ipre

institute for personal robots in education
2007 annual report

The People of IPRE

Georgia Tech:

- Associate Professor **Tucker Balch**, IPRE Director

- Professor **Mark Guzdial**, Co-PI for Curricula

- **Keith O'Hara**, Ph.D. Student, Instructor & Robot Hardware and Software Design

- **Jay Summet**, Postdoctoral Fellow, Curriculum Development & Evaluation

- **Monica Sweat**, Instructor, Curriculum Evaluation

- **Daniel Walker**, Research Scientist, Lead Hardware Design Engineer

- **Ben Axelrod**, IPRE Fellow 2006-2007, M.S. Student

- **Gaurav Gupta**, IPRE Fellow 2007-2008, M.S. Student

Bryn Mawr College:

- Associate Professor **Douglas Blank**, IPRE Co-Director, Lead Software Design

- Professor **Deepak Kumar**, Co-PI for Curricula

- **Ashley Gavin**, IPRE Fellow, 2006-2007

- **Mansi Gupta**, IPRE Fellow, Summer 2007

- **Natasha Eilbert**, IPRE Fellow, 2007 - 2008

Microsoft Research :

- **Dr. Stewart Tansley**, Senior Research Program Manager, Microsoft Research

- **Jared Jackson**, Research Software Development Engineer, Microsoft Research

institute for personal robots in education

Advisory Board

Associate Professor Zach Dodds
Harvey Mudd College

Professor Mark Guzdial
Georgia Institute of Technology

Professor Deepak Kumar
Bryn Mawr College

Assistant Professor Fred Martin
University of Massachusetts Lowell

Professor David Miller
University of Oklahoma

Associate Professor Illah Nourbakhsh
Carnegie Mellon University

Assistant Professor Elizabeth Sklar
Brooklyn College, CUNY

Dr. Stewart Tansley
Microsoft Research

Professor Manuela Veloso
Carnegie Mellon University

Table of Contents

About IPRE

The Institute for Personal Robots in Education (IPRE) was created to make education more fun and effective through the use of personal robots. Robots can provide a tangible and personal means to engage a student in engineering, science and math education.

IPRE's mission is broad: to employ robots in education at all levels from middle school to graduate school. Our initial target, however, is introductory undergraduate computer science. IPRE's investigators expect to show that by empowering every student with their own personal robot, purchased with the class textbook, we can improve retention in and attraction of students to computer science.

Our effort includes 5 main components:

1. **Curriculum:** We will develop curricula that can be quickly and easily adopted and revised by others;
2. **Community:** We will foster an open community for sharing ideas and helping others adopt our curricula and technologies;
3. **Assessment:** We must assess the impact and effectiveness of the tools we develop and deploy, sharing the results openly with the academic community;
4. **Hardware:** We will develop and deploy very low cost robots for teachers and students, in order to make our approach scalable, robust for classroom and personal use, with engaging features yet simple and approachable for all;
5. **Software:** We will develop an easily accessible software environment for students to learn program their robots and teachers to teach the curriculum.

An important component of our concept is that the robots for these courses must be reliable and inexpensive so that every student can have one, and trust that the robot accurately executes the student's programming. IPRE will keep the barrier to entry low for those professors interested in trying something new, with little or no specialist knowledge of robotics or TA support. We believe robots are an attractive but additional way to introduce more students to programming in these challenging times for computer science educators, and need to be made easy to adopt and deploy compared with other contemporary approaches.

IPRE is hosted at Georgia Tech with Bryn Mawr College, and initially supported by seed funds for 3 years from Microsoft Research and the schools themselves. As well as developing the technology to be used in classrooms, leveraging the best available commercial and non-commercial components as appropriate (including MSRS, Microsoft Robotics Studio), IPRE will rigorously assess the effectiveness of the approach. The results of this research will be widely published through regular academic channels.

institute for personal robots in education

Executive Summary

We've had a spectacular year of successes, including:

- Creation of a curriculum and on-line text for undergraduate Introductory Computer Science (CS-1) using personal robots. http://wiki.roboteducation.org

- Release of Myro, an easy to use Python-based software platform for programming personal robots specifically in a learning context

- Deployment of an inexpensive personal robot for education in courses at Georgia Tech and Bryn Mawr College at about the price of a textbook

- Personal robot-based CS-1 pilot courses taught in "production" classes at Georgia Tech and Bryn Mawr College

- Rigorous assessment tools developed and deployed, and initial results recorded and submitted for publication

- Invitations to present our ideas at leading CS education conferences and workshops in the US and beyond

- Coverage of our work in top-tier media including The Chronicle of Higher Education, Computerworld, The Associated Press, Industry Week, the Red Herring, and the Atlanta Journal Constitution

- IPRE received an award for Technical Innovation from the Association for the Advancement of Artificial Intelligence

Re-Invigorating Introductory Undergraduate Computer Science

Undergraduate computer science is initially IPRE's top focus. Computer science has seemingly lost its appeal to many of today's students, and evidence suggests that personal robots perhaps can help them find it again. Even as computing has permeated every aspect of our lives, computer science as a field of study is often seen as disconnected from these same lives. To reestablish the connection, the Institute for Personal Robots in Education is developing a personal robot, software, and curricula to help teach introductory computing courses.

Our vision is that the text for an introductory course would come shrink-wrapped with a ready-to-run personal robot in the same price range as current CS-1 texts. Having an artifact — in this case a robot— provides intrinsic motivation to both the instructor and the student to explore the science and engineering behind it, and the tasks to which it can be applied. Externalizing software into a physical form can also be a way to encourage social interaction, as there is a tangible entity a group can interact with. Students engage in learning computing for reasons that are very different from those traditionally identified: such as fun, curiosity, diverse range of applications that portray computing as a helpful discipline, and to show off to family and friends. A project of such undertaking requires careful attention to all aspects of the course design: the robot, software, course materials, and the lab environment.

The core aspects of our approach include:

A personal robot

Every student gets her/his own personal robot. Our vision is
that students registered in a CS1 course taking this approach
will go to the bookstore and purchase his/her own personal
robot, just like they would purchase a text for the course (See
Figure 1). Additionally, since the student owns the robot
it can be personalized and used in future classes, or later traded
as with traditional textbooks. The class text can be optionally
bundled with the robot or made available on-line, according
to the professor's preference.

Let the needs of the curriculum drive the design of the robot

The design of the personal robot is motivated by the
requirements of our new curriculum and is an outcome of
feedback obtained from students in our pilot course offerings.
We place the robot firmly in the role of a motivational context – teaching robotics is not our intent; teaching
computing concepts by illustrating them in a robotic context, is the approach.

Use tools that are easy to use, scale with experience

We want the students to use the tools (computer, programming language, editing environment, etc.) to be
such that they are not designed specifically (and only) for use in CS1. We want the entire programming
environment to be pedagogically scalable. This way, concepts acquired in the new CS1 easily carry over
into more advanced computing situations without the need to change the programming environment. Later, we

believe this notion could be extended into earlier years, where the same basic environment could be used in pre-university classes, scaling into and through university years.

Robot as a peripheral

Rather than the student directly programming the robot at the robot's level (often conventionally a primitive microcontroller), the student uses the full power of a modern personal computer for development and debugging. In this mode, the desktop or notebook computer commands and queries the robot over a wireless tether. In a sense, the robot's "brain" is virtualized across he wireless link, providing a subtle learning point in itself to the student about distributed systems programming from an early stage.

Create an accessible, engaging environment for a new, diverse population of students

It is well acknowledged today that the introductory computer science curriculum is broken and is in need of a major overhaul. With the wide acknowledgement also comes a wide range of proposed 'solutions'. Ours is one of them. We are taking the issue of accessibility to a wider population as our primary goal. In addition, we have been able to introduce many sophisticated topics using a simple environment. Often, really engaging CS topics (such as AI and robotics) are reserved for only advanced students, and many novices never stick with the field long enough to study these. We hope to allow introductory students to experience some of the excitement such topics bring to computing. On the other hand, we are also mindful of the adverse affects the use of certain technology, and pedagogical examples can have on people from different gender and backgrounds. Our curricular materials attempt to address these issues as well.

Computer Science programming

While programming is central to our approach to CS1 we are also conscious of avoiding the misperception that programming is all there is to computer science. Students from the new CS1 should come away with a solid understanding of the broad scope of computing, and the proper role of programming within it.

Make computing a social activity

There is an explicit attempt in our approach to make computing a social activity. By this we mean that we will strive, in our curriculum, to make every aspect of the learning process a collaborative and social activity. Students learn from each other and by working on their robots in their own environments (dorm hallways, dining halls, study spaces, labs) and interacting with others in meaningful ways. Such activities also serve as a type of "viral marketing" to build greater awareness of CS to a much broader segment of the student population across campuses.

Make computing a medium for creativity

Creativity is central to both software design and robot design and we intend to include several creative aspects into the curriculum. Examples include exercises that demonstrate robot behaviors like dances, choreographed movements, music and song generation, movie making, game playing, robot application design, etc. Most exercises will be open ended (i.e. correctness of the output of a program is not determined by a limited set of outputs) and encourage students to experiment, play, and be creative.

Performances vs. competitions

All robot exercises in the course include demonstrations. However, the demonstrations are depicted and evaluated as performances and not as competitions among peers. We have found that competitions tend to attract only a small population of the student body and serve to deter many students away. However, a non-competitive, collaborative, and social environment encourages learning and motivates students to strive for higher goals. If an individual professor chooses to take a competitive approach, many of our materials and technologies are still nonetheless appropriate and usable, it is just not our recommended approach.
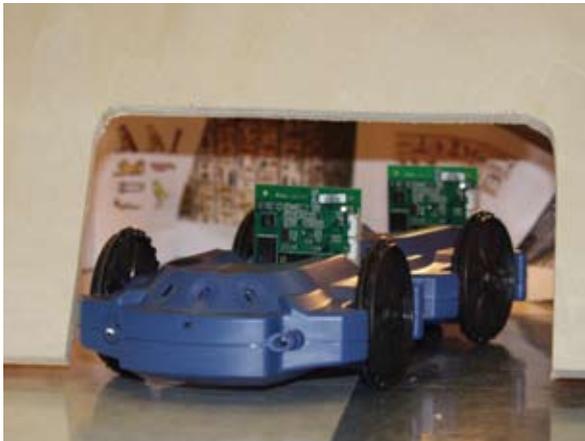
A Personal Robot-based Curriculum for CS-1

The curriculum for a CS1 course based on personal robots has been under active development. Planned curricular materials include a CS1 text, Myro (software) reference materials (including API documentation), and extensive but easy-to-manage materials for instructors. Most of the effort in this first year was spent on the design of the core course text. With the ongoing development of Myro, the API reference is also developing simultaneously. The most important aspect of our curriculum design is to embed the personal robot into a CS1 course in a way that seems most natural and inviting for students. This requires a fresh approach to the overall syllabus of CS1 and rethinking of the traditional sequence of topics presented in CS1. Our current incarnation of the text reflects this approach and contains the following topics:

Working title of text: Learning Computing with Robots
Chapter Outline

1. The World of Robots
This chapter introduces the world of robots to students. It introduces the Scribbler robot and Myro software. Students are led through a session where they can connect to the robot, give it a name of their choosing, and move the robot around with a joystick – the initial "out of box" experience for the student is therefore way beyond traditional "hello world".

2. Robots: Personal or Otherwise
This chapter introduces basic features of mobile robots: movement. Students are also introduced to the Python language, its IDE, and how Myro abstracts robot movements into simple Python commands (like forward, backward, stop, etc.). Students also learn how to write simple Python commands that implement robot moving behaviors, and in the process also learn the basic units of a program: the function. Students write programs that contain functions which are basically abstract behaviors composed using basic Myro movement functions.

3. Building Brains

This chapter introduces some basic programming concepts: the notion of a program (introduced as the brain) of the robot, the use of names in a program to represent things (values, parameters, and functions). The chapter also introduces simple iteration so that repetitive robot motions can be easily programmed.

4. Sensing the World

This chapter introduces the sensory interface to a robot. It also introduces other output and input modalities (like choice dialog boxes and speech as an output behavior). Students familiarize themselves with various sensors of the robot and the kinds of values they return.

5. Making Decisions

This chapter introduces decision making (if-statements) so that combined with sensory information, students can write programs to control robots that can carry out smarter behaviors. That is, they learn to program using the reactive control paradigm. The chapter also includes some traditional computational examples so that students realize that they are learning general problem-solving and computational techniques (and not just robot programming). That is, the same concepts learned to control robots are used in standard computing applications. Students also learn the math library and other arithmetic expressions.

6. Behaviors

This chapter introduces the idea of creating/programming robot behaviors using the Braitenberg paradigm. That is, the same behaviors that can be programmed using decision making structures can be accomplished by means of simple mathematical transformations. Students learn to build/program several Braitenberg-style robot behaviors and indulge into a little bit of synthetic psychology.

7. Control Paradigms

This chapter formally introduces the two kinds of robot control paradigms they have been using. It then discusses the advantages and shortcomings of these paradigms leading up to the introduction of behavior-based control as a general paradigm for building robust robot behaviors. The programs use a single-threaded model of subsumption architectures which can be implemented in Python using very simple techniques. The only new programming concept

introduced in this chapter is that of using functions as first-class objects in a program.

8. Making Music
This chapter explores sound and music. Some basic music theory is introduced and Myro commands for playing sounds and music are introduced. Students are encouraged to design robot behaviors that are choreographed with music.

9. Communication
This chapter presents an IM (instant message) like chat interface developed in Myro. Using this, robots can communicate with each other using instant messaging as well as by sending each other e-mail.

10. Computing & Computation
The central ideas introduced in this chapter are the notions of an algorithm and that of problem solving in the programming process, the role of a programming language, and the limits of computation. While students have been doing computation all along, this is the first time we bring their experience to a more formal notion of an algorithm and program development. Several examples, robotic and well as non-robotic, are presented.

11. Artificial Intelligence
While students have been working with AI ideas (robot control, etc.) this is where we introduce AI as a study of intelligence (robots being the embodiment of intelligence). We introduce examples from natural language understanding (natural language interaction with a robot), smart game playing (using case-based reasoning) and machine learning (using Myro's neural network modules).

12. Applications of Robots
This chapter presents the current state-of-the-art of robot applications and looks into the future. Students will be

encouraged to think about applications of robots in several domains from a design perspective. For example, design an entertainment robotic application, or design a robotic lawn mower, or other household appliance.

13. Appendix
Resources, bibliography, references, and other items

As mentioned above, the curriculum, text, and the Myro API are under constant revision while our research project is in its early stages. At the end of the first year, we believe we have made great strides in the development of this approach. We now have a concrete initial curriculum and are now engaged in refining and extending these materials based on actual first-hand experiences reported by students, teaching assistants, and faculty.

Future Directions for the Curriculum

Besides the continued development on all fronts (hardware, software, curriculum), we envision several specific directions in the coming year:

- Make the course materials independent of the robot hardware, to allow broader adoption with alternative hardware (e.g. that professors may already have in place)

- Evolve the Myro API to be more abstract and overarching

- Revise the curriculum based on added functionality in Myro and enhanced hardware, such as a camera

- Completing the text for release to the wider community

- Completing the support materials to support wider evaluation and adoption

Software for Personal Robots in Education

The software developed by the IPRE is called Myro. It is designed to directly support the goals of the curriculum by creating an intuitive, easy-to-learn, easy-to-teach, and yet powerful interface connecting student and robot, lightly guided by their professor.

Myro provides a standard API to underlying software and robot hardware (usually invisible to the students). We presently support two Myro implementations: One based on Microsoft Robotics Studio (MSRS), and another based on CPython. The CPython implementation is portable across platforms including Windows, Mac and Linux. The MSRS implementation currently runs only on Windows platforms, but it supports a wide variety of programming languages and robot hardware. We also hope that MSRS will be supported on non-Windows platforms in the future.
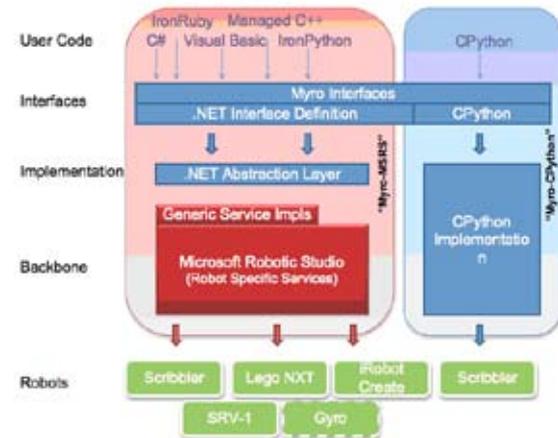
Myro is being developed in two phases: a pilot program, and a final implementation. The pilot program was written in Python and tested on Macintosh, Linux and Windows operating systems. Python is a high-level interactive scripting language that itself exhibits many of our pedagogical goals, and is rapidly gaining popularity as a first programming language across the academic community.

During the pilot program we have developed a set of functions, objects, and nomenclature that attempts to allow the student to quickly pick up the language and syntax, and be able to jump to the heart of issues in computation. Python is open source software, written in the C programming language. Myro is also developed in the open with feedback from the academic community, and with open source licenses.

The Myro pilot application-programming interface (API) covers many topics not typically available to introductory students of computing. Myro covers the following topics:

- Movement controls
- Sensor readings
- Multimedia and Image Processing
- Webpage Design
- Communication and Instant Messaging
- Music and Tone Generation
- Text-to-speech
- Artificial Intelligence

These topics go hand-in-hand with the goals of the curriculum. To get a sense of the style and scope of programs that can be written in Myro, here is a sample program:

```
from myro import *
initialize()
brightness = []
for side in range(4):
        forward(1, 0.5)
        turnLeft(1, 0.3)
        brightness.append( getBrightness(1) )
picture = takePicture()
for pixel in getPixels(picture):
        if pixel == blue: setColor(pixel, pink)
sendPicture(picture, "mypicture", "PassW0rd")
speak("I have completed my mission. Reward me!")
```

This is a real Myro program – nothing has been left out for simplification. The obvious security hole in the sendPicture() method is usable as a trustworthy computing talking point with the students. It loads the Myro libraries, connects to the robot wirelessly, traverses a square while sampling the brightness in four directions, takes a color picture, does some image processing, and sends the picture to the robot's webpage for viewing. Finally, it speaks its completion message. As can be seen, Myro allows students to cover a wide array of topics from a consistent and focused interface.

By using Python, students can develop their programs interactively at first, by entering their code line-by-line and inspecting and manipulating results. This is due to Python's dynamic nature. The Myro pilot is completely written in pure Python. This makes Myro very portable across operating environments. However, there are some downsides to the pilot. For example, because Python is interpreted, there are certain functions that would be too slow to attempt in real-time. Also, even though Python is an excellent choice for a first-year language, not every instructor could use it. For these reasons, and others, the final Myro implementation will allow a variety of languages.

The final implementation of Myro will likely be a re-implementation of the Myro pilot functionality in Microsoft .NET using C#. We will then use the arsenal of languages available in .NET which can now access C# functions. This includes Microsoft's new IronPython and IronRuby, among other languages like Visual Basic, C++, and, of course, C# itself. By utilizing Mono, an open source implementation of .NET, we believe that Myro will remain cross-platform, but will also be language independent. In addition, Myro will take advantage of Microsoft's new Robotics Studio providing broad access to robot hardware, an excellent high-fidelity simulator, and other features[1]. It is planned that the Myro programs from the pilot will be able to run, unchanged, in the final implementation of IronPython running on .NET.

Furthermore, advanced users will be able to write code in the lower-level C# system so that Myro programs will be as fast as any. This will allow to the software platform to be extended for use by expert users in other courses or in research, for example.

---

1 Use of Microsoft Robotics Studio's native distributed and concurrent service-oriented architecture as a beginner programming environment will be evaluated.

Assessment: Is Our Approach Effective?

We undertook a pilot evaluation of our offering of the robotics CS1 at both Bryn Mawr and at Georgia Tech during the Spring 2007 semester. This was a particularly opportune time to study the Georgia Tech class because the same lecturer was in both our robotics and non-robotics introductory course sections. We studied both sections using the same survey instruments.

Our evaluation had three stages:

1. We conducted a midterm survey to gather open-ended comments on what students thought about the classes.

2. We used the survey comments to develop an interview script that we used with three students in the robotics section of the course at Georgia Tech.

3. We analyzed the interview scripts to identify themes—opinions or attitudes that we wanted to explore in the course. We constructed a final survey which was completed by participants on both campuses and in both versions of the course.

A significant difference between Bryn Mawr and Georgia Tech offerings of the course at the time of the initial survey is that the Bryn Mawr students were more likely to be interested in computer science. During the Spring semester, the Georgia Tech CS1 course tends to be populated by non-computing majors. That influenced the interview and final survey responses.

Table 1 summarizes the Bryn Mawr College (BMC) and Georgia Tech (GT) student responses to the final survey. On these survey statements, a "1" indicates strong agreement through to a "5" which indicates strong disagreement. For most statements, BMC and GT students responded similarly.

- All students enjoyed using the robot and found it easy to get working. Students were comfortable working with the robot. Both groups of students found that personalizing the robot improved the course overall for the students.

- All students found the homework challenging. BMC students were significantly more likely to report spending extra time on homework because "it was cool."

- Students did want to see more features in the robot and felt that it would have enhanced the class for them.

- The demographic differences in the two schools influenced the final survey on several statements. GT students were less likely to see the content of the course as being important, more likely to find robots scary, less likely to tell other people about the robots, and less likely to want to work with robots more. GT students thought that the class would be easier without the robot—but not "better." BMC students did not think that the robot was scary and reported being even more excited about computer science after the robotics course than before.

There were very few differences between the robotics and non-robotics sections at Georgia Tech (Table 2). The only significant difference (at p=.066) is that the non-robotics section was more likely to talk to friends about the course.

We were pleased to note that there were also very few differences between women and men at GT (Table 3). Women reported finding maintenance of the robot slightly more of a problem. Men complained that the robot was under-featured. Women were more interested in seeing and using a robot with a real use, like for science exploration.

Table 1: Comparing Bryn Mawr College (BMC) and Georgia Tech (GT) responses to final survey

Statement	BMC Average	GT Average	Chi-Square (if p<0.10)
11. I enjoyed using the robot in this class	1.59	2.83	p=.011
13. It was easy to get the robot working from my laptop.	2.59	2.75	
14. I have imagined the robot when working through unrelated code in my head.	3.59	3.75	
15. I have found the homework challenging.	2.27	2	
19. There was at least one homework that I spent extra time on because I thought it was cool.	2.27	3.0	p=.045
20. What I learned in this class is important to my future career.	2.91	3.25	p=.006
21. What I learned in this class is important for me in my daily life.	3.09	3.58	p=.052
24. I am more excited about computer science now than I was before this class.	2.27	3.25	
26. I would like to take more computer science.	2.72	3.42	
27. I think that some people who take this class will find the robot scary.	4	3.08	p=.066
28. I liked telling other people that I was working with robots.	1.77	2.58	p=.005
30. Most people will prefer not to work with the robot in this class.	4.14	3.25	p=.040
31. I am comfortable working with my robot.	2.18	2.42	
32. Using the robot was frustrating.	2.82	3	
36. I would like to work with robots more.	2.5	3.25	p=.054
37. If the robot used in this class did more, I'd play with it more outside of homework.	2.32	2.58	
38. If the robot used in this class had more features, I would have enjoyed using it more	2.13	2.42	

Statement	BMC Average	GT Average	Chi-Square (if p<0.10)
39. Personalizing my robot made the course better for me.	2.54	2.75	
42. I would like to work more with robots in the future.	2.86	2.75	
44. The class would be easier without the robot.	4	2.92	p=.039
45. The class would be better without the robot.	4.19	3.42	

Table 2: Comparing Georgia Tech Robotic and Non-Robotic Sections

Statements	Robot Section	Non-Robot Section	Chi-Square (if p<0.10)
10. I discuss difficult assignments and/or detailed lectures with friends	3.17	2.75	p=.066

Table 3: Comparing Georgia Tech Female and Male Students

Statements	Male Students	Female Students	Chi-Square (if p<0.10)
17. Maintaining the robot has not been a problem	1.63	1.5	p=.072
33. Our current robot doesn't do much	2.5	3	p=.050
50. I would have liked to see and maybe use a robot designed for use in exploration by scientists in this class.	2.63	2.25	p=.080

Future Evaluation

We are undertaking a further evaluation of the course at both Georgia Tech and Bryn Mawr during the Fall 2007 semester. We anticipate some changes in attitudes as we improve the course through iterations and as the robot becomes more powerful (an issue explicitly mentioned in the surveys). We also expect better attitudes toward computer science, since the GT Fall students will include CS majors.

We are extending the evaluation in a couple directions. We are explicitly exploring known predictors of computing ability in our early survey. We are planning to have some measures of introductory computing learning on the final survey that we can use to get some benchmark of learning. We will based these measures on the work of Allison Tew and Mark Guzdial at Georgia Tech on measuring CS1 learning in a language independent manner.

Last but not least, we expect to see the materials evaluated at a number of institutions beyond Georgia Tech and Bryn Mawr during the coming year. We will offer our assessment instruments so that evaluations can be readily compared across the experiences of different schools, enabling the pooling of our research results to share with the academic community at large. Further details on this will be made public in the coming months.

Hardware: A Personal Robot for CS Education

There is a lack of truly low-cost robots (~$100) available which satisfy the functional requirements of the challenging educational environment. This environment demands a robust device that balances functionality with simplicity, reliability and security. There are many factors to consider. In reviewing the available options, IPRE is investigating the construction of a purpose-built robot hardware platform for the CS classroom called "Gyro". This work is ongoing. In the meantime, we came across the Parallax Scribbler robot, originally designed for children to explore robotics. We soon found, however, that this robot, if appropriately enhanced, meets most of the needs of a good CS classroom robot.

ROBOT UPGRADE MODULE INSTALLED ON A PARALLAX SCRIBBLER ROBOT

institute for personal robots in education

The robot upgrade module is an adjunct device by which low cost robots are reprogrammed to support Microsoft Robotics Studio (MSRS) and expanded to provide the capability that instructional lessons depend upon. The specific areas of the robot which are upgraded include computation, sensors and wireless communication. By this mechanism, the basic off-the-shelf Scribbler robot from Parallax, Inc., can be readily adapted to the CS classroom through the simple act of plugging in our upgrade module (see figure).

Operation of the robot upgrade module is fully automatic. Once attached to the scribbler robot the module uploads the MSRS device server into the scribbler's internal program memory. The module then provides a wireless link between the student's computer and the scribbler robot.
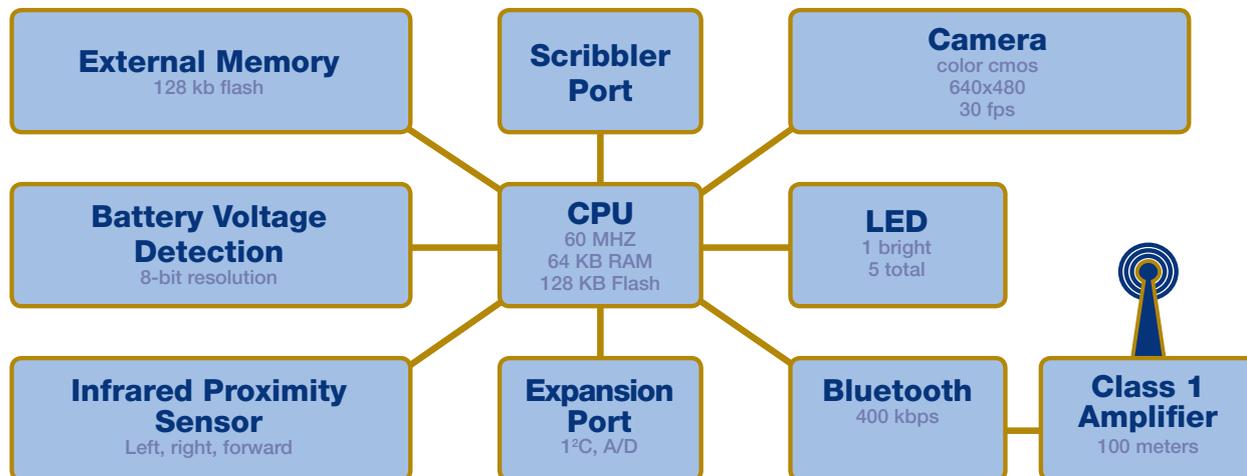
Most contemporary small mobile robots include a complex onboard computer which increases the price of the robot and becomes obsolete before the rest of the robot. An obsolete computer renders the entire robot much less useful, even though there is still value in the robot's mechanical systems. Our robot upgrade module bypasses the robot's onboard computer and controls the robot directly from a desktop computer. By leveraging the power of desktop computers and MSRS, our module allows for the use of extremely cheap base robot platforms and extends their utility and lifetime significantly.

A robot must satisfy a minimum set of capabilities in order to be useful as an educational tool. A rich suite of sensors provides an engaging input stream for student written programs. Wireless control increases the fun factor and helps maintain student interest.

Our robot upgrade module functions as a hardware component of MSRS and removes barriers to adoption for the software for small mobile robots that cannot natively run MSRS onboard (which is most of them, since natively running MSRS requires either full Windows or Windows CE on the robot today).

Development of the module centered on reducing its cost. To provide the desired capabilities within the cost constraint required minimizing the number of components and eliminating any pre-engineered modules. The camera and Bluetooth radio are available as prebuilt modules which ease the development effort but incur a cost penalty. For our robot upgrade module each subsystem is designed from scratch and uses the lowest-level chipsets available.

External Memory	Scribbler	Camera
128 kb flash	Port	color cmos
		640x480
		30 fps

Battery Voltage	CPU	LED
Detection	60 MHZ	1 bright
8-bit resolution	64 KB RAM	5 total
	128 KB Flash	

Infrared Proximity	Expansion	Bluetooth	Class 1
Sensor	Port	400 kbps	Amplifier
Left, right, forward	I^2C, A/D		100 meters

MAJOR COMPONENTS OF THE ROBOT UPGRADE MODULE

institute for personal robots in education

Publications and Presentations

- SIGCSE 2007 – Presentation by Stewart Tansley, Tucker Balch, Douglas Blank, Deepak Kumar, and Mark Guzdial at the Microsoft Vendor Session on the launch of IPRE.

- AAAI Spring Symposium Series, March 2007, Stanford, CA – Paper titled, Advanced Robotics Projects for Undergraduate Students, by Douglas Blank, Deepak Kumar, James Marshall, and Lisa Meeden. Presented at the Symposium on Robots and Robot Venues: Resources for AI Education.

- Deepak Kumar and Mark Guzdial participated in a panel on Computing Education at the New Face of Computing Symposium, GeorgiaTech, March 2007.

- Deepak Kumar gave an invited talk titled, Rethinking Computer Science Education, at the University at Buffalo's 40th Anniversary Symposium, Buffalo, NY, April 2007.

- Deepak Kumar participated in a panel on Approaches to CS1, at the Microsoft Faculty Summit, Seattle, WA, July 2007.

- Stewart Tansley, Tucker Balch, and Deepak Kumar gave a presentation titled, IPRE – One Year On, at the Microsoft Faculty Summit, Seattle, WA, July, 2007. Also gave a demo at the DemoFest at the same event.

- Stewart Tansley presented Microsoft Robotics Studio as a Robot Education Platform, Workshop on Research in Robots for Education as part of the Robotics: Science and Systems (RSS) conference, June 2007.

- Douglas Blank, Robots make computer science personal, Communications of the ACM, Volume 49, Issue 12, December 2006, Pages: 25 – 27.

- Doug Blank gave invited talks regarding IPRE at the Microsoft Latin America Summit in Chile, the University of Memphis, and at Sarah Lawrence College.

- Doug Blank presented Personal Robots for CS1, Workshop on Research in Robots for Education as part of the Robotics: Science and Systems (RSS) conference, June 2007.

Additional sources: http://myro.roboteducation.org/robobiblio/

Visibility in the Media

- ZDNet.com, 31 January 2007: "Georgia Tech, Microsoft bring tabletop robots to CS program."

- Industry Week, 1 February 2007: "Robots, Computers And Education: Students assigned their own personal robots" by John Teresko.

- San Jose Mercury News, 31 January 2007: "Robots not just for geeks" by Dean Takahashi

- USA Today, MSNBC.com, Associated Press, & others, 28 May 2007: "Robot curriculum attracts programmers" by Greg Bluestein.

- Red Herring, 12 July 2006: "Microsoft Unleashes Its Robots."

Further Information

http://roboteducation.org
All of our materials and ongoing research is published here.

- **To contact IPRE:**
  Institute for Personal Robots in Education
  School of Interactive Computing, Georgia Institute of Technology
  85 5th Street NW Suite 232
  Atlanta Georgia, 30308
  Phone: 404.385.2861
  Fax: 404.601.3185

  Bryn Mawr College
  Park Science Building, Room 250
  101 North Merion Ave.
  Bryn Mawr, PA 19010
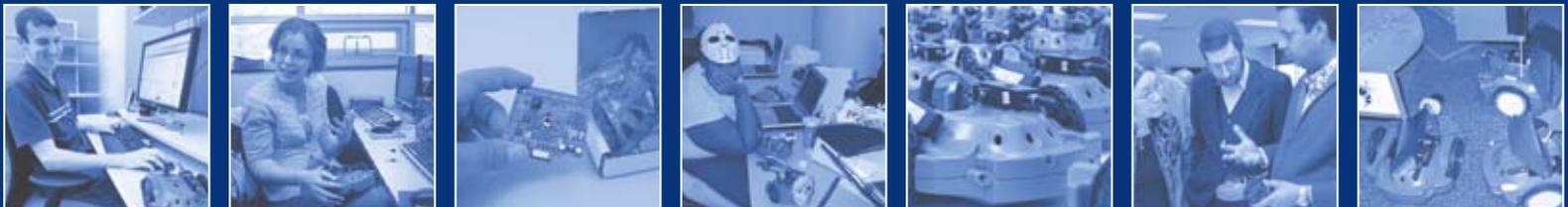 Phone: 610.526.5024

ipre

Institute for Personal Robots in Education
School of Interactive Computing • Georgia Institute of Technology
85 5th St. NW • Atlanta, GA 30308

T: 404.385.2861
F: 404.601.3185

Bryn Mawr College
Park Science Building, Room 250
101 North Merlon Ave • Bryn Mawr, PA 19010

T: 610.526.5024

The Institute for Personal Robots in Education (IPRE) was created to make
education more fun and effective through the use of personal robots. Robots can
provide a tangible and personal means to engage a student in engineering,
science and math education.

www.roboteducation.org