Low-Cost Interfaces for Learning Programming

Authors

Anjali Kulkarni, Center for Mechatronics, IIT Kanpur, anjalik@iitk.ac.in
Sarala Verma, Centre for Mechatronics, IIT Kanpur, saralav@iitk.ac.in
Ritu Raj Singh, Electrical Engineering Dept., IIT Kanpur, rituraj@iitk.ac.in
Abhaya Agarwal, Computer Science and Engineering Dept., IIT Kanpur, abhaya.agarwal@gmail.com
Amitabha Mukerjee, Computer Science and Engineering Dept., IIT Kanpur, amit@iitk.ac.in

Abstract *—This work describes an effort to introduce hands -on learning in schools. In India, access to computers at the school level is very limited, and learning the planning structures relevant to programming becomes difficult. This paper emphasizes the develop ment of low cost interfaces for learning programming using standalone programmable blocks and keyboard interfaces, which are both upgradable to an integrated programming environment based on Java. These tools have been developed under the "Build Robots Cr eate Science" (BRiCS) program for providing school children with experience of tangible programming and integrating their programs later under a graphical programming environment. All tools developed in this program are open source.*

Index Terms *— Educational tool, Hands on learning, PrISM, Programming block, BRiCSpe*

INTRODUCTION

Learning can be made interesting and stimulating by hands on experiments. Educational robotics has become an important tool for driving hands-on education [1-5]. In widespread tests conducted at schools across India as part of the “Build Robots Create Science” initiative [6], we find children responsive to the emotionally appealing topic of Robotics. To deploy hands-on practices we have developed three modules of low cost interfaces for learning programming. Two of these modules don’t need computers. These low-cost devices can be used by children to program interactive “hands-on” systems with simple sensors and motors that the child himself can build. The three modules are as follows:

- **Programmable Blocks**: Consists of a set of stackable blocks which can be mounted one on top of another to construct a program for a general purpose robot.
- **PrISM (Programmable Interface for Sensors and Motors)**: Implements a set of simple programming constructs on a Keyboard to program a general purpose robot.
- **BRiCSpe**: A computer based programming interface which provides interoperability between PBlocks and PrISM and supports GUI programming of these devices. (Figure 1)

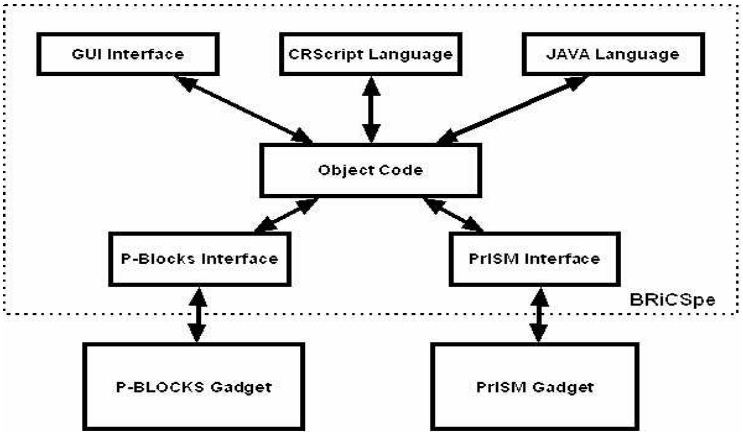

FIGURE 1
OVERVIEW OF BRICSPE INTEGRATED PROGRAMMING ENVIRONMENT. PROGRAMS WRITTEN USING PRISM OR P-BLOCKS CAN BE UPLOADED INTO BRICSPE AND CONVERTED TO OBJECT CODE, WHICH PROVIDES INTEROPERABILITY.

PROGRAMMABLE BLOCKS

Programmable Blocks [7,8] is a system consisting of physical blocks which are used to program and execute a robotic motion program. Each block represents a particular command and the program is implemented by stacking the blocks one on top of another. The supported commands are Move Forward, Move Backward, Turn Left, Turn Right, Sensor and Repeat – these are implemented for a general purpose robot having two motors, and two sensors (e.g. a Light and a Touch sensor). Though P-Blocks are sufficient to program the robot, the system also supports a computer programming interface using BRiCSpe. Figure 2 shows the Block Diagram of the Programmable Blocks.

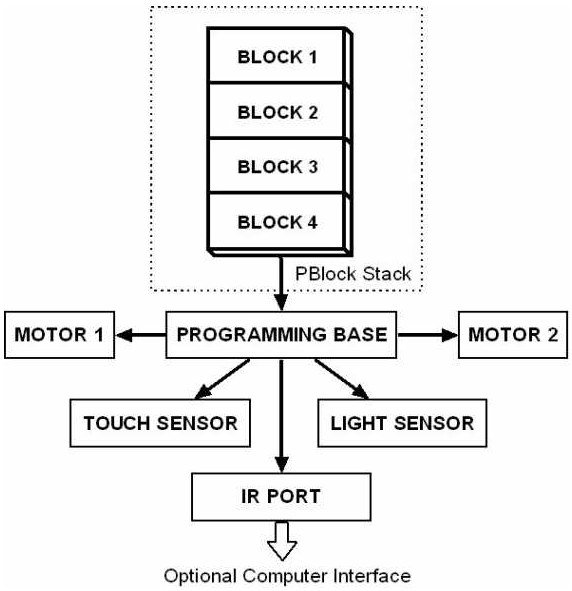

FIGURE 2
BLOCK DIAGRAM OF PROGRAMMABLE BLOCKS: THE MOTORS AND SENSORS ARE MOUNTED ON THE PROGRAMMING BASE, WHICH CAN THEN BE CONTROLLED BY STACKING THE BLOCKS.

WHY PROGRAM USING BLOCKS?

Programmable Blocks implement the concept of Tangible Programming [9-11]. *Tangible Programming* is the use of physical modules as building elements to construct a small program in real world. It is a better approach when coding simple sequential programs as is illustrated with the help of a comparison between a C program and an equivalent modular program in Figure 3. The modular program has been designed for a toy that teaches simple addition, subtraction to kindergarten kids. The addition result is displayed on the LCD display connected at the base. As is obvious, the modular program promotes *Hands on Learning* , reduces complexity and hence decreases the age when kids can be introduced to programming. Using the tool, even Kindergarten kids can get an early taste of programming. Concepts of Repeat, If-Then-Else, and Variables can be easily grasped as compared to the present teaching techniques.

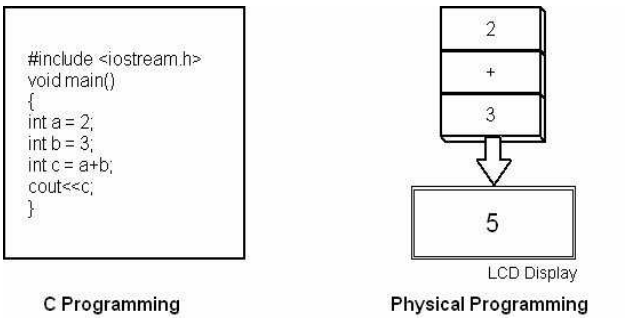

FIGURE 3
COMPLEXITY COMPARISON: C PROGRAMMING VS TANGIBLE PROGRAMMING

P-BLOCK

The P-Blocks form an integral part of the design and function as an elementary module for programming. Issues like size, cost, flexibility and packaging have been taken special care in designing the P-Blocks. The latest design of P-Block consists of an 8 bit Berk stick mounted on an 8.0cm X 2.5cm PCB. The 8 bits are used to specify the command along with the argument. First 4 bits denote the command while the next 4 bits denote the argument. For example, a block which represents a Forward motion command for 2 seconds will have the first 4 bits denoting the "Forward" motion command while the last 4 bits will represent two in binary arithmetic.

The block design is flexible since both command and argument can be changed. Each block can thus function as a multipurpose block which might represent a forward motion for one setup while could perform the function of a sensor on a different one, by configuring the 8 binary switches. The overall design too is flexible because of the freedom to stack in whatever sequence one desires. The cost of each block is less than $0.50, while the programming base costs $7.0. However, the current design of P-Blocks is not suited to children of very small ages because setting of the binary switches is a big impediment. Currently we are replacing the binary switches with a pointer knob to overcome this problem. Figure 4 and 5 show the two developed versions of P-Block.

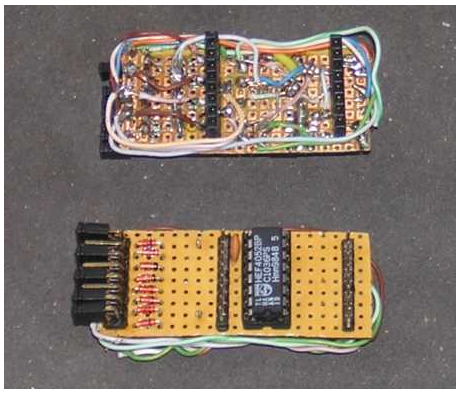

FIGURE 4
P-BLOCK PROTOTYPE WITHOUT CASING

FIGURE 5
EARLIER VERSION OF P-BLOCK WITH CASING

PROGRAMMING AND EXECUTION

The P-Blocks are first stacked on top of each other and then mounted on the Programming Base. Depending upon the command the configuration bits of each block are set. Figure 6. shows a sample modular code for moving a robot in a square.

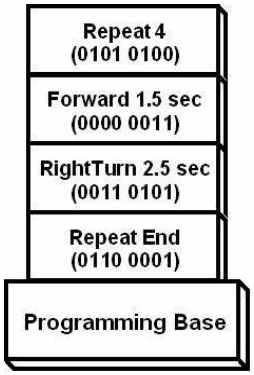

FIGURE 6
SAMPLE PROGRAM FOR MOVING ROBOT IN A SQUARE

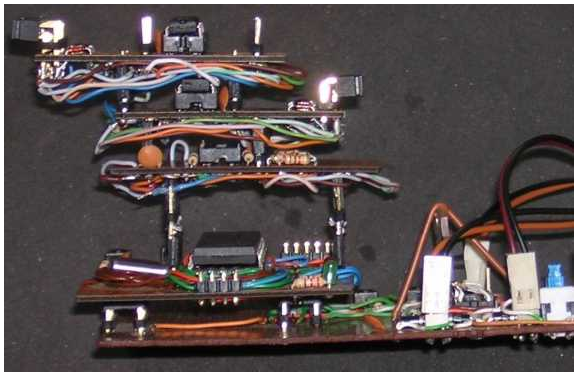

FIGURE 7
PROTOTYPE P-BLOCKS STACK MOUNTED ON THE PROGRAMMING BASE

Once the Block stack has been mounted on the Programming Base as shown in Figure 7, the Read Button on the base is pressed. The PIC Controller [12] at the Base scans through the data bits of each block. Pressing the Execute Button, starts the execution of the commands which are stored in the controller's EEPROM.

PRISM

PrISM [7] is a keyboard programmable device used to program small general purpose robots having 2 motors and 3 sensors (Touch, Light, and Sound). Program is keyed in using a Keyboard Matrix as shown in Figure 9, which is to write the program constructs. Aimed at eliminating the use of computers for learning programming, it is targeted for the age group 8-13 years. The system consists of a microcontroller as the control unit, motor driver circuits, liquid crystal display and the matrix keyboard. The functional overview is as shown in Figure 8.

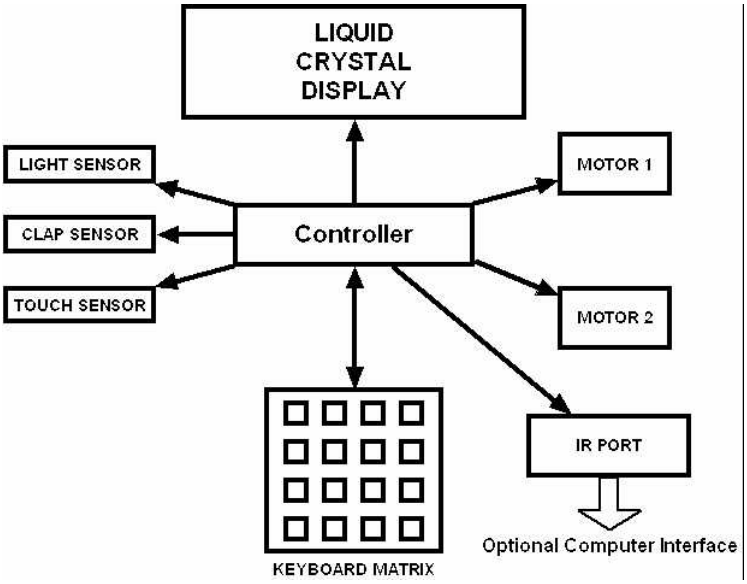

FIGURE 8
BLOCK DIAGRAM OF PRISM

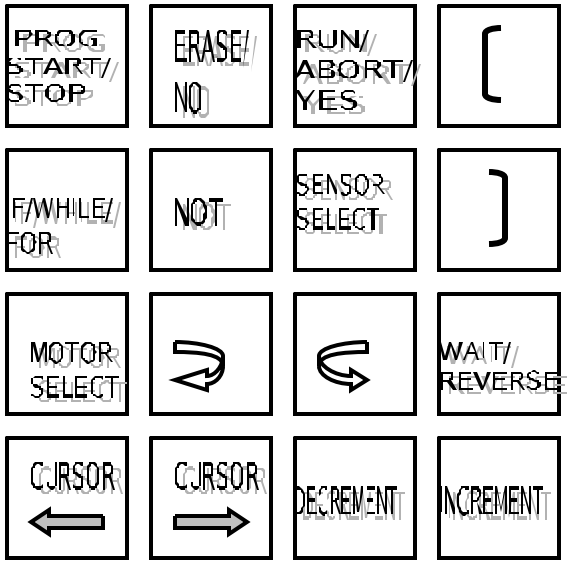

FIGURE 9
PRISM KEYBOARD LAYOUT

Figure 10 shows the photograph of the LCD module displaying typical commands used for programming. The physical structure of the toy robot can be varied depending on the application, that is it can be a car or a crane built with 2 motors and sensors. The approximate cost of the PrISM hardware is around $60.

FIGURE 10
SNAPSHOT OF THE LCD DISPLAY

Derived key functions can control two motors to rotate the robot in clockwise, anti clockwise directions, stop or reverse the current direction, for a specific time period. The Sensors can be sensed for being active high or low. The language developed for PrISM has three basic constructs:

- Motor Statement: Which defines the motor number, direction and duration, written as,
  (M1 → 05) to indicate Motor 1, clockwise 05 secs
- IF Statement: Which defines two different actions based on the Sensor status,
  IF (NO) S1 (Motor Stmt 1) (Motor Stmt 2)
  (NO) indicates an active low Sensor.
  The (Motor Stmt 1) is executed if the sensor condition is True and (Motor Stmt 2) if False.
- FOR Statement: Used for looping.
  FOR 03 (Motor Stmt) indicates that the specified Motor Stmt is executed 03 times.

The PriSM interface was tested on 10-12 year olds. In programming with the PrISM, although the kids started slowly but once they were adept with the programming constructs, they could easily interface motors and sensors. The toy was able to keenly capture their attention

EXAMPLE PROGRAM TO MOVE A ROBOT IN A SQUARE

```
START PROG
    FOR 04 (( MB → 05)(M1→02))
    END PROG
```

This program runs both the motors forward for 05 secs and one motor for 02 secs, repeating it 04 times thus completing a square.

BRiCS PROGRAMMING ENVIRONMENT

The software interface, also known as "BRiCS Programming Environment" or BRiCSpe in short, provides a higher level interface for programming various kinds of robotics controllers like Lego RCX[13], PRiSM and P-Blocks. It is aimed at supporting various modes of programming that may be suitable for different age groups.

FEATURES

The PrISM and P-Blocks described in previous sections are stand-alone devices. They alleviate the need of a computer to program simple robots. So a natural question arises that why should we build a programming environment for them which needs a PC to run. The idea here is to provide interoperability between the various hardwares available to a user. For example, the user may have written a program using PRiSM. Now he can import this program into BRiCSpe, edit it using any of the modes of programming available and then download/run it on any of the controllers that are supported. This frees the process of learning from the details of underlying hardware.

BRiCSpe supports various different modes of programming. On the one hand this multimodal programming experience makes user concentrate on the actual concept rather than only understanding the nuances of one programming method. On the

other hand, it also makes the user experience richer by letting him work in a way that is most comfortable to him. In order to make it accessible to even those who are not comfortable in English, we support a Hindi interface also.

PROGRAMMING MODES

BRiCSpe provides 3 modes that can be used based on the extent of familiarity with regular programming languages.

- **Visual Programming Mode:** The visual programming mode uses the concept of blocks that can be put together to form a complete program. There are many types of blocks. Some blocks represent various high level tasks like turning the motor on and off, setting the direction of motor etc. Then there are blocks that provide simple programming language constructs like loops and conditionals. The editor takes care that any arrangement of blocks that can be put together represents a syntactically correct program. This mode is ideal for those with no programming experience and children in lower age group. Figure 10 shows a snapshot of the Visual Programming Mode.

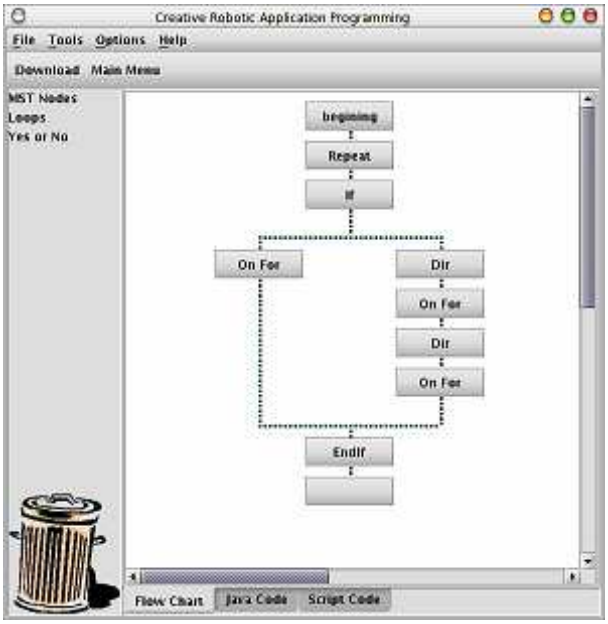

FIGURE 10
VISUAL PROGRAMMING MODE IN BRICSPE.

The design of the workspace improves upon the other currently available products. It incorporates various suggestions made by a study done on young children at the Industrial Design Centre, IIT Bombay by members of BriCS. The children were shown sample programs (e.g. make the robot move along a square) and asked to modify them (e.g. Make it move along a "E" shape). The points of difficulty of the children were noted. This identified and suggested solutions for many of the common problems that children face while working with existing interfaces. For example, one source of common confusion among the users is the difference between normal conditionals and sensor conditions in interfaces like Lego Mindstorms. We have tried to reduce the confusion by moving the sensor conditionals to separate panels.

- **Script Programming mode:** The other two modes involve writing programs in actual programming languages but depending on the level of proficiency, one can choose to either use the script language, called CRScript or the Java languages. CRScript has an easy to learn syntax which lets the user concentrate on the program logic. There is a one to one correspondence between one statement in CRScript and one block in the visual-programming mode. So in principle, this mode provides no extra power to the user in terms of the valid programs he can create. This mode is useful for those who have a little understanding of basic programming constructs. Figure 11, shows a snapshot of CRScript

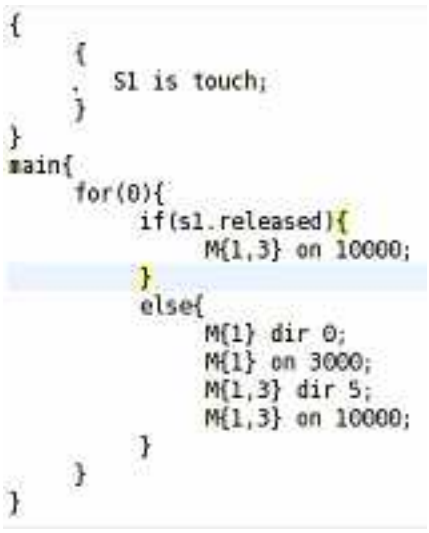

FIGURE 11
PROGRAMMING EXAMPLE IN CRSCRIPT

- **Java Programming Mode:** Java mode allows the user to program using the full features of Java language. This mode is much more powerful than the previous two modes and involves concepts of classes, threads etc that are quite advanced in nature. A basic requirement is the availability of a JVM for the target platform. Also the details of API may vary across different implementation of JVMs. Hence programs written is this mode are quite platform specific. We currently use lejos [14] which is a open source implementation of a subset of JVM for Lego RCX.

CONCLUSION

In this paper we have discussed two potential low cost learning tools for Hands on Learning. They do not require the use of a computer and are cheap. Hence these can be readily used in rural schools of India, where facilities like computers and electricity are not common. Use of P-Blocks and PrISM can make learning fun. They implement the idea of robotics because it is of general interest of children and the very idea of being able to instruct a robot, makes children ecstatic. It is fun accompanied with learning, which sows the early concepts of programming and debugging. With the support of advanced features like BRiCSpe it is desired to further utilize the concept of tangible programming using GUI interfaces and add more features without increasing the cost on basic hardware. Considerable inputs from children have gone into the design of all three interfaces. Our tests on children were positive and we were able to capture their attention, though a lot of improvements still need to done.

ACKNOWLEDGEMENT

This work has been made possible by support from the Department of Science and technology (DST, New Delhi), and also the Ministry of Information Technology under the Media Lab Asia programme. We acknowledge help received from Vibhor Jain, Nitigya Kuchhal, Abhinav Agarwal, Kushagra Singhal, Arpit Patni, and other members of the BRiCS team in the fabrication and testing of the P-Blocks and PRISM systems. We would also like to thank Aniruddh Joshi (Professor, IIT Bombay), Mrityunjay Gautam, Siddarth Wardhan, Utsav Maitra, Kunal Gaurav, Rachna Agarwal and Atul Patil who worked to identify the problems in the User Interface of Lego Mindstorms and also helped in development of BRiCSpe.

REFERENCES

[1] Kantor, G, Manikonda,V, Newman, A. and D.P. Tsakiris, "Robotics for high school Students in University Environment", *Computer Science Education Journal Special Issue on Robotic Education* , 1997.

[2] Kumar, Amruth N. "Using Robots in Undergraduate AI course", *IEEE Front iers in Education* , *Reno Nevada* , Oct 2001

[3] Martin, F. , Butler, D. and W. Gleason, "Design, Story-Telling, and Robots in Irish Primary Education", *IEEE Systems, Man, and Machine conference* , *Nashville TN* , 2000.

[4] Nourbakhsh, Illah R., “When students meet robots, IEEE Intelligent Systems and Their Applications”, Vol. 15, No. 6, p. 15. 2000

[5] Amitabha Mukerjee, “Build Robots Create Science - A constructivist education initiative for Indian schools”, *Dyd '02, Bangalore* , December 1-2, 2003

[6] Mukerjee Amitabha, Verma Sarala, Sinha Nikhil, "BRiCS- An Indian Experiment in Educational Robotics", *Proceedings of the International Conference on Engineering Education ,Valencia, Spain* , July 23-27, 2003.

[7] Mukerjee Amitabha, Verma Sarala, Singh Rituraj, "Build Robots Create Science (BRiCS) – Digital Learning Tools without a Computer", *Proceedings of the International Seminar on Downsizing Technology for Rural Development, Bhubaneswar, India,* October 2003.

[8] Invention Disclosure form, Title of Invention: Programmable Blocks. (Filed with the Ministry of Commerce through the Dean R&D IIT Kanpur)

[9] Mukerjee Amitabha, Sharma Gaurav, Prakash Manu, "Programming using Stackable Bricks", *Dyd 02, Bangalore, I ndia,* December 1-2, 2002.

[10] 'Tangible Programming Bricks: An approach to making programming accessible to everyone,' T. McNerney, S.M. Thesis, MIT Media lab, 2000.

[11] 'Electronic Blocks: Tangible Programming Elements for Preschoolers', P Wyeth and G. Wyeth, *Proceedings of the Eighth IFIP TC13 conference on Human -Computer Interaction* (Interact 2001).

[12] Microchip Technologies. www.microchip.com

[13] Robotics Invention System. Lego Mindstorms. www.legomindstorms.com

[14] lejos, java for RCX. http://lejos.sourceforge.net