Int. J. Engng Ed. Vol. 16, No. 3, pp. 181–192, 2000
Printed in Great Britain.

0949-149X/91 $3.00+0.00

LEGO Engineer and RoboLab: Teaching Engineering with LabVIEW from Kindergarten to Graduate School*

BEN ERWIN, MARTHA CYR and CHRIS ROGERS
Center for Engineering Education Outreach, Tufts University, Medford, MA 02155, USA.
E-mail: crogers@tufts.edu

For the past 6 years, faculty members at Tufts University have developed two different software packages between LabVIEW™ and LEGO™ data acquisition systems. These packages allow us to teach engineering with both LEGO bricks and LabVIEW to students from 5 to 50 years old. The versatility of the hardware and software allow a wide variety of possibilities in what students can build and program—from robots and remote sensing devices to kinetic sculptures. As students design and build their projects, they are motivated to learn the math and science they need to optimise their project. Both college students and kindergartners respond to this motivator. In the paper, we explain how we designed software to complement these projects in allowing automation and animation. The software uses LabVIEW, extending its capabilities to kindergartners and LEGO bricks. Finally, we will show how we have used LabVIEW and LEGO data acquisition to teach elementary school science, freshman engineering, instrumentation and experimentation, and how college seniors and graduate students have used both the hardware and software to solve various data acquisition problems.

THE HISTORY

IN THE SUMMER of 1993, when searching for a cheap data acquisition system for an undergraduate course at Tufts University, we stumbled across the Control Lab Interface from LEGO™ DACTA. The building capability of the LEGO bricks, coupled with the computer interface resulted in an ideal combination for fast, creative, and fun experiment design and construction. The Control Lab Interface connects to the computer through a serial port and controls LEGO motors and lights and reads from LEGO sensors. These fit the bill quite well and so we started working on LabVIEW™ drivers for the box and called it LEGO Engineer. Soon after, the first LabVIEW student version came out and we quickly adapted the code to work with the student version. Coupling this with a web site and a curriculum, we convinced NASA to fund a full-scale project in local elementary schools. This resulted in over 100 teachers working together to build curriculum and ideas and—to date—over 4000 students. We have worked with schools all over the United States and people have down loaded the drivers from our web site from almost every continent.

Kindergartners have used LabVIEW and the LEGO bricks to build their town and automated a bus to stop at each house [1]. Along the way they learned cartography, practiced their budding reading and writing skills and had animated discussions about friction and design optimisation to improve the performance of their busses. Likewise, NSF funded the introduction of the LEGO Engineer software into the college curriculum to teach experimentation [2]. College juniors learned about statistical analysis, sampling theory, and report writing while enjoying the versatility the LEGO bricks have to offer. College seniors went on to build a computer-controlled milling machine with three degrees of freedom as a capstone design experiment. The user could draw the hull of a boat on the computer, and the milling machine would cut it out of balsa wood. One graduate student built a drop tower out of LEGO bricks and used the LabVIEW server subroutines to allow anyone anywhere to control the tower over the web. A drop tower allows a scientist to view behaviour in the absence of gravity since as the experiment drops, everything in the experiment drops at the same acceleration. When users logged in, they could hoist a small chamber containing the desired experiment. The students built four different chambers: a simple candle (the flame goes from teardrop to round in a zero gravity environment), water in a test tube (in the absence of gravity, the water surface becomes highly curved as the water 'walks' up the walls), two opposing magnets (balanced only in a gravity environment since gravity opposes the magnetic repulsion), and a small scale (so that one could see the weight of an object to zero in the zero gravity environment). Two accelerometers placed on the bottom of the chamber were used to control the ascent rate of each side of the chamber to ensure that the

* Accepted 9 September 1999.

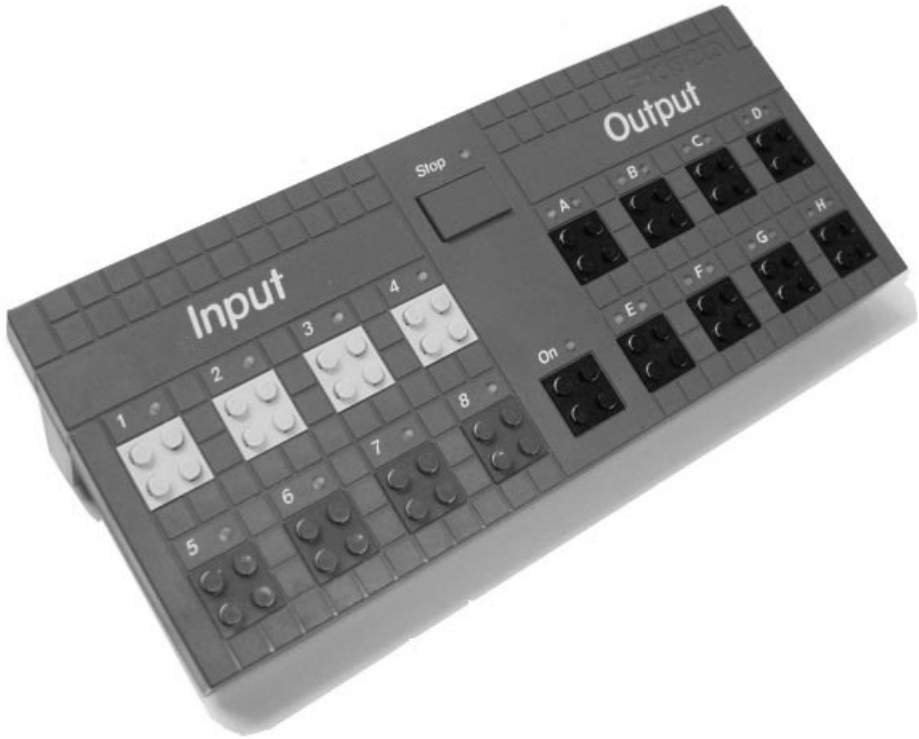

Fig. 1. The control lab interface box.

chamber rose evenly. Then the user could drop the chamber and the LabVIEW server would send back sequential video images of what happened inside the chamber as it dropped.

LEGO Engineer was so successful that LEGO, Tufts, and National Instruments set up an alliance to generate the educational software for their next generation data acquisition system, the RCX. The RCX was different from the previous interface in that it was an independent microprocessor embedded in a LEGO brick. The computer was used only to down load the control code into the microprocessor. After that point, the user is independent of the computer. This resulted in RoboLab; a library of subroutines (VIs) and virtual instruments (panels) powered by LabVIEW 5.01 that has gone into over 1000 schools since it shipped in September of 1998. Since then, RoboLab has been used in classrooms from preschool to college [3].

The main goal of this paper is to explain the programming philosophy we used in developing these two software packages. The ultimate goal was to develop software that has a low entry level (can be used by kindergartners) and high ceiling (can be used by college students) without either extreme feeling overwhelmed or limited. We will outline first the design philosophy and some of the ways we used LabVIEW G code to interface LEGO bricks with students and then present how these students have liked it so far; students ranging from elementary school to college. LEGO Engineer is freeware and is available from the WWW at http://ldaps.ivv.nasa.gov/LEGOEngineer/ or from the accompanying CD. RoboLab is sold by PITSCO LEGO DACTA for $25 [4].

THE HARDWARE

Before presenting the software, it is important to understand the capabilities of and differences between the two sets of LEGO hardware. The hardware must be similar to the software: low entry level, few limits, very capable, and cheap. The LEGO material fits this limitation well. One can build simple cars or complex machines with the same bricks, sensors, and motors. All bricks, motors and sensors are common to both data acquisition platforms. The first, the Control Lab Interface, is an extension of the computer. It connects through a serial port and therefore is limited to one experiment per computer. The second, the RCX, is an autonomous microprocessor and only relies on the computer to down load the control code. Therefore one can have many more experiments than computers.

Fig. 2. LEGO output devices.

Fig. 3. The LEGO sensors.

Control lab interface

The LEGO control lab interface (CLI), a $250 data acquisition device that connects to the serial port. Figure 1 shows the interface box. LEGO Engineer (LE) is a set of LabVIEW virtual instruments and subroutines that control this box. The CLI has 8 output ports (black) and 8 input ports (yellow and blue). The 8 output ports supply a variable voltage out to LEGO motors, lights or sound makers (dubbed 'sound blasters' by third graders)—see Fig. 2. The input ports are multiplexed to a 10 bit A/D with a 0–5 V range. The upper (yellow) inputs are used for resistive sensors such as the touch sensor (a switch) and the temperature sensor. The lower (blue) inputs can use powered sensors and are used for an angle sensor and a light sensor. One can adapt other sensors by simply externally powering the sensor and then using the upper ports to measure the voltage drop across the sensor. LEGO is currently developing an adaptor that will allow one to power and read the sensor from the lower ports. Figure 3 shows the available LEGO sensors. With a little ingenuity one can build force sensors, displacement sensors, bar code readers, and a number of other sensors with these basic sensor bricks.

The RCX

The RCX (Fig. 4) evolved from work done at the MIT media lab and is the basis for a whole new line of LEGO bricks in toy stores called Mindstorms™. It costs a bit less that the CLI (about $120) and has only three input and three output ports. Because the RCX is autonomous, the student can walk up to the computer, down load a program, and then walk away and execute the program. The program can control a robot, an intelligent house, or act as a data collection device, measuring anything from light to acceleration. The RCX has a 10–bit A/D, four timers, 32 internal variables, and three pulse-width modulated output ports. The input ports on the RCX are identical to the lower blue ports on the CLI. The same sensors and output devices work on both hardware platforms. RoboLab is a revised version of LabVIEW 5.01 and a set of subroutines and virtual instruments that allow the user to program the RCX. RoboLab was designed for LEGO specifically for the school environment and differs from the Mindstorms™ software in the toy stores in that it has a lower entry and a higher ceiling. Also, since it is based on LabVIEW, it is cross-platform and can run on both PCs and Macs.

The software interface

Our general philosophy in the development of software for both hardware sets was to introduce the user to programming through levels. At the introductory level (front panel programs), students

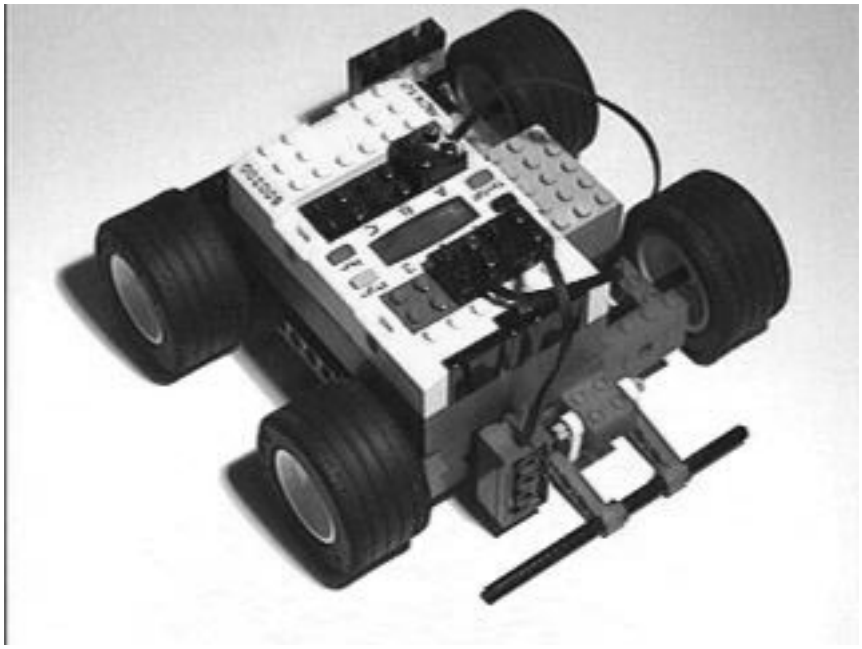

Fig. 4. The RCX.

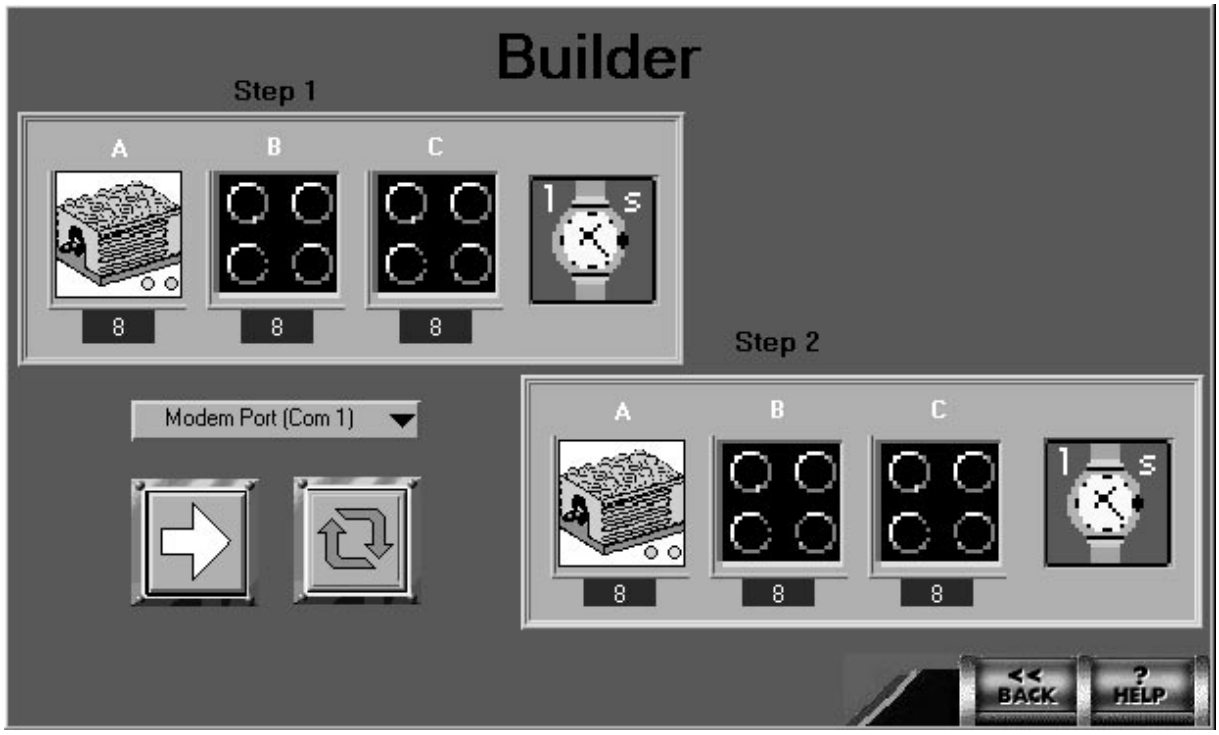

Fig. 5. Sample front panel.

pick and choose from a preselected set of choices. At the higher level (block diagram programs), the students actually write G programs by stringing subroutines together in a LabVIEW diagram. Each of these levels have sub-levels as well to keep the user from getting overwhelmed early on. LEGO Engineer and RoboLab differ some in how they present the levels and what is capable in each level, but the basic philosophy is the same: learn program structure first through picking from a few choices and then apply this knowledge to more complicated programs in the higher level. The introductory level has the further advantage that in classrooms where projects are limited to 45 minute time blocks, the students can actually build something and animate it in a single time block. The main difference between the two software products is that RoboLab is a LEGO product that requires no knowledge or ownership of LabVIEW (although the seasoned LabVIEW user can port RoboLab into LabVIEW and use both together) and LEGO Engineer is freeware and meant for users who already own LabVIEW (or RoboLab).

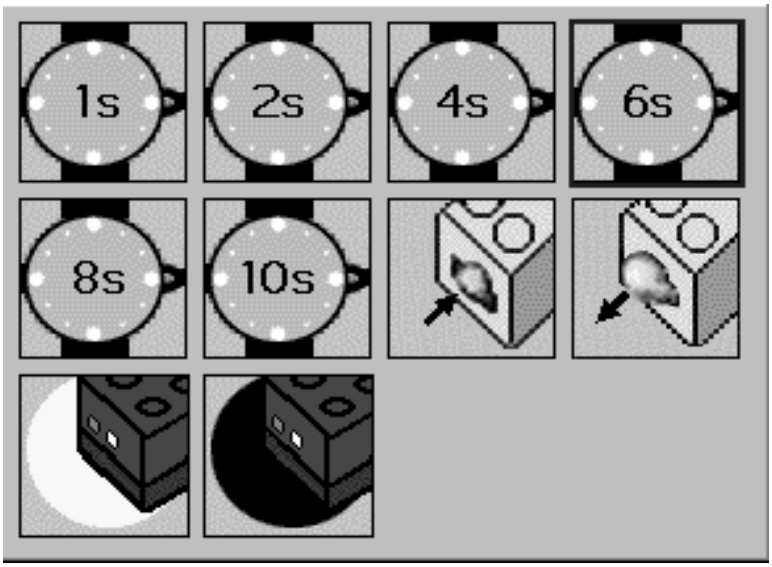

Fig. 6. Possible criteria.

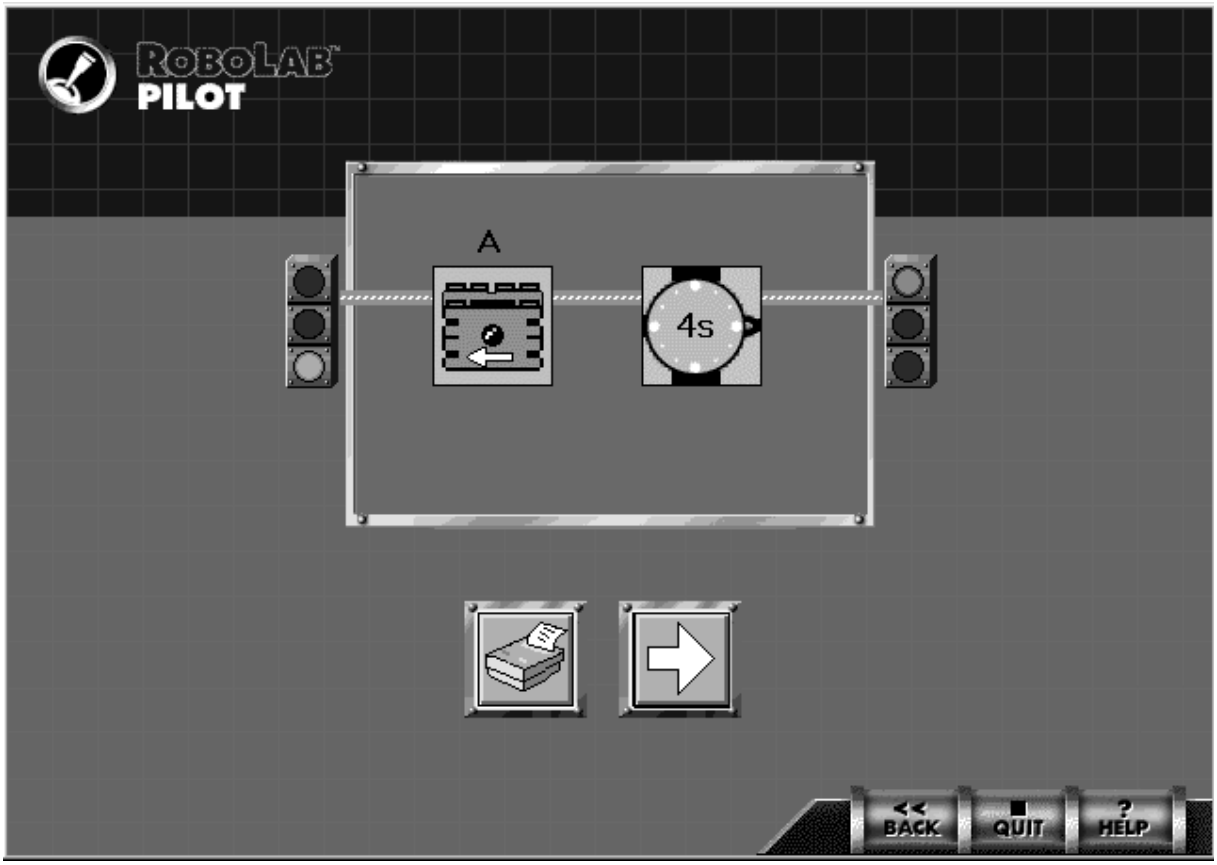

Fig. 7. Pilot 1 panel.

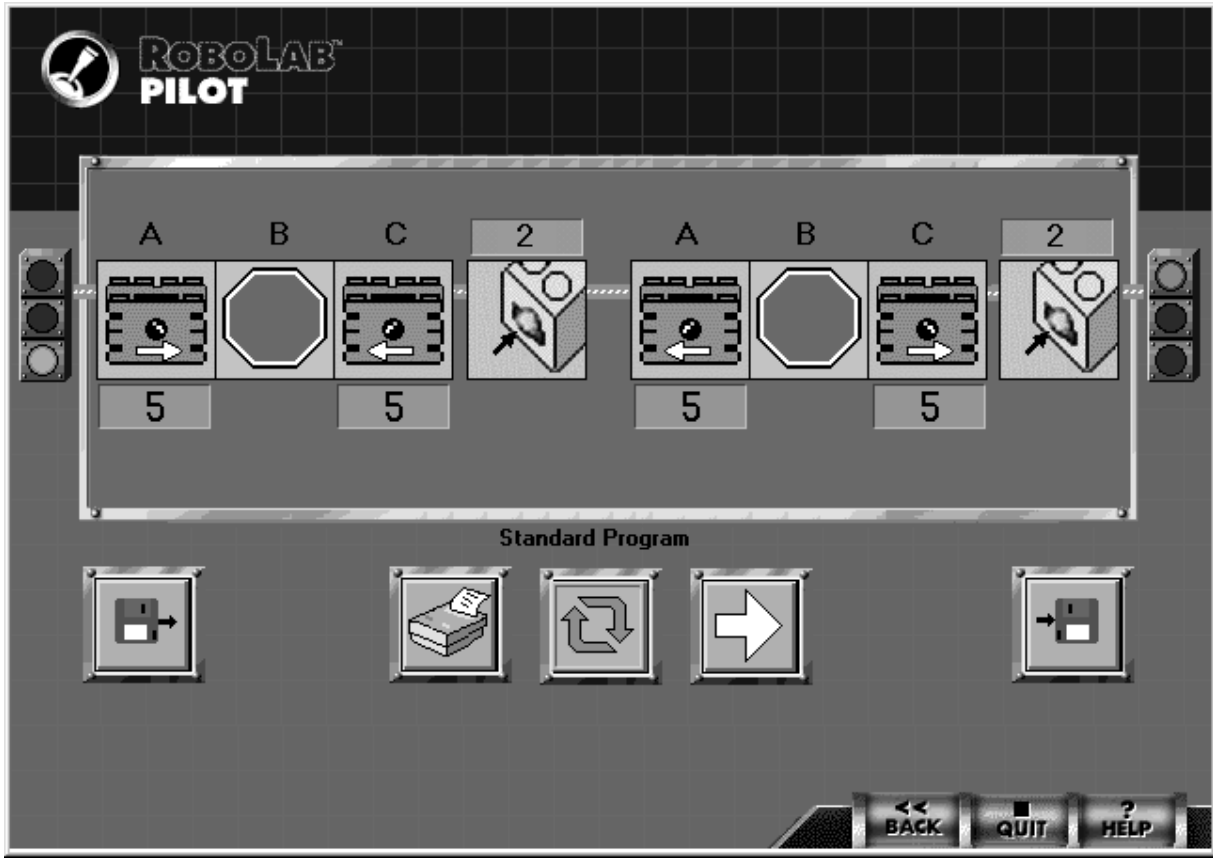

Fig. 8. Controlling a bumper car.

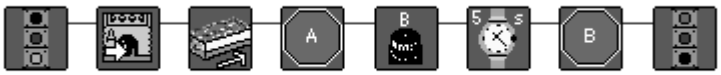

Fig. 9. Engineer level programming.

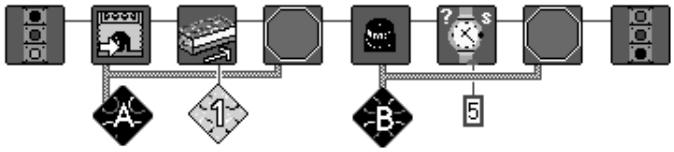

Fig. 10. Using modifiers.

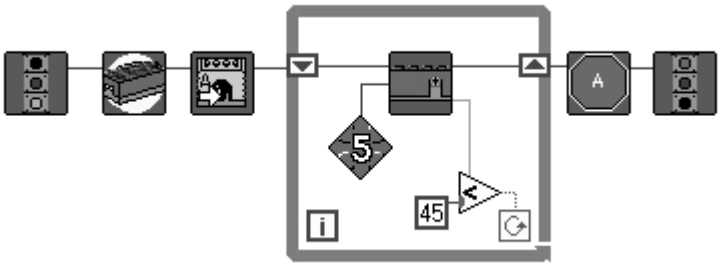

Fig. 11. The code for the automatic garage door.

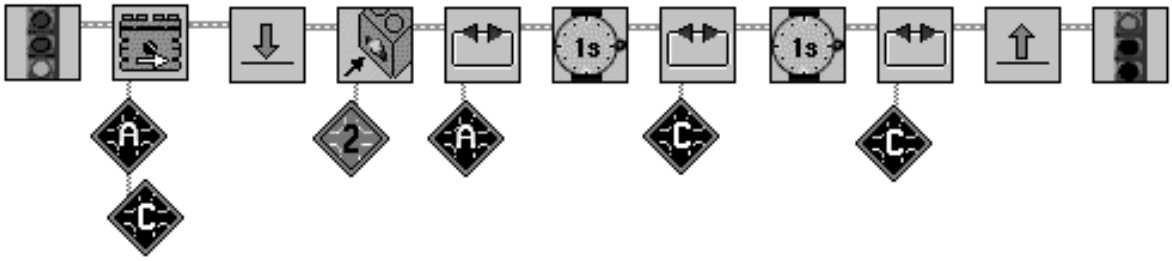

Fig. 12. The bumper car in inventor level.

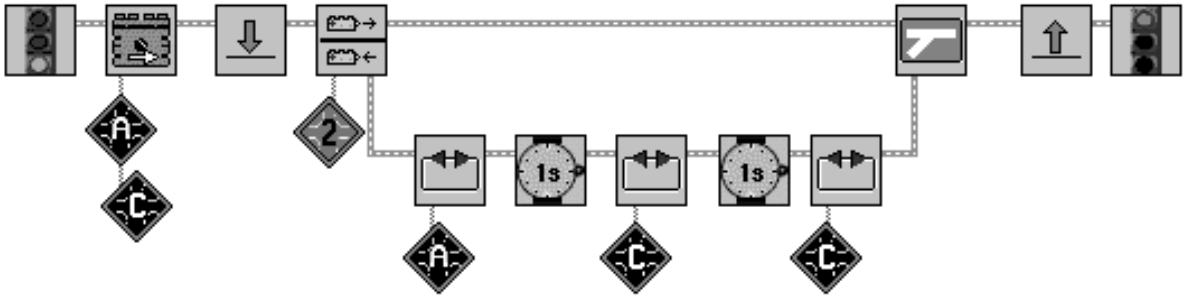

Fig. 13. The bumper car using structures.

Front panel programs

The front panel programs in LEGO Engineer are called the 'Builder Programs' and allow the user to quickly select from a number of choices. For instance, the program in Fig. 5 will turn on the motor connected to channel A at a speed 5 (out of 8), then wait for the touch sensor to be hit before turning the motor off and turning on the sound blaster for 5 seconds. This could be the code for a car that drives until it hits the wall—when it stops and plays a victory dance. Clicking on the picture of the motor (for instance) and selecting a different option (Fig. 6) can change any of these settings. This simplistic interface does not allow for a very high ceiling in terms of the software capability but it does allow the students to get something to work immediately. Further, it starts to teach them logical thinking: something happens, you wait for a condition, and then something else happens. They get used to the idea of program sequence and will use this in the higher level programs where the ceiling is much higher.

LEGO Engineer has a number of Builder programs with more available from the web. RoboLab, on the other hand, has four levels of

panel programming, called the 'Pilot level'. Pilot 1 introduces the concept of programming by only allowing the user to control one motor (see Fig. 7). The capabilities increase until Pilot 3 allows the students to build and program a number of simple robots. Figure 8 shows the program that would allow a bumper car to continually bounce from wall to wall. Pilot 4 gives the user the ability to program an infinite number of sequential steps, but usually by this time the student is ready to move on to G programming. The Pilot and Builder programs are an excellent way to show how the computer can control the LEGO bricks and how one can start to teach programming skills. The limited capability allows that every program will work—it might not do what was intended—but it will run. There are no syntax errors and, more impressively, there is no language. Because it is all graphical, kids can write programs before they can read. Very young kids (3 years old is the youngest) have successfully programmed cars and smart houses with this style of programming.

Block diagram programming

Most kids quickly leave the sheltered environment of the panel programs to the increased capability of the diagram programming. Again both sets of software differ in the presentation but the basic philosophy is the same: all programming can be done from the diagram alone. That is the user builds up a continuous thread of commands that defines the program. Figure 9 shows a sample program in LEGO Engineer that does the same thing as Fig. 8. These programs can become more complicated when one starts to use modifiers to the string. Modifiers either define the port or give a value. Figure 10 shows the same program as Fig. 9 only using modifiers. Most VIs (in both LE and RoboLab) have the same connector setup: a string across the top to define precedence in execution and defining modifiers (ports, motor speed, etc.) on the in the lower right. There are two basic differences between LE and RoboLab, however. The first is that LE has only one diagram programming level. RoboLab, on the other hand, has four. This is a direct result that RoboLab was written for those not already using LabVIEW, whereas LE has no such limitation.

The second difference between LE and RoboLab is that the Control Lab Interface is in constant communication with the computer and therefore in LEGO Engineer, all structures and conditional statements are done using standard LabVIEW structures. For instance, Fig. 11 will wait for a car to approach the garage door before opening it. The angle indicator dictates how long to run the motor to open the door. Since the RCX is not in constant communication with the computer, we could not use standard LabVIEW structures in RoboLab. Therefore, although the modifier and stringing part are similar (Fig. 12 shows the same program as Fig. 8—only for the RCX), the structures are done entirely differently. Figure 13 shows again the same program only this time using structures. With the RCX, the user can have as many forks and loops as he/she wishes. One can also take advantage of the multitasking environment of the RCX to do two things at once. Further, one can use the on-board variables on the RCX to do real-time calculations. Figure 14 shows a sample program for an intelligent house. The first task simply checks the light outside and when it gets dark, turns on the outside lights. The second task measures the temperature in the room. If it starts to get too hot, it turns the fan on. The hotter it gets, the faster the fan spins. The red container (an on-board variable) can be used to convert the temperature reading into an appropriate blade speed for the fan. This program complexity is beyond the grasp of most elementary school students, but has been extensively used by middle school, high school, and college students.

THE SOFTWARE IN THE CLASSROOM

The elementary school

Together, LEGO and LabVIEW can change the nature of the learning process in the classroom, and the way in which K-12 students view tech-

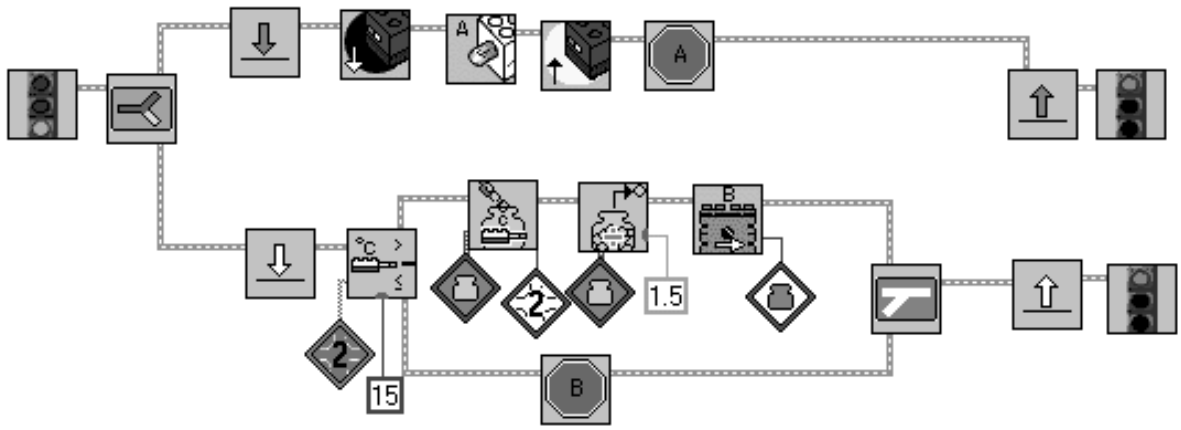

Fig. 14. The smart house.

nology and engineering. Kids (and adults) learn something by experiencing it. They learn about writing by writing their own stories and they learn about engineering and design by designing their own mechanical creations and computer programs. LEGO materials and the LabVIEW programming environment are the tools that allow them to do so. The true power of these tools is not simply to enhance the pre-existing curriculum, but actually to transform the learning environment into one of an inventor's workshop or engineering design firm. In the 'workshop' environment, students work on designing their own personally meaningful creations from their own ideas. In the 'design firm' environment, an entire class of students (with its designated field experts) need to work together to solve a complex design problem. In both environments, the latter of which will be discussed below, engineering isn't viewed as something that is purely competitive, or only 'for boys', since everyone is able to explore his/her own ideas about technology and bring personal expertise and interest to the group. We have on the Web over 50 pages of what students did and what they thought of it [5]. Below, we highlight two different learning environments: the classroom and an after-school program.

Lincoln Brooks School

In an effort to teach kindergartners about forces (including the concept of friction), we brought a few RCX bricks and a computer into a classroom. We first discussed forces by pushing each other and then pushing pillows on a carpet and then on a smooth floor. They then built Duplo cars that were pushed using RCX-based pushers. The pushers were simply an RCX with two motors in the back directly connected to the tires. By placing these pushers behind their vehicles, students quickly started to argue about pusher designs and where it was best to push. We then talked about why some designs could not be pushed on the carpet, where others could. This discussion spawned a number of design modifications and pretty quickly students decided that they wanted to change the speed and direction of their pusher and so were motivated to start programming. After showing them once how to work the Pilot 2 program, they had no problem making their cars move in all directions and speeds (although as fast as possible for as long as possible was the favorite). Because they could independently control the speed of each wheel, a number decided to make their car turn. We worked more closely with a few kindergartners, who quickly learned how to do diagram programming. One kindergartner made a little robotic car that would follow you around if you sped up, so would it. He simply had the light indicator brick controlling the speed of the motors, when the sensor no longer read the reflection off of the person, the car would go faster. In general, the kindergartners had little difficulty programming (the earliest we have worked with is 3-year-olds). The panel programming quickly taught them how to think logically and the basics of programming structure. Armed with this, they have little difficulty moving into the diagram programming environment. Interestingly, however, the younger students seem much more interested in the physical construction rather than the programming. If the program basically works, that is good enough. College students, on the other hand, will often spend far longer on the code than the construction with the theory that if

Fig. 15. The glass sorter.

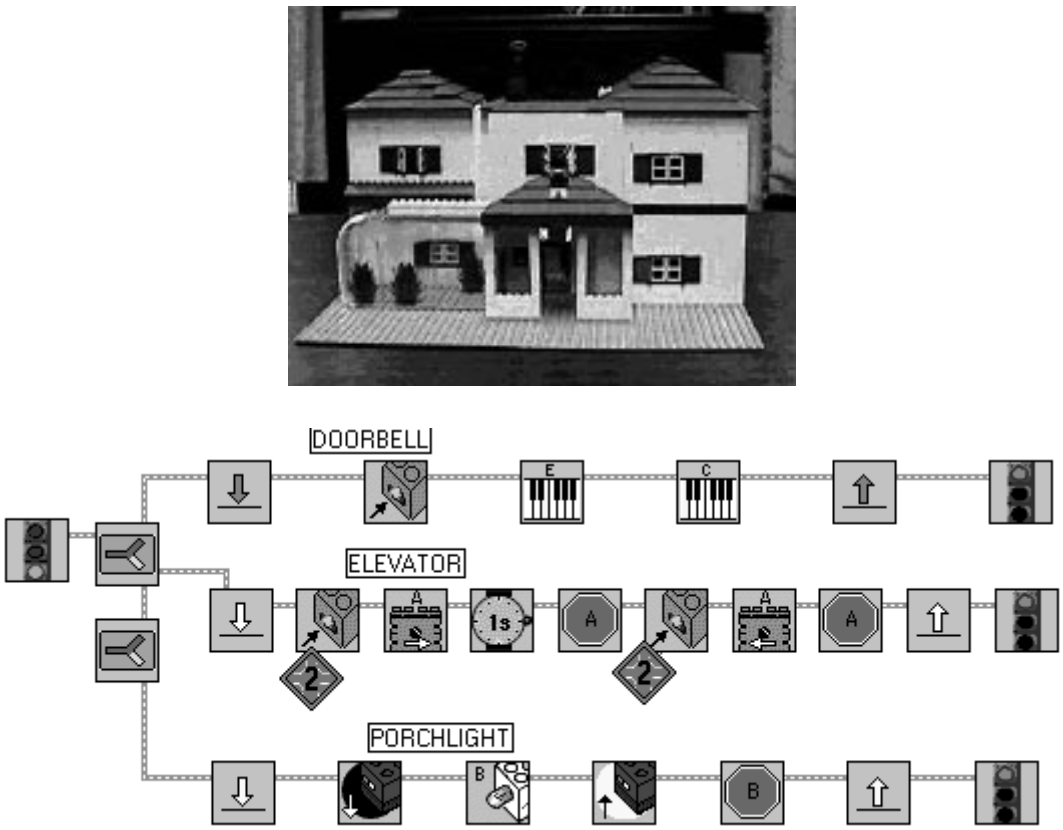

Fig. 16. The intelligent house.

the structure basically works, that was good enough.

In fourth grade, the science teacher decided to take an engineering approach to teach her class about recycling. Not only did the students read and discuss the recycling process, they collectively designed an entire recycling center out of LEGO beams, motors, axles, and other parts. One team of students decided to separate blue bottles (blue LEGO pieces) from other ones using the LEGO light sensor. Their design involved one motor running a conveyor belt, and a second motor connected to a rack and pinion device that knocked the blue LEGO pieces off the conveyor belt and into a bin. The conveyor belt and the LEGO Engineer program for it are shown in Fig. 15. For these fourth-graders, the unit on recycling was unforgettable. After the recycling center was built, they wrote stories about it and displayed them both for the rest of the school. The lesson was able to capture the complete attention of some students that were otherwise bored with the traditional style of learning. For the group that programmed the conveyor belt, the ability to have a command over technology—to be a designer and not just a user—was the most powerful lesson.

Fig. 17. Charlotte's Web.

The Paraclete Center

The Paraclete Center is an after-school center in South Boston whose main goal is to provide an academic working environment where students can become involved in a number of different projects with mentors. We challenged the kids to design an intelligent house (Fig. 16) as a true systems engineering project. During a brainstorming session, everyone talked together about what kinds of features the building might have: a porch light, a moving garage door, a doorbell, a ceiling fan, a security system, an elevator, etc. (top-down design). They had to make trade-offs as to which components to include based on the limited number of sensors (bottom-up design). Next, the students decided which was to be their area of expertise: programming, structures/electronics, or architecture/landscaping. These categories corresponded to the interests of the particular students involved.

These students ran into genuine systems engineering issues, such as when one group is

seemingly not able to begin on their design until they have information from another group. What really needs to happen in a situation like this, of course, is that both groups need to begin by making guesses on what the other group will do. The programming expert realised on his own that he must begin writing a program for the elevator before he even knew what it looked like, in order to save time. 'I'll just guess that he is just going to use one motor, and use a touch sensor for the button that controls it,' he said, and started to write the program.

Other examples of elementary school engineering

Teachers have used the LEGO bricks and our software in a number of innovative ways. Some second grade teachers have built models of Zuckerman's Farm from E. B. White's Charlottes Web (Fig. 17). Others have taught cartography though building a town. Some have built volcanoes and others space stations. First graders had 'slow car' races (whose goes the slowest) and marketed LEGO snowplows. All of the teachers have found that the students—almost universally—really enjoy the designing and building process. The LabVIEW component allows them to animate their product. First graders have made spiders that have flashing eyes and moving pincers, kindergartners programmed a bus to drive through their town, third graders automated their cars. Each time, the students viewed the programming as just a tool to make their construction do something. Thus the computer went from being the conventional focus of the lesson (how many schools have computer time—or the students 'go to computer'?), to where it should be—one of many tools to solve a problem.

In an effort to connect the home and the school, we offered a parents night last year to train parents in LEGO Engineer. Thirty parents came that night to build a small town. By the end, some had buildings with automatic lights, another had a swing set, and a third had a waste disposal system. In all cases, the parents, like the kids, got excited about their creations and learned a lot of engineering principles as their escalator went too fast or the door to the fire station did not raise high enough to allow the truck to pass under.

College engineering

Both LEGO Engineer and RoboLab have already become an integral part of the education of a mechanical engineer at Tufts University. The LEGO bricks and software provide a cheap and flexible workbench of tools from which the students can build something new and different every year. Student support is extremely high with students often staying late to add another turret onto their car. In particular, we have taken it into three main areas so far: entry-level engineering, experimentation, and senior design. We have even used RoboLab and the LEGO bricks as a mechanism to improve undergraduate advising (see reference [3] for details). At Tufts, we have a number of engineering courses that are designed for freshman engineers and liberal arts majors. These courses are meant to give the students a feeling for engineering, giving them a reason to take the two years of math and science that are about to follow [6]. To date we have offered two LEGO/LabVIEW-based courses for liberal arts students and first-year engineers, one with LEGO Engineer and one with RoboLab. The first, entitled 'The Way Things Work', introduces engineering to majors and non-majors through understanding how everyday objects work. Building an egg-beater out of LEGO bricks, for instance, teaches concepts of gearing, stability, and strength in construction. By the end of the semester, students were building new inventions, which ranged from a clock tower with a working clock to a gadget that automatically poured soda from a soda can. Again, LEGO Engineer allowed the students to actually make the pouring mechanism work. This class was very popular among the students as they felt that they got to actually learn some engineering early on in their college career.

In the other freshman course, we taught elementary robotics using RoboLab and the RCX. Here, students competed on a weekly basis with competitions varying from navigating a maze to laser tag—where every robot tries to hit other robots with an infrared pulse—without getting hit themselves. In the final competition, students had to navigate a maze, find a candle, and snuff it out. Three of the groups succeeded in doing that and submitted their robots to a national fire fighting competition [3]. Through the competitions, students learned a number of lessons, with probably the most important being that simplicity wins every time. This applied to both the robot design and the RoboLab code written to control the robot. Where some had programs that used almost all of the memory of the RCX, the winners were usually the ones whose codes were simple and elegant. A robot does not need to be intelligent to navigate a maze—it just needs to be determined. At this level, all students were using the highest

Fig. 18. CNC milling machine.

programming level (Inventor 4) and often pushed the sensors and the RCX to their limits. Again, student evaluations were unusually positive (4.9/5.0 in their final evaluations).

All mechanical engineering juniors are required to take a course in experimentation methods. This course has used LEGO Engineer for four years now with great success. LEGO Engineer provides a fun and interesting way to getting the students to learn LabVIEW programming. This course conventionally begins with students building and programming a joystick-controlled car. They race these cars through mazes and those with the best code and the quickest figures win. The competition pushes students to get a better understanding of the programming language. After racing cars, they move on to learning how sensors work and how to connect them to the computer. After running a number of smaller experiments, meant mainly to teach writing and presentation skills, they then end with a final project. These projects vary from year to year with last year being an experiment to determine the force required to pull two LEGO bricks apart. They had to design, build, and document an experiment that could take this measurement repeatably and accurately. They succeeded using pneumatic power to pull and a pressure transducer to measure the required pressure. This course is more fully outlined in reference [2]. This year, the same course was taught, only with RoboLab and the RCX. Instead of joystick cars, they built a robot that would drive around on top of a table without falling off. Instead of pulling two bricks apart, they are building a fully automated TANG™ dispenser. Their program brings the water up to a desired temperature, pours a designated weight of TANG crystals into a paper cup and then adds a certain amount of the warm water to the cup. The final mixture is then weighed and sent out to the operator. Through this manufacturing station (a mixture of LEGO bricks and wood), they learned about measurement uncertainty, statistics, sampling theory and sensor design. In fact, the scale that weighs the final mixture is a strain-gage system they made themselves. They also wrote reports on each portion of the process, analytically modelled the process, and gave presentations of their findings.

Finally, we have also used LEGO Engineer in a senior design project. As previously mentioned, five seniors built a three degrees of freedom computer-controlled milling machine. The mill is designed to cut boat hulls out of balsa wood (Fig. 18). By restricting the students to LEGO bricks, they were forced to think thoroughly through their design: LEGO bricks do not absorb vibrations as well as cast iron; there are only certain diameter gears available; and how it is difficult to get millimetre positioning accuracy with building blocks rather than welded steel. On the other hand, the bricks gave them the advantage that they could produce something that was not just a paper design in a semester timeframe. By prototyping their milling machine, they were able to test out a number of different designs and clean up many aspects of their final design. Their final design included over $1,000 worth of LEGO bricks and over 15 subroutines in the code.

CONCLUSIONS AND FUTURE DIRECTIONS

LEGO Engineer and RoboLab are two programming environments that extend the capabilities of LabVIEW to allow kindergartners and college graduate students to program side-by-side. We have used the environment successfully with all ages and have found that the graphical format of LabVIEW allows us to concentrate much more on the science or engineering being taught and a lot less on checking syntax of the programming. The computer becomes another tool rather than the central focus of the project. The software and the hardware have a very low entry level but a very high ceiling. Kindergartners are using the same tools to build rocket ships or cars as graduate students who have used the RCX to measure accelerations on NASA's zero-gravity KC-135, or to measure lift and drag on aircraft in a small wind tunnel. The engineering associated with the LEGO bricks, from building a town to measuring acceleration, has received almost unanimous support from the over 4000 students and 100 teachers that have used it.

We are currently creating software that is designed specifically for using the RCX in scientific laboratory experiments for middle school and high school. Future versions of LEGO Engineer will be incorporating the latest features of LabVIEW—such as the ability to run programs over the Internet. With help from corporate and government sponsors and local engineers and scientists, we hope to bring these technological tools to more students so that the average high school graduate has a firm understanding of engineering design and control.

REFERENCES

1. B. Erwin, M. Cyr, J. Osborne, C. Rogers, Middle school engineering with LEGO and LabVIEW, *Proc. NIWeek*, Austin TX, (1998).
2. M. Cyr, V. Miragila, T. Nocera, C. Rogers, A low-cost, innovative methodology for teaching engineering through experimentation, *J. Eng. Educ.*, **86**, 2 (1997).
3. S. McNamara, M. Cyr, C. B. Rogers, B. Bratzel, LEGO bricks sculptures and robotics in education, *ASEE Proc.*, (1999).

4. URL http://www.lego.com/dacta/robolab
5. URL http://ldaps.ivv.nasa.gov
6. Martha Cyr, C. B. Rogers, *Enhancing Education with LEGO bricks and Paperclips*, FEDSM98–5137 (1998).

Ben Erwin is Curriculum Coordinator at the Center for Engineering Educational Outreach at Tufts University. He holds a BS in Aerospace Engineering from MIT and an MAT in Physics Education from Tufts University. He teaches middle school technology education classes in Weston, MA. His professional interests are in teaching engineering and design to young students, and using systems engineering as a model to engage students in complex design projects. His particular LEGO claim-to-fame is the automatic bubble-blowing machine.

Martha Cyr is the Director of the Center for Engineering Educational Outreach at Tufts University (http://www.ceeo.tufts.edu). She received a Ph.D. in Mechanical Engineering from Worcester Polytechnic Institute. In addition to running numerous workshops for K-12 teachers and students, she teaches thermodynamics and a class called Children, Technology, and Society, which combines traditional STS curriculum with student teaching at Tufts.

Chris Rogers is a professor of Mechanical Engineering at Tufts University. He received his Ph.D. in Mechanical Engineering from Stanford University. He teaches numerous courses that allow students to learn about engineering through experience—from building robots to musical instruments. He also visits his elementary school once a month to play tag at recess and build in different classes. He recently received the Carnegie Massachusetts Professor of the Year. His office is full of LEGO constructions and his laboratory is completely LabVIEW-driven (even the accounting is a virtual instrument).