A Behavior-Based Approach for Educational Robotics Activities

Pablo De Cristóforis, *Associate Member, IEEE*, Sol Pedre, *Student Member, IEEE*, Matías Nitsche, Thomas Fischer, Facundo Pessacg, and Carlos Di Pietro

***Abstract*—Educational robotics proposes the use of robots as a teaching resource that enables inexperienced students to approach topics in fields unrelated to robotics. In recent years, these activities have grown substantially in elementary and secondary school classrooms and also in outreach experiences to interest students in science, technology, engineering, and math (STEM) undergraduate programs. A key problem in educational robotics is providing a satisfactory, adequate, easy-to-use interface between an inexpert public and the robots. This paper presents a behavior-based application for programming robots and the design of robotic-centered courses and other outreach activities. Evaluation data show that over 90% of students find it easy to use. These activities are part of a comprehensive outreach program conducted by the Exact and Natural Science Faculty of the University of Buenos Aires, Argentina (FCEN-UBA). Statistical data show that since 2009 over 35% of new students at the FCEN-UBA have participated in some outreach activity, suggesting their significant impact on student enrollment in STEM-related programs.**

***Index Terms*—Behavior-based robotics, educational robotics, mobile robots, robots programming interface, STEM outreach.**

I. Introduction

EDUCATIONAL robotics proposes the use of robots as a teaching resource that enables inexperienced students to approach topics in fields unrelated to robotics. One of its aims is to aid students in building their own representations and concepts of science and technology, through the construction, handling, and control of robotic environments, as well as through collaboration teamwork. The main idea is that knowledge is constructed rather than discovered, and that students' learning significantly improves when they are actively involved in building something meaningful to themselves. These approaches are based in educational theories such as Piaget's constructivism [1] and Papert's constructionism. In particular, Papert's work is closely related to robotics, science, and technology, including the development of Logo [2] and the founding ideas of LEGO robotic kits [3].

The use of education robotics in elementary and secondary school classrooms has grown substantially in recent years, driven by the availability of robots and programming platforms. A recent review in the area yielded 197 published works over the last 10 years [4]; the results indicate that educational robotics is a valuable tool for improving learning, but this needs to be more strongly established through further experience and, above all, through empirical evidence. Another point demonstrated in this study is the lack of empirical research using robots other than LEGO in education since 90% of the works discussed used some of the LEGO kits.

Educational robotics has also been used in outreach experiences to interest students in science, technology, engineering, and math (STEM) undergraduate programs. The number of students enrolling in these undergraduate programs around the world appears to be declining, or at least is not growing at the required rate. In Argentina, only 5% of undergraduate students go into engineering careers upon graduation, and each year only one engineer graduates per 6700 inhabitants. This is very low compared to countries such as China (1 in 2000), Germany or France (1 in 2300), and even other Latin American countries such as Mexico, Chile (1 in 4500), or Brazil (1 in 6000) [5].

The lack of STEM graduates has led governments around the world to institute educational plans and led many universities to institute experimental outreach programs. In the US, the Academy of Robotics of Carnegie Mellon University, Pittsburgh, PA, has a broad offering of outreach activities, such as competitions, camps, and courses to foster teamwork in project development [6]. In Europe, at the University of Reading, Reading, U.K., outreach activities include the use of robots in interactive talks at schools, competitions, and exhibitions and a collectable fortnightly magazine that sold 23 million copies and contributed to 500 000 domestic robots being built worldwide [7]. In the Czech Republic [8] and Spain [9], remote robotics laboratories have been developed that enable an alternative approach. These e-learning systems allow students to run robotic applications on real hardware without being present in person. In Latin America, since 1999 the University of Chile, Santiago, Chile, has developed robotic-centered outreach activities including robotics courses based on LEGO Mindstorms [10] and the use of social robots to give motivational talks to schoolchildren [11]. In Colombia, since 2004 the University of Valle, Cali, Colombia, has offered an introductory course in robotics and has designed and tested eight-day courses using LEGO Mindstorms [12].

In Argentina, the Exact and Natural Science Faculty of the University of Buenos Aires (FCEN-UBA), Buenos Aires, has a comprehensive outreach program that includes science weeks, exhibitions, talks in schools, scientist-for-a-day programs, and several eight-week courses from different disciplines [13]. The Robotics and Embedded Systems Laboratory (LRSE) of the FCEN-UBA participates in this initiative, developing robotic-centered courses, talks, and exhibitions targeted toward computer science careers. In contrast to most of the previous studies mentioned, these activities use low-cost robots fully developed in the LRSE specifically for outreach programs,

Manuscript received May 21, 2012; accepted August 07, 2012. Date of publication November 12, 2012; date of current version January 30, 2013.

The authors are with the Departmento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Buenos Aires 1428, Argentina (e-mail: pdecris@dc.uba.ar).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TE.2012.2220359

research, and teaching in university curricula [14]. This further attracts schoolchildren, who see that their following a STEM-related career could result in them developing robots in Argentina.

One of the obstacles facing the LRSE in developing these outreach experiences was the lack of a satisfactory application for programming the robots. This application had to be easy to use for students with no previous knowledge of robotics or programming and had to be platform-independent. By developing such an interface, the LRSE proposed a solution for a key problem in educational robotics: an interface between an inexpert public and the robots.

This paper describes this application and the design of robotic-centered courses and other outreach activities such as talks and exhibitions. Evaluation data on the application and the courses is provided along with statistical data showing the impact of these activities on student enrollment in the STEM programs at the FCEN-UBA.

II. Related Work

This section discusses various applications for programming robots in the educational context.

Many interfaces have been designed for university-level or older high school students and implemented as extensions to existing programming languages. One example is Pyro [15], a Python-based programming framework that allow students to write platform-independent robot programs. Other interfaces include Not-Quite C (NQC) [16] based on C, BrickOS [17] based on C++, and leJOS [18] based on Java. These three are specific for the LEGO Mindstorms kit. All these interfaces require programming experience or interest in learning a particular programming language. This makes them unsuitable for educational robotics courses for middle or high school students who cannot handle any imperative or procedural programming concepts, such as loops, conditions, forks, or variables. Microsoft also offers the commercial tool Microsoft Robotics Developer Studio [19]. This is a visual programming interface based on a data-flow approach, but it also requires knowledge of programming concepts.

There are also several graphical environments for simulated robots aimed at middle schools, such as Start Logo [20], Squeak Etoys [21], and Scratch [22]. These are easy-to-use programming interfaces, getting inexperienced students off to make a quick start, but they are designed only for particular simulated environments. Maybe the most frequently used programming interface in instructional settings at the K–12 level is RoboLab [23] for the LEGO Mindstorms robot, a graphical environment in which students are given palettes of icons that they can drag and drop on a canvas. The icons represent robot components like motors and sensors, as well as abstract programming structures such as loops and counter variables. However, this interface is particular to the Mindstorms robot, and it also uses imperative programming structures that add complexity to the robotic environment.

III. Application: Requirements, Design, and Didactic Approach

A. Requirements and Design

The design of the new application, called Easy Robot Behavior Programming Interface (ERBPI), follows a certain set of requirements derived from the shortcomings observed in similar previously described programming environments. These requirements are the following.

1) *Ease of use:* The user is not required to have any previous programming or robotics knowledge. The interface must be intuitive and easy to learn, providing all the tools to program robot behaviors graphically.
2) *Platform independence:* The application must work with a variety of robots and simulators and be easily extendible to control new ones. The various bodies, sensors, actuators, low-level commands, and protocols required to communicate with different robots must be abstracted. Moreover, users must be able to test and run the same behaviors on multiple robots.
3) *Flexibility:* Students from a wide range of backgrounds and teachers with a broad range of goals must be able to use the programming interface effectively; the interface must accommodate different levels, curricular needs, academic subjects, and physical environments for instruction.
4) *Portability:* The application must work on various operating systems and platforms to accommodate the various hardware and software available in schools or other educational institutions.

As shown in Section II, most of the existing applications address the robot programming task using the imperative paradigm. In order to meet the Ease of Use and Flexibility requirements, the design of ERBPI abandons this classical approach and takes a behavior-based approach. This allows the user to define behaviors (as opposed to explicit programs) and encapsulates the low-level step-by-step programming. The application implements the idea of Braitenberg vehicles, where the robot's sensors and actuators are simple components that can be connected by various mathematical transfer functions to achieve a reactive behavior [24]. Since this alone is not enough to address several commonly used behaviors for teaching purposes, a subsumption architecture is also included [25]. This architecture allows complex behaviors to be built by combining simpler ones. To further accommodate the Ease of Use requirement, the user interface is completely graphical and is encapsulated in a separate module (i.e., the GUI module). In Section III-C, a brief description of the GUI can be found.

The Platform Independence requirement is addressed by introducing an abstraction layer that hides the particularities of each robot and simulator and exports a common interface to the rest of the application. In this manner, the application works with an "abstract robot," leaving the connection and control of each particular robot to a special module: the *Robot Abstraction Layer* (or RAL). Hence, there is one implementation of the RAL module per robot that the ERBPI can control. Each RAL module implements the following common interface: initialize and deinitialize the robot, get the frequency on which the robot can accept commands, get the list of the robot's sensors and actuators, and get and set the sensors and actuators values, respectively. In order for this design to work regardless of the sensor type used in each robot, the RAL normalizes the values for sensors to a [0, 100] "activation" range, and the values for actuators to a $[-100, 100]$ "motion" range.

Part of this Platform Independence requirement is that the application can be easily extended to control new robots. Hence,

the RAL module is implemented as an operating-system dynamic library that can be redefined in execution time. Moreover, all the GUI configurations, including which robots, sensors, and actuators the application can manage, are defined through XML files. Thus, to add a new robot for ERBPI to manage, the developer just has to implement its RAL module and change the configuration file for the GUI. There is no need to recompile or change any code in the application.

The previously defined modules (GUI and RAL) are related by means of a third central module: the CORE. This module interprets the behavior defined in the GUI and executes it in the robot. To achieve this, it periodically asks the RAL for the robot's sensor state, applies the appropriate transfer functions to calculate the actuator values, and tells the RAL to set these values to the robot's actuators.

To address the Portability requirement, the implementation of the whole application uses only portable libraries to access operating-system-specific functionality. Moreover, the GUI module is programmed in Java, which runs on multiple platforms. The application is compiled for each platform and distributed in a self-contained package. Finally, ERBPI is implemented under the GNU GPL (ver. 2) license.[1]

B. Behavior-Based Approach

The design of ERBPI abandons the classical imperative paradigm and takes a behavior-based approach [26]. This didactic approach allows inexperienced students to program robots intuitively and easily, using Braitenberg vehicles and a subsumption architecture to effectively achieve complex and challenging behaviors.

A Braitenberg vehicle [24] is a conceptual model of a mobile robot. These vehicles enable the development of experiments to progressively illustrate the capacities of various simple agents or creatures and analyze emergent behaviors. The vehicles are composed of three elements: 1) wheels (generally two); 2) sensors of different kinds (heat, light, proximity, etc.); 3) connections between sensors and wheels (which can be a excitatory, inhibitory, or other mathematical function). In consequence, vehicles are completely reactive, requiring no type of explicit internal processing and no knowledge of the environment or its own previous states.

While this definition enables relatively complex behaviors to be obtained combining different kinds of sensors and defining a network of interconnections, when trying to approach a multigoal task, another layer needs to be added to keep the model simple and didactic. This layer is the subsumption architecture, where the desired behavior is decomposed into several simpler reactive behaviors. These simpler behaviors can be defined separately, and reused, facilitating the development of more complex behaviors.

The subsumption architecture is modeled with a Finite State Machine, where each state is a basic reactive behavior defined with the Braitenberg model, and a transition between two states occurs when a certain condition with respect to sensor values, timers, and counters becomes true.

C. How to Program a Behavior

To program a behavior, the user interacts with the Graphical User Interface module. This interface has two views. The main view, shown in Fig. 1, controls the whole application and allows the state machine that represents the subsumption architecture to be graphically defined.

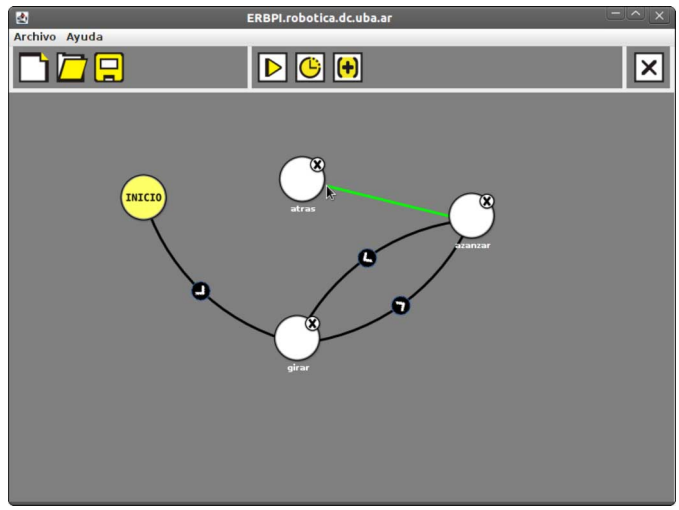


Fig. 1. Screenshot of the main view of the Graphical User Interface. The user can define several simple behaviors on the work canvas and connect them by using the mouse to define transition between behaviors. Global timers and counters can also be defined. The upper panel shows the main control functions. *(left) new, open, save. (center) play, new timer, new counter. (right) close.*

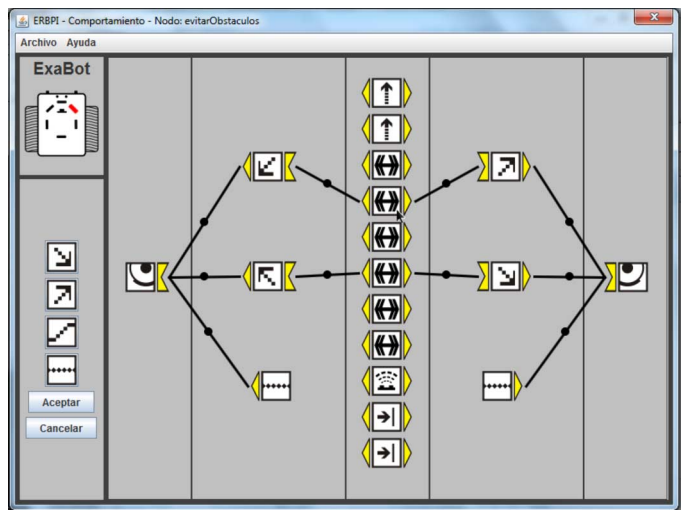
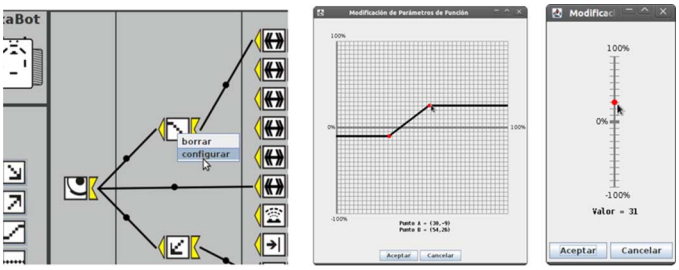


Fig. 2. Screenshot of the Braitenberg view. The left panel shows the robot schema and the transfer functions that can be used (top-down: inhibitory, excitatory, broken, constant). In the center of the work canvas, all the sensors are shown (top-down: two line-following sensors, six infrared telemeters, one sonar, two bumpers). The mouse is over the right-front telemeter, which is colored differently in the robot schema (top left). The wheels are shown on each side of the work canvas. The programmed behavior is a simple explorer that moves around at constant speed and can avoid obstacles.

To start, the user opens the main view, selects a robot or simulator to work with, and defines an empty state by clicking on the work canvas. By double-clicking the new state, the other view of the GUI module pops up and allows the user to define the basic behaviors using the Braitenberg model. The Braitenberg view enables the user to drag and drop elements (sensors, actuators, functions) to a work canvas, and then connect them using the mouse, as shown in Fig. 2.

Functions that connect sensors and actuators can be configured with a pop-up configuration window, shown in Fig. 3. The Braitenberg also shows a schema of the selected robot. When

[1]The complete code can be found at http://robotica.exp.dc.uba.ar/trac/erbpi.



(a) (b) (c)

Fig. 3. Pop-up configuration windows for functions: (a) right-click on the function to open the configuration window, (b) configuration for broken function, and (c) configuration for constant function.

the mouse goes over the sensors or actuators in the canvas, the corresponding element is colored in the robot schema, so the user can easily visualize which sensors are connected to each function and their influence on the simple behavior that is being programmed. The user can save this simple behavior or load a previously defined one using the File tab of the menu. When the user clicks the accept or cancel button on this view, the focus goes back to the main view.

In the main view, shown in Fig. 1, the user can define several simple behaviors and connect them using the mouse. Global timers and counters can also be defined. To define when a transition between two states occurs, the user can set conditions on the timers, counters, and robots sensors. These conditions use the logical functions $=, \neq, >, \geq, <, \leq$. The main view also allows behaviors to be opened or saved, and their execution started and stopped.

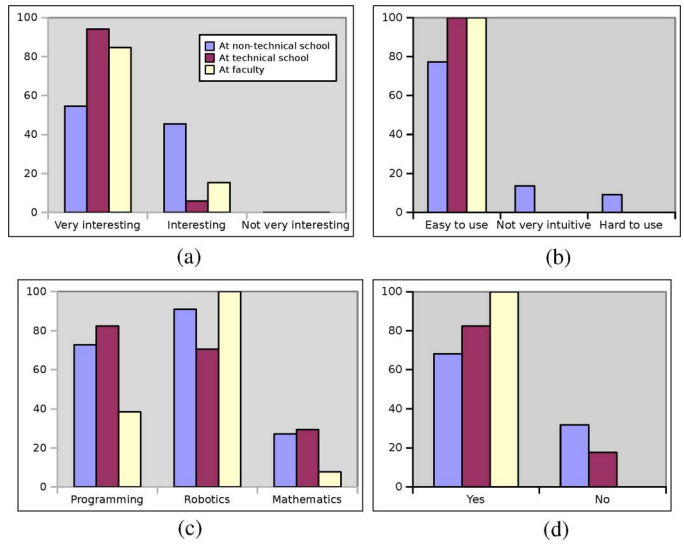
IV. Educational Robotics Activities

Since 2009, various versions of ERBPI were used in courses designed for high school students, and in exhibitions, workshops, lectures, and other outreach activities. Four robotic platforms were used in these activities: the Khepera robot, the YAKS Khepera simulator, the ExaBot robot [14] and a particular configuration of the Player/Stage simulator for the ExaBot robot. The ExaBot is a mini robot completely developed in the LRSE that includes a wide range of sensors (a ring of infrared telemeters, a sonar, two line-following sensors, two bumpers, encoders and a camera). It also includes a PC104 ARM-based embedded processor that provides good processing power and a wi-fi connection. Its sensing and processing capabilities are reconfigurable, making it suitable for different outreach, teaching and research activities.

A. Courses

Since ERBPI allows a wide range of behaviors to be defined using various mathematical and logical concepts, and because it also works with several simulators and robots, it can be used for a wide range of courses. Since 2008, the LRSE has used ERBPI in two types of courses: a short two-day course and a long eight-week course.

1) Short Course: The short course consists of two 3-h-long meetings, and covers the basic concepts of behavior-based robotics. For this course, students use only the Braitenberg view of ERBPI to achieve relatively simple behaviors. The course follows a hands-on approach, programming easy behaviors from day one in groups of two or three students. Each group alternately explains to the rest how they programmed the behaviors and why they made each decision. The course is summarized in Table I.



(a) (b) (c) (d)

Fig. 4. Results of the questionnaire for the short courses. (a) What is your opinion of the course? (b) What do you think of the robot programming interface ERBPI? (c) Was the course useful for learning about any of the following topics? (d) Would you like to find out more about programs at the FCEN-UBA?

TABLE I
Schedule of Short Course

Day	Tasks
1	Introduction to behavior-based robotics and Braitenberg vehicles. Easy problems with simple excitatory and inhibitory connections and one type of sensor. Problems to solve: following a light spot, exploring how to avoid obstacles. Robots: YAKS Simulator and Khepera robot.
2	Introduction to more complex mathematical functions (i.e, broken function), sum of functions and multisensor robots. Problems to solve: the "drunk-guy" (follow a wall at a certain distance and enter the first door), and line-following (follow a line on the floor and stop at a wall crossing the line). Robots: Player/Stage simulator and Exabot.

This course was conducted three times: once in July 2010, in the computer labs of the FCEN-UBA, and twice in May 2012. Twenty students selected from seven technical high schools in Buenos Aires, Argentina, participated in the course. The student's programming experience varied from little to some experience with procedural languages like C, C++, or Pascal. There were also a few students with some electronics background. None of the students had experience with robots.

In the second experience, the course was taken to a high school, and conducted it on-site. The class was a complete 11th grade class of 22 students. Since this was an ordinary non-technical school, the students had no knowledge of programming nor robotics, and only standard mathematical and scientific training. In the final experience, the course was conducted at a technical high school, where the students had some experience with programming languages and electronics, but none with robotics. The course consisted of 17 students from the 7th–12th grades.

After each course, a questionnaire was given to the students to elicit their views on the course and ERBPI. The responses are shown in Fig. 4.

2) Extended Course: The extended course was designed to be completed in eight 3-h-long meetings. The course covers

TABLE II
SCHEDULE OF EXTENDED COURSE

Day	Tasks
1	Introduction to behavior based robotics and Braitenberg vehicles. Easy problems with simple excitatory and inhibitory connections and one type of sensor. Problems to solve: following a light spot and avoiding obstacles. Robots: YAKS Simulator and Khepera robot.
2	Introduction to more complex mathematical functions (i.e., broken function), sum of functions and multisensor robots. Problems to solve: the "drunk-guy" (follow a wall at a certain distance and enter the first door), and line-following (follow a line on the floor and stop at a wall crossing the line). Robots: Exabot and Player/Stage simulator.
3	Introduction to the subsumption architecture to decompose multi-task behaviors into simpler ones. Problems to solve: the "drunk guy also avoids obstacles" (follow a wall at a certain distance and enter the first door, avoiding any obstacles that may appear across the path). Robots: Player/Stage simulator.
4	Mid-course experience sharing and competition. Resolve the "drunk guy also avoids obstacles" behavior with the ExaBot. A competition is held between groups, in which the robot that enters a door in the least amount of time wins.
5	Introduction to counters. Use counters to trigger transitions between simple behaviors. Problem to solve: the "drunk guy" enters the third door, avoiding the obstacles that he meets. Robots: YAKS, Khepera, Exabot and Player/Stage.
6	Introduction to timers. Use timers to trigger transitions between simple behaviors. Problem to solve: "line prisoner" - the robot explores an enclosed area bounded by a line on the floor. Robot: Exabot.
7	Introduction to logic clauses with multisensor conditions. Integration problem to solve: "robot sumo" - explore an enclosed area bounded by a line and push out any object the robot sees. A competition between groups is performed and the robot that pushes the most objects out of the arena in a given time wins. Robot: ExaBot
8	Integration, results and conclusions. Each group creates a poster that synthesizes the activities they did in the course. Discussion and experience sharing between the whole class.

TABLE III
RESULTS OF EXTENDED COURSES

Question The workshop helped you to...	Year	Answer Yes	Partially	No
choose your program	2009	65%	26%	9%
	2011	63%	32%	5%
know the Computer Science program in college	2009	78%	22%	0%
	2011	77%	23%	0%
meet the university environment	2009	83%	17%	0%
	2011	77%	23%	0%
know what computing people do and how they do it	2009	78%	13%	9%
	2011	68%	32%	0%
learn about issues related to computing	2009	91%	9%	0%
	2011	91%	9%	0%

the concepts of behavior-based robotics and the several experiments with simulators and real robots. The main goal of this course was for the students to develop different behaviors using the scientific method, encouraging them to propose hypotheses, contrast the expected results with those obtained in the testing phase, and then propose explanations and changes to the original robot control. For this course, the full version of ERBPI with both Braitenberg and subsumption approaches was used. The course followed a hands-on approach, programming behaviors from day one in groups of two or three students. Each group alternately explained to the rest how they programed the behaviors and why they made each decision. This course is summarized in Table II.

This course was conducted twice: from May to July in 2009 and from August to September in 2011, both times at the computer labs of the FCEN-UBA, each time with 15 students from various high schools. None of the students had experience with programming or robots. After the course, a questionnaire was given to the students to elicit their views on the course, the results of which are shown in Table III.

TABLE IV
IMPACT OF OUTREACH ACTIVITIES IN UNDERGRADUATE ENROLLMENT FOR COMPUTER SCIENCE AND ALL PROGRAMS

Year	Program	Enrolled students	Participants in some outreach activity	Percentage
2009	CS	106	48	45%
	All	789	278	35%
2010	CS	149	53	36%
	All	840	316	38%
2011	CS	163	57	35%
	All	923	306	33%

B. Workshops, Exhibitions, and Other Outreach Activities

ERBPI was also used in shorter courses, lectures, exhibitions, and other outreach activities for high school students and the broader public.

The shorter course is a one-day workshop during Computer Science Week [27]. Each year, over 1000 students from more than 50 schools participate in this Week, attending talks, workshops, and exhibitions related to the Computer Science program at the FCEN-UBA. The one-day workshop starts with an introductory talk about behavior-based robotics. Then, students form groups of three or four and use ERBPI to program a simple behavior, such as following a line on the floor or obstacle avoidance on the simulator and on real robots. This 2-h-long workshop was held in 2009, 2010, and 2011. The average number of participants varied between 20 and 30 students.

ERBPI, the ExaBot, and other robotic experiments were also used as outreach tools in several science exhibitions. The most important of these were ExpoUBA [28], Tecnópolis [29], and TEDxRiodelaPlata [30]. ExpoUBA was a three-day exhibition held by the University of Buenos Aires during October 2010, commemorating the 190th anniversary of the university. Over 70 000 people and more than 600 high schools visited the exhibition. The authors' robotics stand was part of the Science Plaza. As a result of an enrollment program at the exhibition, more than 7000 senior high school students were enrolled for further study at the University of Buenos Aires. Tecnópolis is a science and technology mega exhibition (based in Argentina and the largest in Latin America) that took place for the first time from July to November in 2011, and will continue in 2012. Over 1 million people visited Tecnópolis. The robotics stand was part of Innovar Competition, where the LRSE introduced the ExaBot as a new mobile robot prototype for educational robotics purposes. Various behaviors for controlling the robot with ERBPI were demonstrated. Also, in 2012, the LRSE was part of TEDxJovenRiodelaPlata (TED—Technology, Entertainment and Design) conference and exhibition.

Finally, ERBPI was also used in many lectures and talks in high schools to show how simple it can be to program an "intelligent" behavior for a autonomous robot.

C. Impact of Outreach Activities in Enrollment

When students complete the registration process to enter the Faculty, they also complete a survey indicating whether they had participated in any outreach activities of the FCEN-UBA.[2] Table IV presents the results of this survey.

These results show that more than 35% of the students enrolled in the Faculty, and particularly in the Computer Science undergraduate program, had participated in some outreach activity. This means that the activities described in this paper are

[2]Over 90% of the enrolled students complete the form each year.

part of a comprehensive outreach program that reaches thousands of high school students and has an impact in the enrollment in STEM undergraduate programs, particularly those at the FCEN-UBA.

V. Conclusion

In this paper, an easy-to-use application for programming robots was presented, and the design of robotic-centered courses and other outreach activities discussed.

The design of the application abandons the classical imperative programming paradigm to take a behavior-based approach. Following this approach, ERBPI allows the user to define simple behaviors graphically using the Braitenberg model and then combine them using a subsumption architecture. Hence, a student with no background in programming or robotics can easily start programming robots after a short explanation. The application is also designed to work with different robots and simulators and to be easily extendible to program new robotic platforms. It runs in both Linux and Windows to accommodate different computers available in educational facilities.

The design and development of various robotic-centered courses using this behavior-based approach with low-cost robots designed at the LRSE was also presented. The two-day course was conducted in the FCEN-UBA and in two different high schools: a technical-oriented one and a normal one. The evaluation data from these courses show that 90% of students find the tool easy to use, 75% thought the course was very interesting, and 81% wanted to participate in other outreach activities to find out more about the programs at the FCEN-UBA. Even in the nontechnical school, 77% of the students found the tool easy to use. The eight-week course was conducted twice at the FCEN-UBA. Over 60% of the students felt the course helped them to choose a program, and 77% found out about the existence of the Computer Science program. Other outreach activities such as talks and exhibitions that reached thousands of students and general public are also described.

These activities are part of a comprehensive outreach program conducted by the FCEN-UBA, the largest science faculty in Argentina. Statistics show that over 35% of students enrolled since 2009 at the FCEN-UBA, and in particular at the Computer Science undergraduate program, participated in some outreach activity. These results show a significant impact of these outreach activities on student enrollment in STEM-related programs.

References

[1] J. Piaget and B. Inhelder, *The Child's Conception of Space*. New York: Norton, 1967.

[2] S. Papert, D. Watt, A. di Sessa, and S. Weir, "Final report of the Brookline Logo project: An assessment and documentation of a children's computer laboratory," MIT Study and Research in Education, 1979.

[3] S. Papert, *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books, 1980.

[4] F. B. V. Benitti, "Exploring the educational potential of robotics in schools: A systematic review," *Comput. Educ.*, vol. 58, no. 3, pp. 978–988, Apr. 2012.

[5] Ministerio-Educación, "Plan estratégico de ingeniería 2012–2016," Apr. 2012 [Online]. Available: portal.educacion.gov.ar

[6] Carnegie Mellon University, Pittsburgh, PA, "Carnegie Mellon Robotics Institute," Apr. 2012 [Online]. Available: http://www.education.rec.ri.cmu.edu

[7] R. Mitchell, K. Warwick, W. Browne, M. Gasson, and J. Wyatt, "Engaging robots: Innovative outreach for attracting cybernetics students," *IEEE Trans. Educ.*, vol. 53, no. 1, pp. 105–113, Feb. 2010.

[8] M. Kulich, J. Chudoba, K. Kosnar, T. Krajnik, J. Faigl, and L. Preucil, "SyRoTek—Distance teaching of mobile robotics," *IEEE Trans. Educ.*, 2013, to be published.

[9] C. A. Jara, F. A. Candelas, S. T. Puente, and F. Torres, "Hands-on experiences of undergraduate students in automatics and robotics using a virtual and remote laboratory," *Comput. Educ.*, vol. 57, no. 4, pp. 2451–2461, Dec. 2011.

[10] J. R.-D. Solar and R. Avilés, "Robotics courses for children as a motivation tool: The Chilean experience," *IEEE Trans. Educ.*, vol. 47, no. 4, pp. 474–480, Nov. 2004.

[11] J. R.-D. Solar, "Robotics-centered outreach activities: An integrated approach," *IEEE Trans. Educ.*, vol. 53, no. 1, pp. 38–45, Feb. 2010.

[12] E. Milena, J. Jojoa, E. C. Bravo, E. Bladimir, and B. Cortés, "Tool for experimenting with concepts of mobile robotics as applied to childrens education," *IEEE Trans. Educ.*, vol. 53, no. 1, pp. 88–95, Feb. 2010.

[13] FCEN-UBA, Buenos Aires, Argentina, "Dirección de orientación vocacional," Apr. 2012 [Online]. Available: http://www.fcen.uba.ar/dov

[14] S. Pedre, P. de Cristóforis, J. Caccavelli, and A. Stoliar, "A mobile mini robot architecture for research, education and popularization of science," *J. Appl. Comput. Sci. Methods*, vol. 2, no. 1, pp. 41–59, 2010.

[15] D. Blank, D. Kumar, L. Meeden, and H. Yanco, "Pyro: A python-based versatile programming environment for teaching robotics," *J. Educ. Resources Comput.*, vol. 3, no. 4, p. 3, Dec. 2003.

[16] D. Baum, "NQC," Apr. 2012 [Online]. Available: bricxcc.sourceforge.net

[17] M. L. Noga, "brickOS," Apr. 2012 [Online]. Available: brickos.sourceforge.net

[18] J. Solorzano, "leJOS," Apr. 2012 [Online]. Available: lejos.sourceforge.net

[19] Microsoft, Redmond, WA, "Microsoft Robotics Developer Studio," Apr. 2012 [Online]. Available: http://www.microsoft.com/robotics

[20] MIT, Cambridge, MA, "MIT's Scheller Teacher Education Program (STEP)," Apr. 2012 [Online]. Available: education.mit.edu/drupal/starlogo-tng

[21] A. Kay, S. Papert, and K. Rose, "SqueakEtoys," Apr. 2012 [Online]. Available: http://www.squeakland.org

[22] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond, "The scratch programming language and environment," *Trans. Comput. Educ.*, vol. 10, no. 4, pp. 16:1–16:15, Nov. 2010.

[23] Tufts University, Medford, MA, "RoboLab," Apr. 2012 [Online]. Available: http://www.ceeo.tufts.edu/robolabatceeo

[24] V. Braitenberg, *Vehicles: Experiments in Synthetic Psychology*. Cambridge, MA: MIT Press, 1986.

[25] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Robot. Autom.*, vol. 2, no. 1, pp. 14–23, Mar. 1986.

[26] R. C. Arkin, *Behavior-Based Robotics*. Cambridge, MA: MIT Press, 1998.

[27] FCEN-UBA, Buenos Aires, Argentina, "Semana de la computación," 2012 [Online]. Available: http://www.dc.uba.ar/sdc

[28] UBA, Buenos Aires, Argentina, "ExpoUBA," Sept.–Oct. 2010 [Online]. Available: http://www.uba.ar/expouba

[29] MINCyT, Argentina, "Tecnópolis," Jul.–Nov. 2011 [Online]. Available: http://www.tecnopolis.ar

[30] TED, "TEDxRiodelaplata," Apr. 2012 [Online]. Available: http://www.tedxriodelaplata.org

Pablo De Cristóforis, (A'12) photograph and biography not available at the time of publication.

Sol Pedre, (S'12) photograph and biography not available at the time of publication.

Matías Nitsche, photograph and biography not available at the time of publication.

Thomas Fischer, photograph and biography not available at the time of publication.

Facundo Pessacg, photograph and biography not available at the time of publication.

Carlos Di Pietro, photograph and biography not available at the time of publication.