Learning by creating: Interactive Programming for Indian High Schools

Niloy Gupta[1], Tejovanth N.[2], Preeti Murthy[3]

National Institute of Technology Karnataka, Surathkal, India

[1]niloy_gupta@ieee.org [2]tejovanth.n@ieee.org [3]preeti.murthy@ieee.org

Abstract— **In this paper we discuss results and observations based on empirical studies of introducing programming using Scratch–Arduino to high school students. We analyse the programming experience of students across diverse educational and economic backgrounds, culture and region. Learning of key programming and electronics concepts was measured during the exercise. Results indicate that these fundamentals can be imparted at high schools in the Indian educational context. We find that the introduction of logic programming and computer-hardware interfacing at the high school level is advantageous in terms of creating an interactive environment fostering learning and creativity.**

***Keywords*— Novice programming, embedded programming, K-12 Education, Scratch, Arduino, Computer science education, Indian high schools.**

INTRODUCTION

The Indian high school education system has recognized computer science as an essential subject in the K-12 curricula. Educational boards such as the Central Board of Secondary Education (CBSE), Council for the Indian School Certificate Examination (CISCE) and various state boards have included computer science or computer applications as an optional subject in the pre-university curriculum [11,12]. Educationists across the world have acknowledged the importance of introducing logic programming at the high school level as a means of improving student's analytical and quantitative skills. However, it has been observed that no K-12 curriculum includes hardware interfacing concepts. Also, especially in India, many students from backward economies find programming, in the traditional way using C/C++/Java, challenging because of lack of exposure and programming construct [4]. Most schools use technology to teach content but very few schools provide the opportunity to learn programming, especially in lower income groups. [2,6,7,10]

In this paper we discuss the results of introducing elementary programming and hardware interfacing to students of age groups 15-18 using the Scratch-Arduino platform. Scratch is a programming language designed for novice programmers [1,13]. Whereas, Arduino is an open-source electronic prototyping platform based on flexible, easy-to-use hardware and software. We conducted a series of workshops for over 150 students in which we developed various applications involving use of programming logic and interfacing concepts. The results of the empirical studies indicate a sharp increase in the logic, creativity and learning abilities of the students. Especially, we have observed that programming with Scratch-Arduino is a very effective method of inculcating a spirit of creativity among students. This is evident from the myriad applications that were developed by the students during the study.

The paper is divided into 4 sections. Section II introduces the Scratch-Arduino platform and explains its benefits. Section III discusses the experimental setup and observations recorded. In Section IV we analyse the results. We conclude the discussion with recommendations and future scope in section V.

THE SCRATCH-ARDUINO PLATFORM

The Lifelong Kindergarten Group at the MIT Media Laboratory in collaboration with Yasmin Kafai's group at UCLA has created Scratch, a programming language and environment aimed for children.

There have been various developments in the area of programming environments for children, the popular and widely used one being Logo [8], which is still being used in many schools in India. Logo brings with itself the advantage of being simple to understand and teach. Logo is an effective tool used for teaching logic, arithmetic and geometric concepts. However, there is lack of user interaction because of the command line interface.

Scratch improvises on the ideas of Logo and replaces command line by a drag-and-drop approach. This greatly simplifies the programming process as it eliminates syntax errors. Along with this Scratch gives immediate feedback after placing a block of code, thus facilitating rapid application development. Using Scratch one can develop rich applications such as animated stories, games and interactive presentations.

As of December 9, 2007 the Scratch online community registered over 56,352 projects, 915,489 scripts and 53,639 members.[16] A good number of users are educators who create projects in Scratch, even though many of them know other programming languages. Visitors come from 213 countries, mainly from the United States. But a good number come from UK, Canada, Australia, Japan, Germany, Brazil, Spain, France and India.

Thus, Scratch is slowly getting accepted as a medium for learning in numerous high schools. Few urban schools in India are aware of the Scratch project and are experimenting with it.

Table I. Age wise performance of the students.

Age	Control Set	Experimental Set	Gain
14-16	24.19	65.48	41.29
17-18	36.72	63.86	27.14

Table II. Gender wise performance of the students.

Student	Control Set	Experimental Set	Gain
Girls	28.83	60.75	31.92
Boys	32.80	65.24	32.44

However, no curriculum includes interfacing hardware to a computer system and developing embedded system applications. The reason primarily being the microcontroller programming is too complex to teach to a high school or pre-university student. Embedded programming involves sound knowledge of programming concepts and electronic fundamentals. But Scratch also presents a simple solution, whereby students can make many electronic projects by interfacing Scratch to a microcontroller.

The most popular one is the PicoBoard designed by the creators of Scratch themselves. Using which students and hobbyists have created many kit-projects. But in our experimental setup we have interfaced Scratch to an Arduino using the S4A (Scratch 4 Arduino) platform.

S4A was developed by Marina Conde, Victor Casado, Joan Guell, Jose Garcia and Jordi Delgado with the help of Smalltalk Programming Group Citilab [14]. Our choice of Arduino over PicoBoard is due to fact that with PicoBoards projects are limited in scope. Moreover Arduino being an open hardware platform gives the liberty of modifying the design and structure to suit one’s purpose. The major reason for the choice is that electronic and interfacing ideas can be explained and introduced better with Arduinos.

S4A provides high level user interface with features such as interaction of different plates by user events. The simple and interactive nature of the S4A platform makes it very appealing to novice programmers and the drag-and-drop feature eliminates the cumbersome task of establishing/terminating port connections and sending/receiving data and syntax errors. This removes the initial roadblocks and hesitation when one begins to work with microcontrollers.

We employed these two technologies in teaching programming and electronic concepts to students during our study.

EXPERIMENTAL SET-UP

Many educational boards such as CBSE and CISCE, have are encouraging educators to use computers as a tool for teaching. This is the primary motivation for our empirical study, whereby we selected a group of 150 students from different educational backgrounds (educational boards), income groups, gender and age group (15-18yrs) and conducted a series of tutorials and workshops. Few of the students had prior elementary programming experience but none of them had previous knowledge of embedded programming or microcontrollers. Interfacing was limited to students above grade 10, due to their lack of prior knowledge in elementary concepts of electricity.

The tutorials and workshops were designed to give practical experience to novice programmers and involved developing applications. Students were asked to develop their own ideas and create animated games, videos and educational applications. This gave us the opportunity to evaluate their creativity skills in terms of user interface design, character design and game/application design. The study resulted in the creation of applications in the form of animated stories in regional languages, joystick games using switches and LEDs. Some of them being an interactive calculator, animated stories “Hare & The Tortoise” in regional language, buzzer games in which LEDs would glow based on right and wrong answer. Some of them incorporated user input from switches.

Along with this we conducted a controlled experiment, where students exposed to Scratch-Arduino were tested against students who were not exposed to the workshop. Both the sets had the same amount of diversity. These tests measured the logic and creative abilities of the students.

At the end of the series of experiments we collected formal and informal feedback about their experience.

In Table I. we compare the performance of the experimental and control group in the diagnostic tests. Table II shows performance based on gender.

Fig.1 illustrates educational board wise performance. Fig. 2 depicts concept based improvement.

ANALYSIS & DISCUSSION

Our analysis revealed a great degree of sustained involvement and interest among the students of Scratch-Arduino. Most of the students required very little tutoring, thus the learning process was self motivated. However, with reference to microcontroller coding, the students required constant guidance, as most of them were new to the concept of interfacing. However since it was a project development based tutorial, students overcame their initial hesitation and came up with interesting ideas for buzzer and circuit games.

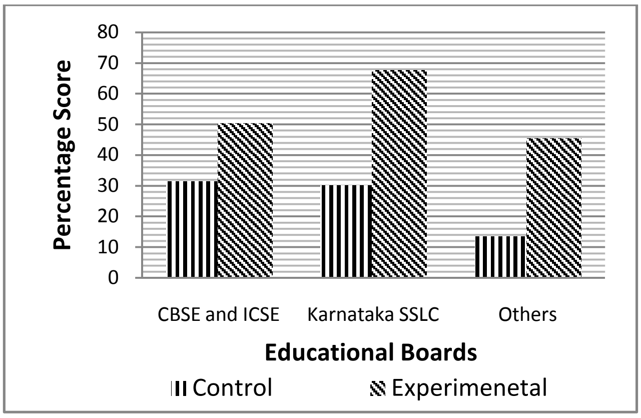

Fig. 1. Educational background wise performance of students.

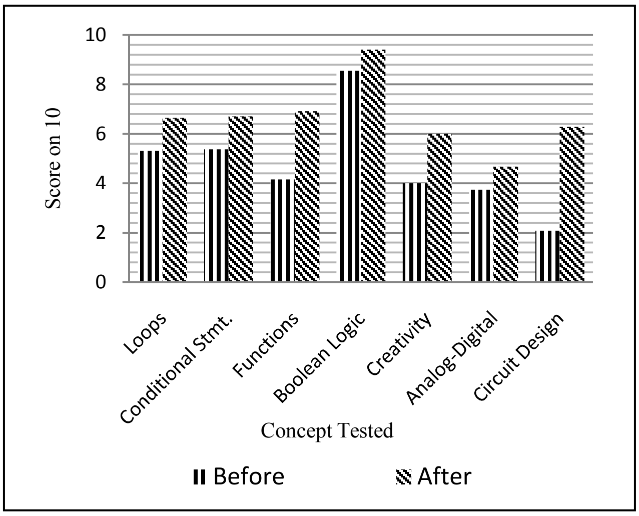

Fig. 2 Graph depicting performance of the students on selected programming and hardware concepts.

The creative energies were revealed in the intricate manner in which they designed the user interface and characters. After the experimental session, 97% of the students asked for the Scratch-Arduino software so that they could use it at home.

Whether given a problem statement or asked to come up with one, it was very evident that the students were enthusiastic in collaborating with one another. Based on feedback we found out, that the opportunity to transform their imagination into animation on the screen or tangible games using interfacing, really excited them.

However, results indicated that interfacing as a concept was not as successful with the children of the age group 15-16 as it was with 18 year olds. The reason primarily being that, students were not introduced to basic electrical and physics concepts.

All the more, since the workshop also dealt in developing simple educational applications students found it interesting to apply their classroom knowledge in developing fun applications and games.

We also found that a project development approach to teaching programming and interfacing concepts was a very rewarding one. It is evident from the experimental results which indicate a sharp rise of concept understanding and internalization, reinforcing previous theoretical learning and encouraging curiosity and inquisitiveness.

CONCLUSION

From our empirical results and analysis we have come to a conclusion that teaching Scratch at a high school level will be an asset to the learning environment. However interfacing should be limited at the pre-university level at grade 12, where students will have stronger theoretical concepts of basic electronics and mathematics. But an introduction of such hardware interfacing projects will surely add value to the existing curriculum.

Also many educational projects can be used by the school in educating junior classes. Projects such as animated stories, mathematic games and interactive puzzles can be developed by senior high school students and deployed in lower classes. Hardware projects can give rise to many in-house products that can be used by kindergarten and primary classes. Interactive hardware and circuits can be interfaced with Scratch-Arduino to create learning aids.

Also, the wealth of online literature and help available is sufficient to sustain a healthy curriculum.

But on a wider perspective the task of transforming the Indian high school student from a consumer of educational content and technology into a creator of one, is challenging and will surely redefine the nature of the Indian high school education system.

ACKNOWLEDGMENT

We would like to thank the NITK IEEE Student Branch for their help in conducting the study.

REFERENCES

[1] Begel,A (1996). LogoBlocks: A Graphical Programming Language for Interacting with the World. Unpublished Advance Undergraduate Project Report, MIT Media Lab

[2] Bobbie, P.O.; Uboh, J.; Davis, B.; , "A project in embedded systems design and development: a partnership with area high school scholars," Frontiers in Education, 2004. FIE 2004. 34th Annual , vol., no., pp. F4D- 14-17 Vol. 2, 20-23 Oct. 2004

[3] Conrad, J.M.; Andrews, D.L.; Butler, D.; Casady, W.; Coleman, M.; Gordon, M.; , "Introduction to engineering concepts for high school teachers and students," Frontiers in Education Conference, 1993. Twenty-Third Annual Conference. 'Engineering Education: Renewing America's Technology', Proceedings. , vol., no., pp.688-693, 6-9 Nov 1993

[4] Rigby, K.; Harrell, D.; , "Issues in developing a high school pre-engineering program," Frontiers in Education, 2005. FIE '05. Proceedings 35th Annual Conference , vol., no., pp.S2F-27, 19-22 Oct. 2005.

[5] Groover, T.R.; Kabara, J.F.; , "Work in progress - The design and implementation of a pre-college computer science curriculum for underrepresented high school students," Frontiers In Education Conference - 2007. FIE '07. 37th Annual , vol., no., pp.T3A-22-T3A-23, 10-13 Oct. 2007

[6] Levy, D.; , "Introducing computer science to educationally disadvantaged high school students - the Israeli experience," Human Centric Computing Languages and Environments, 2003. Proceedings. 2003 IEEE Symposium on , vol., no., pp. 269- 270, 28-31 Oct. 2003

[7] Goode, J., Estrella, R., & Margolis, J. (2006). Lost in translation: Gender and high school computer science. In J.M. Cohoon & W. Aspray (Eds.) *Women in IT: Reasons on the Underrepresentation* (pp. 89-114). Cambridge, MA: The MIT Press.

[8] Guzdial, M. (2004). Programming environments for novices In S. Fincher and M. Petre (Eds.), *Computer Science Education Research* (pp. 127-154). Lisse, The Netherlands:

[9] Kelleher, C. & Pausch, R. (2005). Lowering the barriers to programming: a taxonomy of programming environments and languages for novice programmers. ACM Computing

[10] Werner, L. L., Campe, S., and Denner, J. (2005). Middle school girls + games programming = information technology fluency. Proceedings of the 6th Conference on Information Technology Education SIGITE '05 (pp. 301-305). New York, NY: ACM Press.

[11] CBSE Syllabus, http://www.cbse.nic.in/currisyllabus.htm

[12] ICSE Syllabus, http://www.cisce.org

[13] John h. Maloney, et al, Programming by Choice: Urban Youth Learning Programming with Scratch, SIGCSE'08, Proceedings of 39th Technical Symposium on Computer Science Education, ACM

[14] S4A, http://www. Seaside.citilab.eu/scratch/arduino

[16] Andrés Monroy-Hernández, Mitchel Resnick, MIT Media Labs, Magazine Interactions, Pencils Before Pixels: A prima in hand generated sketching, April 2008, ACM