Using Robotics for Teaching Computing, Science, and Engineering at a Distance

BLAINE A. PRICE[*], ANTHONY HIRST[†], JEFFREY JOHNSON[‡], MARIAN PETRE[*], MIKE RICHARDS[*]
Departments of Computing[*], Telematics[†], and Design & Innovation[‡]
The Open University
Milton Keynes, MK7 6AA
UK
Contact Author: *B.A.Price@open.ac.uk*

ABSTRACT

The abstract nature of computing, science and engineering can make them difficult subjects to teach in any environment; teaching at a distance introduces additional challenges. This paper presents on-going work into new distance education courses using commercially available robot kits to introduce fundamental concepts in computing and engineering. It discusses why robotics provides leverage on teaching these subjects, the choice of platform, and how 'teamwork' will be supported in this context.

KEY WORDS: Curriculum design, Universities without Boundaries, Artificial Intelligence, Mobile Communication and Computing, University Education, Education of Children

1. INTRODUCTION

Teaching abstract science and engineering concepts to novices always presents special challenges, but doing this at a distance introduces additional difficulties for both the student and the organisation. The Open University has been teaching all of its undergraduate students using *Supported Open Distance Learning*[1] for over 30 years. Even given high-quality text and multimedia presentations supported by BBC television programs and video; students have difficulties with some of the more abstract concepts in the sciences, engineering, and computing.

A number of researchers have shown that robotics can be motivating and beneficial when teaching science and technology [1]. We agree that robots are a powerful way to motivate learning. The construction and programming of robots uses a wide range of scientific and engineering principles [2] necessitating teamwork, planning and record keeping, all of which are essential for general science, software engineering, systems engineering and everyday life. So how can these issues be handled effectively in distance education?

Robotics offers particular advantages for teaching computing and engineering subjects at a distance because of the transparent nature of the robot: it is self-demonstrating. Using a robot as a teaching medium allows students to have a 'conversation' with the device under construction. A student can ask questions of a robot and observe the answers:

- *What happens if I use a different gear ratio?*
- *What happens if I set a light threshold to a different value?*
- *What happens if I have these two complementary (or even opposing) behaviours active act at the same time?*

In the absence of a traditional instructor, the robot can provide students with a form of interactive learning experience at a time and place of *their* choosing. As Beer *et al* point out, *"the real world rather than a professor decides whether a particular engineering design or a certain scientific hypothesis is correct"[1]*. The challenge is to guide students *effectively* through the universe of possibilities, to encourage exploration while minimising time-wasting dead-ends.

The Open University (OU) has taught subjects related to robotics for many years, and we are beginning to formulate a new robotics curriculum for our students. This is strictly within the context of distance education: In addition to our own curriculum development, in collaboration with the international RoboFesta[2] and RoboCup[3] movements, we plan to support teachers who are teaching robotics in schools.

In this paper we present a rationale for using robotics when teaching subjects at a distance. Additionally, we include some examples of the activities novice students will undertake. We conclude with some of the open

[1] *Distance Learning* means that students study at a distance from the institution providing their courses. *Supported* means the students have a local tutor for telephone and email contact, able to help, who provides face-to-face tutorials, and grades their assignments. *Open* means there are no pre-requisites for entry.

[2] http://www.robofesta.net, www.robofesta-uk.org

[3] http://www.robocup.org

questions and challenges raised by using robotics in the curriculum and show where robotics may be used to help teach other subjects.

2. ROBOTICS HOME EXPERIMENT KITS

The Open University has always tried to give its students an experience of laboratory work comparable to that provided in other universities. We have a long tradition of designing Home Experiment Kits, (HEKs), which include everything students need to conduct hands-on experiments. For example, as part of a Mechatronics course, from 1994 to 2002, we distributed a Lego-based robot designed and manufactured for students. Using a borrowed HEK, a student could conduct various hands-on experiments in perception, cognition, and execution, including intelligent control.

Annual refurbishment costs and limits on the number of kits in stock meant that the Mechatronics HEK was expensive and limited student numbers. Yet our experience with Lego-based teaching materials has made us well disposed toward the Lego Mindstorms[4] Robotics Invention System™. The set includes a programmable, industry-standard microcontroller with ready access to I/O ports. It was developed from a prototype produced at the MIT Media Lab and is expandable. Additionally Lego Mindstorms is widely used in other universities [1].

We are currently investigating the possible use of Lego Mindstorms as the basis of a set of HEKs that students would keep as part of the course materials; retaining the HEKs for further courses. Lego Mindstorms would form the hardware platform for undergraduate courses in robotics, engineering, and computing at first, second, and third levels[5].

At Level 1 students would perform some elementary but exciting experiments, and be introduced to the basic technologies of robots, and hence basic technology. At Level 2 robotics would be the motivating force for a course in electronics, instrumentation, control, and communications. The students would design and build add-on hardware to increase the functionality of the Lego Mindstorms brick, which by default can only read three sensors and drive three actuators. 'Instrumentation' may include machine vision, and there are exciting possibilities for building robots using the extended functionality. At Level 3, a new course in intelligent machines would replace the existing Mechatronics course.

There are obvious benefits to having the same HEK platform for at least three courses. First, the learning overhead is reduced, because the students can transfer their skills with the HEK from one year to the next. Second, the total cost is reduced, with the HEK for each course costing the equivalent of one third of a Lego Mindstorms kit. This is less than the refurbishment costs for the current Mechatronics HEK. Other gains include reduced overall development costs for the University, especially software, and the development of consistent user interfaces. As an additional benefit the students keep *their* HEK at the end of their studies.

We expect this robotics curriculum to be popular. The OU system makes it feasible to attract and support many thousands of students each year. For example, each of our introductory level courses in technology and other disciplines regularly attract 8,000 to 13,000 students from across Europe and other countries.

3. ROBOTICS TASTER COURSE

As part of the development of the new curriculum, we have devised an introductory 'taster' course; encouraging students to experiment with our materials and report their experiences. The course consists of approximately 10 hours of study per week for 10 weeks. Each week of study is broken down into units lasting no more than 2-3 hours each. This course will have no prerequisites other than basic numeracy, literacy and IT skills (such as using a computer to install software and connect to the Internet). Students will build robots, construct robot behaviours using software objects, test them, and record their observations to be shared with others. At the end of each unit students will be able to watch (by CD-ROM video, streaming Internet, or television) an exciting real world example of the technology they have just taught themselves.

The first section of the course encourages students to explore the issues surrounding robot navigation in a controlled environment (such as their kitchen or bedroom). Students explore the difference between remote-controlled and autonomous robots. We conclude this section by looking at real-world examples such as bomb-disposal robots, robot soccer, and a proposed planetary rover.

The next part of the course introduces a few of the basic mechanical concepts behind robot locomotion. Students build a number of example robots, test them, and record their observations using a scientific method. Students are encouraged to ask questions such as *"how can we build robots with legs?*" Based on their experimental evidence, students will write a short justification for the design of a locomotion system for a robot intended to explore an unknown environment.

Later sections of the course expand on the links between biological systems and robotics. A series of lessons will compare and contrast the sensors that have evolved in the natural world and their man-made equivalents. Students begin giving behaviour to their robots by adding

[4] http://mindstorms.lego.com

[5] Equivalent to first, second and third year undergraduate courses at a conventional university.

independent, yet communicating objects (later they will discover that they have learned object-oriented programming). This is an ideal point to move on to the question of 'thinking machines' and an introduction to artificial intelligence. This discovery is integrated with an examination of human and machine memory, learning, and simulating intelligence.

The course concludes with a choice of assessed projects, each of which is meant to have a real-world significance, such as a can-collecting robot or a robot soccer player.

After this 'taster' course, students will be able to study traditional courses in computing, engineering and artificial intelligence, or continue to use their kit with dedicated robotics courses at higher levels.

One of the biggest concerns within distance education is student retention, both within courses, and continuing to study at higher levels. By using the proven medium of intelligent mobile robots, we believe we motivate students to complete their courses. Our goal is to make learning so enjoyable that students will want to continue the 'taster' course, and move on to other studies (some of which will involve this kit that they will already be familiar with).

4. WHY USE ROBOTICS TO TEACH?

Mature students are often inhibited from experimenting freely with unfamiliar equipment for fear of breaking it. Our work with both children and adults has shown that building and programming robots made from children's building blocks is both inviting and non-threatening. Because Lego Mindstorms is designed for use by 8-year-olds, it is built to withstand all forms of abuse. Students recognise this and tend to be more willing to explore and 'play' – and hence venture beyond prescribed activities.

Some have argued that robotics is a passing fashion, but our research shows that it is significantly different from other technologies used for teaching [3]. We know of no other medium that can simultaneously and transparently support the teaching of algebra and trigonometry, design and innovation, electronics and programming, forces and laws of motion, as well as materials and physical processes. Robotics itself is multi-disciplinary, encompassing subjects such as mechanical engineering, electronics, control, communication, vision, real-time parallel computing, and systems design. All these are relevant in our teaching.

4.1 Teaching computing through robotics

Introducing students to core computing concepts has always been a challenge. Traditional methods of teaching computing tend to favour abstractions, and students often have difficulty reasoning about program behaviour and recognizing the relevance of their activities. Many computer courses have concentrated on providing learners with some skill in a programming language. This can be problematic as general-purpose languages are complex, in order to afford necessary richness to the programmer. Unfortunately for the novice, this often means '*you need to know a lot to do a little*'. (In contrast, languages intended to teach newcomers are often very constrained and of limited use at higher levels. *'You don't need much, but you can't do much either.'*) Either students have to learn the syntax before they can write any programs (which is frustrating), or they have to enter code that is effectively meaningless to them. Furthermore, many languages require the users to type in a large amount of code to produce relatively trivial results.

Students' first programs typically print a few words, sort some numbers or draw a square. These programs are rarely useful and are often frustratingly limited. In contrast, a robot can be given interesting behaviour with relatively little effort. Who wants to print 'hello, world' when they can get a robot to zip around a room following a light source?

Traditional methods of teaching computing tend to concentrate on abstract concepts and procedures. Students often have difficulty extrapolating program behaviour from these concepts. When students write programs and have to debug them, they must often resort to placing statements throughout the program to print its internal state – a task requiring additonal overheads for the student. By comparison, much of a robot's state is evident from its behaviour.

An object-based approach is now considered the basis of sound software engineering. Our experience in teaching computing [4, 5] using object oriented programming has shown that it is easier to represent and present complex behaviours to novices [4]. Object principles are highly abstract, and, even though our existing teaching uses familiar objects (such as frogs on the computer display) as examples to explain behaviour, it is easier to explain object concepts when you can hold the physical objects and observe their behaviours first-hand. A robot is a real-world object that can send and receive messages, just like a computing abstraction of the robot. The difference between the two is that students can observe the effects of their actions directly, as opposed to interrogating a software object.

4.2 Teaching engineering through robotics

Students appreciate mechanical engineering issues when they are faced with problems and have to overcome them. Simple scientific principles such as conservation of momentum can be taught through concrete example and experience. For example, students are given very fast but weak motors and are tasked with moving a heavy load. By putting a large gear on the wheel and a smaller gear on the motor the student produces a slow but powerful output with high torque.

One early experiment that students undertake in the taster course is to study the behaviour of a robot on a sloping surface. An incline is built by propping a plank of wood or a hard-backed book at an angle; the steepness of the slope can be easily adjusted. The student would use a simple robot at differing angles of slope, recording their results at each stage.

When they have found a maximum slope that the robot could climb, the student would substitute softer tires for the default wheels, or replace the wheels with caterpillar tracks and re-run the experiment. Later the student examines the effect of different gear ratios.

By the end of the experiment the student should be able to make predictions about the behaviour of the robot under differing conditions and identify suitable terrains for different robot designs.

We wrap each of our exercises in such a procedure to encourage scientific thinking. Students are encouraged to hypothesize the behaviour of the robot they build, record principled observations, and make conclusions as to the reasons behind the behaviour. In the case of gears, students should be able to eventually conclude a relationship between the torque and the size of the gears. We then present them with a steeper slope and ask them to predict what size gear will be necessary for their robot to climb the slope. The students then test their prediction and observe the results.

Students are often taught programming and design on powerful computers with large amounts of memory and broadband connections. In the real world, embedded systems require the developer to work with resource constraints such as the limited memory, power and input/output limitations. The robot brings home these limitations in a concrete fashion.

Using a robot as the teaching domain forces students to 'take ownership' of their learning and how they apply their knowledge to achieve a particular goal. We have observed that students are often more committed to getting something that exists in physical space to work, than something in logical space (i.e., a tradition al computer program).

5. TEAMWORK AT A DISTANCE?

Almost all the benefits of using robotics for teaching apply to distance learning. There are of course some disadvantages peculiar to distance learning. The most obvious example is that students study at home and do not have daily face-to-face interaction with other students. Surely they must miss out on an essential benefit of teaching with robotics reported by most researchers: learning how to work in teams?

5.1 Teaching teamwork on Technology courses

Teamworking has been addressed directly by the course TU170 *Computing with Confidence,* where it is an integral part of the course. TU170 is an Internet-based 'taster' course, intended to provide basic computer and learning skills for students new to the OU. Part of the philosophy of the course is that students work together to create a support network.

The course begins with a face-to-face tutorial at which students get to know their tutor and other students. This is followed by intense interaction through the electronic conferences and email. This course actually teaches some of the theory of group dynamics.

Like conventional students, distance students become confused, fall behind their peers, lose files, and so on. And, like conventional students, they help one another by offering advice and sympathy. Some of the groups work very well, some do not, for a variety of reasons. Perhaps a leader has not emerged, or perhaps too many.

An interesting aspect of group work is the subtle interventions of the tutor. He or she knows the desired outcome of the activity, and can see where the group may be going astray. The tutor also knows that a major part of the learning process is for students to apply the team-working principles they are learning to find their own solutions. The tutor is generally only required to give a hint toward the correct solution, or perhaps to suggest that the students revisit a particular piece of study to get the weaker groups back on track. Even if this interaction does not resolve the problem, it encourages the students to reflect on their experience, and learn for themselves how their team might have had a better outcome.

5.1 Teaching teamwork on Computing courses

The previous team-working example focussed on *information sharing*, which is ideal for remote collaboration. Can the same success in remote collaboration be achieved in *design-and-build* projects?

Since 1995 we have implemented a variety of CSCW techniques in our teaching of Computing; including asynchronous electronic text conferencing, synchronous text and video conferencing, and fully electronic student assignments where tutors mark up a student document and return it to them [6] [7] [8]. We have surveyed a range of student project work in computer science and engineering involving team working, including projects incorporating robotics [9] [10] [11].

This experience demonstrates that team working succeeds if:

- *The project is well organized and orchestrated*: the task must be clearly defined and marked by deadlines and deliverables, guidance should be given to

participants on how to approach the project, including both the task and the group interaction, the group must be supervised by a person qualified to resolve problems, and there must be mechanisms for resolving problems that arise.

- *The group should be of the right size:* too large a group and some will not contribute as they can rely on the work of others; too small a group and there may be a shortage of skills or diversity of input, making it difficult for the group to maintain momentum.
- *The problem being solved is engaging*: The problem should be both rich and reasonably solvable. It should be sufficiently general as to interest the vast majority of participants, be directly relevant to the other course material, and add value to the remainder of the course.
- *There is sufficient reward for individual contribution:* Some, or all of the mark for the project should come from the student's contribution to the group. There should be rewards for participation even if the task is only partially completed or not completed correctly.
- *The problem being solved is conducive to a group working solution:* some problems are better solved by individuals, designers of group working should not try to forced unsuitable problems into the group working environment.

Our experiences of remote team working have illuminated five basic models that illustrate different degrees of collaboration and can be related to the learning of team working skills. These are summarized in Table1.

Collaboration Model	**Description**	**students experience:**
co-operative problem-solving	tightly-coupled, synchronous activity; approximates a face-to-face environment, with all the problems that entails	teamwork throughout the development process
divide-and-conquer	Mixes synchronous communication (usually used for planning and resolving integration issues) with asynchronous communication and off-line development	collaboration throughout the development process: planning, negotiation, role identification, discourse
component handover	Loosely-coupled asynchronous activity where each student completes a part and hands it on to the next student to work on	negotiation and critical skills; development using others' products
component critiquing	Loosely-coupled asynchronous activity where each student reads and openly criticizes the code of another	focuses on critical and discourse skills
individual projects that interact after completion (e.g., in competition)	Mixes asynchronous (used to code/build robot) with synchronous (each student competes against another, e.g. football)	project work is independent, but students are able to compare and discuss designs/implementations based on public performances

Table 1: Some Models of Remote Teamworking

We propose to use a mix of loosely-coupled approaches for our robotics curriculum.

5.3 Proposed team-working in robotics

We propose to try the following model in our third-level course on designing intelligent machines. We shall require students to:

- conduct a group design project;
- distribute subtasks in the team;
- conference and manage the team project;
- build, test, and critique subsystems;
- assemble the subsystems into the whole robot;
- build and test the whole robot;
- compete with other teams.

How can we do these things? The first three are well-established in our system through courses like TU170. The others have been demonstrated through other remote computing projects [10]

The robot design tasks will be formulated so that they can be broken down into subtasks. For example, one student might be responsible for the sensing subsystem, while another might be responsible for designing the gear train and propulsion subsystem. Other students may be the team programmers, and so on. Each student builds a piece of the whole and presents it to the team for criticism, before repeating the design cycle. This is directly comparable to tasks in traditional universities and in industry.

Each student will build and assemble all the parts at home using the instructions and program code provided by their

fellows. Every student would have a more-or-less identical copy of the group robot that can be used to perform a task. Establishing that the copies *are* comparable will provide practical lessons in benchmarking and testing. Experience leads us to believe that this is a feasible approach; we plan to evaluate it in detail in a series of experiments with remote students.

Finally, how can the team's robots compete remotely? We expect to set the students a task that can be monitored objectively by their own computer in their own home, with results being assembled over the Internet. Again there are exciting possibilities for a Web-enabled competition finals day, with the teams competing in real-time against other teams.

Assessment is an important question, especially in distance teaching. We agree with Beer *et al* that the result of the competitions is not the important thing; it is the keeping of design notebooks, the quality and originality of the designs, the quality of the analysis, and the students' reflections on the team-design process. We already have well-established procedures for assessing these factors.

6. SUMMARY

In this paper we have discussed a number of issues surrounding the teaching of computing, science, and engineering at a distance and we have identified robotics as a useful teaching vehicle. The benefits of using robotics include:

- Robotics can be used to teach many aspects of the computing, engineering and science curricula at all educational levels. It encourages students to expand their learning beyond the immediate confines of any particular discipline.

- Robotics encourages the exploration and understanding of abstract concepts in these disciplines and may well prove to be a superior method of teaching more difficult ideas.

- Robot kits can be used for distance learning education provided they are used alongside suitable, dedicated materials.

- Our experience with other courses has demonstrated the feasibility and benefits of remote team working; a robotics-based curriculum has the potential to combine the best aspects of individual and team-based learning to give students experience of real-world software and hardware engineering.

We believe all of this is feasible, and will make hands-on experimental team robotics available to tens of thousands of students who could get this experience by no other means.

REFERENCES

[1] R.D. Beer, H.J. Chiel, and R.F. Drushel, Using Autonomous Robotics to Teach Science and Engineering, *Communications of the ACM*, *42*(6), 1999, 85-99.

[2] E. Wasserman, Why Industry Giants Are Playing with Legos, *Fortune*, *144*(10), 2002, 101-106.

[3] J. Johnson, Children, Robotics, and Education, *Proc. The Seventh International Symposium on Artificial Life and Robotics (AROB 7)*, Beppu, Oita, Japan, 2002. 491-496.

[4] R. Griffiths, S. Holland, M. Woodman, M. Macgregor, and R. H., Separable UI Architectures in Teaching Object Technology, *Proc. Thirtieth International Conference on Technology of Object-Oriented Languages and Systems, (TOOLS'99)*, Santa Barbara, 1999.

[5] M. Woodman, R. Griffiths, H. Robinson, and S. Holland, An Object-oriented Approach to Computing, *Proc. ACM Conference on Object-oriented Programming, Systems and Languages (OOPSLA '98)*, Vancouver, 1988.

[6] M. Petre, L. Carswell, B.A. Price, and P. Thomas, Innovations in large-scale supported distance teaching: transformation for the Internet, not just translation, *Journal of Engineering Education*, 1999.

[7] R. Griffiths, M. Woodman, and H. Robinson, Group Working for Budding Software Developers, *Proc. EdMedia*, 1999.

[8] M. Petre and B.A. Price, Programming practical work and problem sessions via the Internet, *Proc. ACM SIGCSE/SIGCUE Conference on Introducing Technology into Computer Science Education (ITiCSE '97)*, Uppsala, Swedon, 1997.

[9] S. Fincher, M. Petre, and M. Clark, *Computer Science Project Work: Principles and Pragmatics*, (London: Springer-Verlag, 2001). 267.

[10] M. Daniels, M. Petre, V. Almstrum, L. Asplund, C. Björkman, C. Erickson, B. Klein, and M. Last, RUNESTONE, an International Student Collaboration Project., *Proc. Foundations In Education*, Tempe AZ, 1998.

[11] M. Last, V. Almstrum, C. Erickson, B. Klein, and M. Daniels, An International Student/Faculty Collaboration: The RuneStone Project, *Proc. ITiCSE 2000*, Helsinki, Finland, 2000.