Measuring the Effectiveness of Robots in Teaching Computer Science

Barry Fagin
Department of Computer Science
US Air Force Academy
USAFA, CO 80840
barry.fagin@usafa.af.mil

Laurence Merkle
Department of Computer Science and Software Engineering
Rose-Hulman Institute of Technology
Terre Haute, IN 47803-3999
merkle@Rose-Hulman.edu

Abstract

We report the results of a year-long experiment in the use of robots to teach computer science. Our data set compares results from over 800 students on identical tests from both robotics and non-robotics based laboratory sessions. We also examine the effectiveness of robots in encouraging students to select computer science or computer engineering as a field of study.

Our results are negative: test scores were lower in the robotics sections than in the non-robotics ones, nor did the use of robots have any measurable effect on students choice of discipline. We believe the most significant factor that accounts for this is the lack of a simulator for our robotics programming system. Students in robotics sections must run and debug their programs on robots during assigned lab times, and are therefore deprived of both reflective time and the rapid compile-run-debug cycle outside of class that is an important part of the learning process. We discuss this and other issues, and suggest directions for future work.

1 Introduction

Educators have thought about robots in the classroom for as long as they have thought about robots: their potential as teaching tools and as motivators has long been recognized. For most of our lifetime, however, economic constraints prohibited extensive deployment of robots in all but the most rarefied environments. Initial attempts to capitalize on robots as teaching tools had to rely on software models [14].

Within the past few years, however, improvements in performance and cost have changed the picture dramatically. Robotic systems, both customized and mass-produced, are now sufficiently affordable, powerful, and reliable to be deployed in the college and even the high school classroom. Interest in the use of robots as educational tools has exploded within the past few years [1, 8, 10, 11, 12, 14]. We believe this trend will only increase, as robots continue to get better and cheaper.

But while educators interested in robotics have developed important prototype systems and reported on their deployment in the classroom, very little is known about their effect on learning. The computer science education community has organized panels and workshops on the subject to facilitate the exchange of ideas [1,9], but quantitative studies that assess how robots affect learning are missing from the literature.

We report one such study here. We were among the first institutions to deploy robots in the classroom, so we have considerable interest in this question. The existence of a core course at our institution that all students must take provides a large sample population from which to draw conclusions, and the systematic use of databases for student information and test scores provide us with solid, reliable data for analysis. In the sections that follow, we present the basic parameters of the study, a statistical analysis of the data, and comment on subjective feedback measures. We then present our conclusions, and discuss future work.

2 Experimental Parameters

Our study analyzes data from the 2000-2001 academic year offerings of our core computing course, required for all our students and normally taken in the freshman year. This course was taught to 938 students in 48 sections of 15-20 students each. Nine of these sections were designated as "robotics" sections, where we provided laboratory instruction using Lego Mindstorms ® robots and the Ada/Mindstorms programming environment [4]. We tracked student performance on all exams, as well as their rank in the course after grades were assigned. Additionally, our students declare a major no later than the middle of their sophomore year, so we now have data on the effectiveness of robots in encouraging the selection of

computer science or computer engineering as a field of study.

2.1 Programming Environment

Our classes used Lego Mindstorms robotics kits and the Ada/Mindstorms 2.0 programming environment., available at http://www.usafa.af.mil/dfcs/adamindstorms.htm. A screen shot of the programming environment is shown in Figure 1. It has an easy to use GUI, and runs on any Windows PC.

The coding flow of the programming environment is shown in Figure 2. Programs are written in an Ada subset plus an API of Mindstorms-specific function calls, and compiled with the Ada/Mindstorms compiler. This compiler is a fully validated Ada compiler, with additional logic to check that the program uses only those constructs supported by the Ada/Mindstorms subset. These constructs are necessarily much smaller than the full Ada language, due to both Ada's considerable expressive power and the hardware limitations of the Lego Mindstorms platform.

After the user's Ada program has been validated, it is translated into Dave Baum's NQC language [8], a C-like language for Mindstorms programming. NQC code is assembled into binary bytecodes and downloaded into the Lego Mindstorms RCX module, the central component of the Mindstorms system.

For more information on Ada/Mindstorms, see [3, 4, 5, 6].

2.1 Testing and Assessment Methods

Both the robotics and the non-robotics sections were taught introductory programming using Ada. Our core computing course contains six laboratory exercises, emphasizing the standard topics usually taught in an introductory course.

Students in this course could earn up to 1000 points, not counting extra credit, based on the following scale:

labs 1-5	150	5 labs @ 30 points
lab 6	40	
practica	160	30, 50, and 80 points
midterms	250	2 @ 125 points
final project	100	
final exam	250	
other	50	

Programming practica were in-class programming assignments that had to be completed in a specified time period.

Since the labs, practica, and final project were different for the robotics and non-robotics sections, we did not compare student performance on those exercises. Our analysis is confined to midterm exam scores, final exam scores, and class rank at the end of the semester.

Midterms and the final consisted of three sections: multiple choice, short answer, and programming. For both robotics and non-robotics classes, the multiple choice and short answer portions of the test were the same. For the programming portions, only slight modifications were made in a few instances to ensure that correct answers did not require concepts that robotics students had not been exposed to. For the vast majority of programming questions, we used identical problems.

3. Analysis – Objective Measures

This section describes the statistical analysis of our experimental data, along with the results. The statistical techniques used well known in the literature. We recommend [2] as a useful reference.

3.1 Student Performance

We want to measure the effect of being in a Robotics section on performance. Our analysis considers the effect on both raw exam scores and on residuals after the effect of student GPA is removed via linear regression. This was done to guard against the possibility that unequal distribution of student academic ability might affect the results. The statistical test chosen for analysis is the Kruskal-Wallis *H* Test.

Our results are unequivocally negative. Strictly speaking, the KW test is only used to determine if the scores from the two populations are different. However, *in every case* where a difference was detected, the scores in the Robotics sections were worse. All of these differences remain after attempts to compensate for the correletated effects of GPA.

The relevant sample sizes are presented in Table I. Each semester, approximately 5% of the students are excused from the final exam on the basis of their performance during the semester. Those students are obviously excluded from the analysis of performance on the final exam. They are also excluded from the analysis of overall performance on exams, but they are included in the analysis of performance on the graded reviews.

The results of the analysis for the Fall 2000 and Spring 2001 semesters are summarized in Tables II and III, respectively, while the results of the analysis for the complete academic year are summarized in Table IV. In each case, following the procedure for the Kruskal-Wallis test, student scores were ranked from highest to lowest, the rank values were summed, and the KW test performed to generate an H value. H is then used to calculate a p-value. For tables II-IV:

μ_1 is the mean score for Robotics students

μ_2 is the mean for other students

H is the H-value of the KW test

p is the p-value of the KW test

R_1, R_2 are the residuals from a linear-regression function used to remove the effects of GPA.

In this context the p-value is the probability that it would be an error to conclude that there is an effect on the exam score in question (i.e. the higher the p-value, the less evidence of a difference). p-values less than .005 appear as 0 in the tables. Negative values for residuals indicate scores below what a linear function based on GPA would have predicted.

For the Fall 2000 semester, raw scores are available for both graded reviews, both the multiple choice and short answer portions of the final exam, and final course standing. As shown by the p-values in Table II, the raw scores of the Robotics students are significantly different from those of the other students on both portions and the total of the final exam, as well as the overall course standing.

For the Spring 2001 semester, raw scores are available for both the multiple choice and the short answer portions of both graded reviews and the final exam, as well as for final course standing. There are significant differences between the Robotics scores and the non-Robotics scores on the short answer portions and the overall scores of all three exams as well as the final course standing, but not on the multiple choice portions of the exams. With the exception of the overall score on the final exam, the same differences are present in the GPA-adjusted scores.

The raw scores from the Fall 2000 semester are not directly comparable to those from the Spring 2001 semester. However, the GPA-adjusted scores from the two semesters are directly comparable. Significant differences exist in all of those scores.

3.2 Field of Study

We also attempted to determine the effect that being in a Robotics section has on a student's likelihood of declaring a major of either Computer Science or Computer Engineering. Our data showed no statistically significant results for either the Fall or the Spring semester. Over the course of the complete academic year, the cadets in the Robotics sections were slightly less likely to declare the Computer Science major. For more details, the reader is referred to [7].

3.3 Analysis – Subjective Measures

In addition to objective measures, we also looked at subjective measures including student self-assessments of their learning, how students rated the course, and so forth. We obtained quantitative subjective measures through course critique surveys, distributed at the end of every semester as part of our institution's standard course evaluation protocol.

While quantitative data were important, we were also interested in qualitative feedback. To this end, we selected a few sections of our course, both with and without robotics, for special "focus group" sessions. Focus group sessions at our institution are led by a trained educational assessment specialist, who asks questions and leads brainstorming sessions to determine student perceptions of their classroom experience. The instructor does not attend, to ensure that students will respond honestly. The session is recorded and transcribed, and then analyzed for common themes to pinpoint both strengths and weaknesses.

The results of our student surveys were consistent with the data of the previous section. On the four most significant survey questions where students were asked to rate course content, the average ratings for the non-robotics sections were higher than those in the robotics classes. The difference ranged from 2.2% for "relevance and usefulness" to 7.6% for "instructor effectiveness".

On the qualitative side, an analysis of the focus group transcripts revealed some important threads. We saw that the perceived advantages of robots in the classroom were also felt by students. Words like "interesting", "fun", "challenging", and "relevant" kept recurring in the discussion.

Unfortunately, brainstorming sessions on the weaknesses of robots revealed just how important the lack of a simulator and the corresponding reduction in reflective problem-solving time was.

Since we (and probably most institutions considering robots) could not afford to purchase kits for every student, and since the inventory control problems are painful to contemplate, we required all robots to remain in the labs. In an environment like ours where student free time is at a minimum this effectively meant that most students could work on their problems only during their assigned lab time. Students were keenly aware of this, and saw it as a big disadvantage.

Our data show the students' most significant concern was their lack of ability to work on their programming assignments in their rooms. Comments like "hard to work on", "time consuming", and similar phrases recurred with disturbing frequency. The more typical complaints you might expect with robots, such as logistical and mechanical issues, were far less significant. This is in some sense encouraging, because time constraint issues can be solved in software, through the use of a simulator. Solving serious mechanical or logistical issues inherent in the robots themselves would be much more difficult.

For a more detailed discussion of the subjective feedback received, see [7].

4 Conclusions and Future Work

The most important goal in introducing robots into the computer science classroom is to improve student learning.

Attempts to assess the effect of robots should look for evidence of improved learning on tests and problems. Even if no such evidence is found, the case for robots might still be compelling if they improve student retention, attract more people to the discipline, and enhance the classroom experience. Clearly these goals were not achieved in our experiment.

Our results showed worse results in the robotics vs the non-robotics sections. At least in our environment, the use of robotics deprives students of the opportunity to work on their code back in their rooms, on their own time, and to practice the write-run-debug feedback loop that appears to be an important part of the learning process. When we first conceived of this project 2 years ago, we wanted to deploy the software in the classroom quickly, and therefore made the conscious decision not to include a simulator in the first releases of Ada/Mindstorms. Our hope was that the learning advantages of robots would outweigh the disadvantages of a restricted feedback loop for programming. Our results do not support this hypothesis.

Instructor experience may also play a part. Student feedback metrics improve with instructor experience. We collectively had several years teaching the "old" version of our computing course, with no more than one semester experience teaching the robotics sections. While we were careful to work through all the labs, issue kits to all robotics instructors, and make sure all exercises were carefully worked through and understood before issued to the students, it is difficult to believe we were completely successful in negating lack of instructor experience with the robots as a factor in student learning.

The next step in this research is to uncouple the effects of robotics from the effects of reduced access to the programming feedback loop by adding a simulator to Ada/Mindstorms. Because of the enormous size of the design space (Which robots should we simulate? What environment will they operate in?), this presents significant challenges. Based on the results we have seen, our goal is to produce a simulator that runs quickly, is easy to use, and reliably replicates the behavior of simple Mindstorms robots so that students can have a high degree of confidence that once their program works on their computer it will work in a robot. At the same time, we would like the simulator to function with different robot designs operating in different environments, to maximize the programs usefulness to educators and to enhance the "fun factor". This work is currently in progress [6].

5 Acknowledgements

Funding for this work was provided by the Institute for Information Technology Applications, USAF, whose support is gratefully acknowledged.

References

[1] AAAI 2001 Spring Symposium on Robotics and Education, March 2001, numerous authors.

[2] Allen, A. 1990. *Probability, Statistics, and Queuing Theory with Computer Science Applications*, 2nd Edition, Academic Press, San Diego, CA.

[3] Fagin, B. Using ada-based robotics to teach computer science . In *Proceedings of the 5th International Conference on Innovation and Technology in Computer Science Education*, Helsinki, Finland, June 2000, 148-151. Available *http://www.faginfamily.net/barry/Papers/ITICSEWeb/using_ada.htm*

[4] Fagin, B. An ada interface for lego mindstorms. *Ada Letters*, vol. 20 no 3, 20-40. Available *http://www.faginfamily.net/barry/Papers/AdaLetters.htm*

[5] Fagin, B., Merkle, L., and Eggers T. Teaching basic computer science concepts with robotics using ada/mindstorms 2.0. in *Proceedings of SIGADA* '01, Bloomington, MN, October 2001, 73-78. Available http://www.acm.org/sigada/conf/sigada2001

[6] Fagin, B. Ada/Mindstorms 3.0: A computational environment for introductory robotics and programming, *IEEE Robotics and Automation* special issue on robotics and education, to appear.

[7] Fagin, B. and Merkle, L. Quantitative analysis of the effects of robots on introductory computer science education, *ACM Journal of Educational Resources in Computing*, June 2002 (submitted).

[8] Baum, D. 2002. The NQC Web site. Available http://www.enteract.com/~dbaum/nqc.

[9] Beer, R., AND Chiel, H. 1999. Using autonomous robotics to teach science and engineering, *Communications of the ACM* vol. 42 no. 6, 85-92.

[10] Congdon, C., Fagin, B., Goldweber, M., Hwang, D., Klassner, F. 2001. Experiences with robots in the classroom, panel presented at 32nd SIGCSE Technical Symposium on Computer Science Education.

[11] Flowers, T. and Gossett, K. 2002. Teaching problem solving, computing, and information technology with robots, unpublished paper, project info. available at http://www.dean.usma.edu/dean/ComputingAtWestPoint/Robots.htm.

[12] Harlan, R et. al. 2001. The khepera robot and the kRobot class: a platform for introducing robotics in the undergraduate curriculum. In *Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education*, Charlotte, NC, February 2001, 105-109.

[13] Klassner, F. 2002. A case study of LEGO Mindstorms' suitability for artificial intelligence and robotics courses at the college level. . In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer*

Science Education, Northern Kentucky, February 2002, 8-12.

[14] Pattis, P. 1981. *Karel the Robot: A Gentle Introduction to the Art Of Programming*, 2nd ed., John Wiley and Sons,.

[15] Wolz, U. 2001. Teaching design and project management with lego RCX robots. In *Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education*, Charlotte, NC, February 2001, 95-99.

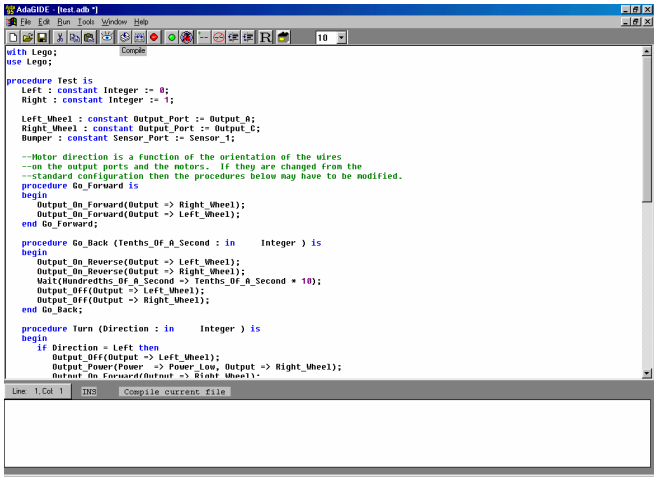

Figure 1: Ada/Mindstorms Programming Environment

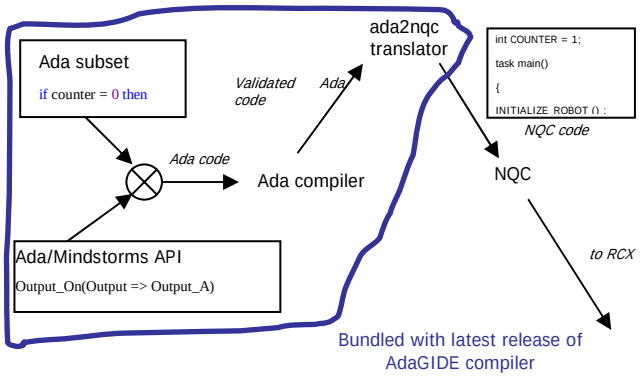

Figure 2: Ada/Mindstorms Code Flow

Table I. Sample Sizes

Population	Graded Events			
	Fall 2000		Spring 2001	
	Midterms	Final Exam and Course Total	Midterms	Final Exam and Course Total
Robotics	53	50	130	125
Non-Robotics	444	417	311	292

Table II. Fall 2000 Data

	Raw scores				GPA-adjusted scores			
Event	μ_1	μ_2	H	p	R_1	R_2	H	p
Exam 1	88.1	91.2	.72	.40	0.0	0.0	.25	.62
Exam 2	107.5	112.0	1.96	.16	-0.1	0.0	.64	.42
Final MC	109.4	114.8	3.79	.05	-0.3	0.0	2.93	.09
Final SA	77.4	83.8	6.02	.01	-0.2	0.0	3.73	.05
Final total	186.7	198.5	7.10	.01	-0.3	0.0	6.09	.01
Course total	764.6	802.7	4.14	.04	-0.2	0.0	3.84	.05

Table III. Spring 2001 Data

	Raw scores				GPA-adjusted scores			
Event	μ_1	μ_2	H	p	R_1	R_2	H	p
Exam 1 MC	44.9	45.5	.59	.44	0.0	0.0	.17	.68
Exam 1 SA	53.0	57.3	27.24	0	-0.4	0.2	24.98	0
Exam 1 Total	97.9	102.9	17.82	0	-0.3	0.1	16.74	0
Exam 2 MC	38.7	38.5	.003	.95	0.1	0.0	.33	.57
Exam 2 SA	54.2	57.4	7.25	.01	-0.2	0.1	5.71	.02
Exam 2 Total	92.9	95.9	4.76	.03	-0.1	0.0	3.06	.08
Final MC	124.1	125.8	1.77	.18	-0.1	0.0	.53	.47
Final SA	77.0	80.0	4.45	.04	-0.1	0.1	3.43	.06
Final total	201.1	205.7	4.10	.04	-0.1	0.0	2.1	.15
Course total	846.4	872.5	8.24	0	-0.2	0.1	6.47	.01

Table IV. Complete academic year data

Event	R_1	R_2	H	p
Exam 1 Total	-0.2	0.1	8.87	0
Exam 2 Total	-0.2	0.0	11.67	0
Final MC	-0.1	0.0	4.44	.04
Final SA	-0.1	0.0	5.62	.02
Final total	-0.2	0.0	6.15	.01
Course total	-0.2	0.1	11.88	0