Students Learn Programming Faster Through Robotic Simulation

By Allison Liu, Jeff Newsom, Chris Schunn, and Robin Shoop

ROBOTIC systems are everywhere—we just don't call them robots. We call them cell phones, bank machines, cars, microwaves, the Internet. Robotic technologies are ubiquitous and are making it easier for people to drive cars, access money, find restaurants via their cell phone, or cook their food using a microwave. Robotic systems are possible because of computer science and sensing.

Schools everywhere are using robotics education to engage kids in applied science, technology, engineering, and mathematics (STEM) activities, but teaching programming can be challenging due to lack of resources. This article reports on using Robot Virtual Worlds (RVW) and curriculum available on the Internet to teach robot programming.

It also reports the results of a research study that compared the test scores of students learning to program LEGO and VEX robots using virtual robots versus physical robots. The study was conducted by Carnegie Mellon University's Robotics Academy and the University of Pittsburgh's Learning Research and Development Center, and the results proved to be positive from both a teaching and learning perspective. This study builds on numerous studies involving teaching novice programming using robots (Roberts et al., 2011).

A LEGO Mining Bot

The Brains of the System

The brains of robotic systems are driven by computer science (CS) and the computational thinking that CS education produces. CS will play a key role in nearly all future innovation, including advancements across all STEM fields, but the United States has entered a significant national decline in the number of college graduates with basic and advanced CS-STEM degrees. This downward trend is particularly pronounced in CS (CRA, 2008).

Reports indicate that the U.S. increasingly does not meet its own demand for CS professionals, and as a consequence has to rely on foreign-born talent (National Science Board, 2010; ScienceDaily, 2007). Since many secondary schools in the U.S. offer few if any computer science courses (Gal-Ezer & Stephenson, 2009; Robelen, 2010), this trend will likely continue. For example, a 2009 survey showed a drop in the percentage of secondary schools providing CS courses from 78% in 2005 to 65% in 2009 (CSTA, 2009).

Scene from the Operation Reset programming game

In part, the focus on high stakes testing topics coupled with increased emphasis on Advance Placement courses has squeezed out coursework in many areas, including computer science. In addition, the sophistication of systems that students use on a regular basis is so far

Allison Liu is a graduate student, the University of Pittsburgh; Jeff Newsom is a technology teacher, Penn Trafford High School, Pittsburgh, PA; Chris Schunn is a cognitive scientist, the University of Pittsburgh; and Robin Shoop is a former technology teacher and currently the director of Carnegie Mellon Robotics Academy.

FTC Ring It Up competition simulation

beyond the level of sophistication offered in many computer science classes that the motivation to learn programming has been reduced. *Early programming environments more similar in sophistication to current gaming platforms could be both transformational for motivation, and well-aligned with state and national learning frameworks.* Students would want to play these games and formal and informal education systems would want to offer them.

Robotics is an activity that excites students to consider CS-STEM careers (Ante, 2007), but schools typically don't have enough robots so that each student can have their own robot to work with to learn programming and it is difficult to assign robotics-related homework. Over the last two years the Robotics Academy, with support from Robomatter Inc., has developed a combination of game-like Robot Virtual Worlds (RVW) (www.robotvirtualworlds.com) and simulations of traditional classroom programming activities that allow students to use the same code on a physical robot that they use in the RVW environment.

For the study, one class used physical VEX robots, the other used virtual VEX robots.

The Study: Physical vs. Virtual Programming

Public high school students from two elective programming classes participated in the study, and the same teacher taught both classes. One class completed a ROBOTC programming course using physical VEX robots (the Physical class), while the other class completed a ROBOTC programming course using virtual VEX robots (the Virtual class). Thirteen students were in the Physical class, and 17 were in the Virtual class. Both classes consisted primarily of freshmen and sophomores with little or no prior programming experience.

Both classes completed the same pre-test and post-test online. The pre-test and post-test contained the same 50 items, and both classes completed the pre-test around the same date. Eleven students in the Physical class and 15 students in the Virtual class completed both the pre-test and post-test, and were included in the analyses. Analyses investigated whether there were learning differences between students who interacted with physical robots versus students who interacted with virtual robots.

Analyses

Three analyses were performed on the data. First, students' total scores on the pre-test were compared with their total scores on the post-test. To control for students' differing pre-test scores, an ANCOVA (analysis of covariance) was run using condition (Physical or Virtual) as the independent variable, post-test score as the dependent variable, and pre-test score as the covariate.

Second, we examined whether learning differed across topic subcategories. Four sub-categories were defined, into which all problems on the pre-test and post-test could be placed:

- Algorithmic thinking: Problems that involved thinking through the process of the programming problem (e.g., planning the program, using pseudocode, predicting what a program would do) or more abstract concepts of programming. Example: "Given the program above, the robot will do _____."
- General programming: Problems that involved syntax or concepts that are applicable to multiple programming languages. Example: "If the condition of an If statement is true, then all of the code inside of its curly braces will run: True/False."
- ROBOTC syntax: Problems that involved ROBOTC syntax or the ROBOTC program (e.g., how to use menus in the ROBOTC application). Example: "To make the robot stop, you set its motor values equal to _____."
- Physical robot: Problems that involved the physical VEX robot's functioning. Example: "The VEX Ultrasonic Rangefinder (sonar sensor) measures distance using _____."

The number of problems in each

Table 1—Number of Sub-Category Problems and Sub-Category Alphas

Problem Sub-Category	Number of Problems	α
Algorithmic Thinking	4	0.54
General Programming	13	0.56
ROBOTC Syntax	37	0.84
Physical Robot	19	0.81

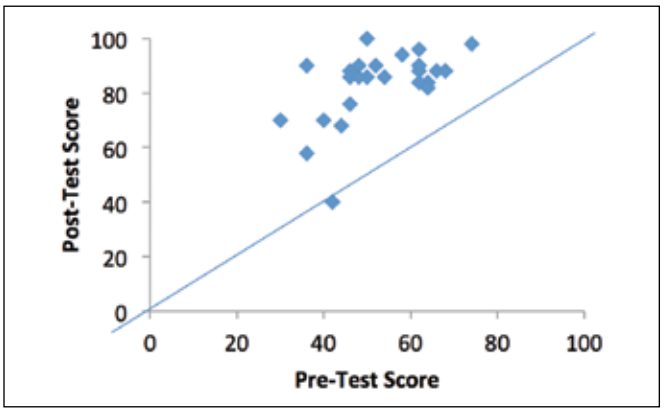

Fig. 1—Pre-test score vs. post-test score. Points above the line improved on the post-test compared with pre-test.

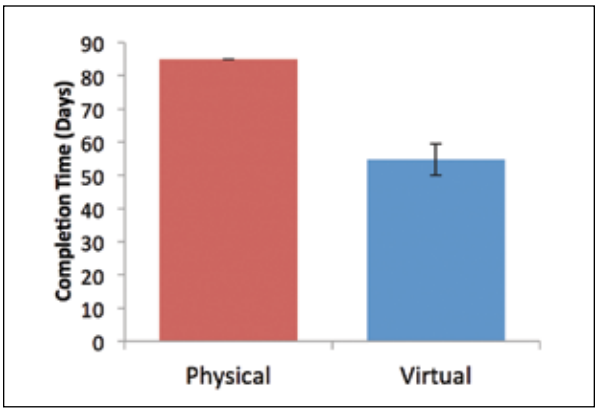

Fig. 2—Days taken to complete the course, separated by condition

subcategory and the Cronbach's alpha (α; calculated using both conditions' post-test scores) for each category are shown in Table 1. Note that the sub-categories were not mutually exclusive; that is, the same problem could fit into multiple subcategories.

Due to the uneven number of problems in each category, we used the proportion of correct answers within each category as a measure of accuracy. An ANCOVA was performed to control for pre-test scores, using condition as the independent variable, post-test score as the dependent variable, and pre-test score as the covariate.

Third, we looked at the number of days between participants' pre-test attempt and post-test attempt. This was used as a measure of the time needed to complete the course, to see whether one condition required less time than the other to learn the same amount of information. A one-way ANOVA (analysis of variance) was performed, using condition as the independent variable and number of days as the dependent variable.

Results

Overall Scores

No differences were found between the Physical and Virtual classes in their overall post-test scores (with pre-test score controlled) [$F(2, 23)=0.19$, $p=0.67$] when pre-test scores were controlled. Both classes began with similar pre-test scores and ended with similar post-test scores. Figure 1 shows that overall learning gain did not differ by pre-test score, as almost all participants improved regardless of their pre-test score. The average pre-test and post-test scores for both classes can be found in Table 2.

Table 2—Averages (and Standard Deviations) of Pre-Test Score, Post-Test Score, and Time Taken, Separated By Condition

Condition	Pre-Test Average	Post-Test Average	Average Time Taken
Physical	50.2 (SD=11.2)	82 (SD=10.6)	85.0 (SD=0.0)
Virtual	55.9 (SD=11.5)	84.5 (SD=14.6)	54.7 (SD=18.2)

Table 3—Average Proportion Correct (and Standard Deviations) of Each Sub-Category, Separated by Condition

Condition	Algorithmic Thinking Pre-Test	Algorithmic Thinking Post-Test	General Programming Pre-Test	General Programming Post-Test	ROBOTC Syntax Pre-Test	ROBOTC Syntax Post-Test	Physical Robot Pre-Test	Physical Robot Post-Test
Physical	0.67 (0.27)	0.88 (0.20)	0.54 (0.12)	0.79 (0.11)	0.49 (0.10)	0.81 (0.11)	0.42 (0.12)	0.82 (0.16)
Virtual	0.80 (0.29)	0.95 (0.14)	0.59 (0.12)	0.86 (0.15)	0.51 (0.11)	0.83 (0.16)	0.51 (0.14)	0.86 (0.17)

Sub-Category Scores

With pre-test score controlled, the two classes did not show any learning differences across the four subcategories of algorithmic thinking [$F(2, 23)=0.061$, $p=0.81$], general programming [$F(2, 23)=1.3$, $p=0.27$], ROBOTC syntax [$F(2, 23)=0.079$, $p=0.78$], or physical robots [$F(2, 23)=0.11$, $p=0.74$]. The average pre-test and post-test scores (and standard deviations), measured as proportion correct, for each sub-category for the two classes can be found in Table 3.

Time Taken

The average time taken for both classes to complete the programming course can be seen in Table 2. The Physical class took significantly more time than the Virtual class [$F(1, 24)=30.3$, $p<0.001$]. All students in the Physical class completed the course in the same amount of time, as working with Physical robots did not afford them the same freedom of students in the Virtual class of working independently through the course.

Thus, due to heterogeneity of variance, we also ran a t-test with equal variances not assumed, which was significant [t(24)=6.5, p<0.001, d=2.2]. Overall, the Physical class took an extra 30.3 days (approximately one month) to complete the course than the Virtual class (see Fig. 2).

Summary

Both the Physical class and the Virtual class showed equal learning gains, as their overall post-test scores were the same (controlling for pre-test scores). The type of learning did not differ between the two classes either, as evidenced by the equal learning gains seen across all four sub-categories. *However, the Virtual class did show a time reduction benefit, as they completed the course about a month earlier than the Physical class, with no effect on their overall learning.* This suggests that working with the virtual robots allowed students to learn more efficiently.

The teacher's informal observations support this conclusion. The teacher noted that students in the Physical class had to deal with the additional mechanical issues that came from working with a physical robot. Consequently, the teacher spent much more of his time in the Physical class helping students with communication problems between the robot and computer. In the Virtual class, the teacher and students were able to focus their time on programming instead of the mechanical side.

Comparison with Other Virtual Classes

To confirm that the learning gains and time savings seen in the Virtual class were consistent, we also looked at two additional classes that completed the same programming course with virtual VEX robots. A graph comparing the three courses' pre-test scores, post-test scores, and days to complete the course can be seen in Fig. 3. One class (Class 2) had 23 students who completed both the pre-test and post-test, and the other class (Class 3) had 13 students who completed both the pre-test and post-test.

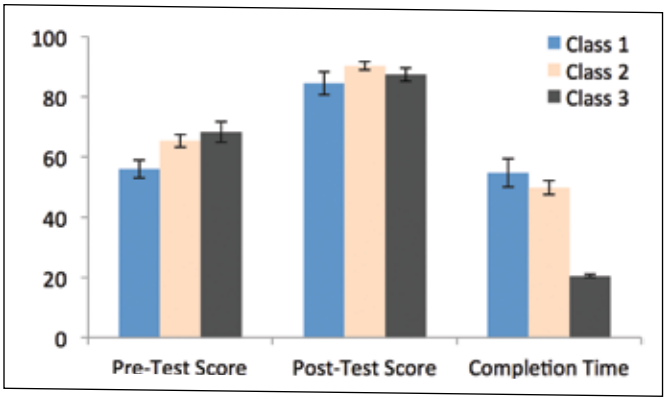

Fig. 3—Average pre-test score, post-test score, and days to complete the course, separated by class

We ran a paired t-test for each class to compare total pre-test scores to total post-test scores. Average pre-test score and post-test score for both classes can be seen in Table 4. Both Class 2 [t(22)=-14.4, p<0.001] and Class 3 [t(12)=-7.1, p<0.001] significantly improved their scores on the post-test, suggesting that the course led to comparable programming learning gains across the three Virtual robot classes.

To compare the time taken by each class to complete the course, we ran a one-way ANOVA, using Class (Class 1 being the Virtual class in the study above, Class 2 being the class of 23 students, and Class 3 being the class of 13 students) as the independent variable and number of days as the dependent variable. Average number of days taken to complete the course can be seen in Table 4. The ANOVA was significant [F(2, 48)=31.6, p<0.001], and a post-hoc contrast showed that Class 3 took significantly less time than the other two classes. Again, due to heterogeneity of variance between Class 3 and the other two classes, we also ran an independent t-test comparing the amount of time taken by Classes 1 and 2 (combined) and Class 3, which was significant [t(49)=13.2, p<0.001]. This suggests that Virtual robots in all three Virtual classes allowed students to complete the course in significantly less time than the Physical class in the study above.

This article will be continued in the next issue of *Tech Directions*. To learn more about RVW go to www.robotc.net.

Table 4—Average Pre-Test Score, Post-Test Score, and Days Taken to Complete the Course (and Their Standard Deviations), Separated by Class

Class	Average Pre-Test Score	Average Post-Test Score	Average Days Taken
Class 2 (N=23)	65.3 (SD=10.5)	90.3 (SD=6.7)	49.8 (SD=12.3)
Class 3 (N=13)	68.3 (SD=12.3)	87.4 (SD=8.1)	20.5 (SD=1.7)

References

Ante, S. E. (2007, April 13). Building robots builds scientists: FIRST's robotics competition helps inspire middle and high school students to pursue careers in math and science. *Bloomberg Businessweek.* http://www.businessweek.com/technology/content/apr2007/tc20070413_582820.htm.

CSTA (Computer Science Teachers Association). (2009). CSTA National Secondary Computer Science Survey: Comparison of 2005, 2007, and 2009 survey results. http://csta.acm.org/Research/sub/CSTAResearch.html.

Gal-Ezer, J., & Stephenson, C. (2009, Spring). The current state of computer science in U.S. high schools: A report from two national surveys. *Journal for Computing Teachers.*

Major, L., Kyriacou T., & Brereton, P. (2011). Systematic literature review: Teaching novices programming using robots. 15th Annual Conference on Evaluation & Assessment in Software Engineering, pp. 27–28.

National Science Board. (2010). Preparing the next generation of STEM innovators: Identifying and developing our nation's human capital. Publication NSB-10-33 of the National Science Foundation. http://www.nsf.gov/nsb/stem/innovators.jsp

ScienceDaily. (2007, February 22). Computer science trouble lies in education, not jobs, professor says. *ScienceDaily.*