Common Errors, Successful Debugging, and Engagement During Block-Based Programming Using Educational Robotics in Elementary Education

Chrysanthos Socratous, Cyprus Interaction Lab, Cyprus University of Technology, chrysanthos@cyprusinteractionlab.com
Andri Ioannou, Cyprus Interaction Lab, Cyprus University of Technology, Research Center on Interactive Media, Smart Systems and Emerging Technologies (RISE), andri@cyprusinteractionlab.com

Abstract: The study aimed to understand the effect of a structured versus an unstructured Educational Robotics (ER) curriculum on student's (a) quantity and type of programming errors in block-based programming, (b) ability to find and debug errors, and (c) engagement in the learning process. We worked with a sample of 35 elementary school students in two groups, experiencing one of the teaching approaches: structured versus unstructured ER curriculum. We employed a mixed-method approach based on video and screen recorded data from eight ER lessons, post-experience focus-group interviews, and post-experience questionnaires on students' engagement and debugging skills. Findings show that there is a list of errors commonly made by students in both groups. Moreover, the unstructured curriculum group was associated with significantly higher amount of errors than the structured curriculum group. Also, a structured curriculum was related to significantly greater efficiency in finding and debugging errors. However, the results revealed that students in the unstructured curriculum outperformed the structured curriculum in terms of their engagement levels.

Introduction

Educational Robotics (ER) has gained much attention in education as innovative learning tool that allows students to develop higher-order thinking skills. During the last two decades, many researchers and instructors have fruitfully used ER in several contexts and disciplines for the teaching of particular content, or for supporting learning of transversal skills such as problem-solving (Lindh & Holgersson, 2007), metacognition (Socratous & Ioannou, 2019), collaborative knowledge construction (Socratous & Ioannou, 2018), computational thinking (Constantinou & Ioannou, 2018) creativity (Sullivan, 2011), and collaboration (Ardito, Mosley, & Scollins, 2014). ER may also support other crucial learning processes such as robot programming, which demands the planning of complex sequences of actions before the execution of the program. During programming, students must set the goal, conceive the sequential steps required to accomplish that goal, program actions, and check performance. These processes require various complex cognitive and metacognitive functions, essential for student's cognitive development, including logical reasoning, decision-making, problem-solving, and sequential thinking.

Although ER has much potential to assist in teaching, learning gains are not guaranteed just by the simple application of robotics; there are several factors that can determine the outcome (Benitti, 2012). Such a factor is the level of structure that educators adopt in their activities. To date, there is relatively little work focused on the level of structure that educators should adopt in their ER activities. Therefore, very little is known about the impact that different teaching approaches might have on students' learning.

In this work, we compare two groups of students, each experiencing one of the teaching approaches: structured versus unstructured ER curriculum. We evaluate the outcomes of the learning experience in terms of the quantity and type of programming errors that students make in ER activities, students' ability to find and debug errors, and students' engagement with the experience. Three research questions (RQ) framed this investigation:

- RQ1: Are there differences between the groups in terms of the type and the number of programming errors?
- RQ2: Are there differences between the two groups in their ability to identify and debug errors?
- RQ3: Are there differences in students' level of engagement between the groups?

Framing

Papert's constructionist framework is one of the most popular approaches to teaching and learning in our days. Most of ER research draws on a constructionist framework and student-centered learning environments for the integration of robotic technologies in education (Ioannou & Makridou, 2018). The constructionist framework

builds on Piaget's ideas on how knowledge is constructed in an individual's mind focusing more on the ways that internal constructions are supported by constructions in the world. Therefore, constructionism suggests that students learn better when participating in the process of building their own meaningful projects that reflect the experience of solving problems (Papert & Harel, 1991) without top-down instruction (Lee, Sullivan & Bers, 2013). In general, learning that is driven by problems allows the student to build his/her own knowledge (Ioannou, Socratous, & Nikolaedou, 2018).

Overall, constructionism has received various critiques. One of the essential critiques is that learners' constructing their knowledge takes too long and wastes valuable learning time along the way to discovering new knowledge. Learning time can be used more efficiently through direct instruction and guided practice (Anderson, Greeno, Reder, & Simon, 2000). In this case, an instructionist curriculum suggests a more structured learning environment with less freedom for children to explore and construct their own knowledge. In a constructionist teaching approach, the role of the instructor is to provide an environment that allows students to explore and experiment with their ideas to gain understanding through the process. In an instructionist approach, the role of the teacher is to provide information to students.

Background work

Block-based programming

About 20 years ago, the MIT Media lab introduced the idea of block-based programming. In contrast to text-based programming, block-based eliminates the need to learn the syntax of a language and enables learners to converge their mental energy on the logic of their programs rather than the semantics. This idea made teaching and learning the fundamental concepts of computer science accessible to younger learners with no previous coding experience. At the same time, this idea generated a new stand of research on scaffolding novice programmers with the use of direct manipulation interfaces (Weintrop & Wilensky, 2015); this became more popular with the utilization of robots in education, and therefore block-based programming is often paired with ER.

Comparing different teaching approaches in ER research

There is some work on different teaching approaches in ER and their effects. A study by Lee, Sullivan, and Bers (2013) examined the effect of teaching using a constructionist teaching approach versus an instructionist on promoting peer-to-peer collaborative interactions. The results showed that using a structured curriculum was associated with less collaboration compared to using an unstructured curriculum. Hence, according to the authors, to encourage collaboration, a less structured learn-by-doing approach might be more beneficial. A study by Atmatzidou, Demetriadis, and Nika (2018) investigated the development of students' metacognitive and problem-solving abilities in ER tasks performing different levels of metacognitive guidance (strong and minimal) in two groups. Both groups used worksheets. The strong guidance group used structured worksheets to lead them gradually to solve the problems and prompted them to follow specific metacognitive and problem-solving strategies. The minimal guidance group used worksheets of increasing difficulty. Their results suggested that strong guidance had a more positive effect on students' metacognitive and problem-solving abilities.

Engagement in constructivist learning environments

There is a widespread assumption that ER can be useful in increasing engagement and motivation in learning, independent of the teaching approach. However, there are no studies in the field of ER, describing how different teaching approaches may impact students' engagement. There is prior work documenting the link between student-centered learning environments and students' engagement without the use of ER. In general, previous studies found positive to mixed results concerning the relationship between student-centered environments and student engagement. For instance, Wu and Huang (2007) observed that in a student-center environment students reported significantly higher emotional engagement than students in a teacher-center environment. A study by Hampden-Thompson and Bennett (2013) aimed to explain the disparity in students' reports of engagement in science, across various science teaching and learning activities and showed that students had greater levels of emotional engagement when higher levels of interaction, hands-on activities, and applications were used.

Methods

Participants

The study involved 35 elementary-school children (third-graders, 20 boys, and 15 girls) in two classes of the same school in Cyprus. The classes naturally formed the comparison groups: the structured curriculum group (16 third-

graders, 9 boys and 7 girls, 4 groups) and the unstructured curriculum group (19 third-graders, 11 boys and 8 girls, 5 groups). For the ER experiences, students worked in groups of 3-4 students with different genders and abilities to allow different problem-solving approaches to develop. Three of the participating students (two in the unstructured curriculum group and one in the structured curriculum group) had previous experience with ER and programming, while four were students with special educational needs and learning difficulties (two in each group). The implementation was employed in 10 sessions of STEM problem-solving activities for three months.

Procedures and activities

Both groups participated in two introductory sessions and eight 80-minute sessions of problem-solving on a given challenge for each session. A teacher and an educational technologist designed the learning environment and activities.

On each group's desk, there were pencils, rubbers, rulers, a tablet, an EV3 robot, blank paper for the unstructured curriculum group and worksheets for the structured curriculum group.

Sessions 1-2 (Introductory Sessions). The first two sessions were introductory lessons to ER with primary activities to support students to get familiar with the Lego EV3 environment. Essential programming details were explained in both groups by presenting examples (directional commands, sensors, loop, wait for, and conditional logic). This design choice allowed considerable top-down instruction in both groups, but it was a conscious decision to ensure that both groups had basic knowledge to cope with the rest of the sessions.

Sessions 3-10 (80-minutes sessions on challenges). The next eight lessons for both groups were grounded on the same content. Each session started with five minutes of introduction to the learning goals and a brief description of a given challenge.

The structured curriculum group received the instructionist approach to teaching in which they were instructed how to program their robot, based on pre-designed tasks and using worksheets as a reference. In detail, the teacher presented students with a worksheet and a mat for the final challenge. First, students had to complete the activities of the worksheets with the help of the teacher and then continue with the final challenge. The teacher's role was to provide information, ideas, and feedback as needed. The worksheets were used to help the students cope with the challenge of the session. In sum, the worksheets and they teacher's guidance pushed the students to implement ideas that were relevant to the challenge. This was, in some way, preparatory work for the final challenge of the session.

The unstructured curriculum group received the constructionist approach. Namely, this group did not have structured activities but instead could investigate ideas linked to the challenge, within their 80 minutes sessions. In every session, a reminder was made by the teacher that they could use existing ER material by Lego Education (i.e., videos and programming examples). Students in this group did not use any worksheets and did not participate in any precursor activities for the final challenge. The teacher's role was to facilitate and scaffold student's thinking. He was assisting the students with hints, prompts, and feedback but without providing any information or answers. The groups should experiment and try to solve it on their own.

The activities are described below.

In Session #3, students had to program a robot to move accurately on 150cm line (i.e., a line following challenge), using rotations, degrees, or seconds. The worksheet for the structured curriculum group asked the students to complete a table by taking measurements of various values. Once they had completed the worksheet, they could proceed to completing the line following challenge. In Session #4, the worksheet for the structured curriculum was designed to help students to practice different turns and help them to understand how the turning variable is related to the distance. Students had to execute several programs associated with turns and explore the output of their applications. The goal of the final challenge was to program the robot to move on a square without a gyro sensor. In Session #5, students were exposed more in activities related to turns. The worksheet for the structured curriculum was a combination of the two previous worksheets about distance and turns. The new concept was the different kinds of turns, such as spin turns, pivot turns, and smooth turns. The goal of the final challenge was to program the robot to move in a path with different kinds of turns to arrive at the final destination.

In Session #6, students were tasked to use the robot's motor with the cargo deliver attachment to move objects. The worksheet for the structured curriculum had two sub-tasks with increasing difficulty as introductory activities for the final task. The first sub-task instructed students to program the robot to move a block that was opposite to the starting position and the second sub-task instructed students to move a block that was located at a random place on the mat. During sessions #7 and #8, students investigated the use of the color sensor and the concepts of loops and wait/until. The worksheet for the structured curriculum asked the students to place the color

sensor close to several objects of the classroom and observe the reading value through the programming interface. The purpose was to program a robot to move on a mat with red and black lines: "When the robot sees a red line, it stops for 1 second and says *red*, then continues until it finds a black line, stops at the black line and says *black*, then reverses and moves forward until it finds the black line for ten times." The final challenge for session #8 was to program an autonomous robot that could move on their desk without falling down for one minute.

In session #9, students had to program the robot to move and stop at an object. The worksheet for the structured curriculum asked the students to place the robot opposite of several objects in the classroom and measure the distance between the robot and the object using the ultrasonic sensor. For the final challenge, they had to program a robot that could move into the classroom without hitting any objects. In session #10, students used the worksheets to explore the concept of conditional logic. They were asked to program the robot to say "red" when the color sensor detects red color and says "no" when the color sensor was not detecting a red color. This activity was designed to help students to respond to the last task and complete the line following challenge (see Figure 1).

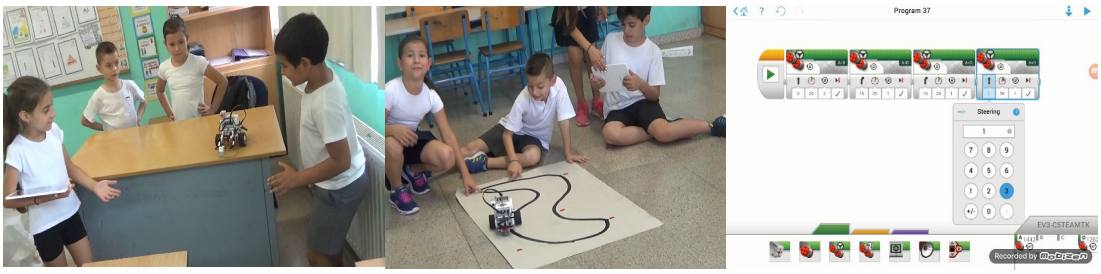

Figure 1. Activities and screenshot of video-recorded tablet screen while programming.

Data collection, instrumentation, and analysis

Data were collected from classroom recordings (one camera in each class), tablet screen, and audio recordings for each team, post-debugging test, post-engagement survey, and focus-group interviews. Tablet screens and discourse of the nine teams were recorded using Mobizen Screen Recorder (see Figure 1).

After the learning experience, students completed a test on debugging, which was developed based on the categories of the common programming errors and was comprised of ten tasks. The test provided students with a scenario with the purpose of the program and a screenshot of the EV3 programming interface with a proposed program with an error. Students had to find the error (by circling the error block or the group of blocks) and describe, writing on the paper, how they could fix it. Each correct task was scored with a total of ten marks (five marks for finding the error and five for a correct proposal to overcome the error). The maximum possible score for the test was 100 marks.

A post-experience survey aiming at evaluating students' engagement was administered. Students' engagement was measured using a 5-point Likert scale with 33 items derived from the Math and Science Engagement Scales (Wang, Fredricks, Hofkens, & Linn, 2016). The scale is comprised of four subscales: (a) Cognitive engagement (e.g., I try to understand my mistakes when getting something wrong), (b) Behavioral engagement (e.g., I keep trying even if something is hard), (c) Emotional engagement (e.g., I enjoy learning new things about STEM), and (d) Social engagement (e.g., I try to understand other people's ideas in STEM class). The four dimensions have internal reliabilities of alpha: .74, .75, .72, .77 respectively.

One week after the learning experience, 16 students participated in semi-structured focus-group interviews (four focus-groups, 50-60 minutes each). The focus-group interviews were organized into two sessions. The first session was contributing to answering RQ1 and enriching our understanding of students' common errors during programming. We used the 40% of the screen recorded data to identify the common programming errors first, and then we designed the first session of the focus-groups based on the common errors derived from the video analysis of the screen recordings. Part of the interviews included questions that prompted students to remember about errors that they have encountered and how they manage to overcome these errors (e.g., What difficulties did you face during programming?, How did you overcome the difficulties?). The second session was aimed at providing additional evidence for students' engagement (e.g., Do you think robotics activities were useful to you? [Cognitive engagement], How did you feel, while you were working on robotics activities in this class? [Emotional engagement]). The focus-groups data were video-recorded and transcribed for data analysis using a thematic analysis approach. The data analysis was conducted by two researchers, working separately. At the completion of the coding, the inter-rater reliability was assessed to 0.736 (Cohen's Kappa).

We used open coding for the screen-recorded and the audio data. We narrowed our analysis to errors associated with programming; we did not code videos related to assembly. Around 40% of the video was coded

by the first researcher, with a second researcher independently coding the same units. Inter-rater reliability between the two raters was found to be high (Cohen's Kappa = .88). Therefore, the first researcher finished coding the complete dataset. The data were analyzed using independent samples t-test to investigate any differences between the two groups on the total number of errors. For the analysis of the data derived from the post-debugging test, and the engagement survey, we also used independent samples t-test.

Findings

Common programming errors (bugs)

We found six common errors in the programming (see Table 1). First, students often failed to define the correct values for a variable, such as defining the accurate distance to move forward. Second, participants made errors in selecting the correct block or the exact sequence of blocks. Sometimes students omitted blocks required for the robot to operate as planned. Robots could not function as expected as a result of omitting loop, move block, turning block, or other commands. Third, students selected inappropriate variables of a block. They picked the correct block, but they made errors in selecting which variable of the block they should use. That is, different variables in the same block led to many errors until students realize the difference between the variables of the same block. Fourth, students selected motors and sensors that were not consistent with the robot. For example, the right motor was attached to the port A of the robot, but the motor selected was motor C. The same happened with sensors; the ultrasonic sensor was connected with a cable in port 4, but the chosen port in programming was port 3. Fifth, students improperly defined conditionals. They had difficulty to understand conditional logic such as if/then or wait until making several mistakes and causing bugs to their programs. Last, students do not recognize a program without an error. They looked for errors when a failure occurred due to other issues. They were seeking for errors in their program when the error was on the assembly of the robot or the positioning on the mat.

Table 1: Common programming errors

Common errors	Description	Example
1. Error in defining value in variables of a block	Not accurate or wrong calculation of the value of a variable	Wrong calculation of the distance variable
2. Error in selecting the correct sequence of blocks	Selecting an inappropriate block or omitting a command	Omitting a turn command
3. Error in selecting the appropriate variable of a block	Selecting a different variable in the same block	Instead of distance in rotations, they chose the engine power
4. Error in selecting a motor or a sensor that is compatible with the robot	Connecting the motors or the sensors in wrong ports	The program was reading port A, but the motor was connected to port C
5. Error in defining conditions	Strangle to understand conditional logic	Strangle to debug an error related with conditionals
6. Error in identifying a program without an error	Do not recognize a program without an error	Looking for an error when the problem was on the assembly

Differences in students' common programming errors between the groups

For both groups, the most common errors were errors of category 1 (error in defining the value of a variable). Students in the two groups produced a significant number of errors trying to define a value in variables of a block. Regarding the frequencies of each type of errors (see Table 2) we observed that in the structured curriculum group, the errors that were dealing with knowledge about the programming interface such as categories 2 and 3 were less than the unstructured curriculum group (41.66 % versus 31.70%). We also noticed that in the last sessions, the errors coded in categories 2 and 3 became less frequent for both groups as students became more familiar with the programming interface. Furthermore, from the significant amount of errors in category 5, it seems that the students in the two groups strangled to define and understand conditional logic, such as if/then and wait until.

We observed a big difference in the number of errors between the two groups. Students in the unstructured curriculum generated more bugs (72) compared to the structured group (41). Therefore, we calculated the average number of errors per session to examine if there were differences between the groups. The unstructured group showed a higher mean number of errors per session. An independent sample t-test was administered to compare mean error scores between the groups. The results revealed that the unstructured

curriculum group (M = 9, SD = 2.12) had a statistically significant higher mean number of errors than the structured group (M = 5.13, SD = 1.90); t (14) = 3.60, p = 0.003. These results suggest that an instructionist teaching approach is more likely to make students produce fewer programming errors than a constructionist one.

Table 2: Comparison of the common programming errors of the two groups

Common errors	Structured Group		Unstructured Group	
	N of errors	%	N of errors	%
1. Error in defining value in variables of a block.	10	24,40%	24	33,33%
2. Error in selecting the correct block or the exact sequence of blocks.	7	17,07%	16	22,22%
3. Error in selecting the appropriate variable of a block that is needed for the robot to operate as intended.	6	14,63%	14	19,44%
4. Error in selecting a motor or a sensor that is compatible with the robot.	6	14,63%	5	6,95%
5. Error in defining conditions.	8	19,51%	11	15,28%
6. Error in identifying a program without an error.	4	9,76%	2	2,78%
Total	41	100%	72	100%

Finding and debugging a programming error

The comparison of students' performance on the post-experience debugging test showed that students in the structured curriculum group outperformed their counterparts in the unstructured group (see Table 3), in terms of finding ($t(33)=2.17$, $p<0.01$) and debugging ($t(33)=3.58$, $p<0.001$) errors in a program. These results suggest that a more structured learning environment may be more effective in promoting students' abilities in terms of finding and debugging programming errors.

Table 3: Comparison of students' debugging test scores between the two groups

	Structured Group			Unstructured Group		t
	Mean	SD		Mean	SD	
Post-debugging test scores	72.8	9.91		60.63	11.07	3.26**
Finding the error	39.73	6.17		32.18	5.85	2.17**
Debugging	33.16	4.65		28.45	6.30	3.58*

Note. *p<.05, **p<.01.

Students' engagement

Data from the students' engagement survey showed that the students in both groups appreciated and enjoyed the ER learning experience with mean scores well above the mid-point (see Table 4). Differences between the groups were examined via an independent sample t-test. Results showed that the unstructured curriculum group (M=3.91, SD=0.38) outperformed their counterparts, who participated in the structured curriculum (M=3.65, SD=0.33), in terms of their emotional engagement; this difference was statistically significant; $t(33) = -2.17$, $p=0.037$. Students of the unstructured curriculum group (M=4.02, SD=0.44) also had a statistically significantly higher mean score on the dimension of social engagement than the unstructured group (M=3.45, SD= 0.37); $t(33) = -4.53$, $p<0.001$. There were no significant differences in students' cognitive and behavioral engagement between the two groups.

Table 4: Comparison of students' engagement between the two group

	Structured Group			Unstructured Group		t
	Mean	SD		Mean	SD	
Cognitive engagement	3.74	0.41		3.66	0.57	0.53
Behavioral engagement	3.82	0.38		3.76	0.38	0.50
Emotional engagement	3.65	0.33		3.91	0.38	-2.17*
Social engagement	3.45	0.37		4.02	0.44	-4.53**

Note.*p<.05, **p<.01.

The focus-groups data provided additional evidence of students' engagement, demonstrating a positive level of engagement. Students showed their cognitive engagement by describing the exchange of ideas, explanations, and directions in their groups. One student noted: "*It was a nice activity. It required to discuss and explain our ideas to others. Then we agreed for a plan, and we had to apply it"* (Unstructured group, Participant 5). Students showed their high level of behavioral engagement, reporting their full participation without distraction. One participant noted: *"I was focused on what I was doing at any time; I haven't disoriented from other things"* (Structured group, Participant 2). The students demonstrated their emotional engagement in the form of expressing happiness and interests. One participant noted. "*We felt happiness! When you see the robot to complete the activity, you feel happy"* (Unstructured group, Participant 3). One other noted, *"It was exciting, and I loved the robots. I asked my mother to buy it for me to practice at home"* (Unstructured group, Participant 2). Students of the unstructured group were referred more often on positive aspects of social engagement. We identify two sources of social engagement. The sense of good teamwork and the expressions of respect of the ideas of other group members. For example, one participant noted: "*I felt very comfortable with my team. We worked very well together, and we knew our responsibilities"* (Unstructured group, Participant 8).

Discussion

This study posits that a critical factor for the successful integration of robotics in education is the teaching approach. The role of the teaching approach for successful technology integration remains relatively unexplored in the area of ER. To our knowledge, this is the first time that a study investigates this topic, as prior studies focused on the level of guidance in a constructionist learning environment (e.g., Atmatzidou, Demetriadis, & Nika, 2018), or on the differences on the social aspect of learning such as collaboration and social interaction (e.g., Lee, Sullivan & Bers, 2013); it, therefore, represents a novel extension of prior work in this area.

First, to answer RQ1, a list of the common programming errors during blocks-based programming was produced. The list can be used by educators and researchers in teaching students how to debug programming errors. Four of the six common errors (i.e. #1, 3, 4, 6) were similar to those in a study by Kim et al., (2018) with early childhood pre-service teachers. Therefore, a great similarity appears about the type of errors that novice learners produce regardless of their age. Second, subsequent analysis of students' errors in the two groups showed a significant difference in the number of errors produced. Students who participated in the constructionist class involved in a significantly higher amount of errors than students who participated in the instructionist class. These results demonstrate the superiority of the instructionist approach against the constructionist as being more effective in teaching students essential aspects of the programming interface and make students become familiar with the functions of the programming environment. It makes sense; if you are allowed to explore on your own, you will make more errors. The procedure of solving an error (debugging) is considered as a problem-solving situation which students should experience and resolve productively. On the other hand, if students exposed to too many programming errors, this might become a source of frustration with a negative impact on their engagement.

Third, the results of the debugging test indicated that the structured group outperformed the unstructured group in a statistically significant degree in terms of finding and debugging an error. We can assume that the direct instruction (more instructive role of the teacher and the use of the worksheets) gave this group an advantage in terms of debugging. On the other hand, as the students in the unstructured group were exposed more in programming errors, one would expect to have learned how to debug the errors. The failure in this situation was not productive in terms of debugging. The students of the instructionist condition were stronger debuggers because they have stronger content knowledge, as a result of the direct instruction, and not because they were better at debugging. Summarizing, these results suggest that students could benefit more through an instructionist teaching approach in order to become better debuggers. Fourth, when focusing on the experienced engagement, students who participated in the unstructured curriculum group reported statistically significant higher levels of emotional and social engagement than students of the structured group. This finding is aligned with prior research efforts indicating that students, who participate in student-center environments, might experience higher levels of emotional engagement when compared to teacher-center approaches (e.g., Wu & Huang, 2007). We also found that students of the unstructured curriculum group had a grader level of social engagement than students of the structured curriculum group. This finding is consistent with that of Sullivan and Bers (2013), who showed that using an unstructured ER curriculum was linked with more collaboration than a structured curriculum.

Conclusions

This study shows that students express higher levels of engagement in an unstructured ER curriculum, while the structured ER curriculum produces better results in terms of finding and debugging an error. The main finding is that different teaching approaches to technology curriculum design can have a significant impact on various

aspects of learning. It is uncertain which method would be the best for enabling the development of students' skills in programming and problem-solving as both approaches seem to have some advantages. Therefore, we suggest that it would be better for teachers to design a curriculum with a combination of teaching approaches such as unstructured and structured activities in order students take advantage of the benefits of both approaches.

References

Anderson, J. R., Greeno, J. G., Reder, L. M., & Simon, H. A. (2000). Perspectives on Learning, Thinking, and Activity. *Educational Researcher, 29*(4), 11–13. https://doi.org/10.3102/0013189X029004011

Ardito, G., Mosley, P., & Scollins, L. (2014). WE, ROBOT: Using Robotics to Promote Collaborative and Mathematic Learning in a Middle School Classroom. *Middle Grades Research Journal, 9*(3), 73.

Atmatzidou, S., Demetriadis, S., & Nika, P. (2018). How Does the Degree of Guidance Support Students' Metacognitive and Problem Solving Skills in Educational Robotics? *Journal of Science Education and Technology, 27*(1), 70–85. https://doi.org/10.1007/s10956-017-9709-x

Socratous, C., & Ioannou, A. (2018). A Study of Collaborative Knowledge Construction in STEM via Educational Robotics. In J. Kay & R. Luckin (Eds.), *Rethinking Learning in the Digital Age: Making the Learning Sciences Count, 13th International Conference of the Learning Sciences* (ICLS) 2018 (Vol. 1, pp. 496-503). London, UK: ISLS.

Socratous, C., & Ioannou, A. (2019). An Empirical Study of Educational Robotics as Tools for Group Metacognition and Collaborative Knowledge Construction. In *13th International Conference on Computer Supported Collaborative Learning* (CSCL) 2019, Volume 1 (pp. 192-199). Lyon, France: International Society of the Learning Sciences.

Benitti, F. B. V. (2012). Exploring the educational potential of robotics in schools: A systematic review. *Computers and Education, 58*(3), 978–988. https://doi.org/10.1016/j.compedu.2011.10.006

Constantinou, V., & Ioannou, A. (2018). Development of Computational Thinking Skills through Educational Robotics. In EC-TEL *(Practitioner Proceedings*).

Hampden-Thompson, G., & Bennett, J. (2013). Science Teaching and Learning Activities and Students' Engagement in Science. *International Journal of Science Education, 35*(8), 1325–1343.

Ioannou, A., Socratous, C., & Nikolaedou, E. (2018, September). Expanding the curricular space with educational robotics: A creative course on road safety. In *European Conference on Technology Enhanced Learning* (pp. 537-547). Springer, Cham.

Ioannou, A., & Makridou, E. (2018). Exploring the potentials of educational robotics in the development of computational thinking: A summary of current research and practical proposal for future work. *Education and Information Technologies, 23*(6), 2531–2544.

Kim, C. M., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science, 46*(5), 767–787. https://doi.org/10.1007/s11251-018-9453-5

Lee, K. T. H., Sullivan, A., & Bers, M. U. (2013). Collaboration by Design: Using Robotics to Foster Social Interaction in Kindergarten. *Computers in the Schools, 30*(3), 271–281.

Lindh, J., & Holgersson, T. (2007). Does lego training stimulate pupils' ability to solve logical problems? *Computers and Education, 49*(4), 1097–1111. https://doi.org/10.1016/j.compedu.2005.12.008

Papert, S., & Harel, I. (1991). Situating Constructionism. *Constructionism, 36*(2), 1-11.

Sullivan, F. R. (2011). Serious and playful inquiry: Epistemological aspects of collaborative creativity. *Journal of Educational Technology & Society, 14*(1), 55–65.

Wang, M. T., Fredricks, J. A., Ye, F., Hofkens, T. L., & Linn, J. S. (2016). The math and science engagement scales: Scale development, validation, and psychometric properties. *Learning and Instruction*, *43*, 16-26. https://doi.org/10.1016/j.learninstruc.2016.01.008

Weintrop, D., & Wilensky, U. (2015 June). To block or not to block, that is the question: Students' perceptions of blocks-based programming. In *Proceedings of the 14th International Conference on Interaction Design and Children* (pp. 199-208). ACM. https://doi.org/10.1145/2771839.2771860

Wu, H. K., & Huang, Y. L. (2007). Ninth-grade student engagement in teacher-centered and student-centered technology-enhanced learning environments. *Science education, 91*(5), 727-749.

Acknowledgements

This work has been partly supported by the project that has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 739578 (RISE – Call: H2020-WIDESPREAD-01-2016-2017-TeamingPhase2) and the Government of the Republic of Cyprus through the Directorate General for European Programs, Coordination and Development.