Gradus Vol 10, No 2 (2023)
ISSN 2064-8014

USING INNOVATIVE DEVICES IN INTERMEDIATE LEVEL OF PROGRAMMING EDUCATION

Éva Krisztina Szerémi 0009-0003-3593-8787 [*], *Attila Pásztor* 0000-0001-7354-5114

Department of Information Technology, GAMF Faculty of Engineering and Computer Science, John von Neumann, Hungary

https://doi.org/10.47833/2023.2.CSC.014

Keywords:
programming,
programming education,
elementary algorithms,
programmable robots,

Article history:

Received	15 September 2023
Revised	15 November 2023
Accepted	17 November 2023

Abstract
Several Hungarian and foreign research and experiments have already proven the effectiveness of programming education supported by innovative devices. These experiments have largely focused on the beginning learning cycle. With our short-term research we wanted to investigate whether the positive consequences of the use of these devices can be experience at the intermediate level of programming learning. In the framework of the course for second semester students BSc students in Computer Engineering at the John von Neumann University, they could try out the practical implementation of elementary programming theorems with programmable robots. The focus of our study was to investigate whether the tool-based form of teaching had a positive effect on the completion of the compulsory Algorithms subject. This study is the first step in a series of research studies.

1 Introduction

For the effective use of today's modern computing tools, it is essential for students leaving secondary and higher education to acquire the basic skills of programming, algorithmic thinking and the understanding and application of algorithms.

Even before the turn of the millennium, efforts to use concrete tools in the teaching of programming and algorithms had already begun. Simulation tools were used to make abstract algorithms informative by using objects on the screen. Such tools included Papert's turtle [1], Karel the robot [2] or the Spider World [3] used in Dalbey and Linn's studies. The programming teaching trends of the era were systematized by Brusilovsky and his colleagues, who highlighted the role of mini-languages for controlling various learning tools in the initial learning of programming [4].

Several experiments have been carried out to test the use of tools that enable specific operations at different ages. Experiential learning about the behaviour of specific robots is already possible at kindergarten age, as can be seen from the studies of Levy and Mioduser [5] or Istenes and Pásztor [6]. However, most studies use tools that can be applied in secondary and college education. The understanding of the hardware and software built by students and the failure modes were investigated by Kamada and colleagues [7] and Kurebayashi and colleagues [8], respectively. They mainly aimed at deepening the knowledge elements related to the technical tools using modern IT tools. Their results showed that the students enjoyed building and programming robots, and that the students in the experimental group were more successful in understanding the operation of embedded systems.

The use of LEGO's programmable tools, the Mindstorms robot family, in programming education is illustrated in several studies. Sartatzemi and colleagues [9] investigated the possibilities of teaching the basics of programming using robots in Greek secondary schools. Their pilot experiment was successful in improving students' programming skills despite the low number of

[*] Corresponding author.
E-mail address: szeremi.krisztina@nje.hu

lessons, but also showed that the understanding of concepts often remained incidental. Wu, Tseng, and Huang [10] investigated whether the effectiveness of teaching programming would change if programmes simulating its activities were used instead of physical robots.

It is worth mentioning that there is a growing body of research on supporting the learning of students with special educational needs with robotic devices, as discussed in more detail in Aknai & Fehér[11]

In many countries around the world, robots are used in higher education to develop students' algorithmic thinking, to develop programming skills and to make education more practical. In addition to the Lego NXT and EV3 robots mentioned above, TETRIX, Vex, TRIK robots are used in many places, while in other countries the use of Arduino robots is being promoted [12][13].

At the Department of Information Technology of GAMF Faculty of Engineering and Computer Science John von Neumann University we have been using programmable mobile robots for the education of engineering students for almost two decades. In the beginning we used RCX and later NXT robots mainly to educate beginner programmers. We hoped that robot-assisted programming education would make the courses more practical and effective, and that the playful teaching would motivate the students.

Research has also been carried out by Pásztor et al. [14] to demonstrate the effectiveness of robot use. Our control group experiments have shown that the use of robots in programming education can provide more hands-on programming and increase attitudes towards programming and promote a positive programming self-concept. The results of the experimental and control groups were compared at the beginning and at the end of the educational semester. Only the experimental groups used robots during the process. The measurements of the experiment demonstrated that the use of innovative tools did not lead to a significant increase in programming skills. However, in our study, new tools such as model robots can contribute to the development of learning motives, especially the programming self-concept. This could strengthen and broaden learners' motivational base and positively influence their further performance in programming.

2 Robot-assisted education for intermediate programmers

All of the above-mentioned studies focus on the early stages of programming education. We were curious to see whether the use of these tools has a similar positive impact on the next stage of programming education, whether it has an impact on student motivation, on the completion of concurrent subjects, or on the self-image of the programmer/engineer where the aim is to further develop the existing basic programming knowledge. The study is entirely pilot in nature, in terms of the appropriateness of the method itself and the research direction to investigate the above, and is the first measurement in a longer series of studies involving 10 students.

Our study was conducted among second semester BSc students in Computer Engineering who had already successfully completed *Programming I* in the autumn semester, which is followed by two compulsory courses in the spring semester - *Programming II* and *Algorithms and Data Structures*. In the foundation course, the following knowledge and concepts are acquired: algorithm, variables, programme structure, programming steps, keywords, operators, logical operations, type conversion, instruction repetition, loops, preprocessor, arrays, character arrays, text handling functions, input output handling functions, pointers, indirection, dynamic array, structure, file handling, functions, scope of variables. You will learn and practice some elementary algorithms in C++ in the first semester, but you will be introduced to the full range of algorithms in the second semester in the course *Algorithms and Data Structures*.

The University's mentoring programme provides first-year students with professional mentoring in the spring semester, during which they carry out a research project or a project assignment and present the results of their semester's work to their first-year peers and tutors at the HET Mentoring Conference. As part of this programme, we have launched an elective project work course called *Elementary Algorithms in Practice with Robots*. The aim of the course was to simulate real-life situations using model robots in which robots perform their tasks using an elementary algorithm, thus showing the real role of algorithms in programming and increasing student motivation for the subject of *Algorithms*.

For these reasons, the role of the instructor has changed to that of facilitator. Although at the beginning of the semester there was some theoretical knowledge (project, project life cycle, aspects and tools of project planning, list of elementary algorithms, professional presentation and lecture) to be given to the students, but during most of the semester the task was mainly to focus the attention of the teams on the actual tasks and the whole process. In order to present the project in a high quality at the conference, the task list included a video presentation of the implementation.

During the tutorial we used NXT Mindstorm robots, programmed in NXC in the Bricx framework. This was chosen because the language itself is a C-based programming language, so it uses many of the same or very similar syntax and control structures as the C++ programming language we had previously studied. Before starting the project, we had to introduce students to the robots themselves, their construction, their operation, the framework used to control them and the features of the programming language. [15]

The students worked in two teams on two separate projects. Three criteria were set for the tasks: 1. the programme must use at least two or more elementary algorithms (preferably different for each team), 2. it must implement communication between robots, 3. it must simulate a realistic, real-life situation.

One team's project is called *Boss in the Warehouse!* which involved modelling forklift truck routes. It was based on the premise that time is an essential element of efficiency in the life of a company, especially when it comes to the work of a manager, not to mention communication between organisational levels. Starting from a given point, one robot measures the distance between storage locations for a given type of product, following a set of routes, then sorts (using bubble sorting) and selects the closest one (with minimum selection) to the starting point. It sends the result to another robot - it is the boss, which, when searching for a given product, already uses the shortest path.

The other team's project is called *Something Dangerous!* It is based on the fact that there are countless situations where it is important to map an area, but it would be out of reach or too dangerous for humans to do so, and robots are used for these tasks. These jobs can serve humanitarian, environmental or research purposes. When a robot is in a defined area, it maps the amount of different hazardous substances (four types) in the area (by counting) and then transmits the property value of the most hazardous item (by selecting the maximum) to another robot, which searches for it (by searching) and removes it from the area.

3 Research findings

In the full survey - interview and questionnaire - 75% of the participating students took part. The survey covered three areas. Firstly, student knowledge of the subject *Algorithms and Data Structures*, secondly, student attitudes and attitudes towards the optional subject and the compulsory subjects *Programming II* and *Algorithms*, and thirdly, the impact of the subject *Algorithms in Practice with Robots* on previous knowledge of the subject, completion of parallel subjects, and the image of the programmer/engineer.

The results of the student knowledge test were as follows: the end-of-semester score in *Programming I* was 4.0 for the sample, while for the whole cohort of BSc Computer Engineering students it was 3.72, i.e. those who took this optional subject had a 0.28 better average in the prerequisite subject. For the two second semester subjects studied parallelly, the same figures are as follows: for *Programming II*, the end-of-semester score of the sample is 4.5 and the end-of-year score is 4.17, i.e. the previously existing difference has increased by 0.05 to 0.33 for those who took the subject. In the subject *Algorithms and Data Structures*, the sample scored 3.33 and the cohort 3.34, i.e. the students tested were 0.01 below the average of those who took the subject. However, performance on the practical part of the subject shows a different picture. The sample group all scored above 90% in the practical final exams, while the average for the year group was between 70-80%. The discrepancy is partly explained by the fact that the *Algorithms* course had a structure of 2 hours of lecture and 2 hours of lab per week and, as discussed above, the advertised course was highly practice oriented for 2 hours per week. On the other hand, the lecture was more extensive in its subject matter than the practical part, so the theoretical part of the lecture required students to cover more knowledge that was not covered in the lab. Regarding the compulsory *Algorithms* course, 46% of the respondents stated that they had experienced some difficulties in completing the course, while 50% had experienced some level of success in learning the course. In the same areas, the results are quite different for the optional subject. 29% of respondents had difficulties in completing the subject, but 79% had experienced success. This result is in line with the results of research on the positive impact of tool-supported programming education.

The results of the analysis of students' attitudes towards the subjects further confirm these experiences. In the lab classes of the subject *Algorithms and Data Structures*, the average number of absences was 2 times per semester, while in the subject *Elementary Algorithms in Practice* it was 0.02 times per student. In the former, 50% did not review the previous material before the labs, 50% only before the exams, in the latter 100% of the students reviewed the previous material. The same result of 100% was about the extent to which they felt the subject was useful in their training, while for the compulsory subject 1/3 did not see it as useful at all and 2/3 saw it as partially useful in their engineering training. Nevertheless, 50% were fully satisfied and 50% were partially satisfied with their end-of-semester performance.

An analysis of the impact of the optional subject in the context of professional mentoring on the parallel subjects yielded somewhat surprising and unexpected results. Concerning the successful completion of the *Algorithms and Data Structures* course, 67% of the respondents felt that the course had contributed to success. From the point of view of the fact that the subject was actually launched as a support subject in addition to the compulsory subject, this result may seem low, but if we look at the facts mentioned earlier, in terms of the themes of the two subjects, this is a satisfactory result.

The most unexpected result was the response to the basic course *Programming I*, as 83% of the students said that this course helped them to organise and deepen their knowledge. This fact probably contributed to the fact that the semester results for the *Programming II* subject showed a further increase in the sample group's performance compared to the year group. Thus, the *Elementary Algorithms in Practice with Robots* subject not only had a direct impact on the successful performance of this concurrently taught subject, but also an indirect impact through the prerequisite subject.

92% of respondents strongly agreed with the statement that they felt that they had really benefited from completing the subject in their engineering education. This is compounded by the fact that none of the students who took the course had previously used robots in their studies, nor were they familiar with their potential or the programming language used to control them. The survey also revealed that they actually had very little information about how widespread the use of robots in everyday life actually is. Given this, it is not surprising that all of them clearly perceived that their knowledge of both programming and the uses of programming had increased as a consequence of completing the course. However, the fact that 100% of them said that the course helped them to recognise where and how elementary algorithms can actually be applied in real-life computer science work supports previous research in this area that tool-supported programming instruction also helps students to understand how systems work more effectively.

For the free-response question on how you would describe the biggest positive aspect of the subject for you, the answers fall into 4 categories in terms of frequency of responses. Firstly, the teamwork itself, the collaboration on the project, and secondly, learning how robots work and discovering their potential. Thirdly, the participation in a professional conference, the presentation of professional work, and fourthly, the fact that they recognised and understood the importance and role of elementary algorithms in the programming work process.

4 Summary

Overall, despite the small sample size, the study has clearly demonstrated that innovative tools can enhance the effectiveness of education in two ways at the higher, more advanced levels of programming. On the one hand, they directly support the successful completion of subjects with similar content taught parallelly, and on the other hand, they indirectly support the successful completion of subjects through the deepening of existing basic knowledge. It has also become clear that it is worthwhile to continue the study and to extend the research sample, and to investigate with follow-up the short- and long-term impact of the perceived positive student attitudes in the areas of professional development and career loyalty. The very nature of the subject itself is also thought-provoking in terms of the role that a similarly structured subject can play in the development of students who are either struggling or otherwise lagging behind or performing poorly, as opposed to a catch-up, tutoring subject using traditional repetition methods.

References

[1] Solomon, C. J. & Papert, S.: A case study of a young child doing Turtle Graphics in LOGO. In: Proceedings of the June 7-10, 1976, national computer conference and exposition. 1976. pp. 1049-1056. DOI: https://doi.org/10.1145/1499799.1499945

[2] Pattis, R. E.: Karel the robot: a gentle introduction to the art of programming. Wiley & Sons, 1984Hoboken, NJ.

[3] Dalbey, J. and Linn, M.: Spider World: A robot language for learning to program. Assessing the cognitive consequences of computer environments for learning (ACCCEL). Paper presented at the Annual Meeting of the American Educational Research Association, New Orleans, LA, 1995. April. Electronic source: http://www.eric.ed.gov/contentdelivery/servlet/ERICServlet?accno=ED250178 pp. 2-28.

[4] Brusilovsky, P. et al. : Teaching programming to novices: a review of approaches and tools. In: Educational Multimedia and Hypermedia. Proceedings of ED-MEDIA 94 – World Conference on Educational Multimedia and Hypermedia, Vancouver, Canada, June 25-30. 1994. pp. 103-110.

[5] Levy, S. T. & Mioduser, D.: Does it "Want" or "Was it programmed to..."? Kindergarten children's explanations of an autonomous *robot*'s adaptive functioning. International Journal of Technology and Design Education, 2008. **18.** 4. pp.337-359. DOI: https://doi.org/10.1007/s10798-007-9032-6

[6] Istenes, Z. és Pásztor, A.: Using Innovative Devices in the Education of IT from Primary to University Level VII. International Electrotechnical and Computer Sience Conferencee 2007. Portoroz, Slovenia ISSN 1581-4572 Volume B pp.133-136.

[7] Kamada, T., Aoki, H., Kurebayashi, S. & Yamamoto, Y.: Development of an educational system to control robots for all students. In: Mittermeir, R. T. & Syslo, M. M. (eds.): Informatics Education – Supporting Computional Thinking. Poceedings of the Third Conference on Informatics in Secondary Schools – Evolution and Perspectives, ISSEP 2008. Torun, Poland, July 1-4. pp.63-74. DOI: https://doi.org/10.1007/978-3-540-69924-8_6

[8] Kurebayashi, S., Aoki, H., Kamada, T., Kanemune, S. & Kuno, Y.: Proposal for teaching manufactoring and control programming using autonomous mobile robots with an arm. In: Mittermeir, R. T. & Syslo, M. M. (eds.): Informatics Education – Supporting Computional Thinking. Poceedings of the Third Conference on Informatics in Secondary Schools – Evolution and Perspectives, ISSEP 2008. Torun, Poland, July 1-4. 75-86.

[9] Sartatzemi, M., Dagdilelis, V. & Kagani, K.: Teaching programming with robots: a case study on greek secondary education. Lecture Notes in Computer Science 2005, 3746. pp.502-512 DOI: https://doi.org/10.1007/11573036_47

[10] Wu, C., Tseng, I. & Huang, S.: Visualisation of program behaviors: physical robots versus robot simulators. In: Mittermeir, R. T. & Syslo, M. M. (eds.): Informatics Education – Supporting Computional Thinking. Poceedings of the Third Conference on Informatics in Secondary Schools – Evolution and Perspectives, ISSEP 2008. Torun, Poland, July 1-4.2008. pp. 53-62. DOI: https://doi.org/10.1007/978-3-540-69924-8_5

[11] Aknai, D. O. & Fehér, P.: Kalandozások robotméhecskével – problémamegoldás, gondolkodásfejlesztés padlórobotokkal. Debreceni Egyetemi Kiadó – IKT MasterMinds Kutatócsoport 2009.

[12] [12] López-Rodríguez, Francisco M., and Federico Cuesta. :Andruino-A1: Low-cost educational mobile robot based on android and arduino. Journal of Intelligent & Robotic Systems 81.1 2016. pp 63-76. DOI: https://doi.org/10.1007/s10846-015-0227-x

[13] Chebotareva, E; Gavrilova, L.: Educational Mobile Robotics Project" ROS-Controlled Balancing Robot" Based on Arduino and Raspberry Pi. 12th International Conference on Developments in eSystems Engineering (DeSE) 2019. pp. 209-2014 DOI: 10.1109/DeSE.2019.00047

[14] Pásztor, A., Pap-Szigeti, R., & Lakatos Török, E.: Effects of Using Model Robots in the Education of Programming. Informatics in Education, 2009. 9(1), pp.133-140.

[15] Kiss R. – Pásztor A.: Mobil robotok programozása NXC és NXT-G nyelven, Kecskemét, Magyarország, Kecskeméti Főiskola Gépipari és Automatizálási Műszaki Főiskolai Kar, 2009, 106 p.