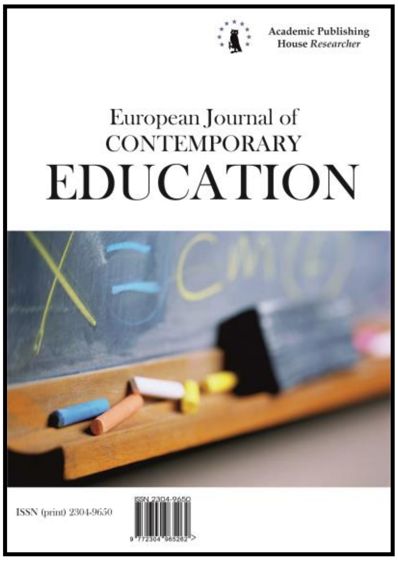
Published in the Slovak Republic
European Journal of Contemporary Education
ISSN 2304-9650
E-ISSN 2305-6746
2017, 6(3): 390-400
DOI: 10.13187/ejced.2017.3.390
www.ejournal1.com



Innovative Methods in Teaching Programming for Future Informatics Teachers

Janka Majherová [a], Václav Králík [a, *]

[a] Department of Computer Science, Faculty of Education, Catholic University in Ružomberok, Slovakia

Abstract

In the training of future informatics teachers the students obtain experience with different methods of programming. As well, the students become familiar with programming by using the robotic system Lego Mindstorms. However, the small number of Lego systems available is a limiting factor for the teaching process. Use of virtual robotic environments seems to be a suitable alternative for dealing with an insufficient quantity of hardware tools. The resulting programs are created and tested in the virtual laboratory and can be subsequently implemented into a real robot model. In such cases, teaching no longer depends on the available number of hardware kits and the form of teaching can be changed from group to individual. This paper describes our experiences with students' learning with the robotic system Lego Mindstorms, programming environments Bricx and virtual educational environment ROBOTC. One approach to making teaching programming language attractive is the use of robotic kits and virtual environments in the classroom.

Keywords: programming of robot, secondary education, future teacher, Lego Mindstorms, Robot Virtual World, ROBOTC.

1. Introduction

In the current education system we can see visible efforts to modernise the education process. Teachers are becoming more interested in continuing their education and implementing projects with the use of digital technology (DT). Schools are refurbishing teaching facilities with modern educational equipment (technology) so that teachers can implement DT support for teaching various disciplines. The advent of interactive technology enhances research in all

* Corresponding author
E-mail addresses: janka.majherova@ku.sk (J. Majherová), vaclav.kralik@ku.sk (V. Králík)

disciplines and levels of education. However, technology alone is not enough. Quality digital content is needed and it is currently available, among other means, via the Internet.

An essential component of the younger generation's digital literacy is the ability to understand information and use it in different formats from the different sources presented by DT. According to the study conducted by the Institute for Public Affairs (Velšic, 2013), the average value of the digital literacy index in Slovakia increased from 0.33 points (on a scale from 0 to 1) in 2005 to 0.47 points in 2013. The study shows that schools have the highest impact on the improvement of digital literacy. In this research the young people of Slovakia had results approximately 60 per cent better than the overall population average, and during the past decade they have improved their skills and abilities in the most commonly available DT. An improvement of 50 per cent is seen in the ability to work with hardware and peripheral devices such as computers, tablets, smart phones, scanners, printers and portable media. On the other hand, slower growth has been observed in the ability to work with software (editing text, tables and graphics, multimedia and web browsers).

The research of e-skills for the job market in Slovakia has shown that the younger generation still lacks many skills and abilities in DT (Velšic et al., 2016). Among the respondents aged 18 to 26 only 14 per cent claimed that they lacked nothing in this respect. Fifty-seven per cent of young people claimed that they lacked such skills as application programming, system design, website development and multimedia. This problem is closely related to the inadequate preparation for problem solving and inadequate development of algorithmic and logical thinking reported by the other 42 per cent of respondents.

Therefore, we focus on preparing future teachers, as well as practising teachers in continuing education programs, for working with the new developments in modern DT. Extending learning objectives to the digital dimension with the appropriate use of ICT increases the effectiveness of teaching (Nagyová, 2015; Hubwieser et al., 2015; Jacková, 2008).

In order to facilitate the continuing and effective integration of DT and increase the quality of learning processes we can explore and observe potential ways to promote and encourage innovation. The use of educational robotic kits stimulates and motivates students. Motivation in this form of education is based on the method in which students, according to their own proposals, use kits to create a device (a robot, a vehicle, etc.). They program certain features, procedures, behaviour or actions to be performed by the device. Such an interconnection of software and hardware shows how the knowledge of a programming language can be applied in practice (in fields such as automation and process control). The price of educational robotic kits is the factor that determines the quantity of such kits in schools. The solution can be the use of virtual laboratories that can simulate a hardware device and real environment.

2. Teaching programming in real and virtual environments

Teaching programming language in secondary school education has certain specific features which are determined by many factors. One of the factors that influences learning is the complexity of the language. This is the moment to ask oneself: when is the best time to start teaching programming and which language is most appropriate for the specific age group of the students? Our requirement for language selection was the facilitation of continuing education for students, taking into account the intellectual development of students within the scope of secondary school education, as well as the possibility of combining the use of these programming resources for both the lower and upper levels of education.

Interactive environments for teaching programming reinforce the role of visualisation. According to Musa et al. (2015), visualisation in educational environments can provide a simple and effective approach to obtaining results, to problem-solving and to discovering the structure of the model during the process of students learning new information. The visualisation of relations and logical connections within a single model allows us to support students' digital and basic competencies in science and technology. Modelling on the lessons of informatics is not only an instrument, but even the very subject of education, when the students, based on gained knowledge and with the help of digital tools, create a model of a certain part of the real world (Majherová, 2007; Gunčaga et al., 2015).

Educational robotic systems offer several possibilities for teaching programming. According to Saleiro et al. (2013) and Benedettelli (2014), the LEGO Mindstorms building kit has proven to be an appropriate tool. In addition to various hardware accessories, it also contains basic software that allows teaching programming to students from the age of 8 years. The development and programming environment NXT-G is an iconic type. Students are not directly confronted with the syntax of the language, but the environment allows students to create an algorithm for a simple program (Fig. 1).

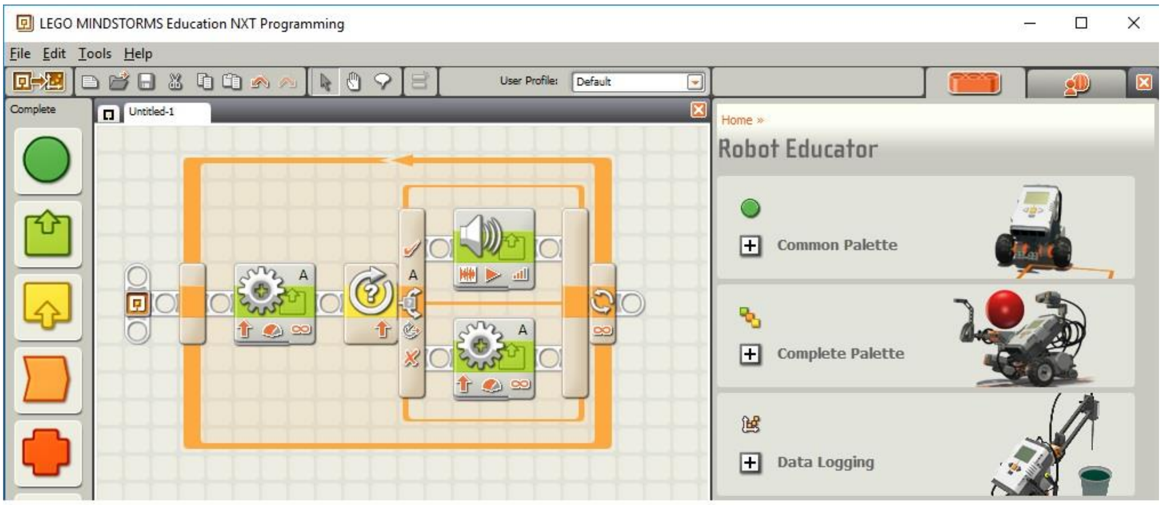


Fig. 1. The Lego Mindstorms NXT environment

Using Lego Mindstorms at a higher level of secondary school education, one can work in the Bricx Command Center (Bricx, 2016) programming environment, which supports multiple programming languages. The advantage of this teaching method is that a student, who was able to create a program in the iconic language and understood the function of iconic commands, can work then with the equal building kit in C language (Fig. 2). The visual connection of brands with the icons of the NXT-G program and commands in the Bricx environment allows the student to assimilate graphical information about the link between the theoretical registration of the solved problem and the icons of commands.

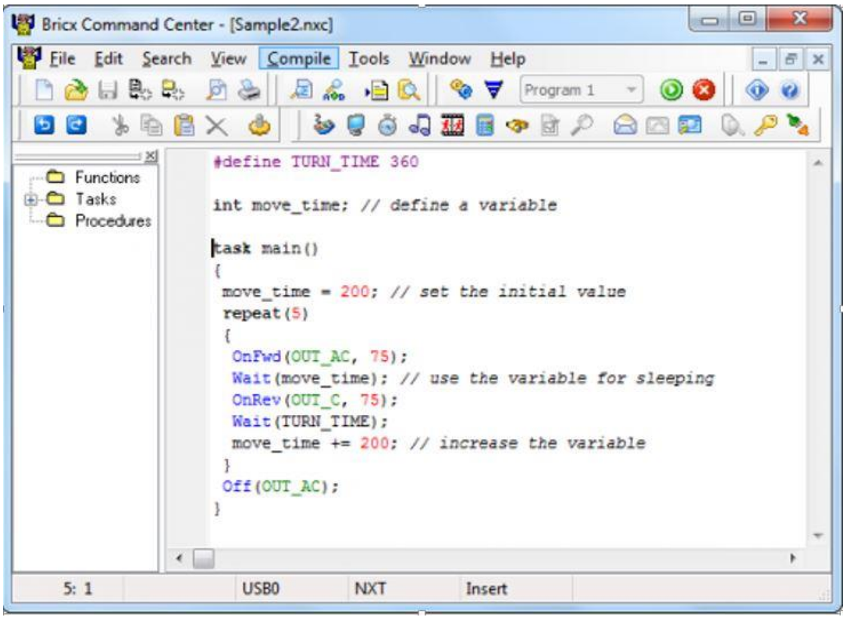


Fig. 2. The Bricx Command Center environment

For programming of microcontrollers it is recommended to make the structure of a programming language as similar as possible to the programming language used in the development of software applications in PC platforms. C language contains libraries for programming sensors and a compiler that will convert the program into the language of a given type of microcontroller. For teaching programming we can use programming language ROBOTC (ROBOTC, 2016). This language is for writing and debugging programs and at software level it offers a comprehensive compiler (real-time debugger).

Liu et al. (2013a), conducted experiments with the ROBOTC and the Robot Virtual Worlds environment (RVW). They wanted to verify how RVW could be used to teach novice programming skills. Students used a combination of the RVW tabletop simulations and the fantasy-based Palm Island programming environment to learn basic programming. One class completed a ROBOTC programming course using physical VEX robots (the Physical class), while the other class completed a ROBOTC programming course using virtual VEX robots (the Virtual class).

According to Liu et al. (2013b), both the Physical class and the Virtual class showed equal learning gains. The type of learning did not differ between the two classes either, as was evidenced by the equal learning gains. The Virtual class did show a time reduction benefit, as they completed the course earlier than the Physical class, with no effect on their overall learning. This suggests that working with the virtual robots allowed students to learn more efficiently in this context, as compared to working with physical robots. The students in the Physical class had to deal with the daily robot setup, additional mechanical issues and the clean-up required when working with a physical robot. Consequently, the teacher spent much more of his time in the Physical class helping students with robot communication and mechanical and class organisation issues. In the Virtual class, the teacher and his students were able to focus 100 per cent of their time on learning programming.

3. The experience

We were looking for tools to support the teaching of programming robots for bachelor degree students who specialise in computer science teaching. The students have the opportunity to familiarise themselves with this technology and way of teaching it, and apply their knowledge in their teaching practice after graduation (Králík, Majherová, 2016).

The content of the robots programming course is adjusted to this goal (Table 1). The course takes place in a computer lab. During the introduction students will explore the history of the development of robotic kits from Lego as well as from other producers. They will become familiar with software tools (languages) used for programming.

Table 1. Content of the course Programming of robot

Topics	Number of hours Full time/ external form	Goals
Robotic kits, software tools	2/1	Introduction to robot programming
NXT-G environment	4/2	to solve basic programming tasks
Bricx environment	6/1	to solve basic programming tasks
Construction of robot	4/1	to construct a robot
ROBOTC language RVW virtual laboratory	4/2	to use a virtual environment for robot programming
Tutorials and instructions	4/1	to use tutorials

Students are gradually introduced to the NXT-G environment and to a higher-level programming language in the Bricx development environment. Students are divided into groups and program a robot constructed from the LEGO Mindstorms kit. In this part of the course students will become familiar with the general requirements of virtual labs as well as with the possibility of using virtual laboratories in teaching programming languages. In the practical part of the course they will become familiar with the ROBOTC language and with the RVW virtual laboratory. From the perspective of a teacher, it is very important to motivate students to study programming. To provide this motivation, games and competitions may be considered. Therefore, students will receive information about the scope and rules of national and international competitions in programming robots, for example First Lego League (FLL, 2016) or Istrobot (2016).

The last part of the robots programming course is focused on working with tutorials and instructions. Prospective teachers will become familiar with video tutorials for teaching support from Lego and free guides on the web. Students acquire teaching methodology and working practices for the programming of robots with the help of various manuals.

4. Experimental Setup

The experiment lasted for 1 academic year 2015/2016 and involved 13 participants. The students were divided into two groups: a full-time students group (8 students) and a group of external students (5 students). We tried to compare two approaches to teaching robot programming. In a full-time form of education we taught programming with the use of a physical model of a robot and the NXT-G programming language, as well as the advanced Bricx Command Centre language. In the distance course we used the virtual robotic environment RVW, as well as the ROBOTC program (Table 2). We tried to observe the sequence of key tasks from simple to complex: from the simple iconic programming language up to the programming of a virtual robot.

Table 2. Methods for the robot programming

	Full-time form	**External form**
hardware	a physical model of a robot	virtual robotic environment
software	NXT-G Bricx Command Centre	ROBOTC program

In order to compare the two approaches, we used a robot constructed according to the instructions designated as “BASE”. This robot has a chassis with two engines and one supporting wheel, and therefore we are dealing with a differential control. The basic sensors from the Lego Mindstorms kit are placed on the robot: touch sensor, light sensor, ultrasound sensor and microphone. This robot can be constructed from the NXT or EV3 Lego kits (Fig. 3). For experiments in the process of teaching programming we use algorithms for the management of a differential gear.

Fig. 3. LEGO Robot “Base” (tutorial NXT-G)

In the context of teaching robots programming in physical environments we use different types of tasks for students. The goal of the project is to compile a program that uses automatic control of the robot’s movements based on values measured by sensors. In this task two sensors are used for the robot's motion control. The first sensor counts the engine turns; the second sensor provides an ultrasonic measurement of the distance.

Task A: Robot with ultrasonic sensor

Construct a robot that can move forward and stop according to a distance measured by the ultrasonic sensor. Attach an ultrasonic sensor to the robot that points forward. Write a program which makes the robot move towards the wall and then turn back, so it stops at the same place it started.

Important note: The program should function properly regardless of the robot's distance from the wall. The robot's distance from the wall is measured as a variable. The correct solution in the NXT-G environment is shown in Fig. 4.

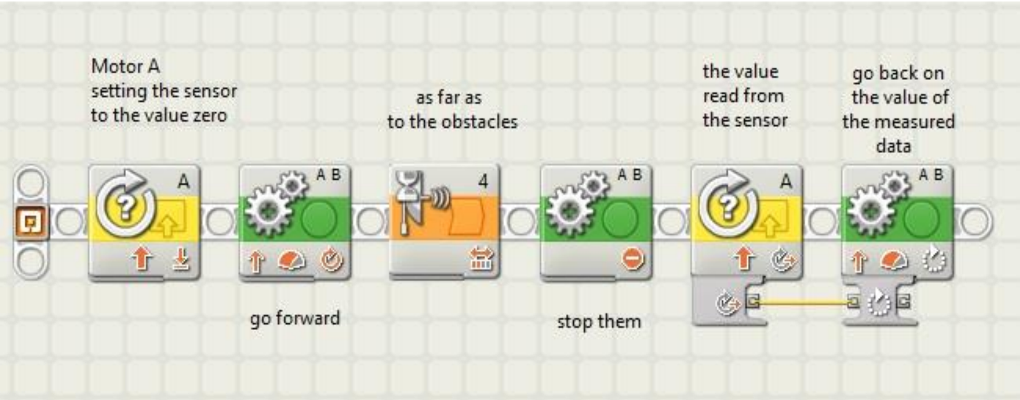


Fig. 4. Correct solution for task A

Task B: Counting lines

For this task we use the already-created chassis from the robot in task A. To the chassis that can move forward and has an ultrasonic sensor that allows measurement of the distance, attach the sensor of light. Its location is important. The measuring part of the sensor has to point downwards. The distance between the measuring part and the pad must be up to 0.5 cm above the pad.

Create a surface with lines according to Fig. 5 to the white paper, measure values for white and black colours using the "View" menu on the NXT brick and write them down.

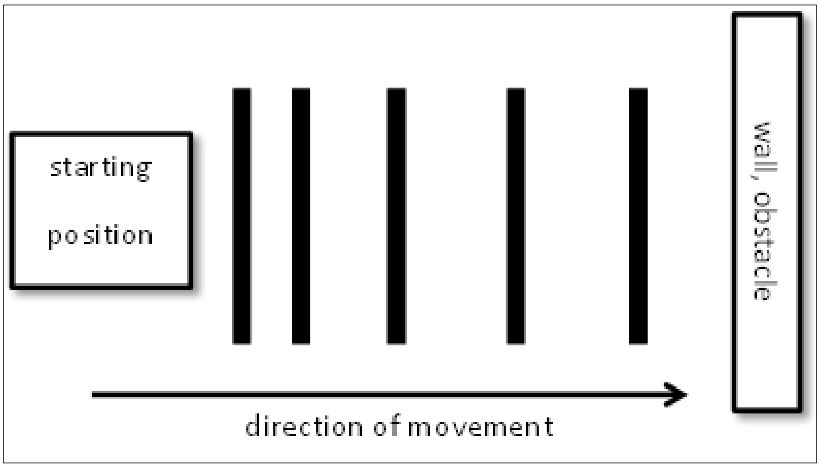


Fig. 5. Drawing of the area

Write a program in which the robot moves forward; with each crossing of the black line the robot should beep, and when it comes to the wall it should stop. Modify the program, so that after stopping it will whistle how many crossed black lines it has recorded.

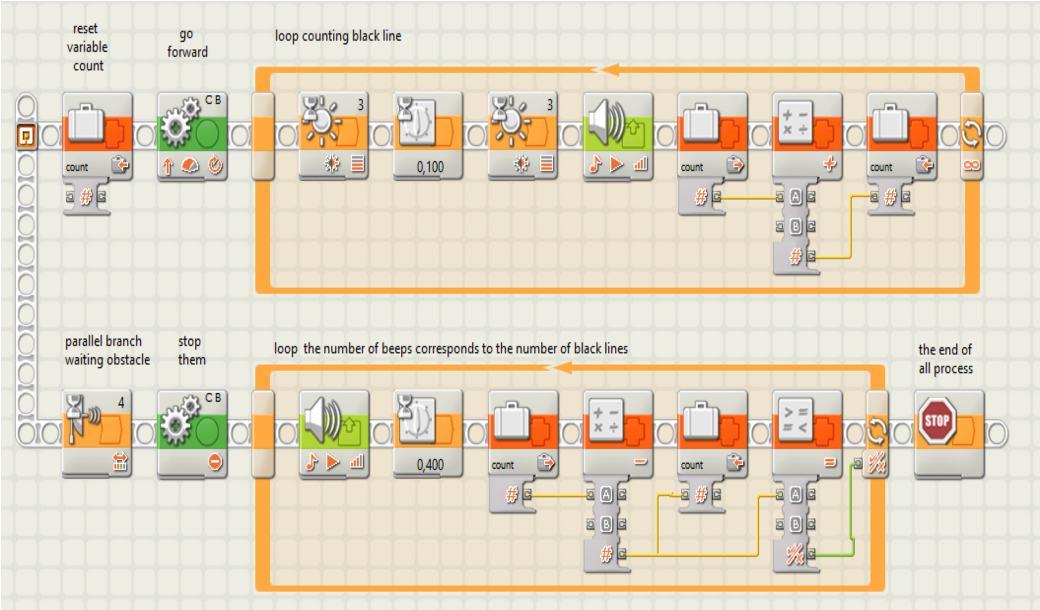


Fig. 6. Solution for task B

Solutions for the tasks A and B (Fig. 4 and 6) are easier to find in a graphical environment. For learning programming it is important to understand by which algorithm the specific task can be solved and through which sequence of commands. The NXT-G environment is graphical and facilitates understanding of the solution. If the assignment is understood in this way, it can then be programmed in a higher-level programming language in the Bricx Command Centre environment. For teaching of programming we use a combination of development environment NXT-G and a higher-level programming language. The advantages of this approach are the visualisation of order sequence and the graphical display of work with variables in the program. Ultimately, this can lead to an effective understanding of work with a higher-level programming language and a faster understanding of the assigned task.

For the second approach to teaching robots programming we used the ROBOTC programming language and a virtual robotic laboratory. ROBOTC, the higher-level programming language, is derived from C++ language, has an online compiler and the capacity to transfer the created program to the robot.

The great advantage of the program ROBOTC is its link to a virtual environment. The virtual model of a robot is identical to the real robot, called BASE. The visualisation of the motion and interactions of the robot in the virtual laboratory take place according to the control programs developed by the students. The visualisation of the motion takes place in the 3D environment and the mutual interactions between the robot and its environment can be observed from different angles.

The advantage to this way of teaching is that we do not need a physical model of the robot, because the verification of the correct operation of the program is performed by using the 3D model of the robot in a virtual environment (Fig. 7).

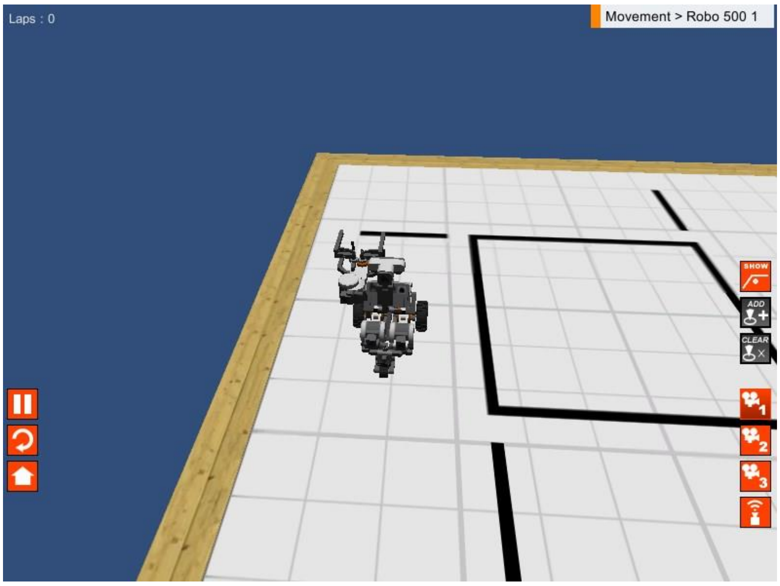


Fig. 7. ROBOTC virtual environment

Programming within a virtual environment is divided into several separate parts which are focused on the movements of the robot, programming the robot sensors and working with variables as well as on controlling the robot. The different programming tasks were prepared in each part of the virtual environment. The correct sequence of the algorithm can be always determined by the interaction between the virtual robot and the virtual environment. Students received visual information about the fulfilment of the task.

5. Results

Full-time students worked only with physical Lego Mindstorms kits in the NXT-G or Bricx Command Center environments in the classroom 2 hours per week. The external students worked with physical models of robots in the classroom 4 to 6 hours, as well as conducting self-directed learning outside the classroom with the use of the virtual environment. In the experiment, we verified two approaches for teaching the programming of robot models in the preparation of future teachers of informatics. We examined their advantages and disadvantages. In a small group of students we assessed qualitative results on the basis of observation and the use of a questionnaire.

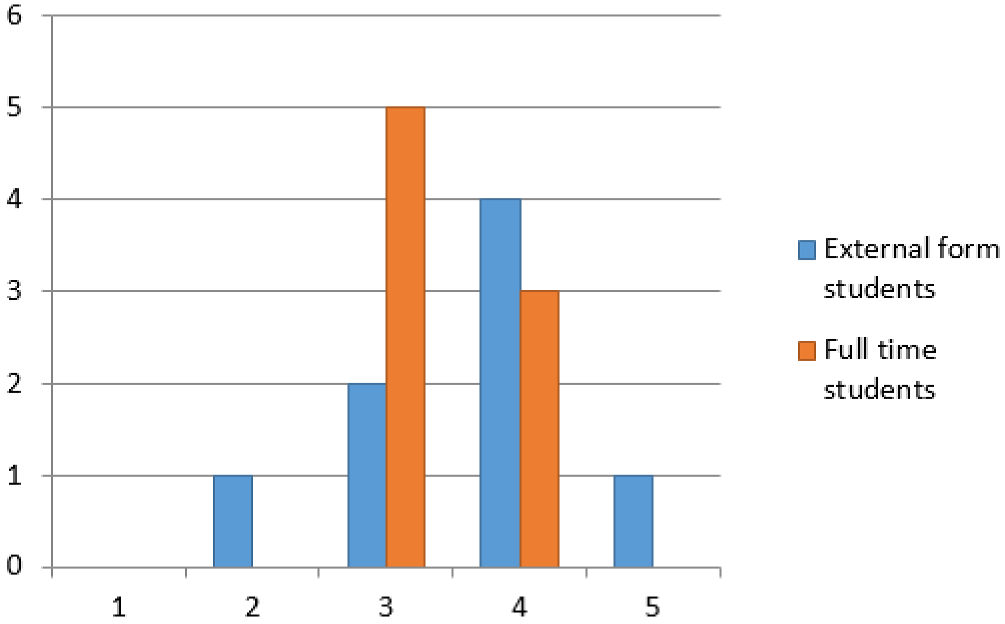
In the robot programming course we worked with 8 full-time students and 5 external students. At the end of the course the students completed a form with the following questions:

- In which programming environment do you know how to work?
- Have you already programmed a robot?
- Did you work with real robot model during the course?
- Did you work in LEGO Mindstorms during the course?
- Did you work in a virtual environment during the course?
- How do you evaluate your skill in creating a program for the robot? (1 – no skill, 5 – excellent skill)
- How do you evaluate your skill in creating a model of a robot?
- How do you evaluate your skill in working in a virtual environment?
- The final question was open: Assess the benefits of working in a virtual environment compared to the real environment for programming robot models.

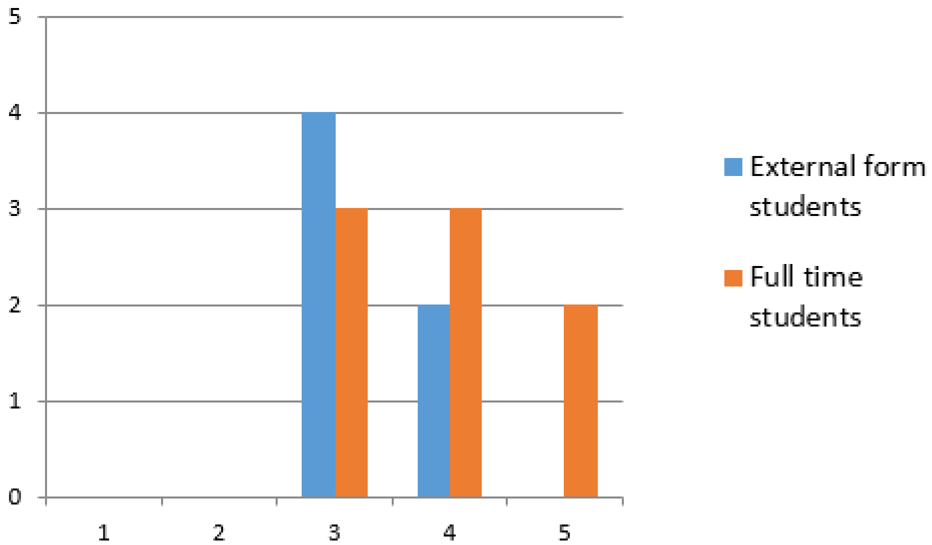
Before this course all students have worked in Pascal and C languages; some also knew the children's programming languages Imagine Logo or Scratch. They had encountered robot programming mainly in college; only 5 students reported experience with robots from secondary school.

Full-time students had to divide the study time allowed for this subject into two parts. The first part dealt with the construction of a physical model of a robot and the second part was devoted to the creation of a program. In most cases, the students focused more on building a robot than on creating a program. Full-time students rated their skills for the robot programming with an average score of 3.4 on a scale from 1 – no skills to 5 – excellent skill (Graph 1). Skills in making robot models were rated with an average score of 4.5 (Graph 2).

External students, with a shorter study time allowed for the subject, had fewer opportunities to work with physical models, so a greater emphasis was placed on work in the virtual environment. They rated their skills in constructing a model of a robot with an average score of 3.3 (Graph 2). But the skills for programming the robot were reported as higher, with an average score of 3.75 (Graph 1).



Graph 1. Skills to Create a Robot Model Program (1 – no skill, 5 – excelent skill)



Graph 2. Skills to Create a Robot Model (1 – no skill, 5 – excelent skill)

In the questionnaire we asked students how they see the benefits of working with a robot model in a virtual environment compared to a physical model. These are examples of their responses:

"...in a virtual environment, we can simulate the movement of the robot before we put it into action in a real environment. Working in a virtual environment is time-efficient and financially more profitable"

"...there is a reduction of cost associated with the production of robots, there is also a free access to the virtual laboratory (anytime, anywhere), there is no risk of injury, the virtual environment provides diagnosis of the source code that controls a particular source or combination of sources, simulates various alternatives, there is a compatibility with a wide range of programming languages"

"...we have the opportunity to participate in robotics exercises without direct access to the robot, the automatic online correction of errors in handling the robots in virtual environment is available"

"... new skills in the area of programming"

"...the virtual environment is simpler and takes less time, each can work independently, can work at home via the network at any chosen time"

6. Discussion and conclusion

From an analysis of learning outcomes, we can make several conclusions about the advantages and disadvantages of using physical and virtual approaches to programming robots.

If we use a method consisting of a combination of the NXT-G environment, a higher-level programming language and a real robot, we observe an excellent visibility of a commands sequence for building a program. This is due to the fact that in the NXT-G environment the orders are created with the help of icons. This method is suitable for simple robot motion-control programs, as in tasks A and B mentioned above. In a more complex program the number of commands increases and they take up space as icons on the screen. At some point, if the program becomes too complicated, the advantage of good visibility is lost. Verifying the proper operation of the program is performed on a physical model of a robot in real life conditions. We consider this method of teaching programming to be time-consuming because it is necessary to create and verify tasks prior to their application in educational practice.

Studying with the LEGO Mindstorms kits is motivating for students due to their interaction with the robot that they have to build. However, we are also concerned by the situation in which the mechanical building of a robot takes up the greater part of the teaching time compared to the time allowed for creating a program. The fact that the kits require regular maintenance, as well as taking up storage space after the end of the course, may also be considered a disadvantage.

We also see benefits in the method of teaching programming with the use of the virtual environment, because it provides a substitute for a physical kit. The use of the programming environment is intuitive. The virtual board on which the robot moves is a standard for all experiments; the parameters do not change. The view of the virtual board can be switched between different appearances. Students learn to program a virtual robot as if they had available a physical robot assembled from a kit. Programming takes place in a higher-level programming language. The program created by students can be sent to a teacher as a text file.

The Robot Virtual World programming environment supports only C language. From this perspective, the program appears to be unsuitable for use in grades K4 to K6. We consider this software to be suitable for the higher grades: K7 to K12. The virtual environment allows students to fully concentrate on the tasks associated with programming. Mistakes caused by faulty parts of the robot (sensors, batteries, etc.) are avoided. In addition to direct instruction, it is possible to use this method for distance learning or for other forms of learning. The virtual environment is the software that fully replaces the Lego kit.

In our future research we will explore programming robotic kits connected Lego with module Arduino and Raspberry. According to Polčin et al. (2016), Šnajder and Guniš (2016), in preparing future teachers of computer science it is important to implement new knowledge, methods and forms within the teaching of programming languages, in such a manner that the teaching is made attractive and is interconnected with application outcomes.

7. Acknowledgements

This work is supported by the Research Grant **KEGA 009KU-4/2017** Innovative methods in teaching informatics in secondary education and by the Grant **KEGA 002UJS-4/2014** Interactive electronic learning materials to support implementation of modern technology in teaching mathematics and informatics.

References

Benedettelli, 2014 – *Benedettelli, D.* (2014). The Lego Mindstorms EV3 Laboratory. San Francisco: No Starch Press. Retrieved from http://robotics.benedettelli.com/publications

Bricx – Bricx Command Centre. Retrieved from http://bricxcc.sourceforge.net/

Firstlegoleague – URL: http://www.firstlegoleague.org/

Gunčaga et al., 2015 – *Gunčaga, J., Janiga, R. and Žilková, K.* (2015). Supporting of simulation and visualization through ICT in the educational process. Information and Communication Technology in Education 2015. pp. 25-33. Ostrava: University of Ostrava.

Hubwieser et al., 2015 – Hubwieser P., Giannakos M. N., Berges M., Brinda Torsten, Diethelm I., Magenheim J., Pal Yogendra, Jacková J. and Jasute E. (2015). A global snapshot of computer science education in K-12 schools. In: ITiCSE-WGP' 15: proceedings of the 2015 ITiCSE on Working Group Reports, July 4–8, 2015, Vilnius, Lithuania. New York: ACM, 2015. pp. 65-83.

Jacková, 2008 – *Jacková, J.* (2008). Learning for Mastery in an Introductory Programming Course. In ITICSE '08 Proceedings of the 13th annual conference on Innovation and technology in computer science education. pp. 352-352. New York: ACM.

Králík, Majherová, 2016 – *Králík, V. and Majherová, J.* (2016). Virtual Robotic Laboratories Robot C in the Informatics Teachers Training. In DidMatTech 2016. Budapest: Eötvös Loránd University.

Liu et al., 2013 – *Liu, A., Newsom, J., Schunn, Ch. and Shoop, R.* (2013). Students Learn Programming Faster Through Robotic Simulation. Techdirections. Vol. 72, n.8.

Liu et al., 2013a – *Liu, A.S., Schunn, Ch., Flot, J. and Shoop, R.* (2013). The role of physicality in rich programming environments. Computer Science Education. Retrieved from http://education.rec.ri.cmu.edu/educators/research/

Majherová, 2007 – *Majherová, J.* (2007). Virtual plants in high school informatics - L-systems. In Auer, M. (ed.): International Conference Interactive Computer Aided Learning 2007, ePortfolio and Quality in e-Learning. CD. Kassel: Kassel University Press.

Musa et al., 2015 – *Musa S., Ziatdinov, R., Sozcu, O. F. and Griffiths, C.* (2015). Developing Educational Computer Animation Based on Human Personality Types. *European Journal of Contemporary Education*, Vol. 11, Issue 1, pp. 52-71.

Nagyová, 2015 – *Nagyová, I.* (2015). Analysis of Students' Behaviour in eLearning Course "Educational Technologies". *European Conference on e-Learning*. pp. 428-434. Hatfield: ACPI.

Polčin et al., 2016 – *Polčin, D., Petrušková, H. and Stando, J.* (2016). Support and Improvement of the Educational Process in Regional Education in Slovakia Through Tablet Classrooms. In The Third International Conference on Computer Science, Computer Engineering, and Education Technologies. pp. 71-78. Łodz: Łodz University of Technology.

ROBOTC – ROBOTC a C Programming Language for Robotics. Retrieved from http://www.robotc.net/

Robotika – URL: http://www.robotika.sk/mains.php

Saleiro et al., 2013 – *Saleiro, M., Carmo, B., Rodrigues, J.M.F. and Buf, J. M. H.* (2013). A Low-Cost Classroom Oriented Educational Robotics System. In Social Robotics: 5th International Conference, ICSR 2013. LNAI 8329. pp. 74-83.

Šnajder, Guniš, 2016 – *Šnajder, Ľ. and Guniš, J.* (2016). Analysis of results in inquiry based informatics education of selected topics. Information and Communication Technology in Education 2016. pp. 169-180. Ostrava: University of Ostrava.

Velšic, 2013 – *Velšic, M.* (2013). Digitálna gramotnosť optikou mladej generácie. [Digital literacy view of the young generation]. Bratislava: IVO – Microsoft.

Velšic, Janotík, 2016 – *Velšic, M. and Janotík, T.* (2016). E-skills a trh práce na Slovensku. [E-skills and job market in Slovakia]. Bratislava: IVO.