# Using robotics to motivate 'back door' learning

Marian Petre and Blaine Price

Computing Department, The Open University, UK
m.petre@open.ac.uk

**Abstract**

This paper suggests that robotics can provide a vehicle for guiding primary and secondary school children toward an effective understanding of programming and engineering principles. It observes that children find robotics stimulating and motivating, and that their interest in, and focus on, *'making the robot do what I want'* leads them 'via the back door' to learn about programming and engineering in a way that is both well-grounded and generaliseable.

These observations arise from empirical studies of children participating in robotics competitions: we conducted observations and interviews with all the participating teams at two robotics events (one regional, one international), and we followed one young robotics team in a case study. The children had almost all built their robots using LEGO MindStorms for specific competitions, with soccer, rescue and dance events. The children typically worked in teams, building robots as an extra-curricular activity supported by a teacher/mentor. The children came from a variety of educational and social backgrounds.

The paper considers what makes robotics motivating to children, including children who are not considered 'technically oriented'. It describes learning that has emerged from children's experiences in building and programming robots. It describes examples of children learning subjects that they previously considered difficult and inaccessible, in order to solve problems in robotics. It describes examples of children identifying and understanding principles, concepts, and elements of practice that are fundamental to programming and engineering. It describes further how secondary school students working in teams learned that this programming and engineering knowledge has a social context.

## Introduction

The fascination of robots for many children (and adults) is evident in the surge of robotics programmes on television, such as Mentorn's *Robot Wars* and the BBC's *Technogames*, in the proliferation of affordable robot toys and construction sets, and in the publication of robotics magazines and websites. At the same time, educators' concerns about falling numbers of science and technology students are now familiar, as are arguments that we need to attract students to these disciplines as early as possible. Many arguments centre on generating and sustaining students' motivation. Numerous writers endorse robotics as an educational vehicle (e.g., Kumar and Meeden, 1998; Beer, Chiel and Drushel, 1999; Nostrand, 2000; Weinberg et al., 2001; Martin, 2001), with a whole literature devoted just to using LEGO MindStorms (e.g., Capozzoli and Rogers, 1996; Klassner and Anderson, 2003), at levels ranging from primary school through university. There are reports of improved performance in mathematics, physics, and engineering design courses resulting from robotics projects (e.g., Nagchaudhuri et al., 2002). However, most of the evidence is situated and anecdotal, based on reports of teachers achieving positive outcomes with individual initiatives.

Jeff Johnson (2002) argues that robotics offers special educational leverage, because it is multi-disciplinary and involves a synthesis of many technical topics, including algebra and trigonometry, design and innovation, electronics and programming, forces and laws of motion, and materials and physical processes. In his analysis, only Meccano (a classic British construction toy) and computer games approach robotics in scope. "Although simulation of physical systems can be very realistic, the pedagogic value of robots lies in making them

work, by using or extending your knowledge to diagnose and fix problems." He considers robots as models of living things with which children can experiment without adult interference or sanction. He argues that robots are a particularly motivating technology – and that they are enduringly so – because they are concrete, complex, and relate to deep human needs.

As an active, problem-based, team-centred approach to learning, robotics relates well to current thinking in engineering and computing education (e.g. Striegel and Rover, 2002; IEEE Problem Based Learning Initiative, 2003, Vandebona and Attard, 2002), where problem-based learning is viewed as integrative and as enhancing team-working skills. Students learn through *doing*; they discover knowledge and develop understanding through their own activities, guided by a teacher – rather than 'receiving' it from a teacher through instruction. Hence hands-on robotics fits well within the constructivist view of learning (Piaget and Inhelder, 1966; Papert 1980; Harel and Papert, 1991) which portrays learning as the acquisition or 'construction' of knowledge through observation of the effects of one's actions on the world.

In robotics, students' learning is concrete, associated with phenomena they create, observe and interact with, and so the abstractions they derive (or apply later) are grounded and relevant. Problems are open-ended, permitting many solutions and many approaches. Hence, robotics affords opportunities for learning problem-solving techniques and processes, integrates a number of domains, exposes realistic constraints and issues, and leaves room for creativity. It is argued that, if supported by:

- guidance,
- introduction to tools,
- access to models and examples,
- suitable task structure, and
- encouragement of self-reflection,

students can generalise and abstract from their experiences and bridge between experience and curriculum. (Jadud, 2000; Miglino, Lund and Cardaci, 1999).

**RoboCup Jnr. and RoboFesta context**

RoboCup, the Robot World Cup Initiative, aims to foster AI and intelligent robotics research by providing a standard, familiar, test problem which promotes exploration of technology and techniques: playing a soccer game. Robocup Jnr. reframes the aim for primary and secondary school children, providing an educational introduction to robotics and to teamwork through competitions in specific challenges:

1. *soccer:* matches between 2-on-2 teams of autonomous robots on a 90 X 150 cm grey-scale pitch
2. *rescue:* autonomous robots race to identify 'victims' in a line-following task incorporating obstacles and uneven terrain
3. *dance:* one or more autonomous robots perform to music in a competition judged for creativity.

RoboCup Jnr. promotes project-oriented, team-based activity intended to encompass a range of theoretical and practical issues: "We use autonomous mobile robots to give students hands-on experience with construction and programming, physics and mechanical engineering, mathematics and artificial intelligence. Through the robotic medium, we can teach about artificial life, evolutionary computation, hardware-software interaction, distributed systems, electronics and communication." (RoboCupJr, 2003)

RoboFesta is "a world-wide educational movement that focuses on bringing science and technology to a general audience through widespread public participation is a range of robotics competitions". (RoboFesta, 2003) It aims to further robotics education at all levels. Its first event was a 'Design a Robot' competition for children held in conjunction with the popular BBC children's television programme *Blue Peter*. That competition attracted over 30,000 entries (Johnson and Hirst, 2001). RoboFesta-MK 2002 was the first of a series of regional robotics competitions intended to promote "robust, hands-on robotics activities." Held in Milton Keynes in July 2002, it involved children aged 12-17 from secondary schools in the town.

Both groups have an educational mission and see robotics as a powerful educational vehicle. This paper reports on studies intended to discover how well such events are able to serve the mission, and to investigate the nature and extent of learning associated with building robots for challenges such as soccer, rescue, and dance. The data collection was motivated by and conducted in collaboration with event organisers, in particular Elizabeth Sklar (Columbia University) and Jeff Johnson (Open University).

**Empirical studies**

Our observations arise from three studies: observations and interviews at two robotics competitions, and one longer-term case study.

We conducted interviews and some observation with 17 (out of 25) teams at the international RoboCup Jnr. 2001 competition in Seattle, Washington. RoboCup Jnr. 2001 included soccer, rescue, and dance events. All of the teams built their robots specifically for the competition. All but two used LEGO MindStorms, although many of them enhanced MindStorms with other technology (e.g., motors, sensors, dressing). The exceptions used Fisher Technic or built from scratch. The children all worked in teams, typically building robots as an extra-curricular activity supported by a teacher/mentor. We were able to observe the teams in the competition workspace, while they 'tweaked' their robots during the events. Interviews were semi-structured, focused on learning and motivation.

We conducted interviews and some observation with all 13 teams at the regional RoboFesta MK 2002 event in Milton Keynes. RoboFesta MK 2002 included soccer and dance events. Again, all of the teams built their robots specifically for the competition using LEGO MindStorms, with some enhancements. Most of the children worked in teams, although two of the teams were very loosely coupled, comprised of two children each building one of two robots for a soccer team. We were able to observe the teams in their work areas.

In addition, we followed one team of two young children in a case study over two years, through both the RoboCup Jnr. 2001 and RoboCup Jnr. 2003 competitions. The team included a boy, aged 8 when observation began, and 11 by the time they competed in RoboCup Jnr. 2003 in Italy, and a girl, aged 6 to 8. Over the period of observation, the two participated in all three challenges.

*Subjects:*
RoboCup Jnr. 2001: 17 teams, of 2 to 7 members, from 4 countries. Ages from 6 to 18, with 13 all-boy, 2 mixed, and 2 all-girl teams. The children came from a variety of backgrounds, from both state and private schools, including two 'alternative' schools for students who did not perform well in mainstream education.
RoboFesta MK 2002: 14 teams, of 2 to 7 members. Ages from 12 to 14. 10 all-boy, and 4 mixed teams. The children came from a variety of backgrounds, from state secondary schools in Milton Keynes.

*Data collection:*
Observations were collected as contemporaneous field notes. RobotCup Jnr. interviews were captured on videotape and transcribed[1]. RoboFesta MK interviews were captured on audiotape and as contemporaneous notes. Observations during the case study were collected in contemporaneous field notes, photographs, and a development diary.

*Programming context:*
The LEGO MindStorms construction system includes a visual programming language, RCX code. It uses a drag-and-drop interface and a LEGO bricks metaphor for constructing programs. It is a simplified language, providing only basic facilities. Other programming platforms are also available for MindStorms, such as RoboLab, a derivative of LabView, and

[1] Transcripts were kindly provided by Elizabeth Sklar.

'Not-Quite-C' (NQC), a textual language based on C. Programs are created on a PC and transmitted to the MindStorms programmable controller 'brick' via an infrared link.

*Data analysis:*
The material was subjected to a variety of analyses. Some results have been published elsewhere (Sklar, Eguchi and Johnson, 2002). The observations here arise from inductive analysis of all the material.

**What makes robotics motivating to children?**

Despite setbacks, frustration, and occasional tedium, the children in these studies all persisted. The single most-frequently-cited driver, articulated across the age range by more two thirds of the teams, was the desire to build a better robot: "There was always something new you could do with your robot." It came in many forms: "You find…easier ways and easier ways to solve a problem." "Always a new bug." "We saw the bigger robots and began to dream." One important aspect was the determination to finish: "The challenge of getting there in the end." "It's like a video game…You want to be able to have it done." Yet another important aspect was the *open-endedness* of the pursuit: "…at least you can always change things, there's always another goal to reach." "You can always improve it and you can never have it perfect." "You get up to that point and it's good, but then you get let down, and so you think 'Oh, I'll do better next year' and so you do it again." Another contributing factor was the social context: "It's interesting meeting new people and showing how good you can be." "I like the way other people's robots can give you ideas for the next time, and then you get more and more and more. And then if you put all of your ideas together you can get a really neat robot."

The second most-frequently-cited driver was the prize, to travel (for RoboCup Jnr.) and to be selected for the competition (both RoboCup Jnr. and RoboFesta). Other reasons for persisting included friendship, interest in technology, and a desire to study related disciplines such as engineering or computing.

**Evidence of learning from building robots**

Both in their interviews and in their observed, spontaneous discussions, children report learning from building robots. The most compelling accounts were those they made spontaneously to each other, for example: "Oh, yeah, I had trouble with the gear thing, too. We got a book out and figured it out with a bunch of tries. His Dad helped." [age 9] and "I couldn't get the program right until I learned algebra. Then I could write it all down so I could see what was happening and work out where I went wrong. Want me to show you?" [age 14/15] We observed a number of discussions in the work areas in which children shared and compared code, explained techniques, and showed each other useful **representations**, such as truth tables for working out actions associated with different sensor inputs.

***Did you have to learn anything in order to build your robot?***

In the interviews, when asked if they had had to learning anything new in order to build their robot, 21 of the 30 teams reported something specific, often more than one thing. 19 teams named **programming or a programming language** as something they had learned. They acquired more than syntactic knowledge: "…right now I'm doing a programming class and it might not be the exact same programming you get by doing RoboLab, but you get the concept and idea of follow through this and then do this…transformed into different uses or different programs." [age 15] They felt that the experience gave them a more general advantage: "Being on the computer makes it much easier being on the computer at school." [age 7]

7 teams reported that they learned about **gearing and motors**. 5 teams (all teen-age) reported that they learned about **hardware** and **electronics**, e.g.: "…understanding hardware – in order to integrate parts from different sources". [age 18]

2 teams reported more advanced insight into robotics: "…into the difficulties involved in robotics" and "simulation and testing (and the difference between what you see on the screen and robot behaviour)". [age 17-18].

But their reported learning encompassed more than the obvious technology. One team reported emphatically that they learned "to think". 5 teams talked about improved **problem solving skills** ("…figuring out how things work") and associated lessons of persistence ("If you sit there and work there are always new ideas…"). This is predicted in the problem-based learning literature (e.g., Gallagher, Rosenthal and Stepien, 1992) Several felt that the problem solving they learned had helped them in mathematics and science classes. 2 teams described incorporating **improvisation** into their problem solving ("So I think we kind of learned what to do, learned how to improvise…"). Many teams described **learning from mistakes**, but two teams captured attention to obstacles as part of their problem-solving strategies: "Disasters can produce insights." and "Problems can lead to improvements."

A third of the teams talked about **teamwork**. Some had to learn teamwork: "The fact that we have to work together as a team, we have to talk to each other and communicate more." Some children learned about their potential to contribute to, and influence, a team: "I never knew I was so persuasive." Some described **interpersonal skills** they learned in order to sustain the team: "I learned very quickly how to do calming exercises on the drop of a hat." **Patience** was a recurrent lesson.

Children learned, too, about themselves: "I never knew I could think that much. I thought my brain was the size of an acorn." They gained confidence: "Always trust your instincts." and "We definitely got more wiser."

***What lessons would you pass on to someone just starting?***

The children revealed some of their priorities when asked what lessons they would pass on to another student new to the competition. The teams provided 95 pieces of advice, which can be grouped into four general categories: general encouragement and moral support (29 examples), design advice (28), specific and technical advice (21), and teamwork and information sharing advice (17). More detail is given in Table 1. The general support and specific advice are reasonably predictable. Most interesting are the emergent insights about the design process, and the social lessons.

| general category | sub-category | examples |
|---|---|---|
| *general encouragement and moral support* | it's an open-ended problem – you need your own answers (4) | "There isn't any right way." "Build your own robot" |
| | things go wrong (8) | "Lots of things go wrong." "The robots do completely unpredictable things even though you programmed them…that's just the nature of it." |
| | time and patience (8) | "Take your time, don't rush." "…takes a lot of time, a lot of patience" |
| | perseverance (7) | "Try your best." "Practice, practice, practice." "Never stop trying to improve your robot." |
| | dealing with frustration (2) | "If you're too angry/stressy – walk away." |
| *design advice* | plan (7) | plan (7):<br>"Know what you're building; plan it out first" "To have a better, clear concept" |
| | consider alternatives (3) | "Try different ideas before making decisions." |
| | suggested design process (5) | "Try one design, then make amendments." "Try and if it doesn't work just sort of use that idea, take the good bits from that idea, and use those to make a new idea." "Sometimes it's easier to make a new one rather than modify, if you've got the time." |
| | constraint-based design (7) | "Choose actions based on priorities." "I would make sure to bring all the rules because it can affect the design." |
| | learn the tools, the basics | "Get familiar with programming first." "Work out how |

| | | |
|---|---|---|
| | first (6) | to use motion touch sensors." |
| *specific or technical advice* | technical advice (12): | "Have support so it stays together." "Run motors for a fixed time and then stop." "Make sure the cords are covered because they can get ripped off." |
| | choice of technology and tools (5) | "Don't use RoboLab, unless you have to choose over it and the one that comes with the MindStorms kit." |
| | hygiene and backups (4) | "…typing the programs and always keep the programs tidy" "Back up the programs." |
| *teamwork and information sharing* | share information (6) | "Trial it against other teams. Don't be afraid that they're going to steal your design or anything like that. At least if someone else sees your design and likes it they're furthering their own knowledge as well as yours, because you might find something…" |
| | work as a team (7) | "Work together as a team." |
| | be open to advice (4) | "Discuss ideas and listen." "Follow advice!" |

**Table 1:** Advice to new competitors

### *Design principles and processes*

Building robots has clearly conveyed to these children that "You have to do it step by step." [age 12] What is presented in the advice was also reflected in their observed discussions and behaviour in the work areas. Many of them learned the lessons of systematic exploration and testing; systematic behaviour was evident in the work areas. But many had evolved more sophisticated views on design. Many talked about incremental design; some talked about when radical re-design was more appropriate.

The development of attention to constraints was notable. We observed design discussions (and emergency re-design discussions) in which teams mapped out their solution space in terms of the bounds set by the constraints, for example pruning possibilities excluded by the technology (e.g., the limitations of sensors). Teams talked about planning in order to take account of the rules. Teams also talked about what they called "priority-led design", emphasising design priorities ("fast and agile"), team strengths, and perceived technical advantages. They talked about the importance of understanding the operating conditions: "Have lots of practice matches because it's really just trial and error. You've got to know what works in different situations." [age 15]

Many had taken on board the value of planning before action, and of considering alternatives before committing to a particular plan. Not only did they offer it as advice in the interviews, but they reminded one another of it during their activities. This is a step toward expertise; experts are distinguished from novices by the time they spend planning (Allwood, 1986; Kaplan et al., 1986) One team was developing a sense of design elegance: "Keep trying – make it more simple." [age 12-14] For one team, planning increased their sense of control over outcomes: "I think the lucky breaks we got weren't all that much luck because we spent so much time planning and designing and thinking about how it was before we ever made anything." [teen-age]

### *Social lessons*

The organisers of RoboCup Jnr. and RoboFesta have striven to convey an atmosphere of learning and participation within healthy competition: both are primarily educational, rather than competitive, events. The evidence from these cohorts is that they have succeeded. The children talk often about their intrinsic motivation to 'build a better robot', many spend considerable time talking to other teams, and some have explicitly learned that the advantage lies in sharing, rather than hoarding, information. Said one robot builder: "It doesn't matter if you win or not." [age 8]

Many of the secondary school students working in teams learned that programming and engineering knowledge has a social context. They addressed the social context of their own team, reflecting on the nature of decision making within the teams, and on the distribution of

roles and responsibilities – and on the consequences of that distribution of roles. Some children were relieved to rely on others, some talked about what they had learned through collaboration, and a few felt limited by their own role and denied the roles they did not play: "*I* want to program…" [age 9] The children addressed the social context of the competition, almost all of them prioritising participating over winning, and finding that information sharing was advantageous: "Exchange ideas with other teams, especially you can see some tricks of the mid-sized teams that use new robots." [teen-age] They also saw the broader social context, looking at the influence of national culture on design and engineering practices, and on interaction: "…you learn so much just about technology and about your own country…cultural differences" [age 15]

**Directly observed learning**

The observations and case study provided more direct evidence of learning, providing examples of children identifying and understanding principles, concepts, and elements of practice that are fundamental to programming and engineering. In their interviews and observed discussions with one another, many of the children revealed that they had come to terms with topics (such as programming, gearing, and mathematical representations) which they had previously found difficult, in order to make the robot work. That drive to build a functioning robot had carried them into new and sometimes daunting territory. It had helped them to take step-by-step and systematic approaches to learning what they needed to know.

The following sections will describe selected learning episodes observed during the case study of the young (8-11 year-old) robot builder.

***Variables***

RCX code of 2001 vintage did not afford variables, although it did provide counters. One day the child remarked: "What I really need is a place to put things, you know, stuff I want to remember. I don't care where it is, just so I can get it back when I want it. I could just call it back and it would tell me what's inside. Like calling a dog: 'Here Fred' and the stuff comes back."

The child had articulated the concept of a variable, without having been introduced to it, and without knowing its name. Many computer science educators would find this remarkable, since many university students struggle with the concept. In particular, many students have difficulty separating the *concept* of variable from the details of *how it is stored* in the computer. Indeed, some educators have researched novel ways of conveying this concept (e.g., Sajaniemi, 2002).

***Subroutines and the specificity/generality trade-off***

In the course of working with RCX code, the child discovered the sub-routine mechanism, which allows a group of instructions to be named and thereafter to be re-used by name. The child found this mechanism appealing and set about using it. However, he came across a need to re-use some code, but with variations. First, he made the code into a sub-routine. Then he copied the sub-routine, altered it, and gave it a new name. He did that again for another variant, but he became dissatisfied with the repetition. Instead, he tried to identify portions of the code that could be grouped for re-use as a sub-routine while leaving out the pieces that needed to be varied. The effort failed, because there were too many pieces, and he became confused. The child asked: "How can I decide when to bundle it up, and when to do it separately?... Maybe there's a way to do all the different ways inside one [sub-routine]? But then I'd have to give it some way to choose…"

The child, having become dissatisfied with the repetition required by *specific* solutions, was grappling with the overheads associated with making a *general* solution. The specificity/generality trade-off is a classic in engineering, and the circumstances, combined with a desire for economy, had led the child to weigh the trade-off for himself. It also led him to look up different control structures.

***Engineering discipline: one problem at a time, one step at a time***

Faced by a catastrophic bug with a large number of possible causes, the child quit working on the robot. A 'consultant' was called in, a friendly adult computer professional, to help with the debugging. The professional listened to the problem and set out the debugging plan, a series of some 20 tests. "So let's get started." The first two tests failed. The failures agitated the child, but he noticed that the professional was undaunted. "Doesn't it bother you?" the child asked. "No, just because we don't have it working yet doesn't mean we won't fix it," was the reply. After reflection, the child asked: "OK, so it doesn't matter if it doesn't work this time. It only matters what we do next?"

This idea became embodied in the child's practice as a sort of mantra – "It's the next step that matters" – which he would repeat whenever he got stuck. The discipline of systematic debugging, and the patience required to undertake it, are usually lessons hard won. Many students (and professionals) persist in seeking shortcuts despite repeated introductions to good practice.

The significance of these accounts is that, in each case, events drove the child to discover concepts and principles that are often considered difficult. The child not only asked the questions, but also took in the answers, because they had relevance and immediacy for what he wanted to accomplish. Trying to build a robot was a rich enough task to cast him repeatedly into such situations of need, and his desire to 'make it work' provided sufficient motivation for him to persist in addressing those needs.

## Limitations

Our findings are limited by the sample of children we studied. Although we interviewed and observed teams from 4 countries and 20 schools, they were all children who had persisted as far as a robotics event. We didn't have access to the children who quit, or who didn't choose to participate in the first place. Therefore, our conclusions can address only the demonstrated *potential* of robotics as an educational vehicle. Although the fact that the sample was diverse provides evidence that robotics appeals across cultures, institutions, and genders, we cannot address how universal the appeal may be.

## Conclusion

The paper has considered what makes robotics motivating to children, including children who are not considered 'technically oriented'. It has described learning that has emerged from children's experiences in building and programming robots. It has described examples of children learning subjects that they previously considered difficult and inaccessible, in order to solve problems in robotics. It has described examples of children *independently* identifying and understanding principles, concepts, and elements of practice that are fundamental to programming and engineering. It has described further how secondary school students working in teams learned that this programming and engineering knowledge has a social context.

Robotics is not an answer for every one or every problem, but does provide some insight into how the 'right' technology, in the context of problem-based learning, can draw children into learning underlying principles. And it has shown how context, need, and the desire to *'make it work'* draw children to that learning so naturally that they hardly notice the intellectual strides they are making.

## Acknowledgements

This work would of course not be possible without the gracious participation of the children and their mentors, to whom we are grateful. Interviews from RoboCup Jnr. 2001 were

conducted in collaboration with Elizabeth Sklar, Jeff Johnson, and Amy Eguchi. Transcripts of those interviews were kindly provided by Elizabeth Sklar. Interviews from RoboFesta-MK 2002 were conducted in collaboration with Mike Richards and Jennifer Rode. This work has been supported by the Open University Robotics Outreach Group, headed by Jeff Johnson.

**References**

Allwood, C.M. (1986) Novices on the computer: a review of the literature. *International Journal of Man-Machine Studies*, **25**, 633-658.

Beer, R.D., Chiel, H.J., and Drushel, R.F. (1999) Using autonomous robots to teach science and engineering. *Communications of the ACM*, **42** (6), 85-92.

Capozzoli, P., and Rogers, C. (1996) LEGOs and aeronautics in kindergarten through college. In Proceedings of AIAA 19th Advanced Measurement and Ground Testing Technology Conference, (New Orleans, LA, June).

Gallagher, S., Rosenthal, H., and Stepien, W. (1992) The effects of problem-based learning on problem solving. *Gifted Child Quarterly*, **36** (4), 195-200.

Harel, I. and Paper, S. (1991) *Constructionism*. New Jersey: Ablex.

IEE Problem Based Learning Initiative (accessed 1 December 2003) http://www.iee.org/professionalregistration/accreditation/pbl.cfm

Jadud, M. (2000)Teamstorms as a theory of instruction. In Proceedings of IEEE Systems, Cybernetics, and Man 2000 (SMC2000).

Johnson, J. (2002) Children, robotics, and education. In Proceedings of the 7th International Symposium on Artificial Life and Robotics (AROB-7), 491 - 496.

Johnson, J., and Hirst, A. (2001) *The Blue Peter – RoboFesta robot design competition.* Technical Report, Faculty of Technology, The Open University, 23 March 2001. http://robofesta.open.ac.uk/report2/

Kaplan, S., Gruppen, L., Leventhal, L.M., and Board, F. (1986) *The Components of Expertise: A Cross-Disciplinary Review* (University of Michigan, Ann Arbor).

Klassner, F., and Anderson, S. (2003) Lego MindStorms: not just for K-12 anymore. *IEEE Robotics and Automation Magazine*.

Kumar, D., Meedan, L. (1998) A robot laboratory for teaching Artificial Intelligence. In Proceedings of the 1998 ACM SIGCSE Symposium, 341 – 344.

Miglino, O., Lund, H.H., and Cardaci, M. (1999) Robotics as an educational tool. *Journal of Interactive Learning Research*, **10** (1), 25-48..

Nagchaudhuri, A., Singh, G., Kaur, M., and George, S. (2002) Lego robotics products boost student creativity in pre-college programs at UMES. In Proceedings of 32nd ASEE/IEEE Frontiers in Education Conference (Boston, November), S4D-1 – S4D-6.

Nostrand, B. (2000) Autonomous robotics projects for learning software engineering. In Proceedings of IEEE Systems, Cybernetics, and Man 2000 (SMC2000), 724 - 729.

Papert, S. (1980) *MindStorms: Children, Computers, and Powerful Ideas*. New York: Basic Books.

Piaget, J. and Inhelder, B. (1966) *La Psychologie de L'Enfent.* Paris: P.U.F.

RoboCupJr (accessed 1 December 2003)
[http://www.artificialia.com/RoboCupJr/education/index.html]

RoboFesta (accessed 1 December 2003) [http://robofesta.open.ac.uk]

Sajaniemi, J. (2002) A new approach to variable visualization: roles as visualization objects. In Proceedings of the 2nd Program Visualization Workshop (Hornstrup Centret, Denmark, June), 74-82.

Sklar, E., Eguchi, A., and Johnson, J. (2002) RoboCup Junior: learning with educational robotics. In G.A. Kaminka, O.Lima, and P. Rojas (eds) *Proceedings of RoboCup 2002* (Fukuoka Japan, June).

Striegel, A., and Rover, D. (2002) Problem-based learning in an introductory computer engineering course. In Proceedings of 32nd ASEE/IEEE Frontiers in Education Conference (Boston, November).

Vandebona, U., and Attard, M.M. (2992) A problem-based learning approach in a civil engineering curriculum. *World Transactions on Engineering and Technology Education*, **1** (1).

Weinberg, J.B., Engel, G.L., Gu, K., Karacal, C.S., Smith, S.R., White, W.W., and Yu, X.W. (2001) A multidisciplinary model for using robotics in engineering education. In Proceedings of the 2001 ASEE Annual Conference and Exposition.

Wolz, U. (2000) Teaching design and project management with LEGO RCX robots. In Proceedings of ACM 2001 SIGCSE Symposium (Charlotte, NC), 95 – 99.