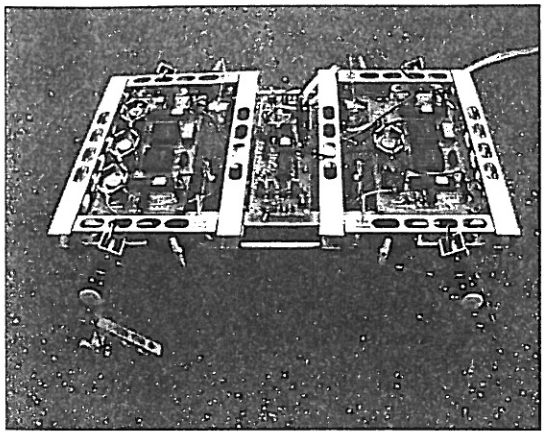

Fig. 2. Exploratores II during movement

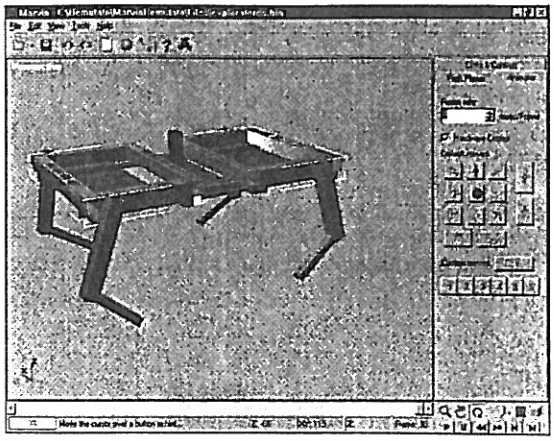

Fig. 3. Simulation of Exploratores II

The objective of the next simulation was a Genghis (Brooks, 1989) walker robot. There was not any speed problem during the test, because the architecture of the robot was relatively simple.

The results of the simulation often deflected from the experience of real life. To bring both them together, there are built such possibilities into the program, with which the users can control their robot in the same way, on the screen and real life. The user must create a new script, describing the communication channels and the given commands. This script describes the hardware commands, delays, and conditions for each frame of the animation

Thus far, in the course of development, we control using our program a manipulator-arm and a four-legs walking robot. There was auspicious experience in both cases.

The first robot was the 4-leg Exploratores II (Vámossy et al., 1988) (Fig. 2 and Fig. 3.), developed for educational and research purposes. Due to high-level commands of the robot, its control was very simple and it follows the changes occurring on the screen perfectly, although for the precise following in real-time the movements of the robot the user has to describe a very detailed animation script and scheduling.

There were several positive experiences on the second attempt: the Rob3 manipulator arm, made for education intentions. It submitted the commands perfectly. The only negative experience was that sometimes the settings deflected from the requirements due to the attrition of construction.

Our great experiences have left us with the wish to continue the development of the hardware controlling, primarily of all with concentrating on to the path planning. Besides this, there are many development possibilities. The authors wish to take the environment, physical effects (frictions, accelerations, gravity, etc.) into consideration for a more realistic simulation also. It would be advantageous to develop a 3D editor (for easier creating, modifying of the robots and terrains), and to support more standard 3D file formats.

REFERENCES

Brooks, R. (1989) A Robot that Walks; Emergent Behaviours from a Carefully Evolved Network, *Neural Computation* 1, 1989 pp. 253-262

Deneb Robotics (1993) IGRIP Homepage, http://www.deneb.com

Fanuc Ltd. *FANUC LR Mate 100 technical documentation*

Festo Didactic (1994) Die Revolution in der Robotersimulation: COSIMIR, Festo Didactic

Füzi J. (1997) *3D Grafika és animáció PC-n (3D Graphics and Annimation: in Hungarian)*, ComputerBooks, Budapest

Poppeova, V. (1998) The Simulation Program of Training Robot SLR 1500, In: *Proc. of the 9th International DAAAM Symposium*, (Ed. B. Katalinic) 22-24th October 1998, Cluj Napoca, Romania

Robot Simulations Ltd. (1998) Workspace Homepage, http://www.rosl.com

Rode N. J. (1994) *Simulation of a Subsumption Architecture Robot: Genghis*, ftp://daneel.rdt.monash.edu.au/pub/techreports /RDT/94-3.ps.Z

Vámossy, Z., A. Molnár, R. Brünner, L. Varga, (1998) Exploratores II, the Four-Legged Mobile Robot, In: *Proc. of the First International Symposium on Climbing and Walking Robots (CLAWAR'98)*, (Ed. Baudoin) BSMEE, Brussel, Belgium