How to Support Students' Computational Thinking Skills in Educational Robotics Activities

Soumela Atmatzidou, Stavros Demetriadis

Department of Informatics, Aristotle University of Thessaloniki, Greece
{atmatzid, sdemetri}@csd.auth.gr

Abstract. The term Computational thinking has received intense attention over the past several years as a fundamental skill, which promotes new ways of thinking to the students across all disciplines of science. The present study describes the implementation and evaluation of developing computational thinking skills in Educational Robotics activities for secondary Technical schools, which focus on the basic skills of CT: abstraction, generalization, algorithm, modularity, decomposition and problem solving. We summarize the results from pre- and post-questionnaires and a series of think-aloud interviews. The results suggest that the students became familiar with the concepts of CT, and integrated them to a satisfactory extent in the process of problem solving in ER activities.

Keywords: Computational Thinking, problem solving, Educational Robotics.

1 Introduction

This work presents and discusses a specific didactic approach to support the development of students' computational thinking skills in activities of educational robotics. Computational thinking is a fundamental skill for everyone and as Wing believes it should be added along with reading, writing, and arithmetic to every child's analytical ability [23]. There has been a growing recognition of the importance of CT for controlling and managing cognitive activities, as well as understanding and solving problems in a wide range of contexts, not only in the field of computer science, but in all disciplines [24].

Robotics can be used as a tool that offers opportunities for students to engage and develop computational thinking skills [14], [19]. Educational robotics is being introduced in many schools as an innovative learning environment, enhancing and building higher order thinking skills and abilities, and helping students solve complex problems [7]. In addition, a guided instructional approach with robots, facilitates teamwork, develops conceptual understanding, enhances critical thinking, and promotes higher-order learning in the domains of mathematics and science [8].

This paper describes the implementation of Educational Robotics activities in a secondary Technical school, and focuses on the development of computational thinking and problem solving skills. Students work in small groups, guided by

Proceedings of 4th International Workshop Teaching Robotics, Teaching with Robotics & 5th International Conference Robotics in Education
Padova (Italy) July 18, 2014
ISBN 978-88-95872-06-3
pp. 43-50

worksheets to solve authentic complex problems through our proposed model for developing CT skills that focus on the following CT concepts: abstraction, generalization, algorithm, modularity, decomposition.

2 Background

Computational thinking (CT) is defined by Wing (2006) as a way of solving problems, designing systems, and understanding human behavior that draws on concepts fundamental to computer science. She argues [23] that CT is a type of analytical thinking that shares many similarities with mathematical, engineering and scientific thinking. CT roots go back, to Papert's work on Logo programming language and the idea of the computer being the children's machine that would allow them to develop procedural thinking through programming [18].

According to Wing [24] the keys of CT are abstraction, decomposition, separation of concerns and modularity. Other researchers support that the keys of CT are computation, communication, coordination, recollection, automation, evaluation, design, algorithm building, conditional logic, debugging, simulation, working effectively in teams and analyzing problems [2], [21]. Lu and Fletcher argue that, teaching CT should focus on establishing languages that can be used to annotate and describe concepts of CT and provide notation around which mental models of processes can be built [17].

Robotics is usually seen as an interdisciplinary activity drawing mostly on Science, Maths, Informatics and Technology and offering major new benefits to education in general at all levels [1], [20]. Drawing on the theoretical perspective of Piaget's constructivism, Papert's constructionism and Vygotsky's collaborative learning, ER activities help students transform from passive to active learners, developing many mental skills as researchers and creating new knowledge. Many studies indicate that ER activities have a positive effect on the development of critical thinking, problem solving and metacognitive skills [3], and also on the learning of a programming language [1]. In addition, a great advantage of using robots is that abstract concepts can be turned into real-world problems and solutions [25].

Studies have focused on the environment of robots, as an appropriate tool for the development of CT. In 2011, a research from National Science Foundation, examined how abstraction, automation and analysis in problem-solving take shape for middle and high school youth. In a robotics project, student programmers needed to think about how the robotic agent would interact within its world and the results indicated that the students were able to use abstraction, automation, and analysis to create original products. Still the field requires systematic assessment procedures. Prior research demonstrates that children as young as four–six years old can build and program simple robotics projects as well as learn powerful ideas of engineering, technology, and computer programming while also building their computational thinking skills [5], [6].

Although, CT is a concept that has received considerable attention over the past several years, the literature on implementing CT in a K-12 setting is still relatively sparse [25]. Little is known about the development of CT in K-12, although recent

Proceedings of 4th International Workshop Teaching Robotics, Teaching with Robotics &
5th International Conference Robotics in Education
Padova (Italy) July 18, 2014
ISBN 978-88-95872-06-3
pp. 43-50

articles begin to describe what it looks like [4]. Furthermore there is little agreement about strategies for assessing the development of CT in young people [2].

We can see that there is a lack of empirical evidence in defining the explicit boundaries of CT [11]. In our study in order to investigate the contribution of ER to the development of CT skills in Elementary and Secondary school's students, we designed and implemented the following CT model of Computational Thinking skills.

Considering the above, we focus on the following research questions:

(a) How can the CT and problem solving skills supported efficiently in educational robotics activities? , and

(b) Which are the appropriate strategies for assessing the development of CT?

3 Method

3.1 Participants

For the purpose of this study we used the Lego Mindstorms NXT 2.0 educational tool. The ER activities took place in a secondary High Technical School in Thessaloniki. We recruited 35 school students (28 boys and 7 girls). The students worked in groups consisting of 3 members. The study was conducted in 11 sessions that lasted two hours each.

3.2 A model for CT skills

In order to operationalize our approach for CT support we need to model this set of skills. A proposed model for CT skills is as follows:

Table 1. A model for CT skills

CT skills	Definitions of CT skills	Guidance for development CT skills
Abstraction	Abstraction is the process of creating something simple from something complicated by leaving out the irrelevant details, by finding the relevant patterns, and by separating ideas from tangible details [22].	1. Separate the important from the redundant information. 2. Analyze and specify common behaviors or programming structures between different scripts. 3. Identification of abstractions between different programming environments.
Generalization	Generalization is transferring a problem-solving process to a wide variety of problems.	1. Expanding an existing solution in a given problem in order to cover more possibilities / cases. 2. Use variables in solution

Proceedings of 4th International Workshop Teaching Robotics, Teaching with Robotics &
5th International Conference Robotics in Education
Padova (Italy) July 18, 2014
ISBN 978-88-95872-06-3
pp. 43-50

Algorithm	Algorithm is a practice of writing step-by-step, specific and unambiguous, instructions for carrying out a process.	1. Explicit wording of the steps of the algorithm. 2. Possibility of different algorithms for the same problem. 3. Effort to find the most effective algorithm.
Modularity	Modularity is the development of autonomous processes, which encapsulate a set of often used commands that perform a specific function and might used in the same or different problems.	Develop autonomous sections of code to be used for the same or different problems.
Decomposition	Decomposition is the process of breaking problems down into smaller parts that may be more easily solved	Breaking apart problems into smaller / single ones that are easier to be solved.

3.3 Learning Design – Implementation

In each session, the students are separated into groups of 3 with each member assuming a role such as analyst, algorithms' designer, programmer or debugger that are alternated in each activity. Students are guided through worksheets in the investigation of authentic complex problems and focus on CT language in order to develop basic skills of computational thinking:

During the sessions, the teacher has the role of the facilitator and the instructor who directs children through appropriate questions and explains and analyzes the skills of CT.

The implemented ER activities were divided into two phases: the "trainings" and the "challenge". The "training" phases consisted of 10 sessions and the "challenge" phase 1 session. At the beginning, we did an introduction on Robots and Lego Edu programming environments NXT-G and handed out an individual pre-questionnaire in order to create the students profile about their experience with computing and robotics tools. In the first 4 sessions we handed out worksheets to students for familiarizing with ER and basic programming concepts. At the core of each worksheet is the understanding and assimilation of the basic CT skills that constitute our computational model. Then we gave the first quiz in order to investigate if students understood the CT skills. In the next 6 sessions the activities had integrated more CT skills in complex authentic problems with graduated difficulty. In the robotic activities the students programmed the robots in authentic scenarios such as an alarm, a car that follows the rules of traffic, a security guard, a recycler, etc. A second quiz and a final questionnaire followed to record the student's views. Finally, a challenge took place and all the groups were required to implement an activity in which the winning group would be the one with the best performance.

Proceedings of 4th International Workshop Teaching Robotics, Teaching with Robotics &
5th International Conference Robotics in Education
Padova (Italy) July 18, 2014
ISBN 978-88-95872-06-3
pp. 43-50

3.4 Data collection

In the present study we used qualitative and quantitative methods. The evaluation tools were:

(a) systematical monitoring of the students' work by taking notes on a structured form.

(b) individual pretest questionnaire given before the sessions for creating the student's profile about computing and experience with robotics tools.

(c) individual posttest questionnaire, given after the completion of the interventions, which documented the students' views and evaluation of their overall experience with educational robotics activities. Both questionnaires used a 5-grade Likert scale (1= 'Not at All Interested', 2 = 'Not Very Interested', 3 = 'Neutral', 4 = 'Somewhat Interested', 5= 'Very Interested').

(d) Quizzes (Quiz1 and Quiz2) given in the 4th and 10th session, in which students were asked to solve problems and cite the CT concepts that they use in the problems (e.g. identify the concept of abstraction between two or more given problems and propose a generalization), and finally

(e) interviews with the students were they described the process that they followed to solve a problem. The assessment of quizzes and interviews was evaluated with graded criteria (rubric) on a 4-point Likert scale (1= 'unsatisfactory', 2 = 'quite satisfactory', 3 = 'satisfactory', 4 = 'excellent').

The evaluations tools focus on five dimensions: (1) the development of computational thinking skills and (2) problem solving skills, (3) the basic programming concepts, (4) the collaboration in the groups, and (5) the educational robotics tools.

3.4 Results

The statistical analysis t-test for paired-samples on Quiz1 and Quiz2, showed that between Quiz1 and Quiz2 they was statistically significant difference a) in the averages of CT concepts, as presented in table 2 and b) in the Problem Solving skills, in table 3.

Table 2. Results of Quizzes for CT concepts

CT skills	Quiz1	Quiz2	Statistics t-test
Abstraction	M=2.37, SD=0.826	M=2,629 SD=0,843	t(34)=-2.491, p=0.018
Generalization	M=2.21, SD=0.949	M=2.59 SD=1.003	t(34)=-2.176, p=0.037
Algorithmic	M=2.43, SD=0.822	M=2.85 SD=0.805	t(34)=-4.606, p=0.000
Modularity	M=2.06, SD=1.056	M=2.77 SD=1.330	t(34)=-3.841, p=0.001
Decomposition	M=2.35, SD=0.928	M=2.97 SD=0.954	t(34)=-3.899, p=0.000
Overall CT	M=2.29, SD=0.667	M=2.76 SD=0.792	t(34)=-5.202, p=0.000

Table 3. Results of Quizzes for problem solving

Skills	Quiz1	Quiz2	Statistics t-test
Problem Solving	M=2.55, SD=0.84	M=2.88, SD=0.89	t(34)=-2.961, p=0.006

Table 4. Results of student's Interview

Skills	N	Mean	Std. Deviation
CT	35	2.58	0.788
Problem Solving	35	2.87	0.944

According to the results of the students' Interview about the solution of a problem (Table 4), results of Quizzes for CT concepts (Table 2), and the systematical monitoring of the students' work, we noticed the following:

(1) Regarding the development of CT skills, we noticed that they have managed to assimilate them in quite a good degree (2.58). Specifically, (a) most of the students (2.38) experienced difficulties in the identification and the description of the concept of abstraction but despite this they were able to identify easily the common programming parts between different scenarios. (b) The students, in the beginning, found it difficult to understand the concept of generalization and to suggest a more general solution. However, at the end of the training, with our encouragement, we observed interesting generalizations in the activities (2.76) and more specifically in the alarm scenario the students had the idea of adding more sensors to activate the alarm in many different cases (e.g. fire). (c) Relating to the algorithm's design, most of the students (2.69) found it difficult to describe the algorithm with clarity and accuracy. They preferred to describe a process in general rather than analyze it step by step. (d) The students, after our encouragement, incorporated the creation of code's modules in their activities (2.26). (e) The students directly acquainted with the process of decomposition and they divided the problems into smaller ones, easily (2.83). Specifically it was documented during the interview that they have applied it to other courses.

(2) In the first sessions, students faced difficulties with the complex problems, however after a few trainings they became familiar with the process of solving them (2.87). It is worth mentioning some answers about the development in problem solving: "Now I think differently and solve problems more easily" and "I changed my way of thinking in problem solving".

The post-questionnaires and the systematical monitoring of the students' work on three dimensions: (a) the basic programming constructs, (b) the collaboration in the groups, and (c) the educational robotics tools, can be summarized as follows:

(a) The students became familiar with basic programming constructs such as repetition and selection and even said they would continue with programming. In particular 21 students of the computer science department stated that they understood better some of the basic programming concepts such as the concept of choice (If ... then... else) and the concept of repetition (For ... Next, Do While ... Loop).

Proceedings of 4th International Workshop Teaching Robotics, Teaching with Robotics &
5th International Conference Robotics in Education
Padova (Italy) July 18, 2014
ISBN 978-88-95872-06-3
pp. 43-50

(b) Additionally, regarding the collaboration, the students liked working in groups and assuming roles. The most popular role, for the students, was the programmer's.

(c) Finally, the students found the activities of ER very interesting and said that they would like to continue engaging with ER afterwards. Specifically they replied: "I'd like to keep working on robotics because it is the job of the future".

3.5 Conclusion

In the present research, we studied the effect of ER activities on the development of CT skills and problem solving. The results showed that the students, during the first trainings, faced difficulties understanding the CT concepts, however as the trainings progressed they started familiarizing and adopt this concepts satisfactorily. From the results of the quizzes and the final problem during the interview, which we evaluated with graded criteria (rubric), we found that the students developed CT skills quite successfully. Specifically, concerning the understanding and assimilation of the CT concepts, the ones that the students became more familiar with and in a shorter time were the Algorithm, Modularity and Decomposition. Abstraction and Generalization, on the other hand, they posed the greatest difficulty. Many students told us that they remember and they use these CT concepts in other courses as well.

From the interviews and the post-questionnaires we observed that the students considered very interesting the activities and important the guidance for problem solving in the worksheets as well as the collaboration within the teams. However, more sessions and more engagement with complicated, authentic problems are required for the students to be able to assimilate and pore over the CT skills.

Our future goal is to improve the proposed model for supporting the development of CT skills, focusing on: (a) enrich the worksheets with targeted activities it guide and support the students (b) increase the number of sessions, and (c) make a wider research about the assessment.

References

1. Alimisis, D. Ed., Teacher Education on Robotics-Enhanced Constructivist Pedagogical Methods. School of Pedagogical and Technological Education (ASPETE), 2009.
2. Astrachan, O., Barnes, T., Garcia, D.D., Paul, J., Simon, B., Snyder, L.: CS principles: piloting a new course at national scale. Proceedings of the 42nd ACM technical symposium on Computer science education, pp. 397-398, New York, NY, USA (2011)
3. Atmatzidou, S. & Demetriadis, S. (2012). Evaluating the role of collaboration scripts as group guiding tools in activities of educational robotics, IEEE International Conference on Advanced Learning Technologies (ICALT 2012), Rome, Italy.
4. Bers, Marina U. (2005). The TangibleK Robotics Program: Applied Computational Thinking for Young Children, 1–20.
5. Bers, Marina Umaschi, Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. Computers & Education, 72, 145–157. doi:10.1016/j.compedu.2013.10.020

6. Bers, P. M. U., Flannery, L., Kazakoff, E., & Crouser, R. J. (n.d.). A Curriculum Unit on Programming and Robotics.
7. Blanchard, S., Freiman, V., & Lirrete-Pitre, N. (2010). Strategies used by elementary schoolchildren solving robotics-based complex tasks: innovative potential of technology. Procedia - Social and Behavioral Sciences, 2(2), 2851-2857.
8. Chambers J. M., Carbonaro M., Rex M., and Grove S., "Scaffolding Knowledge Construction through Robotic Technology: A Middle School Case Study," Technology, vol. 6, pp. 55-70, 2007.
9. Demetriadis, S. (2014). Collaboration scripts to support computational thinking. Future Learning, 1, 61–66.
Retrieved from http://essential.metapress.com/index/R6388263424XT5X0.pdf
10. Grover, S., Grover, S., & Grover, S. (2011). Robotics and Engineering for Middle and High School Students to Develop Computational Thinking, (650), 1–15.
11. Kazimoglu, C., Kiernan, M., Bacon, L., & Mackinnon, L. (2012). A Serious Game for Developing Computational Thinking and Learning Introductory Computer Programming. Procedia - Social and Behavioral Sciences, 47, 1991–1999.
12. Kazimoglu, C., Kiernan, M., Bacon, L., & Mackinnon, L. (2012). A Serious Game for Developing Computational Thinking and Learning Introductory Computer Programming. Procedia - Social and Behavioral Sciences, 47, 1991–1999.
13. Kramer, J., & Hazzan, O. (2006). The role of abstraction in software engineering. Proceeding of the 28th international conference on Software engineering - ICSE '06, 1017. doi:10.1145/1134285.1134481
14. Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., et al. (2011). Computational thinking for youth in practice. ACM Inroads, 2(1), 32-37.
15. Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., . . . Werner, L. (2011). Computational thinking for youth in practice. ACM Inroads, 2(1), 32-37.
16. Lee, T. Y., Mauriello, M. L., & Sopan, A. (n.d.). CTArcade: Learning Computational Thinking While Training AI Through Game Play.
17. Lu, J., & Fletcher, G.: Thinking about computational thinking. ACM SIGCSE Bulletin, 41(1), pp.260–264 (2009)
18. Papert S., Mindstorms: children, computers, and powerful ideas, (2nd ed.). New York, NY: Basic Books, 1993, p. 230.
19. Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. Proceedings of the 41st ACM technical symposium on Computer science education - SIGCSE '10, 265. doi:10.1145/1734263.1734357
Retrieved from http://www.joelonsoftware.com/articles/LeakyAbstractions.html
20. Rogers, C., & Portsmore, M. (2004). Bringing engineering to elementary school. Journal of STEM Education, 5(3&4), 17–28.
21. Settle, A., Perkovic, L.: Computational Thinking across the Curriculum: A Conceptual Framework. Technical Reports (2010)
22. Spolsky, J. (2002). The Law of Leaky Abstractions.
Retrieved from http://www.joelonsoftware.com/articles/LeakyAbstractions.html
23. Wing, J. M. Computational thinking. Communications of the ACM 49(3), pp.33-35 (2006)
24. Wing, J. M. Computational thinking and thinking about computing, Phil. Trans. R. Soc. A (2008) 366, 3717-3725
25. Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T.: Introducing computational thinking in education courses. Proceedings of the 42nd ACM technical symposium on Computer science education-SIGCSE0000 '11, pp.465–470 (2011).

Proceedings of 4th International Workshop Teaching Robotics, Teaching with Robotics &
5th International Conference Robotics in Education
Padova (Italy) July 18, 2014
ISBN 978-88-95872-06-3
pp. 43-50