The Use of Robots in Computer Science Education

Jeremy Coupland
University of Cape Town
CPLJER001@myuct.ac.za

ABSTRACT
Educators have always searched for more effective and efficient teaching methods. Using robotics to teach basic Computer Science concepts is becoming increasingly popular. By examining and analyzing existing studies involving robots in education, I have identified some issues. The structure of the robot-based course was seldom taken into account in the results of each study. Many studies document the features of their specific robot, but do not say how effective these features were for teaching specific concepts. Many of the studies were carried out without a control group to compare whether a robot-based course would be better than a traditionally taught course. It was concluded that, while robots have shown statistically significant improvements in Computer Science education, there is still much left to research with regards to the individual robot components, and how they can be best utilized in the courses.

CCS Concepts
• **Computer systems organization → Embedded and cyber-physical systems → Robotics.**

Keywords
Educational robot, Education, Pedagogy, CS1, virtual environment

1. Introduction
Over the past few decades, Computer Science educators have noticed a decrease in the number of students majoring in Computer Science [5, 10, 20]. Researchers have identified many factors that may be contributing to this decrease, many of them being social factors. Students often view Computer Science as a very asocial subject, others view Computer Science as tedious, boring and irrelevant, with no room for creativity [5]. Other researchers have found that certain programming languages, and the syntax associated with those languages, have been deterring students [3]. A combination of these factors has resulted in a decrease in the number of students majoring in Computer Science, and also an increase in the number of students deciding to drop out or change majors [12]. Because of this, educators have been searching and experimenting with different methods of teaching Computer Science, in an attempt to change the students' and society's perspective.

Many educators have found that teaching Computer Science within a certain context can greatly improve a student's learning experience [6, 21]. This entails teaching students basic Computer Science concepts, but framing them around a real life application, so that the students can see how the concepts are directly applicable to real life problems. Some educators have chosen to use a "games first" approach [22], in which the students are taught basic Computer Science concepts, and then apply them by developing simple games. Another context, on which the paper will focus, is robotics.

A large motivating factor for using robotics as a context arose from a learning theory called Constructionism [23]. This theory states that, among other things, learning happens most effectively in situations where students actively make or interact with physical objects. Some educators have even started using robotics in education at primary school levels [16] to encourage students to learn about Science, Technology and Mathematics. Over the years, as far back as the 80s, educators have experimented with using robots as educational tools in order to improve Computer Science education. The following sections will discuss how robots have been used in education, as well as problems and opportunities that certain educators have identified.

2. Simulators in Education
Before robots could be manufactured and sold relatively cheaply, educators decided to use simulations of robots, rather than actual physical robots. This enabled them to teach within a certain context which provides visual feedback in a cost-effective manner. Simulators have been around for a long time, and are still in use today. Some educators have chosen to develop a completely new simulation environment themselves [2, 18], while others have opted to use publicly available simulators.

One of the earliest simulators is *Karel The Robot* [24], developed by Richard Pattis in the early 80s. In this simulator, robots live in a grid world of streets and avenues. Robots can occupy street corners, which are the intersections of the streets and avenues, and can move in four directions: up, down, left and right. The two remaining types of objects in the world are wall sections and beepers. Wall sections are obstacles that prevent a robot from moving in a certain direction, and beepers are objects that a robot can transport from one location to another. *Karel The Robot* has been used many times over the years [12, 13, 19] and has been used to teach both high school and college level Computer Science. While *Karel The Robot* was originally developed to be more oriented towards C++ programming, it has since been translated to support Java programming as well [13].

In 2000, Stephen Cooper, Wanda Dann and Randy Pausch published a paper on a 3D interactive animation environment that they developed: *Alice* [18]. The authors mentioned *Karel The Robot*, stating that they felt it introduced a high level of code complexity, which they felt could be improved upon in an introductory course. *Alice* was created in order to improve upon

the concept of *Karel The Robot* by creating a 3D environment, where students could create their own virtual worlds. *Alice* teaches programming concepts by providing multiple functions, such as *Move*, *Rotate*, *PointAt*, and *Resize*. The students can then use these functions, among others, on their 3D objects and instantly visualize the execution of their programs.

The final simulation environment is called *Jago* [2]. This simulator was created by Jerry Schumacher, Don Welch and David Raymond, and was designed to work in conjunction with a physical robot, the *LEGO Mindstorms*, which will be discussed later. The environment was created as not every student had access to the physical robots at all times, but still needed to work on assignments. The simulator was designed so that it would mimic the physical robots behavior as closely as possible, so that students could work on assignments at home, and demonstrate them using the physical robots during laboratory sessions. *Jago* is programmable in Java and provides students with instant visual feedback on the execution of their programs.

3. Robots in Education

As technology has advanced over the years, physical robots are becoming more and more feasible to use in education. There are many commercially available robots, which educators have adopted as part of their institution's Computer Science curriculum, while others have chosen to develop their own robot to specifically cater to their needs. There are a large number of robots available for purchase that educators have identified as being relevant as educational platforms, however many of them lack the versatility required in order to have an entire curriculum based around them [17].

An example of a robot that was designed specifically by a group of educators is the *e-puck* [17], designed to teach Engineering and Computer Science concepts. These educators needed a robot that met certain requirements: it must be able to fit on a desk near a computer, the interface must be simple, efficient and intuitive, it must be low cost, and it must be easy to share between other students or lecturers. The *e-puck* is a small robot, similar in size to a computer mouse, but contains a wide variety of features, such as eight infrared proximity features, a 3D accelerometer, three microphones for sound triangulation, a colour camera, a speaker, and a number of LED lights. These features together provide a wide range of possibilities for students and educators.

In 2006, an organization called The Institute for Personal Robots in Education (IPRE) was formed. The IPRE [25] is a joint effort between Georgia Tech and Bryn Mawr College and has produced an entire curriculum based around using robotics as a Computer Science context. The IPRE sells a robot kit which costs roughly the equivalent of an average university textbook [5], which includes the robot itself, the *Scribbler*, as well as a textbook, a gamepad controller, drawing pens and a lunchbox carrying case. The hardware of this robot includes five infrared sensors, three photo sensors, a colour camera, programmable LED lights, a speaker, a pen port, and a Bluetooth wireless communications link [1]. Students are able to program the robot in Python using Myro, a python module developed specifically for this robot.

Another robot which has been used at both high schools [11] and college/university [2, 4, 9] levels of education is the *LEGO Mindstorms*. This robot set contains a large number of pieces, which students can use to construct their own personal robot, and a programmable brick, called the RCX. The RCX can be programmed in Not Quite C (NQC), but has been modified by users to allow Java [2]. The standard *LEGO Mindstorms* kit includes multiple building pieces, motors, touch and light sensors, as well as an infrared attachment to connect to a computer, to serve as a communications link. Many more advanced features are also available for purchase from the *LEGO Mindstorms* webstore [26], including gyroscopes, magnetic compasses, vision sensors and more.

A robot that has been used less commonly is the *Electric Ray Robot*. This robot is Arduino compatible and has been used to teach Arduino programming at a university level [3]. The robot comes equipped with an ultrasonic distance sensor, three infrared sensors, a buzzer and an OLED display. The robot can be programmed using the Arduino IDE, as well as a graphical programming environment, which supports drag-and-drop code blocks, useful for introductory programming [3].

The *Ridgesoft Intellibrain-Bot* is a robot which has been used to teach undergraduate Computer Science [8]. The hardware of this robot includes line (photo-reflector) sensors, infrared range sensors, and ultrasonic range sensors. The robot can be programmed in Java, through an API created by the developers of the robot.

4. Discussion

The robots and simulators mentioned in the previous two sections have all been used in a tertiary education syllabus at some point in time. There are two primary reasons for why educators felt robots were a necessary tool to use in education. The first reason was that many educators felt that the programming skills acquired by the students after completing their introductory Computer Science courses were still not sufficient [12]. These educators felt that using robotics as a context could give the students a better understanding of programming concepts and problem solving skills by providing visual feedback of code execution. The second reason that educators chose to introduce robots into their introductory courses was to try and attract more students to choose Computer Science as their major, or to ensure that current Computer Science majors would not reconsider once enrolled [10, 14]. These two reasons can be classified into two different metrics: the first reason being quantitative, as it can be directly measured by the results achieved by the students at the end of the course, and the second reason being qualitative, as it deals more with the students' attitudes towards the course and Computer Science in general.

4.1 Existing Studies

Since there is no formal evaluation method for measuring the qualitative feedback of the students after the completion of the course, educators have chosen to create their own questionnaires or surveys, or simply interview the students and make notes on their experiences. The creators of the *e-puck* robot [17] evaluated their robot using two ratings by students, the first being whether they thought that the robot was a good tool to illustrate the concepts of the course, and the second being whether the robot performed well. These questions were rated between "totally agree" and "totally disagree", but a justification for the student's answers was never requested. Similarly in other papers, results are simply presented as *in a survey, x percentage of students liked interacting with the robot* [3], or merely stated that the course received favourable reviews [4, 6]. Exactly what these reviews were, or how the students expressed these opinions is never stated, nor the opinions of the students who did not feel that the courses were successful. These results, which overall were mostly positive, provide little insight as to what worked well and what did not, or how to improve the course so that it would be better received by students.

The previously mentioned experiments were all carried out without any control conditions. The results all appear positive, but we have nothing with which to compare. It is very possible that a more traditional Computer Science curriculum without robots may receive even higher student ratings. One such study [10] measured students' experiences in a robotics and non-robotics course, both of which taught the same underlying concepts. The students' experiences were rated using a survey of eight questions, administered to both robotics and non-robotics students at the beginning and end of the course. The feedback showed that on average, robotics students enjoyed the course more than non-robotics students. These results were then statistically analyzed and found to be just shy of being statistically significant. Other feedback from students revealed that having a laboratory environment with the necessary software pre-loaded would be a great improvement, as some laptops had compatibility issues. While these results were not statistically significant, they do provide strong evidence that using robotics as a context for education could improve students' learning experiences.

While qualitative results tend to be more vague and open-ended, quantitative results are easier to measure and interpret. Various studies have measured different quantitative values, such as the percentage of students who passed or failed the course [12], the percentage of students who continued to study Computer Science after the introductory robot-based course [8], or simply the results acquired by the students across all tests administered during the course. One study [12] was able to reduce the failure rate of an introductory Computer Science course by 77%, from a 45.8% failure rate, down to 9.8%. They did this by changing the introductory language to Python, and using a simulator called *Guido Van Robot*, which is a Python based implementation of *Karel The Robot*. While the results were not statistically analyzed, such a drastic decrease in the failure rates is difficult to ignore.

Fortunately, some studies have been done which focus on both the qualitative and quantitative results [1, 7, 9]. One such study [1] involved around 1500 students, 144 of which were taught using the robot-based curriculum. Other students in this experiment were students in one of three other introductory Computer Science courses. It was found that, out of these four courses, the students in the robotic-based course had the highest pass rate of 90.97%, the second highest pass rate of 85.71% was obtained by a course that taught the same concepts without robots. A final exam was administered to robotics and non-robotics students, examining the same concepts, and it was found that on average students in the robotics course performed 10% better than non-robotics students, a result that was proven to be statistically significant. It was also noted that the average number of students enrolled in the second year Computer Science course more than doubled after the robot-based course was introduced.

Lastly, and arguably most importantly, is a study performed by Barry Fagin, Laurence Merkle and Thomas Eggers. This study was conducted in 2001, for which a paper was published [9]. The following year, the authors published a quantitative analysis of their findings [7]. They found that out of the 948 students involved in the experiment, 175 of which were taught using the robot, the robotics students performed worse in every instance where a difference was detected. An extremely thorough statistical analysis of both the subjective and objective measures of this experiment was conducted, the findings of which were published [7] and greatly influenced the way in which the previous studies, discussed above, decided to approach robotics in education.

The objective measures analyzed were all students' exam and test results, as well as the number of students in the robotics and non-robotics course that declared a Computer Science related major at the end of the year. For the students' results, it was found that on average, non-robotics students performed better than robotics students on every test and exam. This result was proved to be statistically significant. The results were analyzed both as raw, unprocessed values, and again after being processed to remove the effect of each individual students GPA, such that each student's individual skills would not skew the data. The number of students of each course declaring a Computer Science related major was also recorded, but the value between them was not statistically significant.

The subjective measures of this study included a four question survey administered to both robotics and non-robotics students. This survey asked students to rate on a scale of 1-5 the course as a whole, the relevance and usefulness of the course, the amount that they learned and the effectiveness of their instructors. In all four questions, robotics students rated lower values than non-robotics. The students were also divided into focus groups, and asked to discuss the advantages and disadvantages of using robots as part of the Computer Science curriculum. The responses were then classified and totaled, the most common advantages being that the robots were creative and fun to work with, and the disadvantages being that the work was very time consuming, and that the robots were inconvenient to work with.

From this study, the authors concluded that one of the biggest disadvantages was that the students did not have access to a robot at all times, since the robots were required to remain in the labs. Without access to the robots, or a simulator, the students were only able to work on assignments during the designated lab sessions. The authors concluded that, if possible, students should be able to take the robots back to their rooms, or if that is not possible, a simulator should be provided, giving the students an opportunity to work from home and reflect on the work covered in classes.

4.2 Analysis of Studies

While a slight analysis of some of the studies has been done in the previous section, there are still some issues which must be addressed regarding all of the studies as a whole. It must be noted that, while they all provide useful information and results, none of the studies follows the scientific method when it comes to testing. Only one study [1] actually draws attention to this fact. In all studies where robotics and non-robotics courses are split, the students are able to choose which course they wish to attend. This skews the data since some students may have different skill sets and be more likely to perform better in one class over another. The only way to obtain fair results would be to populate both courses at random, to achieve a random distribution of students who may or may not have been predisposed to robotics.

Another gap in information common across many of the studies is that few of them detail the structure of the robot-based curriculum. The studies all detail the robot used in the course, and the hardware they come with, but do not go into detail as to how these components were used to improve the students learning experience. While the exact curriculum differs from institution to institution, the basic concepts taught in introductory Computer Science courses all remain the same. Only one paper [9] provided a breakdown of the curriculum, along with examples of how the robot was used to teach each important concept. In other studies, robots were evaluated with regards to qualitative and quantitative feedback, but the manner in which the classes were taught was

never analyzed. Robots are incredibly versatile machines, and can teach different concepts in many different ways. The qualitative feedback gathered in some studies [3, 4, 6, 15, 17] is often too general. Instead of asking students to rate the course as a whole, the course could be broken up into certain sections and rated separately, in order to find out which activities involving the robots were most useful, and which were not.

5. Conclusions

Robots and simulators have been used many times over the past few decades to provide a context for Computer Science education. They have proven that, if used correctly, they can provide a huge benefit to students, and greatly improve their learning experiences. Through analysis of existing studies, it can be concluded that if an institution wishes to teach a Computer Science course using robotics as a context, certain criteria must be met in order to maximize the chances for success: each student must have a personal robot, but if this is not possible, a simulator must be made available, and if possible, a computer lab must be available, with computers preloaded with necessary software. It can be concluded that there is still much left to be researched regarding the usage of specific robot components being used to teach specific Computer Science concepts. This produces an opportunity to evaluate certain features coupled with certain activities, with regard to how engaging they were, and how well they demonstrated the underlying concept. Such information could provide helpful to anybody considering robot-based education in the future.

6. References

[1] Summet, J., Kumar, D., O'Hara, K., Walker, D., Ni, L., Blank, D., & Balch, T. (2009, March). Personalizing CS1 with Robots. In *ACM SIGCSE Bulletin* (Vol. 41, No. 1, pp. 433-437). ACM.

[2] Schumacher, J., Welch, D., & Raymond, D. (2001). Teaching Introductory Programming, Problem Solving and Information Technology with Robots at West Point. In *Frontiers in Education Conference, 2001. 31st Annual* (Vol. 2, pp. F1B-2). IEEE.

[3] Rahul, R., Whitchurch, A., & Rao, M. (2014, December). An Open Source Graphical Robot Programming Environment in Introductory Programming Curriculum for Undergraduates. In *MOOC, Innovation and Technology in Education (MITE), 2014 IEEE International Conference on* (pp. 96-100). IEEE.

[4] Apiola, M., Lattu, M., & Pasanen, T. A. (2010, June). Creativity and Intrinsic Motivation in Computer Science Education: Experimenting with Robots. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education* (pp. 199-203). ACM.

[5] FACTS, Q. (2008). Designing Personal Robots for Education: Hardware, Software, and Curriculum.

[6] Xu, D., Blank, D., & Kumar, D. (2008, February). Games, Robots, and Robot Games: Complementary Contexts for Introductory Computing Education. In *Proceedings of the 3rd international conference on Game development in computer science education* (pp. 66-70). ACM.

[7] Fagin, B. S., & Merkle, L. (2002). Quantitative Analysis of the Effects of Robots on Introductory Computer Science Education. *Journal on Educational Resources in Computing (JERIC)*, *2*(4), 2.

[8] Saad, A., Shuff, T., Loewen, G., & Burton, K. (2012, March). Supporting Undergraduate Computer Science Education Using Educational Robots. In *Proceedings of the 50th Annual Southeast Regional Conference* (pp. 343-344). ACM.

[9] Fagin, B. S., Merkle, L. D., & Eggers, T. W. (2001). Teaching Computer Science with Robotics Using Ada/Mindstorms 2.0. *ACM SIG Ada Letters*, *21*(4), 73-78.

[10] Markham, S. A., & King, K. N. (2010, June). Using Personal Robots in CS1: Experiences, Outcomes, and Attitudinal Influences. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education* (pp. 204-208). ACM.

[11] Bahang, S. M. K., & Pendidikan, B. T. USING ROBOTICS IN EDUCATION: LESSONS LEARNED AND LEARNING EXPERIENCES.

[12] Yadin, A. (2011). Reducing the Dropout Rate in an Introductory Programming Course. *ACM inroads*, *2*(4), 71-76.

[13] Becker, B. W. (2001, February). Teaching CS1 with Karel the Robot in Java. In *ACM SIGCSE Bulletin* (Vol. 33, No. 1, pp. 50-54). ACM.

[14] Malec, J. (2001, March). Some Thoughts on Robotics for Education. In *2001 AAAI Spring Symposium on Robotics and Education*.

[15] Goldweber, M., Congdon, C., Fagin, B., Hwang, D., & Klassner, F. (2001, February). The Use of Robots in the Undergraduate Curriculum: Experience Reports. In *ACM SIGCSE Bulletin* (Vol. 33, No. 1, pp. 404-405). ACM.

[16] Saleiro, M., Carmo, B., Rodrigues, J. M., & du Buf, J. H. (2013). A Low-cost Classroom-oriented Educational Robotics System. In *Social Robotics* (pp. 74-83). Springer International Publishing.

[17] Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A. & Martinoli, A. (2009). The e-puck, a Robot Designed for Education in Engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions* (Vol. 1, No. LIS-CONF-2009-004, pp. 59-65). IPCB: Instituto Politécnico de Castelo Branco.

[18] Cooper, S., Dann, W., & Pausch, R. (2000, April). Alice: a 3-D Tool for Introductory Programming Concepts. In *Journal of Computing Sciences in Colleges* (Vol. 15, No. 5, pp. 107-116). Consortium for Computing Sciences in Colleges.

[19] Stehlik, M., Roberts, J., & Pattis, R. (2005). *Karel J Robot: A gentle introduction to the art of object-oriented programming in Java*. Dream Songs Press.

[20] Lenox, T. L., Woratschek, C. R., & Davis, G. A. (2008). Exploring Declining CS/IS/IT Enrollments. *Information Systems Education Journal*, *6*(44), 1-11.

[21] Yarosh, S., & Guzdial, M. (2008). Narrating data structures: The Role of Context in CS2. *Journal on Educational Resources in Computing (JERIC)*, *7*(4), 6.

[22] Leutenegger, S., & Edgington, J. (2007, March). A Games First Approach to Teaching Introductory Programming. In *ACM SIGCSE Bulletin* (Vol. 39, No. 1, pp. 115-118). ACM.

[23] Papert, S., & Harel, I. (1991). Situating Constructionism. *Constructionism*, *36*, 1-11.

[24] Pattis, R. E. (1981). *Karel The Robot: A Gentle Introduction to the Art of Programming*. John Wiley & Sons, Inc.

[25] The Institute for Personal Robots in Education official website: http://www.roboteducation.org/

[26] *LEGO Mindstorms* official webstore: http://www.robotshop.com/en/lego-mindstorms-parts.html