

VISUAL PROGRAMMING: A PROGRAMMING TOOL FOR INCREASING MATHEMATICS ACHIEVEMENT

By

CHERYL SWANIER *

CHERYL D. SEALS **

ELODIE BILLIONNIERE ***

* Associate Professor, Mathematics and Computer Science Department, Fort Valley State University

** Associate Professor, Computer Science and Software Engineering Department, Auburn University

*** Doctoral Student, Computer Science and Engineering Department, Arizona State University

ABSTRACT

This paper aims to address the need of increasing student achievement in mathematics using a visual programming language such as Scratch. This visual programming language facilitates creating an environment where students in K-12 education can develop mathematical simulations while learning a visual programming language at the same time. Furthermore, the study of visual programming tools as a means to increase student achievement in mathematics could possibly generate interests within the computer-supported collaborative learning community. According to Jerome Bruner in Children Learn By Doing, "knowing how something is put together is worth a thousand facts about it. It permits you to go beyond it" (Bruner, 1984, p.183).

Keywords : Achievement, Creativity, End User Programming, Mathematics Education, K-12, Learning, Scratch, Squeak, Visual Programming

INTRODUCTION

Across the nation, math scores lag (Lewin, 2006). Students are performing lower than ever. Lewin reports, for "the second time in a generation, education officials are rethinking the teaching of math in American schools" (p.1). It is imperative that higher education, industry, and K-12 collaborate to ascertain a viable solution to the problem of low achievement in mathematics education. One way to get students interested in math is to teach them how to creatively code mathematical programs using a visual programming language. Using a visual programming language would allow the students to show their creativity and possibly produce software that would increase mathematics achievement. With the students' involvement, they are able to not only create games and other forms of media; they are also learning mathematical concepts. Not only will the usage of these applications benefit the designers of the software, but also those students that will follow in their footsteps.

This paper gives a brief overview of the visual programming language Scratch and describes a prototype of an implementation of basic math tutorial using Scratch. The prototype demonstrates a visual

programming language as a tool to create mathematical tutorials. Additionally, this paper provides a modicum of information as it relates to the educational value of visual programming to mathematical achievement. Peppler and Kafai (2007) indicate "game players program their own games and learn about software and interface design. Some efforts have integrated the learning of subject matter, such as mathematics and science" (Jenkins 2006, p.1). Jenkins articulates "three issues that policymakers and educators face as they attempt to bridge the gap between those that contribute and those that do not: the participation gap, the transparency problem, and the ethics challenge. These three issues encompass the need to ensure that every young person has access to the skills and experience needed to become a full participant, can articulate their understanding of how media shapes perception, and is knowledgeable of emerging ethical standards that shape their practices as media makers and participants in online communities" (p.1). The work presented in this paper belongs to a broader spectrum of end user programming and learning that could possibly facilitate achievement in mathematics.

Background

Kay (2007) states “globally networked, easy-to-use computers can enhance learning, but only within an educational environment that encourages students to question facts and seek challenges,” (p.3). “A visual programming language (VPL) is any programming language that lets users specify programs by manipulating program elements graphically rather than by specifying them textually. A VPL allows programming with visual expressions, spatial arrangements of text and graphic symbols. Most VPLs are based on the idea of “boxes and arrows,” that is, boxes or circles or bubbles, treated as screen objects, connected by arrows, lines or arcs” (Wikipedia, p.1). According to the Microsoft Robotics Studio (2008), although visual programming was initially developed for novice programmers who had some knowledge about variables and logic, VPL is also for the more experienced programmers. For the purpose of this paper, Scratch, an open-source visual programming language, was chosen for our study as its environment supports user creativity and software development.

Scratch

Scratch is a “new [emerging] programming language that makes it easy to create interactive stories, games, animations, [music, and art]” (MIT Media Lab, 2007, p.1). The software does not require prior knowledge in programming. While observing this programming language, Scratch appeared to be user friendly and encourages computational thinking and creativity when developing learning games.

“Scratch is written in Squeak, an open-source implementation of the smalltalk language” (OLPC, p.1). Squeak is a “media authoring tool” developed by a community of people from Massachusetts Institute of Technology (MIT). It can be used to create your own media or share and play with others” (Kay, 2007, p.1). It allows novice programmers to create tools for learning or play in an environment that is non-threatening and fairly intuitive. Kay believes that “education is a primary focus for many Squeakers who are doing cutting edge research on how computers can be used to enhance and amplify

learning” (p.1).

Scratch consists of a programming language made up of different blocks and an easy to learn graphical development environment that includes a paint application for creating graphics and built-in sound editing capabilities. Scratch blocks are snapped together to create well-structured programs composed of color-coded tags describing motion, looks, sound, pen, control, sensing, operators, and variables. Scratch blocks resemble puzzle pieces in the way that they snap together as shown in Figure 1. Thus, Scratch enforces proper programming syntax and ensures that novice programmers learn the proper way to assemble and formulate programming logic. Scratch projects consist of sprites, costumes, blocks, and scripts. Even though Scratch is a gentle approach to programming, it does not have the advanced functionality of some programming environments like Squeak and Alice 3D. This was observed when trying to develop a math tutorial for an Algebra I class. Despite the lack of advanced functionality, Ford (2008) reports “Scratch provides robust support for performing mathematical calculations” (p.159).

Some of the basic functionality of the tool Scratch include:

adding, subtracting, multiplying, and dividing;
generating random numbers in any range;
performing various types of numeric comparisons;
performing a number of built-in operations. As such, a simple math tutorial was implemented in Scratch to assist students in learning basic math facts.

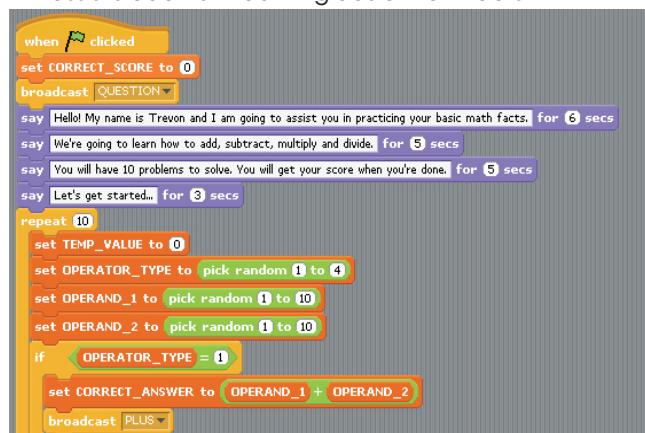


Figure 1. Scratch blocks

The Scratch environment is separated into several task panes as shown in Figure 2. The middle section contains the scripts, i.e. the instructions controlling the actions. The first section consists of blocks to create the scripts. The Scratch blocks are organized as the following: Motion, Looks, Sound, Pen, Control, Sensing, Operators, and Variables. The last pane contains the area where the results are populated as well as the objects, sprites, costumes, and stages. Each sprite consists of three distinct components: i) scripts; ii) costumes; and iii) sounds.

Basic Math Facts Tutorial

Creating tools that facilitate student achievement in math is critical. It is imperative that the authors find ways to generate interest in mathematics and make it more interesting. One way to do that is to develop software that is culturally relevant such that the participants are enthusiastic about learning, and engaged in the learning; thus learning can be achieved.

Scratch implementation assists individual in learning their basic math facts such as multiplication, division, addition, and subtraction as shown in Figure 3. Participants are given ten randomly chosen basic math facts problems. The tutorial is culturally relevant as the tutor is represented according to participants' preferences in terms of race, gender, and clothing. For the purpose of this study, the authors use Trevor, an African-American male as the agent that guides the participant through the tutorial. This feature enables participants to relate with their tutor and therefore allows a safe learning environment. The system

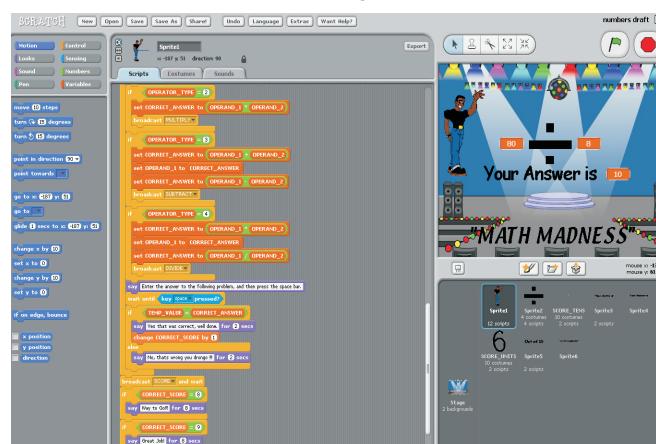


Figure 2. Scratch Environment

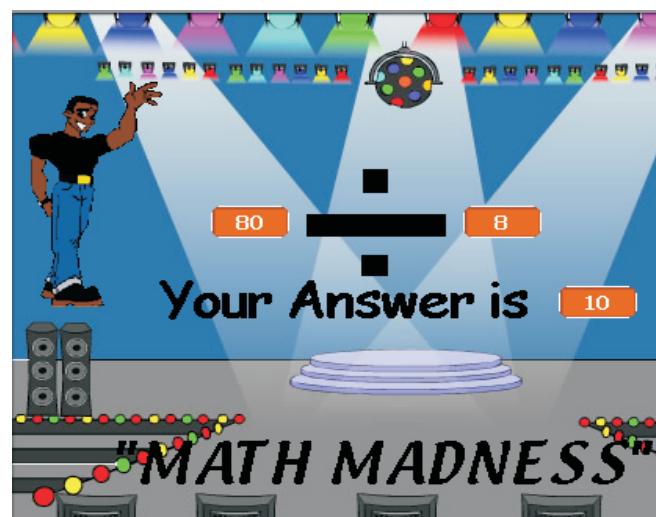


Figure 3. Basic Math Facts Tutorial

immediately informs the participant whether the answer is correct. If not, the system generates a message indicating the answer is incorrect. Furthermore, another math problem is generated when participants press the spacebar. Upon completion of the short tutorial, the score is generated and a message indicating the performance level and instructions for resetting the tutorial are displayed on the screen.

The layout of the basic math facts tutorial consists of objects called sprites, costumes, a stage, and several scripts shown in Figure 4. There are eight sprites and one stage. Sprite1 as illustrated in Figure 3 depicts the agent Trevor. To start a game, the green flag sign must be clicked and to end a game, the red stop sign must be clicked (Figure 2).

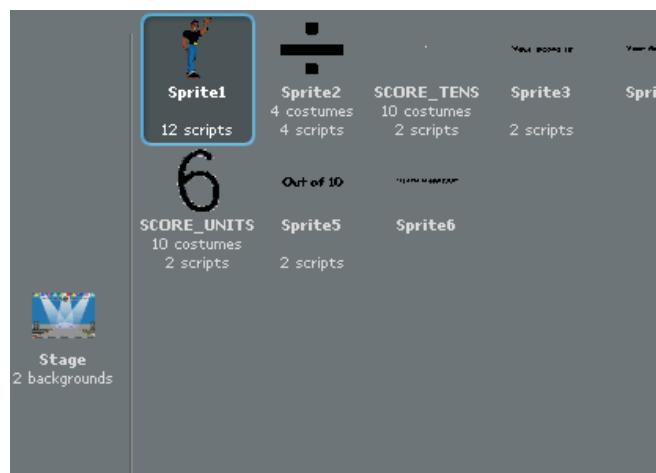


Figure 4. Objects, Sprites, Costumes, Stages, and Scripts

Conclusion

According to Klopfer (2008), et al., "As technologies become more and more advanced, we have empowered students to magically create things themselves on computers. They can manipulate media, explore virtual worlds, and communicate across the globe almost effortlessly. The day when students need to program computers to do interesting things is far behind us. So what is the role of computer programming for students at this point in time? What are the advantages to kids of understanding and controlling the science behind the magic?" (P.1)

As such, it is imperative that government, educational institutions, and industry convene to provide a solution to this nation wide problem regarding mathematics achievement. This community of stakeholders can involve the students in their learning and teach them to program in an environment that is less threatening while promoting achievement in mathematics. This basic math tutorial illustrated in this paper does not involved any advanced mathematical problem solving; however, it demonstrates that visual programming can be used as a fun and easy tool to possibly increase mathematics achievement.

In terms of future work, more research is required to implement advanced topics in mathematic. Visual programming via Scratch can "provide a rich context for learning programming, how to collaborate with others, becoming a member of an affinity group, developing sustained engagement," and a community of learners (Peppler & Kafai, p.7).

References

- [1]. Bruner, J. S. (1983). The New Curriculum: IN THE END. In Search of Mind: *Essays in Autobiography* (pp.177-200). New York: HarperCollins.
- [2]. Ford, J. L. (2008). Scratch™ Programming for Teens. Boston, MA: Course Technology PTR.
- [3]. Jenkins, H., Clinton, K., Purushotma, R., Robison, A., & Weigel, M. (2006). *Confronting the challenges of participation culture: Media education for the 21st century*. Chicago, IL: The John D. and Catherine T. MacArthur Foundation.
- [5]. Kay, A. (2007). Computer, networks and education. Retrieved July 29, 2009, from Squeakland Home of Squeak Etoys Web site: http://www.squeakland.org/school/HTML/sci_amer_article/sci_amer_01.html.
- [6]. Klopfer, E., Resnick, M., Malone, J. Silverman, B., diSessa, A., Gegel, A., & Hancock, C. (2008). Programming Revisited The Educational Value of Computer Programming. Retrieved June 23, 2009, from Microsoft Research Web site: <http://research.Microsoft.Com/~abegel/papers/icls04.pdf>.
- [7]. Lewin, T. (2006). As Math Scores Lag, a New Push for the Basics. *The New York Times*, p. A22.
- [8]. Microsoft Robotics Developer Center. (n.d.). VPL Introduction. Retrieved on July 23, 2009, from Microsoft Robotics Developer Center site: <http://msdn.microsoft.Com/en-us/library/bb483088.aspx>.
- [9]. MIT Media Lab. (2007). Scratch reference guide. Retrieved on July 29, 2009, from Scratch Web site: http://info.scratch.mit.edu/Support/Reference_Guide_1.4.
- [10]. OLPC News.(2007). Learning Squeak from Scratch. Retrieved on June 20, 2009, from OLPC News Web site: http://www.olpcnews.com/software/applications/learning_squeak_scratch.html.
- [11]. Peppler, K., & Kafai, Y. B. (2007). What Videogame Making can teach us about literacy and learning: Alternative pathways into participatory culture. In *Proceedings of DiGRA*, p. 369-376.
- [12]. Scratch. (N.d.). Help Screens for Scratch. Retrieved on July 29, 2009, from Scratch Web site: http://info.scratch.mit.edu/Support/Help_Screens.
- [13]. Wikipedia. (n.d.). Visual Programming language. Retrieved on July 23, 2009, from Wikipedia Web site: http://en.wikipedia.org/wiki/Visual_programming_language

ABOUT THE AUTHORS

Cheryl A. Swanier, Ed.D., is an Associate Professor of Computer Science at Fort Valley State University. She is also a Ph.D. student in the Computer Science and Software Engineering Department at Auburn University. She conducts research in Human Computer Interaction, Learning Technologies, Visual Programming, and STEM Education.



Cheryl D. Seals, Ph. D. is an Associate Professor in the Computer Science & Software Engineering Department at Auburn University. She conducts research in Human Computer Interaction with an emphasis on Visual Programming of Educational Simulations with End User Programming, Intelligent Agent, Usability Evaluations, Computer Supported Collaborative Work, and Minimalism. She is additionally involved in software engineering projects.



Elodie V. Billionniere, M.Ed is a Ph.D. Student in the Computer Science and Engineering Department at Arizona State University. She conducts research in Human Centered Interaction, Learning Technologies, Service Computing, and STEM Education.

