AC 2007-1207: TEACHING STRUCTURED PROGRAMMING USING LEGO PROGRAMMABLE BRICKS

Eric Wang, University of Nevada-Reno

ERIC L. WANG is an Associate Professor of Mechanical Engineering at the University of Nevada, Reno. Dr. Wang has won numerous awards including the Tibbitts Distinguished Teaching Award, UNR's most prestigious teaching award. In addition to his pedagogical activities, Dr. Wang conducts research on sports equipment, biomechanics, robotics, and intelligent materials.

Jeffrey LaCombe, University of Nevada-Reno

JEFFREY C. LACOMBE is an Assistant Professor of Metallurgical and Materials Engineering at the University of Nevada, Reno. In addition to his education-oriented research activities, Dr. LaCombe's research lies in the areas of kinetic processes in materials (such as diffusion and solidification), nanoscale manufacturing methods, and remotely operated aerospace & satellite systems.

Ann-Marie Vollstedt, University of Nevada-Reno

ANN-MARIE VOLLSTEDT is a doctoral student in the Department of Mechanical Engineering at the University of Nevada, Reno. Ms. Vollstedt has MS degrees in both Mechanical Engineering and Secondary Science Education. Her dissertation focuses on methods to increase the cognitive development of engineering students.

© American Society for Engineering Education, 2007

Teaching Structured Programming Using LEGO Programmable Bricks

Abstract

For the first time in nearly a decade, the LEGO programmable brick has undergone a major hardware revision. The LEGO programmable brick has been adopted for a variety of uses in primary, secondary, and higher education. With the introduction of the new hardware, there appears to be a growing interest in using the programmable brick for teaching computer programming to college students. The goal of this project was to develop a set of instructional workshops, online tutorials, and accompanying project-based learning exercises that, combined, teach the basics of structured computer programming.

Traditionally, structured computer programming is taught in an instructor-centric manner using a combination of lectures and programming assignments. The use of the programmable brick facilitates the use of student-centric active/project-based teaching methods. The instructional model described in this paper includes alternating weeks of workshops (i.e. interrupted lectures) and projects, supplemented with online video tutorials for asynchronous learning. The instructional materials include ROBOLAB, which is a graphical programming language, and the programmable LEGO brick.

A series of workshops and assignments have been developed and refined over the past several years and spans both the old and new hardware versions. A series of online tutorials were developed to explain each programming concept and an online learning module, complete with self-study quizzes, was developed to help students transfer the skills learned in the graphical programming environment to the traditional text-based format, such as that commonly used in C programming.

Concept inventories were used to assess student learning and a statistical analysis of student use was performed to assess the utility of each of the online video tutorials. Finally, a control-group study investigated the difference in student learning between exclusive use of an online learning module compared with learning experiences supplemented by in-class instruction. The concept inventory for computer programming was developed and implemented for the first time during the spring 2006 semester in order to assess student learning. The new hardware will be introduced for the first time in the spring 2007 semester. The concept inventories included both ROBOLAB (graphical) and pseudo-code (text-based) questions. The pseudo-code component was deemed important in order to quantify the student's ability to transfer knowledge between domains. The key concepts included in the inventory were: goto's, conditionals, loops, nested structures, variables, functions/arguments, and subroutines/subprograms.

1. Background

There is a vast history of using LEGO® bricks in education. Projects that use the RCX programmable brick have included a wide variety of projects and courses ranging from robot competitions[1-3] to laboratory experiments[4-10] to project based learning[11-17]. There have also been a few recent publications dealing specifically with computer programming[18-20], which is the focus of the study described herein. By definition, the RCX requires programming and almost all the studies found in the literature address programming to some degree. As one recent study points out, because students are used to using fully functional software applications, they are quickly frustrated by their limited ability to write useful programs[18]. None of the literature found provided quantifiable assessment data indicating the efficacy of the approach taken.

With this in mind, the overall objective of this project was to measure the effectiveness of the student centered instructional method used. This paper first describes the methodologies of how we used the LEGO® RCX and NXT programmable bricks in the classroom, the assessment methodologies we employed, and the assessment data itself. Finally, an analysis and discussion of the results is presented.

2. Hardware and Software

After a decade of extensive use, the Robotic Command eXplorer (RCX) programmable brick was recently replaced by the NXT programmable brick (Figure 1). As shown in Figure 1, the RCX has six ports (three input ports and three output ports while the NXT has 8 ports (four input, three output, and a USB port). Both the RCX and NXT are also equipped with LCD screens for displaying useful information, four command buttons, and a built-in speaker for playing sounds.

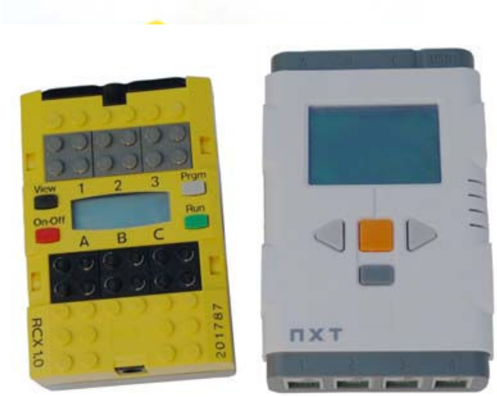

Figure 1: The RCX (left) and NXT (right) programmable bricks

This paper focuses on the use of the ROBOLAB programming language, which can be used to program both the NXT and RCX. ROBOLAB was jointly developed by National Instruments (Austin, TX), Tufts University, and LEGO® Educational (Enfield, CT). As shown in Figure 2, program icons are "wired" together to create a program. Programs created can contain all the typical programming elements such as constants, variables, loops and functions. The primary reason for concentrating on ROBOLAB is that students can learn to program quickly with very little instruction and with no previous programming experience. The added benefit is that they are introduced to LabVIEW, which is the most common data acquisition package currently on the market and is used in most industry and government laboratories.

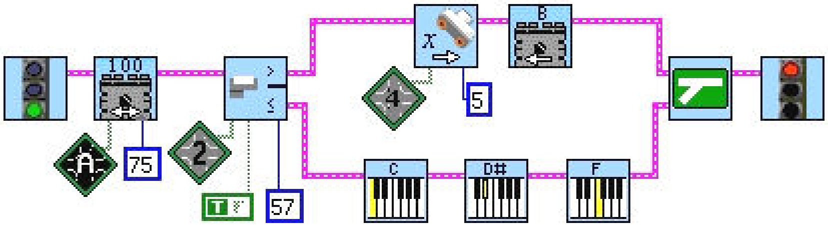

Figure 2: Sample ROBOLAB program.

3. Course Description

The Mechanical Engineering (ME) and Material Science Engineering (MSE) Departments are participating in a multi-disciplinary first-year project funded by the William and Flora Hewlett Foundation[21]. As part of this program, an interdisciplinary freshmen-level course

(ME151/MSE102) is taken by all mechanical engineering and material science engineering undergraduates and some secondary science education majors. Traditionally these three courses have a combined enrollment of approximately 100 students. During the last offering, the education students constituted 4% of the enrollment.

As stated in the course syllabus: the overall goal of the course is for the student to learn the fundamentals of structured computer programming, the design process, and creative thinking. In order to accomplish this goal students create autonomous robots with LEGO bricks and a computer program called ROBOLAB. By the end of this course students should be able to: write ROBOLAB programs that contain structures, variables, and multi-tasking; explain the design process in their own words; and describe several creative thinking exercises.

4. Methods of Instruction

Over the past decade, this course has undergone several major revisions. Since 2005, we have been using a student-centered instructional approach. The course is now held in a computer laboratory with 26 students at a time, working in pairs. The format each week alternates between a workshop and a design challenge (mini design project). A carefully designed scaffolding curriculum is used across the semester.

During the weeks in which a workshop format is used, the class consists of an interrupted lecture where students alternate between listening for brief periods and then actively participating (i.e., programming). The instructors and teaching assistant circulate to help *guide* students rather than *show* them. LEGO® robots were used so that students would have a tangible application for their computer programs. We believe that having an application is inherently more engaging and motivating for students.

Every workshop deals with the development of a line following robot. The robots do not increase in complexity. Rather, the line following algorithm is different each week and, thus, requires an ever expanding set of computer programming skills. The hypothesis was that the use of a consistent framework for each workshop would help students both retain and transfer knowledge from one week to the next.

To provide additional scaffolding, each design challenge was created to develop specific skills that would be required to complete the final competition. Design challenges are essentially open-ended (design) problems that the students must build and program a robot to solve. During the weeks in which a design challenge is held, students bring their robotic designs to class and are evaluated based on performance, creativity and the content of their programs. Students are given at least one week to design, build, program, and test their designs. About half of the design challenges are in the form of robot competitions. The course culminates in a final robot battle.

A concept map (Figure 3) was developed to summarize the activities for the semester. The concept map indicated the seven skills required for the final competition (indicated in 2nd row of boxes), the design challenges which were used to assess the skills (circles), and the underlying workshops (bottom box) which were used to teach the skills. This graphic was shown to students every week to remind them of the overall goal, current position, and the progress that had been made.

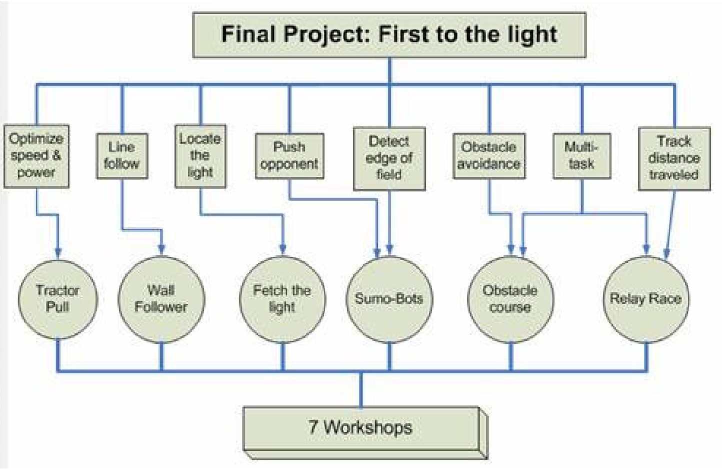

Figure 3: Concept map for the semester.

The student-centered instructional method also included 13 online video tutorials, ranging in duration from just over 2 minutes to just under 7 minutes. Ten of the video tutorials covered programming with ROBOLAB[22], one covered the grading criteria, and the remaining two covered Excel (which was necessary for one of the design challenges). The video tutorials were only accessible through WebCT as Flash videos so that they were neither accessible to anyone outside the course nor downloadable by the students. This was done to allow us to track the usage of the videos (described in the assessment tools below) while still allowing for asynchronous learning.

Finally, in order to help students transfer their knowledge of ROBOLAB to other programming languages, we developed an online learning module for pseudo-code within WebCT. The module contained examples and six self-study quizzes that focused on translating ROBOLAB programs into text-based pseudo-code programs and visa-versa. This learning module included an additional seven video tutorials and 13 audio commentaries.

5. Outcomes

The instructional methods, outcomes, and assessment tools are graphically represented in Figure 4. The measurable outcomes relating to computer programming included: basic knowledge of computer programming skills using ROBOLAB, ability to synthesize that knowledge to solve problems, and the ability to transfer that knowledge to other programming languages.

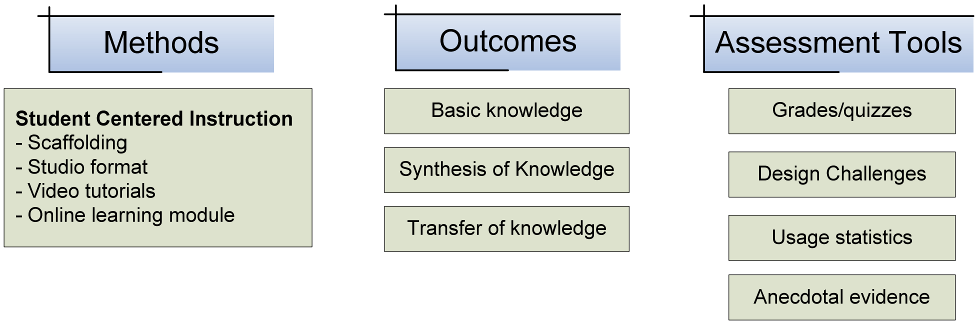

Figure 4: Summary of instructional methods, outcomes, and assessment tools used.

From a pedagogical standpoint, the redesign of this course paid special attention to both the type of assignment, as classified by Bloom's taxonomy[23], and the expected level of cognitive development of our students, as classified by Perry's model[24]. Since this is a first-year course, we expected students to be roughly at Perry's position 2 (dualistic thinkers). With this in mind, we were careful about asking students to synthesize and evaluate. In terms of Bloom's taxonomy, the concept inventories (i.e., quizzes) evaluated basic knowledge and comprehension. The design challenges evaluated application, analysis, and a limited amount of synthesis (which they understandably struggled with). The final competition focused on analysis and synthesis. The reflective reports addressed analysis, synthesis, and evaluation.

6. Assessment Tools

As shown in Figure 4, there were four assessment tools used in this project: grades/quizzes, design challenges, usage statistics, and anecdotal evidence. There were three quizzes during the semester. The first two quizzes, covering ROBOLAB, were organized as concept inventories and were used to assess students' basic knowledge of ROBOLAB. The two concept inventory quizzes covered the following programming concepts: goto's, conditionals, loops, nested structures, variables, functions/arguments, and subroutines/ subprograms.

The third quiz was used to assess the students' ability to translate ROBOLAB programs to/from pseudo-code (i.e., transfer of knowledge). All three quizzes were online quizzes taken during the normal class time in the computer lab. There were at least three alternates for each quiz question so that students in later sections of the course would not have an unfair advantage (i.e., cheating).

To assess the whether or not student could learn pseudo-code using the online learning module alone, a control group study was completed. While all of the students were given access to the online learning module, only half of the students were also given in-class instruction that covered the same material. Since there were three different instructors involved in the course, the students that received the additional in-class lectures were all taught by the same instructor. By comparing the performance on the third (pseudo-code) quiz we can infer whether or not the online module alone is sufficient to teach students how to transfer their ROBOLAB knowledge to another programming language.

Grades for the reflective reports were used to assess the students' abilities to analyze and evaluate their performance. As mentioned earlier, because of the expected Perry's position of the first-year students, the reflective reports were graded quite leniently. The main goal was really to help develop the students' critical thinking skills.

The design challenges were the second assessment tool used. These were used to further evaluate basic knowledge along with rudimentary synthesis of knowledge. The final competition was used to evaluate the overall ability to synthesize knowledge gained over the semester. All of the design challenges we graded on the performance of the robot, creativity of the solution, and content of their ROBOLAB programs.

The method students used to access the online video tutorials (described above) ensured that only students in the class were watching the videos and that the students could not download the videos. Thus, the major assessment tool for the video tutorials was the usage statistics: number of hits and duration per hit. Knowing the length of each video, the number of hits, and the duration per hit, allowed us to deduce whether or not students rewound the videos.

Anecdotal evidence provided by the teaching staff served as the final assessment tool. Anecdotal evidence is primarily in the form of the impressions of teaching staff as to what went right and what went wrong. While quantifiable assessment data is extremely valuable, as evidenced by the literature review, the qualitative data is also of great value.

7. Results & Discussion

A summary of student performance on the first two quizzes are presented in Table 1, organized by concept. Overall, the instructors were quite pleased that 75% of the students could demonstrate basic knowledge of 5 of the seven concepts. It is no surprise that the vast majority of students understood the function of a GoTo. Somewhat surprising, however, is the result that over 80% of the students understood subroutines. Since ROBOLAB makes it very easy to incorporate subroutines, this result may be attributable to the use of this particular programming environment.

Table 1. Results of concept inventories

Concept	**% of correct answers**
GoTo's	82.6%
Conditionals	75.3%
Loops	58.3%
Nested structures	74.5%
Variables	75.5%
Functions/arguments	60.3%
Subroutines/ subprograms	80.6%

The performance for conditionals, nested structures and variables were nearly identical. Students obviously struggled more with loops than conditionals. The fact that students performed better for nested structures (74.5%) than for loops (58.3%) caused some concern because loops are supposedly a simpler concept. However, a detailed analysis of the nested structures questions revealed the same discrepancy between the understanding of loops and conditionals. The results showed that 94% of the students correctly answered questions on nested conditionals, whereas only 55% correctly answered questions on nested loops.

While this helps explain how the results for nested structures can exceed those of loops, this raises another concern; more students correctly answered questions regarding nested conditionals (94%) as compared to non-nested conditionals (75.3%). Again, a detailed analysis of the results indicated that two of the four alternate questions on conditionals had correct response rates exceeding 90% while the other two were near or below 50%. Thus, it appears two of the non-nested conditionals problems were confusing to students (i.e., not well designed questions). This can explain why students performed better for nested conditionals as compared to the easier concept of non-nested conditionals.

While the quizzes were meant to discourage it, it is possible since the students worked in pairs that only one partner did all of the programming. The results of the concept inventories indicate that this was most likely not the case for most teams since the performance was never below 50% for any of the concepts.

As mentioned previously, while the first two quizzes dealt with concept inventories, the third quiz was used to assess transfer of knowledge. A t-test was performed (α=0.05) to compare the scores of the third quiz for those students that received additional in-class instruction (N=42) and those who did not receive any additional instruction (N=56)). The results indicated that there was no statistical difference between the groups (P=0.631). Thus, we conclude that the online module alone was sufficient to teach the students how to transfer their ROBOLAB knowledge into pseudo-code. From an instructional workload standpoint, this is good since the learning module has been developed and can be deployed year after year.

In terms of the design challenges, the anecdotal evidence provides the most useful information. The anecdotal evidence suggests that the notion of using the design challenges and final competition to teach students how to synthesize knowledge worked. We found that the majority of the students could successfully take the knowledge learned in the workshops and apply it to solve the design challenges. The performance on several of the design challenges was quite poor, but we found that students were having far more difficulty devising a workable algorithm than actually implementing an algorithm. In terms of Bloom's taxonomy, their ability to apply knowledge exceeds their analysis and synthesis skills.

We also found that the overall student performance in the design challenges has been much higher since we switched to the workshop (student-centered) instructional format. We attribute this to the fact that the workshops are student-centered. We can both devote more time to struggling students and identify topics that students have difficulty with. The end result is that the basic knowledge is in place before we ask students to apply, analyze, and synthesize new knowledge.

Table 2. Video tutorial usage statistics

Video tutorial topic	Duration	Hits	Average duration/hit	duration ratio
Understanding the ROBOLAB Tools	4:01	22	11:14	2.80
Wiring Basics	4:19	9	7:22	1.71
Using LEGO Motors	2:05	0	n/a	n/a
Defaults & Modifiers [arguments]	6:03	0	n/a	n/a
Jumps [goto]	3:24	20	10:45	3.16
Loops [do while]	5:45	31	5:37	0.98
Forks [conditionals]	3:15	32	4:00	1.23
Nested Forks [nested conditionals]	3:48	55	10:37	2.79
Container Basics [variables]	4:53	45	21:05	4.32
Multi-tasking [threads]	3:38	48	13:18	3.66

Both the concept inventories and anecdotal evidence from the workshops have helped identify topics that students have trouble with. The usage statistics provides yet another method to identify these topics. Table 2 summarizes the usage statistics for the video tutorials. We define the duration ratio as the ratio of average duration/hit to duration of the video. Thus, a duration video of less than 1.0 indicates students did not, on average, watch the entire video. Conversely, a duration ratio of greater than 1.0 indicates that students either paused or rewound the video. With 98 students in the class, the number of hits indicates that even the most popular videos were only watched by just over half of the class. However, since the students worked in pairs, this result does not seem too surprising. Of the ten ROBOLAB video tutorials, two of the tutorials were never viewed. On average, the other 8 videos were watched by about a third of the class.

Of the eight tutorials watched, only one had a duration ratio less than 1.0 and even this case the duration ratio was 0.98, indicating that students watched 98% of the video on average. The remaining seven video tutorials garnered duration ratios ranging from 1.23 to 4.32. On the low end (1.23) it seems likely that the students rewound a small portion of the video to watch again. For the high duration ratios, however, it is hard to believe that the students would watch the same video more than four times, it is more plausible that the students must have paused the video while writing their ROBOLAB programs. Regardless of what caused the high duration ratios, they are a clear indication of which topics the students struggled with.

There is only a very weak correlation between the number of hits and the duration ratio, indicating that the popularity of a particular video is not a predictor of the degree of difficulty of the topic covered in the video. Thus, the number of hits appears to be a good indicator of which topics the students need additional help with and the duration ratio is a good indicator of which concepts the students are having difficulty grasping. Personal conversations with the students revealed that students really appreciated the video tutorials; they asked for more, but we simply did not have enough time to produce additional tutorials.

8. Conclusions

Based on the results of this project, several conclusions can be drawn:

1. Overall, the instructors were satisfied that about 75% of the students could demonstrate basic knowledge of all concepts, with the exception of loops. The results indicate that more time must be spent on loops in the future.
2. The online learning module was sufficient to teach students how to transfer their knowledge in ROBOLAB to other programming languages.
3. The scaffolding technique employed worked to help students apply, analyze, and synthesize knowledge.
4. On average, about a third of the class made use of the online video tutorials and most either rewound or paused the videos, indicating they were interacting with the tutorials.
5. The video tutorials worked quite well by reaching students through a medium that they are accustomed to.

9. Acknowledgments

This work was sponsored by the William and Flora Hewlett Foundation's Engineering Schools of the West Initiative and the Tufts University Center for Engineering Education Outreach.

10. Bibliography

1. Gerovich, O.; Goldberg, R.P.; Donn, I.D.; "From science projects to the engineering bench" IEEE *Robotics & Automation Magazine*, 10(3):9-12, Sept. 2003.

2. Chung, C. J. and Anneberg, L. "Robotics contests and computer science and engineering education" *proceedings of the 33rd ASEE/IEEE Frontiers in Education Conference*, Session F1F, Boulder, CO, November 2003.

3. Self, B., Wood, J. and Hansen, D., "Teaching Undergraduate Kinetics Using LEGO Mindstorms Race Car Competition," *proceedings of the ASEE Annual conference and exhibition*, session 3668, 2004.

4. Cyr, Martha, Miragila, V., Nocera, Rogers, Chris, "A Low-Cost, Innovative Methodology for Teaching Engineering Through Experimentation," *Journal of Engineering Education*, **86**(2):167-171, 1997.

5. Portsmore, M., Cyr. M, and Rogers, C. "Integrating the Internet, LabVIEW, and Lego bricks into modular data acquisition and analysis software for K-college" *proceedings of the ASEE Annual conference and exhibition*, session 2320, 2000.

6. Rogers, C. and Portsmore, M., "Data acquisition in the dorm room: teaching experimentation techniques using LEGO materials" *proceedings of the ASEE Annual conference and exhibition*, session 2366, 2001.

7. Brockman, J., Batill, S., Renaud, J., Kantor, J., Kirkner, D., Kogge, P., and Stevenson, R., "Development of a multidisciplinary engineering design laboratory at the University of Notre Dame" *proceedings of the ASEE Annual conference and exhibition*, 1996.

8. Levien, K.L. and Rochefort, W.E. "Lessons with LEGO - engaging students in chemical engineering course" *proceedings of the ASEE Annual conference and exhibition*, session 1313, 2002.

9. Moor, S.S., Piergivanni, P., and Keyser, D. "Design-build-test: flexible process control kits for the classroom" *proceedings of the ASEE Annual conference and exhibition*, session 1526, 2003.

10. Wang, E., LaCombe, J. and Rogers, C. “Using LEGO Bricks to Conduct Engineering Experiments,” *proceedings of the ASEE Annual conference and exhibition*, session 2756, 2004.

11. Flikkema, P. "Learning Embedded and Real-time systems via low-cost mobile robots" *proceedings of the ASEE Annual conference and exhibition*, session 1332, 2001.

12. Goff, M. and Vernon, M. "Using LEGO RCX bricks as the platform for interdisciplinary design projects" *proceedings of the ASEE Annual conference and exhibition*, session 3425, 2001.

13. Kumar, A. "Using robots in an undergraduate artificial intelligence course: an experience report" *proceedings of the 31st ASEE/IEEE Frontiers in Education Conference*, Reno, NV, October 2001.

14. Shih, A. and Hudspeth, M. "Using the LEGO robotics kit as a teaching tool in a project-based freshmen course" *proceedings of the ASEE Annual conference and exhibition*, session 1353, 2001.

15. Campbell, M.I. "Teaching machine design through product emulation" *proceedings of the ASEE Annual conference and exhibition*, session 2366, 2002.

16. Gage, A. and Murphy, R. "Principles and experiences using LEGOs to teach behavioral robotics" *proceedings of the 33rd ASEE/IEEE Frontiers in Education Conference*, Session F4E, Boulder, CO, November 2003.

17. Tester, J. and Hatfield, J. “The Design4Practice Sophomore Design Course: Adapting to a Changing Academic Environment,” *proceedings of the ASEE Annual conference and exhibition*, 2005.

18. El Naga, H, Stokes, S. and El Naga, N. “The Use of Real Time Operating Systems and Development Tools as a Men to Revitalize Computer Engineering Programming Courses,” *proceedings of the ASEE Annual conference and exhibition*, session 1532, 2004.

19. Schumacher, J., Welch, D., and Raymond, D. "Teaching introductory programming, problem solving and information technology with robots at West Point" *proceedings of the 31st ASEE/IEEE Frontiers in Education Conference,* Reno, NV, October 2001.

20. Plett, G, and Ciletti, M. “Piloting a Balanced Curriculum in Electrical Engineering – Introduction to Robotics,” *proceedings of the ASEE Annual conference and exhibition*, 2005.

21. Wang, E. Velasquez-Bryant, N., Adams, J., Batchman, T., Cantrell, P., Jacobson, E., Johnson, W., Kleppe, J., LaCombe, J., LaTourrette, N., Norris, G., Sparkman, W., and Varol, Y. “First Year Engineering Experience Initiative,” *proceedings of the ASEE Annual conference and exhibition*, session 2171, 2004.

22. http://www.me.unr.edu/lego/podcast.htm

23. Bloom, B., Engelhart, M., Furst, E., Hill., W., and Krathwohl, D., **Taxonomy of Educational Objectives: The Classification of Educational Objectives. Handbook I: Cognitive Domain**, David McKay, New York, 1956.

24. Perry, W. Jr., **Forms of Intellectual and Ethical Development in the College Years: A Scheme**, Holt, Rinehart and Winston, New York, 1970.