A Proposal of Autonomous Robotic Systems Educative Environment

Jorge Ierache, Ramón Garcia-Martinez, and Armando De Giusti

Computer Science PhD Program, Computer Sc. School, La Plata National University
Instituto de Sistemas Inteligentes y Enseñanza Experimental de la Robótica FICCTE
Universidad de Morón
Intelligent Systems Laboratory, Engineering School, University of Buenos Aires,
Instituto de Investigación en Informática LIDI, Facultad de Informática, UNLP
jierache@unimoron.edu.ar, rgarciamar@fi.uba.ar,
degiusti@lidi.info.unlp.edu.ar

Abstract. This work presents our experiences in the implementation of a laboratory of autonomous robotic systems applied to the training of beginner and advanced students doing a degree course in Computer Engineering., taking into account the specific technologies, robots, autonomous toys, and programming languages. They provide a strategic opportunity for human resources formation by involving different aspects which range from the specification elaboration, modeling, software development and implementation and testing of an autonomous robotic system.

Keywords: Robotic, Autonomous Systems, Technologies in Education.

1 Introduction

The development of the technologies applied to education contributes to the learning process; particularly the application of an Autonomous Robotic Systems Laboratory (ARSL) collaborates with different areas in the training process of Information Technology students, from the interpretation of requirements to the autonomous system implementation, enhancing student's creativity as regards physical construction, software optimization and sensor integration, as well as the development of cooperative and competitive environment among robots. Watching how a turtle moves around in our monitor, while avoiding virtual obstacles to reach its goal in the corner of the monitor, does not have the same emotional impact on a student as observing how an Autonomous Robotic System (ARS) can avoid obstacles to achieve its goal in the corner of a room, and interacts with us by means of our mobile phone. We consider that the Computer Engineering, especially those associated with the Autonomous Robot Laboratories become an aid to learning processes in the case of beginner and advanced students; in the contextual framework [1], stated in figure 1, it is considered: [a] Paradigm under which the student carries out his/her work,[b] Methodology that is applied under the selected paradigm, [c] Techniques that facilitate the development of the phases and stages of the methodology applied, [d] Tools on which the techniques are applied, [e] Programming languages, [f] Robots**.** In this order, for

J.-H. Kim et al. (Eds.): FIRA 2009, CCIS 44, pp. 224–231, 2009.

beginners we can consider the imperative paradigm, that of objects, their methodologies and techniques, such as the Nassi-Shneiderman Diagrams [2], UML [3], programming languages like C, particularly NQC [4], and JAVA, in particular LeJOS Java [5], for the development of software running on RCX [6], NXT [7] robots. For advanced students, we consider the multiagent paradigm, with methodologies like MaSE [8], [9], techniques like Agent-UML [10].

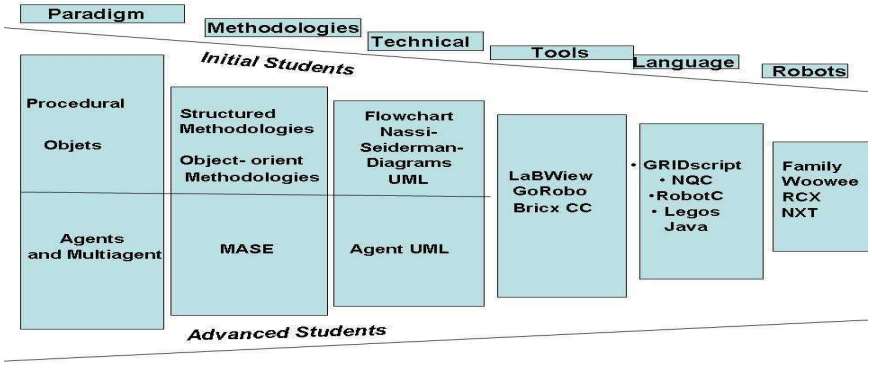

Fig. 1. Contextual Framework Learning

Meanwhile, the ARSL makes it easier to state explicitly the requirements under the IEEE 830 Standards [11], their validation and contextual framework the testing with the application of techniques such as Complexity Menasure [12]. It also contributes to improve the construction processes in a teamwork environment, where students are highly motivated for the development of their robots or pets. Many projects involve a centralized control, the computer instructs motor 1 to start, to turn in a clockwise direction, at half power under a planned action sequence, but the same robot agent can be applied to explore decentralized systems and those of self-organized behaviors [13]. For instance, if we consider an agent that wanders around its habitat, which has lit areas and dark areas, our agent has two rules, one indicating it to move forward when lit areas are detected and the other indicating to move backward when dark areas are detected; the agent wanders about until it reaches a shaded spot, so it moves back until it comes out of that spot and moves forward again; it goes on oscillating at the edge of the shade; in this case, we can consider our robot agent as a creature that detects edges; this capacity is not explicitly stated in its two rulers, in fact it is a group behavior which emerges from the interaction between the two rulers, similarly to the way in which a flock behavior emerges from the interaction between birds [14]. At different moments, students tend to consider their creatures at different levels; they sometimes see them at a mechanistic level, when analyzing how a piece of the mechanism moves another one. At times, they see them at an information level, and they explore how information is transmitted from the computer to the motors and sensors. On another occasion, they see their creatures at a psychological level, by attributing them a certain purpose or personality. One creature wants to go towards

light, another one prefers darkness, another one is afraid of noise. Students go quickly through these levels and learn according to the context situation what level is better; they think about systems in terms of levels [14]. The idea of learning through design is one aspect of what Seymour Papert [15] called "constructionist approach to learning and education". The human beings build their knowledge in a particularly efficient way when they participate in the construction of products they are emotionally involved in.

2 Robots, Languages and Tools

The objective of this section is to give an overview of today´s inexpensive LegoMindstorms, RCX Robotic Kits and of the latest NXT, their programming tools in the Robot C [16], NQC, LeJOS , among others. The RCX is characterized by having: three ports for motors, five slots to keep programs, a Light sensor, which enables to distinguish different levels of light and dark, two touch sensors, which enable to detect three states (pressed, released, bumped); it also has a loudspeaker for sound emission. The program downloading is carried out by means of the infrared tower included in the kit. The communication with other RCXs is possible via their infrared port on the front. The NXT is the new generation of Legomindstorms robots; it is characterized by having higher computing power than the RCX. The NXT includes some functions to test the sensors, to personalize the sounds it may reproduce, three ports for motors, four ports for sensors. It is equipped with a light sensor, a sound sensor, two touch sensors, which enable to detect three states (pressed, released, bumped), an ultrasound sensor functioning as a radar, thus enabling the detection of object, which may be set to detect close or distant objects; it detects objects at a distance from 0 to 255 centimeters with a precision of +-3 centimeters; it also has a high-fi loudspeaker, improved, and three servo motors, that have been improved as regards the RCX version. The servo motors have built-in rotation sensors which enable precise and controlled movements and a perfect motor synchronization. The NXT has a USB port, intended for program downloading. It supports Bluetooth wireless communication, thus enabling both program downloading and interaction with cell phones, PCs and laptops, etc. The communication with other NXTs is also carried out via bluetooth. These robots can be programmable in a native graphical environment, in the case of RCX [17] and LabView [18] in the case of NXT. Regarding LabView, it is worth mentioning that it was developed by National Instruments and used by the NASA to monitor and control Sojourner Rover robot, during the mission to explore the surface of Mars [19]. These environments use blocks which assemble with one another to form a complete program. These blocks include: motor control (forward, reverse, on and off), repetitive cycles (while, repeat), control structures (if else), data collection from the sensors, variable use, constants and timers. In addition to these graphical environments, there exists a series of programs which enable their programming in more traditional codes, such as Java. That is the case of the LeJOS API for the RCX [20] and the LeJOS NXJ for the NXT [21], iCommand is a Java package to control the communications over a Bluetooth connection [22]. One of the mostly used programs, which highly increases programming possibilities is the NQC [23], [17], developed by Dave Baum and used to program the RCX in a language similar to the C one. For the

NXT there exists a program called RobotC [16], which is much more complete than the NQC for the RCX, and includes its own firmware that makes it very powerful. Here follow the most important features of the main programming tools of Lego Mindstorms. Among the languages, the ones that can be mentioned are: NQC for the RCX and Robot C, similar to NQC, for the RCX, however it is much more powerful and enables robot programming in limited C. It includes a firmware, support for Bluetooth communication. This is one of the new languages existing nowadays to develop with NXT Lego Mindstorms. For JAVA NXT programming it can be mentioned the Lejos Java. Its alternative firmware for the NXT is characterized by: [a] enabling program development in JAVA in order to monitor NXT robots, [b]functioning under Windows and Linux, [c] communicating with the NXT via USB, [d] supporting Bluetooth communication; NXT firmware enables a Master/Slave-type set up for Bluetooth communication. Up to three NXTs can communicate via Bluetooth. The new JAVA API for the NXT is called iCommand, that includes, among other features, Webcam Support and Electronic Compass Support.

The Multimodule Robots are introduced as Bioloid Robot kit [24], is characterized by having a total of 18 servomotors, infrared sensors in the head to communicate with other robots and sensors to detect proximity forward and towards its sides, microphone and loudspeaker. Bioloid Comprehensive is the modular robotic platform kit suitable for building advanced robots having up to 18 degrees of freedom like humanoids. It is suitable for learning, hobby, research and competition. The kit is like an upgraded version of Meccano and is made up with many constructive blocks the user may assemble with screws. Its programming language is C.

3 Autonomous Toys, Programming languages and tools

Regarding Autonomous Toys, here follow the most relevant features of “Robosapien” (humanoid robot) and “Robopet, Robotail, Roboraptor” (quadruped robots), Roboquead (hexapod robots) from the Woowee family [25].Moreover, communication interfaces and programming tools are considered, particularly GoRobo. Although they are sold as toys, they offer so advanced features that they become an excellent way of experimenting on robotics. The humanoids robots have stereo sound sensors, infrared vision, and touch sensors to detect obstacles and several degrees of freedom. We can find in this category: [a] Robosapien V1 is a version with less features concerning sensors than the V2, it does not incorporate vision, the displacement capacities are similar in their functionality, though the RS-V1, being smaller, has a better displacement, [b] Robosapien V2, apart from the above-mentioned characteristics, it includes touch sensors in the gloves, and in palms of its hands, thus enabling to take objects. It also has a camera which lets it recognize colors. [c] Robosapien Multimedia increases even more the capacities of the RS-V2; it includes as an important characteristic a mini SD memory, in which it can be directly programmed, by means of a graphical-type code editor, existing only in this version. It has 4 personalities by default, which can be modified by the user. It can also record videos and mp3-format sounds, take pictures, and then reproduce them all on its Liquid Crystal Display (LCD).

The Quadrupeds robots [25] are also equipped with infrared vision, stereo sound sensors and motors. The most relevant ones are: [a] Robopet: apart from the

above-mentioned characteristics, it is able to interact with Robosapien, and also detect edges, for instance table edges. [b] Robotail has a touch sensor on its back, which by being pressed makes the robot have a different behavior. Moreover, when it is "hungry", it becomes very "aggressive" and can only be calmed down by "finding food". [c] Roboraptor is the roboreptile that is able to interact with Robosapien. [d] the arthropods are introduced as Roboquad, which has four legs with a chassis designed to move in any direction at three different speeds, has as a special feature that of identifying motion at a distance of about three meters; once identified, the robot can follow the object movement. It has edge sensors which allow to detect doorframes, table and chair edges.

A "GoRobo" programming environment allows to control most of the above-mentioned robots from the WowWee family (Roboraptor, Robopet, Roboreptile, RSV2 and RS Multimedia). The programming language used is called GRIDscript (Go-Robo ID script) [26]. It uses a simple and consistent programming syntax, based on modern commercial practices of programming products (Visual Basic, C++, etc). GRIDscript uses a basic programming syntax (While/EndWhile, For, If/Else/Endif, Repeat/EndRepeat), for the creation of procedures and the use of variables. The beginners can use this language to define simple procedures which may be later combined to create more complex ones. Moreover, the robot can be programmed to interact with each other, since the software allows the simultaneous control of six of them. The commands are transmitted via an infrared tower that, as an interpreter, sends them to each robot, by identifying the type through an infrared tower, such as USB-UIRT [27] and RedRat3 [28]. Here follow some actions to be carried out with GoRobo: use of conditional and instruction repetition blocks, use of events conditioned by timers, possibility of introduction of random code execution. This language was designed to be suitable for every age and to be used in both an educational and a professional context, where there exists an interaction of classical and formal programming languages with the natural commands of the robots that are used. It also includes a scene editor which can be upgraded with sound. Other programming options, for the Robosapien and the Robopet, receive their commands via IR through a remote control; in this way there exist those that have performed a mapping to hexadecimal of said commands [29], thus making it possible the Robosapien´s programming by downloading the code with the Lego Mindstorms´ IR Tower. The problem is that there is a constraint in the quantity of instructions that can be received by the RS, up to twenty. Another more radical option is the brain transplant to the Robosapien; sometimes it has been decided to replace the Robosapien V1 head with a Palm [30], in this way the problem of the quantity of instructions that can be sent to RS is eliminated, greatly improving the Robosapien V1´s ability to do calculations.

4 An Application Case between Robots and Autonomous Toys

This case simulates the behaviour of a herbivore, wandering along its environment developed with the NXT robot, which in this case, was a carpet outlined by a wooden structure, with green papers distributed at random representing food. So this herbivore (the prey) wandered easily until it found a food area. At this point it stopped to "feed" and it was also able to detect the borders of the habitat, thanks to its touch sensor and avoid them. At the moment the prey, with its sound sensor, detected the sound of a predator, represented by the Robotail (autonomous toy) or something got the

backwards position (detected by the ultrasound sensor) it "scared" failing to eat and beginning to run away at high speed. In this case, you can clearly see two types of behavior, the first one that looks for food and the other one, that flees, both depending on the NXT interaction with its environment and with Robotail (figure 2). The NXT (robot) which represents the prey was programmed with RobotC and the Robotail (toys) which represents the prey was programmed in a GoRobot environment.

Fig. 2. Robot NXT (prey) interacting with the environment and Robotail (predator) TE&ET 07,

5 Autonomous Robot System Development Laboratory

The Autonomous Robot System Laboratory (ARSL) presents an opportunity for students´ learning, particularly in the context of programming robots that work in dynamic and cooperative environments and require the creation of strategies aimed at reaching their goals to confront their opponents, without the action of external supervision. The Autonomous Robot System control programs cannot define explicitly every possible action in view of all the possible situations that may arise in its environment. The robot must not be fully pre-programmed; it must have a cognitive architecture that enables to establish a relationship between its sensory input and its actions on the environment [31]. It must have the ability of generating its autonomous sensorization map–actions to survive and achieve its goals. An Autonomous robot System Laboratory (ARSL) also offers a favorable scenario for the development of applications centered on context where the participation of robot and human players may be of interest in an interactive environment through cell phones, Internet, etc. The initial communication strategy to support the interaction between autonomous agents and human beings is based on the use of the possibilities provided by the wireless Bluetooth communication among agents. The advanced students are also interested in the methodologies in multiagent context, tools, intelligent autonomous systems, Artificial Intelligence concepts, vision and distributed processing [32]. Moreover, an ARSL can include global information from the environment by means of the integration of a vision system that allows the detection and localization of objects and autonomous robots in the scenario; in this case, the complexity level is even higher, thus enabling

that, besides processing the information given by its own sensors, the robot may have information of everything happening around it and be dynamically adapted, interacting among them and with the environment, as well as develop capabilities to facilitate sharing of knowledge between systems of autonomous robots [33], [34].

6 Conclusions and Future Research Lines

The use of robotic technology proposed on the present paper helps the development of different educational experiences such as robot soccer, rescue, navigation and so on. the experiences developed by the students within the robotics laboratory context turn out to be stimulating for them as they can see the result of their work through the action performed by their robots while strengthening the learning process. Furthermore the present paper has been developed on the last five years´ experience with a participation of an average of twenty initial level students per semester, working in teams for the construction of robots, scenarios, software development and tests. On the advanced level an average of eight students worked for two semesters, they developed final works where robots were integrated with the application of intelligent systems techniques and multiagent methodologies.Future research lines are aiming to the development of a framework where different robots are integrated, to the development of interoperating simulation capability between virtual and real worlds in order to support the robots learning scenario.

Acknowledgements

This research is supported by PID A01-007- FICCTE-UM.

References

1. Ierache, J., Bruno, M., Dittler, M., Mazza, M.: Robots y juguetes autónomos, una oportunidad en el contexto de las nuevas tecnologías en educación. In: VIII Ibero-American Symposium on Software Engineering, pp. 371–379 (2008)
2. Nassi, I., Shneiderman, B.: Flowchart techniques for structured programming, SIGPLAN Notices XII (August 1973)
3. UML, http://www.uml.org/
4. NQC – Not Quite C, http://bricxcc.sourceforge.net/nqc/index.html
5. Lejos, Java for Legomindstorms. SourceForge, http://lejos.sourceforge.net/
6. Lego Mindstorms RCX, http://www.lego.com/eng/education/mindstorms/
7. Lego Mindstorms NXT, http://mindstorms.lego.com/
8. DeLoach, S.: Analysis and Design using MaSE and agent Tool. In: Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference, MAICS (2001)
9. Ierache, J.: Elaboración de una Aproximación Metodológica para el desarrollo de Software Orientado a Sistemas Multiagentes (2003), http://www.fi.uba.ar/materias/7570/index.htm
10. Bauer, B., Muller, J.P., Odell, J.: Agent UML: A Formalism for Specifying Multiagent Software Systems. In: Proc. ICSE 2000 Workshop on AOSE 2000, Limerick (2000)

11. IEEE recommended practice for software requirements specifications -IEEE Std 1028-1988, IEEE Standard for Software Reviews and Audits (ANSI) Software Requirements Specifications. IEEE. Std 830-1
12. Mc Cabe, T.: A Software Complexity Menasure. IEEE Transactions on Software Engineering 2(4), 309–320 (1976)
13. Resnick, M.: Tortugas, Termitas y Atascos de Tráfico, Gedisa, Barcelona (2001)
14. Morrollon, M., Segoviano, A.: 1, 2, 3... Logo (Ideas e Imaginación). Centro de Orientación de Sociología y Psicología Aplicada. Cospa, Madrid (1985)
15. Papert, S.: Situating constructionism, en I. Harel y S. Papert (comps.), Constructionism. Abel Publishing, Norwood (1991)
16. Quick start guide, Robotics Academy, Carnegie Mellon University, http://www.robotc.net/
17. Baum, D., Hansen, J.: NQC, http://bricxcc.sourceforge.net/nqc/doc/NQC_Guide
18. National Instruments. LabVIEW, http://www.ni.com/academic/mindstorms/
19. National Instruments LabVIEW Software Monitors Health of Mars Pathfinder Sojourner Rover (1997), http://findarticles.com/p/articles/mi_m0EIN/is_1997_July_18/ai_19593795
20. Lejos RCX, http://lejos.sourceforge.net/p_technologies/rcx/lejos.php
21. Lejos NXJ, http://lejos.sourceforge.net/p_technologies/nxt/nxj/nxj.php
22. Icommand.NXT, http://lejos.sourceforge.net/p_technologies/nxt/icommand/icommand.php
23. Baum, D.: NQC Manual, http://bricxcc.sourceforge.net/nqc/doc/NQC_Manual
24. Bioloid Constructive Kid, http://www.tribotix.com/Products/Robotis/Bioloid/Bioloid_info1.htm
25. WowWee, http://www.woowee.com
26. Go-Robo, http://www.q4tecnologies.com/
27. USB-UIRT, http://www.usbuirt.com/
28. RedRat3, USB Universal Remote Control, http://www.redrat.co.uk/RedRat3/index.html
29. Lego IR-Tower.Trondheim-Bratislava, http://www.robotika.sk/maine.php
30. Sven, B., et al.: Using Handheld Computers to Control Humanoid Robots Proceedings Dextrous Autonomous Robots and Humanoids (2005)
31. García Martínez, R., Borrajo, D.: An Integrated Approach of Learning, Planning and Executing. Journal of Intelligent and Robotic Systems 29, 47–78 (2000)
32. Wooldrige, M., Jennings, N.: Agent Theories, Architectures and Languages: a Survey in Eds. Intelligence Agents 1(22) (1995)
33. Ierache, J., Naiouf, M., García Martínez, R., De Giusti, A.: A Un modelo de arquitectura para el aprendizaje y compartición de conocimiento entre sistemas inteligentes autónomos distribuidos. In: VII Ibero-American Symposium on Software Engineering pp. 179–187 (2007)
34. Ierache, J., García-Martínez, R., De Giusti, A.: Learning Life-Cycle in Autonomous Intelligent Systems. World Computer Congress. In: Bramer, M. (ed.) Artificial Intelligence in Theory and Practice II, pp. 451–455. Springer, Boston (2008)