ISSN 2513-8359

Teaching Computer Science Concepts through Robotics to Elementary School Children*

Mor Friebroon-Yesharim[1]

Mordechai Ben-Ari[1]

[1]Weizmann Institute of Science

DOI: 10.21585/ijcses.v2i3.30

Abstract

Studying computer science (CS) in elementary schools has gained become popular in recent years. However, students at such a young age encounter difficulties when first engaging with CS. Robotics has been proposed as a medium for teaching CS to young students, because it reifies concepts in a tangible object, and because of the excitement of working with robots. We asked: What CS concepts can elementary-school students learn from the participation in a robotics-based CS course? We used two theoretical frameworks to explain possible difficulties in learning: the Jourdain effect, and constructs vs. plans. A taxonomy of six levels was created to characterize levels of learning. The levels were measured using four questionnaires that were based on the taxonomy. In addition, field observations of the lessons were recorded. The population consisted of 118 second-grade students (ages 7-8). Lessons on CS concepts using Thymio educational robot and its graphical software development environment were taught during normal school hours, not in a voluntary extracurricular activity. The syllabus was based on existing learning materials that were adapted for the young age of the students. The analysis showed that the students were very engaged with the robotics activities. They did learn basic CS concepts, although they found it difficult to create and run their own programs. We concluded that the Jourdain effect was not demonstrated because the students understood concepts and constructs of CS; however, they were unable to plan and construct their own programs from the basic constructs.

Keywords: elementary school, computer science, Thymio robot, Braitenberg creatures

1 Introduction

1.1. Research goals and research framework

Studying computer science (CS) in elementary schools has gained become popular in recent years. The goal of teaching CS at a young age is primarily to increase self-efficacy and motivation when engaging with science and technology. However, students at such a young age face difficulties when first engaging with CS. One approach to overcome these difficulties is to use robotics activities, because they reify abstract CS concepts in a tangible object and because of the excitement of working with robots. The goal of our research is to distinguish between the performance of a task and the understanding of the concepts.

The phases of the research were:

1) The development of an age-appropriate syllabus partially based on existing learning materials for the educational robot.

---

* Preliminary results of this research were presented at the International Conference on Robotics in Education, April 2017, Sofia, Bulgaria.

2) A quantitative and qualitative assessment to determine if the use of a robot-based syllabus enables young students to understand CS concepts.
3) The development of taxonomy appropriate for specifying the levels of understanding of young students who learn CS.

Section 1 presents a review of existing literature. The methodology, including the new taxonomy, is described in Section 2. The findings are presented in Section 3, discussed in Section 4 and summarized in Section 5.

1.2. Review of existing literature

The literature review is divided into five sections: (a) learning CS concepts by elementary school students, (b) learning CS by robotics, (c) learning CS with robotics in elementary school, (d) the Jourdain effect, (e) near transfer.

1.2.1. Learning CS concepts by elementary school students

Papert was among the first to propose teaching programming to young children (Papert, 1980). He coined the term constructionism for learning by constructing artifacts such as computer programs. Research has shown that CS studies had some positive effects on cognitive development, thinking skills, problem-solving strategies, creativity, intrinsic motivation, or even social development specifically at young ages (Liao & Bright, 1991; Clements, 2002; Clements & Sarama, 2003). Duncan and Bell (2015) explored and analyzed CS curricula for elementary schools; they found that some countries have already incorporated formal studies as part of the curriculum, while others are limited to informal classes and clubs. Seiter and Foreman (2013) explored the development of computational thinking by elementary school students using Scratch. They found that basic proficiency of algorithmic thinking started in second grade. Clements and Sarama (1997) studied learning with LOGO and concluded that it can provide an evocative context for young children's explorations of mathematical ideas and CS concepts.

Duncan et al. (2014) raised the question: Should eight-years-old students learn to code? They proposed three parameters to establish effective learning: (1) the teachers must be confident and motivated; (2) the learning objectives should be realistic for the age of the students; (3) the development environment should be age-appropriate. They found that the age at which programming should be taught depends on many factors among them the software tools and learning aids, the context and the teachers' training. Armoni and Gal-Ezer (2014) and Duncan et al. (2014) claimed that the advantages of learning CS at such a young age include the capability: to learn quickly, to shape attitudes to programming, to support learning outside of just programming, and to prepare students for future endeavors in computing. The disadvantages of engaging with CS at a young age include the possibility that students will be less confident in their abilities regarding CS or will receive a negative impression of the subject. Furthermore, the students may study fewer hours in core subjects such as mathematics, science and language skills (Duncan & Bell, 2015). The limited time available and the lack of resources could cause problems in the allocation of school resources (Duncan et al., 2014).

1.2.2. Learning CS via robotics

Since the 1980s, both curricular courses and outreach programs on robotics have been developed. A pioneering successful tool for teaching CS with robotics was the environment Karel the Robot (Pattis, 1981). Anderson et al. (2011) claimed that robotic activities are very exciting for the students and that they reify the abstract behavior of algorithms and programs. Robotics provides hands-on experience with real-world problems and can also reduce the level of intimidation that students can encounter. Ben-Bassat-Levy and Ben-Ari (2015) showed that robotic activities can influence both the motivation and the self-efficacy of young students. They found that robotics encourages positive intentions to choose STEM (science, technology, engineering, mathematics) subjects in high school. Markham and King (2010) investigated attitudes and motivation among CS1 students, they found that students who studied with robots had more positive experiences than those who studied without robots. Kaloti-Hallak

et al. (2015) investigated young students who participated in the FIRST® LEGO® League competitions. Their research showed that students demonstrated meaningful learning in computer science and engineering, and that most of the students demonstrated high positive attitudes and motivation for learning robotics. Kay (2011) showed that CS learning by high-school and undergraduate students really improve when robots are used.

1.2.3. Learning CS with robotics in elementary school

Barker and Ansorge (2007) taught 9–11 year-old students and found that the LEGO Mindstorms® robotics kit was effective for teaching STEM concepts. Magnenat et al. (2014) ran a workshop for students aged 8–9 using the Thymio educational robot. They found that while students successfully used trial and error when writing programs that controlled the robot, they only understood a subset of the CS concepts that appeared in their programs. Both these projects involved extracurricular activities.

Several research projects addressed learning with robotics by young children. Martinez et al. (2015) taught robotics to students of a variety of age groups from 3 to 11. They showed that the older students were capable of understanding and applying CS concepts such as loops, parameters, conditions and sequencing, while preschool students understood fewer concepts. Sullivan and Bers (2016) implemented a robotics curriculum in preschool through second grade. They found that younger children were able to master basic concepts of robotics and programming, while older children were able to master more complex concepts. Bers et al. (2014) engaged 4–6 year-old children in robotics activities in order to guide age-appropriate curriculum development. Wyeth (2008) showed that children can learn simple programming concepts related to input and output, and the impact of logic on program behavior. Common to all these research projects is the use of robots specifically designed for young children, in particular, Bers' group used tangible programming (programming using physical blocks), which can no longer be used for older students.

1.2.4. The Jourdain effect

Guy Brousseau proposed the *theory of didactic situations* as a framework for investigating learning, in particular, mathematics (Brousseau, 1997). We were influenced by his discussion of the *Jourdain effect*, the conflation of the performance of a task with understanding the underlying concepts. Here is an example from the New Math curriculum:

> [A] model of this group can be constructed using some transformations of the position of a cup of yogurt … As children were playing with the cups of yogurt, they were performing such transformations … from this, the 'structurally minded' observers were concluding that the children 'have constructed' the group of Klein. But what the children were actually doing had nothing to do with the identification of the group structure in their manipulations. … [T]hey would not have been able to identify the part of their activity called 'construction of the group of Klein' … and they would not be able to produce …, further examples of their activity, now sanctified by a scientific term (Sierpinska, 2003, no page numbers).

1.2.5. Near transfer

It is frequently claimed that learning certain subjects results in transfer of knowledge: a general improvement of cognitive and problem solving abilities. Such claims have recently been made for computational thinking (see the analysis by Tedre and Denning (2016)). Such claims have been repeatedly debunked (Guzdial, 2015) and we make no such claims for learning CS with robotics. However, a classic investigation by Gick and Holyoak (1980) showed that learning in one domain can aid in solving *analogous* problems in a different domain; this is called *near transfer*. While our research is not a comprehensive study of near transfer, we did investigate whether the students were able to transfer their knowledge of robotics commands to a hypothetical command somewhat different from the actual commands.

2 Methodology

2.1. Rationale for the research

There were three aspects to the rationale for this research:

1) The robotics activities were taught in the non-voluntary, non-selective environment of a normal classroom during school hours with just one teacher and one assistant. In previous work (Ber et al, 2014), several research assistants were able to help individual groups of students. Finally, a general purpose educational robot was used, not one specifically developed for the project.
2) We developed a questionnaire based on a taxonomy of learning in order to attempt to distinguish performance from understanding (the Jourdain effect).
3) We wanted to demarcate what this specific age group was able to learn from concepts that were too difficult for them.

2.2. Research question

What CS concepts can elementary school students understand from participation in a robotics-based CS course?

2.3. Population

The research was carried out in four second-grade classrooms of a public school (ages 7–8 years). All the students in these classes participated during normal school hours, so we had no control over the actual ages, genders or abilities of the students. There were 118 students, 72 boys and 46 girls. The first author taught the lessons aided by a research assistant from our department. The class teachers were present, but we did not have time to train them in robotics and CS so they were not involved in teaching. However, they remained in the classrooms, primarily to deal with behavioral problems that occasionally arose.

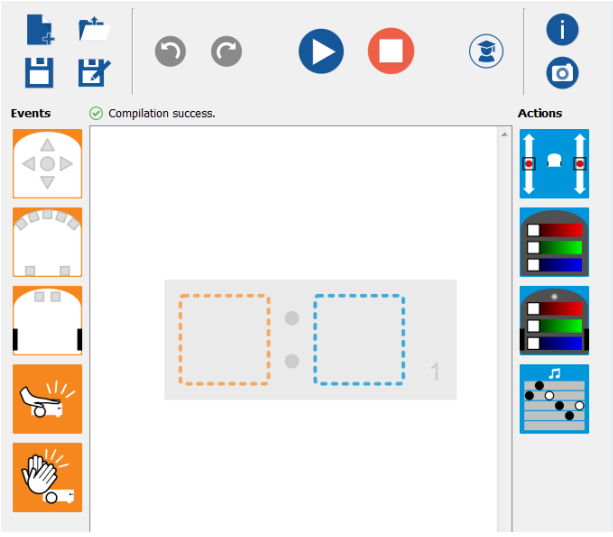
2.4. The robot and its software environment

The Thymio educational robot (Figure 1) is small, self-contained and very robust with differential drive, nine infrared proximity sensors, five touch-sensitive buttons, a 3-axis accelerometer, dozens of LEDs, a speaker and a microphone (https://www.thymio.org/en:thymio).

Figure 1. Thymio robot

The robot is programmed using the Visual Programming Language (VPL) environment (Figure 2) (Shin et al., 2014). Programs are constructed by drag-and-drop of graphical blocks. VPL supports one programming construct: event-action pairs. Event handling is a core CS concept, which has been proposed as the basis of teaching introductory programming (Bruce et al., 2006). In addition to the basic blocks shown in Figure 2, there is an advanced mode that supports additional blocks and advanced versions of basic blocks.

Figure 2. The VPL environment. The events are on the left, the actions are on the right and the central area is used for constructing programs



2.5. The robotics class

The course was taught for one hour a week for 21 weeks during normal school hours. Thirty students shared ten robots. Each lesson began with a short video that introduced a concept. Then the students received a worksheet.

The syllabus was based on existing learning materials for the Thymio (Ben-Ari, 2011; Magnenat et al., 2012), by selecting age-appropriate activities from this material. The first tasks were based on the predefined behaviors of the robot and were intended to familiarize the students with the events generated by the proximity sensors and the buttons, and with the actions of changing the colors of the LEDs. Then the students were introduced to the VPL graphical programming environment. Many of their tasks were to implement *Braitenberg creatures* (Braitenberg, 1984), which were developed during the Programmable Brick project (Hogg et al., 2000) that was the inspiration for LEGO Mindstorms®. Later, the students explored combinations of several actions per event and blocks from the VPL's advanced mode: timer events and actions, accelerometer events and music actions.

In addition to the concepts explicitly expressed in the Thymio robot and VPL, the following CS concepts that appeared only implicitly were taught:

1) Concurrent execution of event-action pairs: for example, in the line-following program, the program simultaneously checks if the robot is leaving the left edge or the right edge of the line.
2) Parameters: setting the color of the LEDs and the power applied to each motor.
3) Writing an algorithm and implementing it in a program.

Table 1 presents the topics and activities that took place in each lesson.

Table 1. The content of the lessons

Lesson	Content
1	Pre-programmed behaviors.
2	Pre-programmed behaviors in groups.
3	The VPL user interface. Events (buttons, sensors) and actions (top and bottoms colored LEDs). First experience programming.
4	The VPL user interface. Programming five exercises.

5	First questionnaire (four questions).
6	The motor action. Programming the Braitenberg creatures.
7	First questionnaire (two questions). Second questionnaire (two questions). Checking the exercises from the previous class. Multiple actions for one event.
8	Second questionnaire (two questions). Braitenberg creatures with multiple actions per event.
9	Checking the Braitenberg creatures exercises.
10	The clap and tap sound events and the sound action.
11	Second questionnaire (two additional questions). Bottom sensors.
12	Programming with the bottom sensors.
13	Programming with the bottom sensors.
14	Programming with the bottom sensors (summary).
15	Third questionnaire (two questions). The advanced mode.
16	Accelerometers.
17	Fourth questionnaire (four questions). Timer blocks.
18	Fourth questionnaire (two questions). Exercises with the timer blocks.
19	Exercises with the timer blocks.
20	Group projects.
21	Group projects.

2.6. The taxonomy

Our method for investigating performance vs. understanding was based on a taxonomy of levels of learning. We developed a new taxonomy appropriate for the context of learning CS concepts.

2.6.1. Justification for developing a new taxonomy

There are several existing taxonomies of levels of learning: SOLO (Biggs & Collis, 1982), Bloom (Bloom et al., 1956), Lister et al. (2004) and the CS-specific taxonomy of Fuller et al. (2007). Meerbaum-Salant et al. (2013) combined the Bloom and SOLO taxonomies, producing a new scale with three categories (unistructural, multistructural and relational) each containing three sub-categories (understanding, applying, and creating). Magnenat et al. (2014) based their work on learning with the Thymio robot on the combined taxonomy. They investigated two age groups: 8-9 and 10-15 years old. They checked the levels of understanding using questionnaires. While most of the students of the older group achieved the level of unistructural understanding, the young age group found it hard to answer the questions because of their limited ability to read. The difficulties they encountered led us to develop a new taxonomy.

2.6.2. The new taxonomy

A student demonstrating the follow behaviors will be considered to have achieved a corresponding level of learning. In ascending order they are:

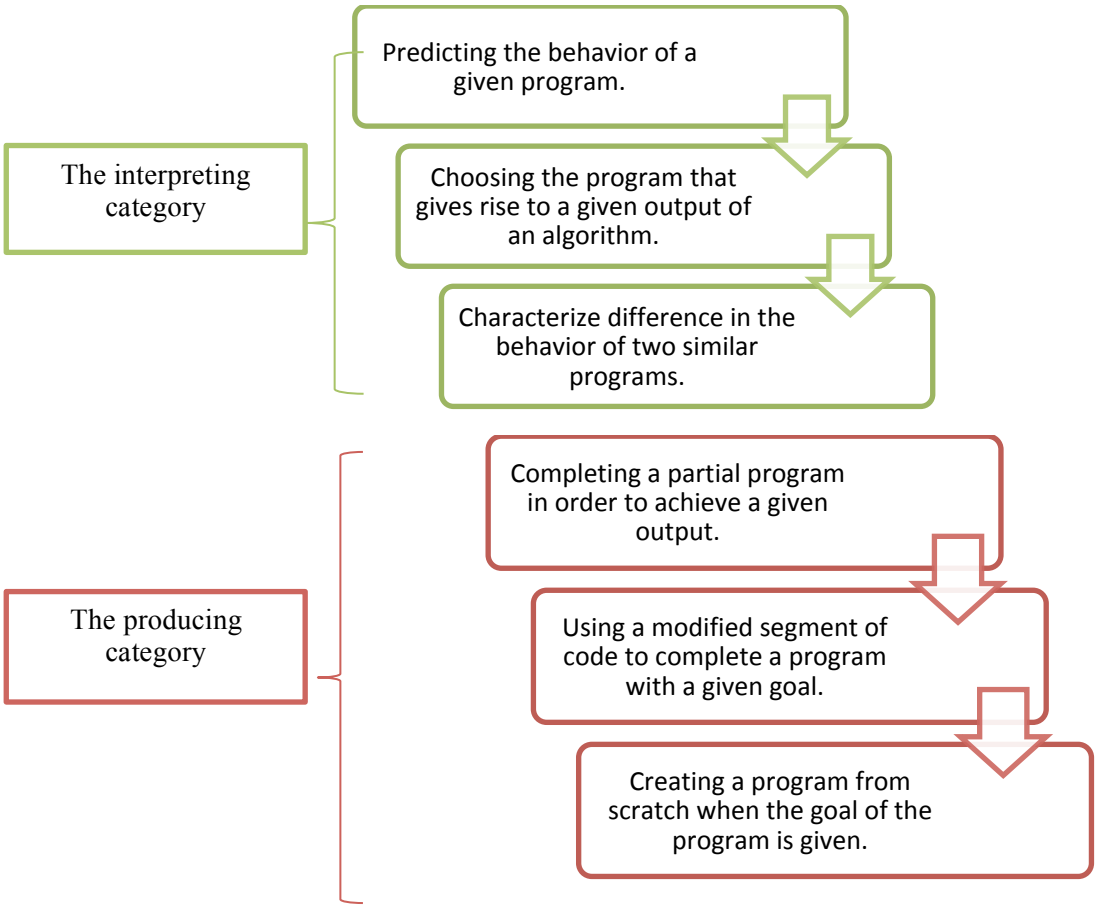
1) Predicting the behavior of a given program.
2) Choosing the program that gives rise to a given output of an algorithm.
3) Characterizing the difference in the behavior of two similar programs.
4) Completing a partial program in order to achieve a given output.
5) Using a modified programming construct to complete a program with a given goal.
6) Creating a program from scratch when the goal of the program is given.

The different types of behaviors can be classified into two categories (Table 2) based on Fuller et al. (2007):

1) Interpreting: to give or provide the meaning of; explain.
2) Producing: to bring into existence by intellectual or creative ability.

The interpreting category includes only behaviors for which the code is given, as oppose to the producing category whose behaviors require code completion. This enabled us to distinguish between students who can only read and understand programs but cannot necessarily write or complete one on their own.

Table 2. The classification into two categories



2.6.3. Secondary classification: The interpreting category

The ordering of the three behaviors in the interpreting category can be justified as follows.

Lister et al. (2004) claimed that the ability to predict the behavior of a given program while tracking and following its instructions is the lowest cognitive level required for a CS student. The second type of

question differs from the first type in that the students are required to choose the right program from several slight different programs. Lister et al. (2004) claimed that students showed less success in this type of questions then in questions predicting the behavior of a program.

Differentiating between two programs is similar to comparing, but it adds the requirement to predict the purpose of the programs. Differentiating also requires analytical skills to identify similarities and differences, which is a higher cognitive process than just predicting, according to the revised Bloom's taxonomy (Anderson et al. 2001). Therefore, this is the most challenging behavior in this category.

2.6.4. Secondary classification: The producing category

In this category, the student is asked to complete a program. Even if the questions are multiple-choice, this category requires the student to analyze the purpose of the program and to choose the most suitable completion. The student must understand the functionality of the missing parts, and then to understand how each possible completion will affect the program.

The second behavior and the question refer to it; the students are presented with modified block whose functionality is explained. They need to complete a given segment of code correctly using this modified block. The cognitive stages are very similar to the first behavior in this category, but, in addition, the student needs to show proficiency in the material he has already learned in order to understand the purpose of the modifications. We consider this process to be near transfer of knowledge and this type of behavior is harder than the first one.

The most challenging type of behavior in this category asks the students to create a program from scratch when the goal of the program is given. This requires the student to use all of the knowledge he has gained so far and to create a new artificat. Both the Bloom (Bloom et al., 1956) and SOLO (Biggs, & Collis, 1982) taxonomies rate this cognitive skill as the most challenging one.

2.6.5. Constructing questions according to the taxonomy

Magnenat el al. (2014) found that elementary school students had difficulty understanding long passages of text; this led them to use graphics and video clips in their questionnaires. We followed their lead in constructing our questionnaires. Graphics is particular appropriate in this context since the VPL programming environment uses graphical elements only with no text. Here is a description of the format of the questions associated with each level of the taxonomy:

1) Predicting the behavior of a given program: The students received a code segment and four photos of the robot and they had to choose the photo that demonstrated the behavior of the robot caused by the code segment.
2) Choosing the program that gives rise to a given output of an algorithm: The students received a short video and had to choose the code segment that gives rise to this behavior.
3) Characterizing differences in the behavior of two similar programs: The students watched two short videos displaying behaviors of the robot and they had to choose among four descriptions the one that describes the difference between the behaviors.
4) Completing a partial program in order to achieve a given output: The students were given a short segment of code and a goal that the program needs to achieve. The students had to complete the missing parts of the code in order for the program to achieve the goal.
5) Using a modified segment of code to complete a program with a given goal: The students were given a modified block whose meaning was explained in the question and they had to complete the code segment in order to implement the behavior demonstrated in a short video or explained in words.
6) Creating a program from scratch when the goal of the program is given: The students were given several event-action blocks and they had to choose and arrange the blocks needed in order to achieve a given goal.

2.7. Conjectures

We proposed three conjectures to explain why some students might achieve only lower levels of

learning, together with criteria to accept or reject the conjectures:

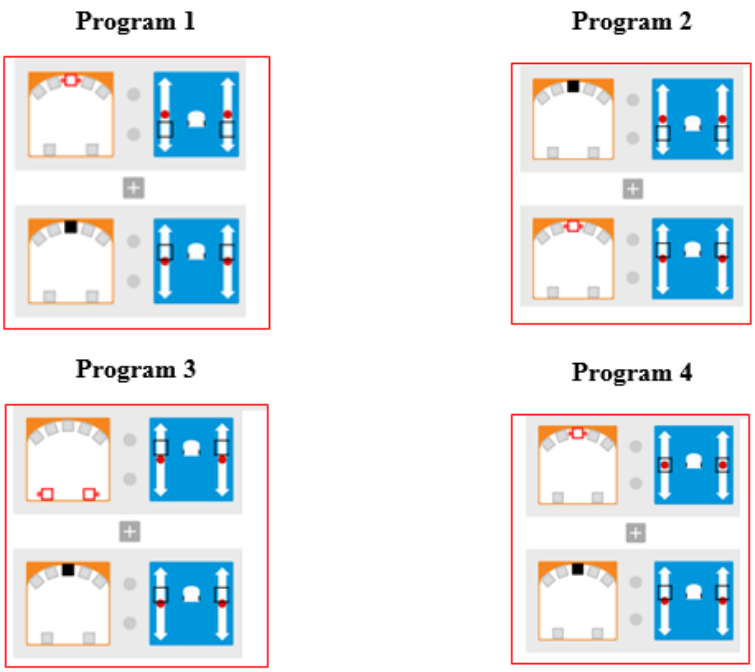
1) **They really don't understand** The simplest possibility is that young students don't understand most of the CS concepts that they are exposed to, and that the success reported in previous research has been misinterpreted. This conjecture will be supported if we find that less than 50% of the students succeed in answering questions even for the lowest levels of learning.
2) **Jourdain effect** The students will demonstrate the Jourdain effect if they successful answer the predicting and choosing types of questions, partial succeed in the answering difference questions, and are unable to answer any of the questions from the second category.
3) **Constructs vs. plans** Soloway and Spohrer (1986) suggest that there is a gap between the ability of novice programmers to understand individual constructs and their ability to plan and implement a functioning program. This conjecture will be supported if students are only able to answer questions from the interpreting category and the first question of the producing category. Students demonstrating the ability to implement plans will be able to answer questions from all the categories.

2.8. Research Instruments

Four questionnaires of six multiple-choice questions each were administrated during the regular lessons. They looked like the usual worksheets and the students willingly participated in solving the problems. After the first experience with the class assignments, it was decided that the maximum number of questions that these young students could deal with in one lesson is four, so the questionnaires were split over two or more lessons. The questionnaires can be found at https://goo.gl/peKBpp.

The topics asked about in the questionnaires are as follows:

1) The first questionnaire



CS concepts: simple event-action pairs.
New events and actions blocks: sensors, buttons, top and bottom colors.

2) The second questionnaire
CS concepts: advanced event-action pairs.
New events and actions blocks: motors.
3) The third questionnaire

CS concepts: multiple actions in one event-action pairs.
New events and actions blocks: tap and clap detection.

4) The fourth questionnaire
CS concepts: line following, concurrent execution.
New events and actions blocks: ground proximity sensors.

After each lesson, the first author recorded her observations and solicited impressions from the research assistant and the class teacher.

In order to fully investigate the ability of students to plan and implement a program, during the final lessons they were asked to come up with their own ideas for writing a program. The programs and observations of the programming process were recorded.

After the final lesson, the first author met with the teachers to discuss their impressions of the students' ability to learn CS and robotics.

2.8.2 Example questions from the questionnaires

Question 2 from questionnaire 2

Watch the video: https://youtu.be/okoLamAY9ac. Which of the following programs causes the behavior of the robot that is shown in the video?

Question 2 from questionnaire 3

Look at the video: https://youtu.be/Vhc33fxR3co. Which of the following programs causes the behavior shown in the video?

Question 5 from questionnaire 1

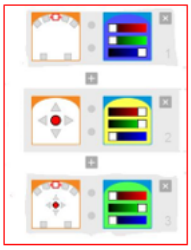
We invented a new event: The center button is touched and at the same time an object is detected by the center front sensor. Here is the block for the new event:

Use the new event to construct a program that does the following:

1) Detecting an object only by the front center sensor causes the top light to display blue.
2) Touching only the center button causes the top light to display yellow.
3) Touching the center button and at the same time detecting an object by the front center sensor causes the top light to display green.

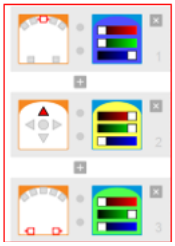
Choose the correct program:

Program 1

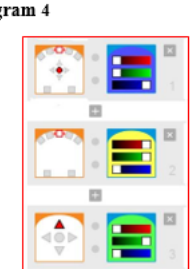


Program 2

Program 3

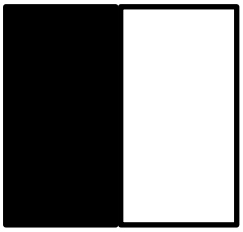


Program 4



Question 5 from questionnaire 4

We want a program that causes Thymio to follow the edge between a white area and a black area:



Here is a description of the program:

1) If Thymio detects white under the right bottom sensor and it detects black under the left bottom sensor, then Thymio moves forwards.
2) If Thymio detects white under both bottom sensors, then Thymio turns left.
3) If Thymio detects black under both bottom sensors, then Thymio turns right

Choose the correct program:

2.9. Data Analysis

The findings from the observations will be integrated with a description of the syllabus, in order to show how the students reacted to each individual topic. The analysis was influenced by Glaser and Strauss (1967) and was done as follows:

1) Collate the observations of the four classes with the same lesson plan.
2) Identify the important events, and the similarities and differences among the four classes.
3) Unify similar events and important concepts.
4) Link the different categories in order to understand the students' capabilities.

The discussion of the focus group with the teachers was recorded and transcribed. The analysis of this focus group was identical to the analysis of the observations of the students.

The project lessons were analyzed in order to find the CS concepts that the students used and to discover the level of understanding that the students achieved.

The questionnaires were analyzed quantitatively using Pearson's chi-squared test, which is used to determine whether there is a significant difference between the expected frequencies and the observed frequencies in one or more categories. Since each question was asked in all of the four classes, and every question had four options, we found the chi-squared test most appropriate for the research needs. The chi-squared test was used to determine whether there was a significant difference between the expected frequencies and the observed frequencies in one or more of the four classes. For each questionnaire, a table is given which indicates the uniformity of the success rates of the four classes. The first column is the question number and the second column is the success rates.

For every question of the questionnaires, the null hypothesis was that there were differences among the four classes; the alternate hypothesis was that there were no differences in the success rates. Questions that did not negate the null hypothesis were marked with a gray background, and the success percentages for those questions will be presented separately for each of the four classes.

When the percentage of the students who answered a question correctly passed 50% were marked with bold face, we took that as evidence that the level of the taxonomy corresponding to that question was achieved by the majority of the students.

In the presentation of the findings, we will emphasize the questions that didn’t negate the expected frequencies and provide explanations in cases of discrepancy.

2.9.1 Reliability

Since the students had no previous background in CS, there was no reason to administer a pre-test. Colleagues were asked to judge that the curriculum and research methods involve well-known CS concepts and are age-appropriate. To check the reliability of the knowledge questionnaires, both colleagues and the teachers examined the questionnaires and expressed their opinions regarding their difficulty.

3 Findings

3.1. Initial experience with the robot

The first two lessons with the robot were designed to give the students an easy start, so they were conducted without a computer. During the lessons the students worked in groups. Each group had to perform several activities using the robot's pre-programmed behaviors (https://www.thymio.org/en:thymiostarting). In the second lesson, the students performed tasks with the help of a discovery kit (https://www.thymio.org/en:thymiodiscoverykitotter).

The students were excited; some girls said that the robot was "cute," while the boys described it as "cool" and indicated that Thymio is "fun." During the performance of the tasks, the students followed the instructions while correcting each other. When they were given open questions, they kept asking "what to write?" It seemed as if they wanted to be correct at every single question.

During class, the students managed to manipulate the robot and to switch between the robot's pre-programmed modes, but they found it hard to describe the differences between the modes. However, they were able to describe the different behaviors, as shown by their terminology: "it made a sound when I put both of my hands," "Thymio drove forward in the direction of the doll!" The students were not able to memorize the different modes, but they knew how to describe the modes during the activities.

3.2. Initial experience with programming

During the third and fourth lessons the students started to program. They were introduced to the VPL environment and used it to learn how to write programs.

The first worksheet with VPL included the following events: pressing the buttons, detecting objects with the front and back sensors, and the following actions: turn on the top and the bottom LEDs. The students were thrilled by their first experience writing programs, they were highly motivated to succeed and readily proceeded from one exercise to the next. The students easily implemented the three tasks with a single event-action pair in this lesson. All the students seemed engaged with the class activities and the worksheet, asking questions and exploring the capabilities of the robot and VPL.

The students frequently asked "where to put it?" with regard to an event or an action; sometimes they used the buttons event without indicating which button, so they tended to ask many questions of the form "why isn't it working?" or "what am I doing wrong?" All the students wanted to be the ones with hands on the robot or the computer.

During the fourth lesson, the students were introduced to exercises that included more than one event-action pair. Initially, the students had difficulty understanding that they should be placed one after another on the screen. Although this was explained to the students, they found it unnatural, and asked "where to put this event?" or they just started a new program for each event-action pair. After sufficient practice they felt more confident with the concept of a program containing multiple event-action pairs.

3.3. The first questionnaire

The success rates of the first questionnaire (administered during lessons 5 and 7) are shown in Table 3. The first questionnaire investigated whether the students could associate an event with an action.

Table 3. First questionnaire, % of students who gave correct answers for each question

Question	Success Rate
1	**82**
2	**87**
3	**69**
4	**93**
5	**74**
6	**51**

For question 1, the α obtained from the chi-squared test was 0.0039; this result is greater than the likelihood ratio chi-squared, which is 0.0009, therefore the null hypothesis (that there were differences among the four classes as explained in section 2.9) cannot be rejected. Question 1 asked about the behavior of a given program. Although different success rates were obtained for the three classes, the success rates of all classes were greater than 50% (Table 4).

Table 4. The success rates of questionnaire number 1, question number 1 in the different classes

	Class A	Class C	Class D
Success Rate	**100**	**85**	**64**

From Table 3 we see that the students answered questions at levels 1 to 5 quite easily: (1) they could understand what a given program does, (2) match a program to an output, (3) characterize the difference in the behavior of two similar programs, (4) complete a partial program, (5) use a modified programming construct. However, for question 6, creating a program from scratch, their success rate was relatively low, but still reached 50%.

3.4. Programming the Braitenberg creatures

During the sixth lesson the students explored the motor block on their own. They copied programs given in the worksheet into the VPL environment, ran the programs and explained the behavior of each program: moving forwards, moving backwards and turning right and left. After understanding the motor block, the students wrote programs to implement the Braitenberg creatures. During the activities with the creatures, it was observed that these exercises helped the students "bond" with the robot and gave it some human qualities. The students began to refer to the robot as a living: "Thymio didn't do what I asked him!", "I miss Thymio," "I love Thymio."

3.5. The second questionnaire

Questionnaire 2 (administered during lessons 7, 8 and 11) investigated if the students can associate an event with an action, but this time they had more blocks at their disposal, which made the programs more complex. The success rates for the second questionnaire are shown in Table 5.

Table 5. Second questionnaire, % of students who gave correct answers for each question

Question no.	Success Rates
1	**52**
2	46
3	49

4	**76**
5	**70**
6	46

The first two questions of the questionnaires were given to the students just after their initial experience with the new motor block, while the next four questions were given after they had much more practice and experience with the robots.

The high success rates for questions four and five compared with those of questions one and two can be explained by the additional practice that the students had, which enabled them to answer questions at the higher levels. The students encountered difficulties answering question 3: it was hard for them to identify the difference between a pair videos showing behaviors of the robot. Nevertheless, in three of the four classes, a majority of the students successfully answered the question. The success rates for question number 6 were low: the students found it difficult to write a program from scratch given the desired behavior of the robot.

The heterogeneity between the four classes was more significant in this questionnaire. For questions 3, 5 and 6, the α obtained from the chi-squared test were greater than the likelihood ratio chi-squared, so the null hypothesis could not be rejected.

Table 6 shows the α obtained from the chi-squared test and the likelihood ratio chi-squared for each of the questions.

Table 6. $\boldsymbol{\alpha}$ from the chi-squared test and the likelihood ratio chi-squared for each of the questions in questionnaire 2

Question	The α result obtained	The likelihood ratio chi-squared
1	0.1089*	0.05
2	0.0890	0.0862
3	0.0133	0.0105
4	0.5188*	0.5061
5	0.0506	0.05
6	0.0312*	0.05

*Because these three questions contained more than one section, the α obtained is the average of the differences between the four classes and actually represents the Generalized Linear Model (GLM) test.

It can be seen that the differences between the α obtained and the likelihood ratio chi-squared obtained are very small, so we unified the success rates of the four classes.

Table 7 shows the success rates of the different classes and where there were differences between the α and the likelihood.

Table 7. The different success rates of questions 3,5 and 6 in the different classes

	Class A	Class B	Class C	Class D
Question 3	**63**	**52**	**58**	22
Question 5	**71**	**93**	**83**	33
Question 6	42	**64**	**58**	26

The low success rates for class D can be explained by significant discipline problems that occurred.

3.6. Event handling with multiple actions

During lessons 7 through 9, the students were taught how to associate multiple actions with an event

using two simple examples. In lesson 8 the students practiced this construct by implementing additional Braitenberg creatures. The students found the transition from one action to multiple actions difficult. Frequently, they duplicated the event and associated it with the second action, asking: "where to put it?" or "what to do with…?". They tended to associate the position of a button with the direction of a movement, for example, they selected an event of touching the front button and then they were disappointed that the robot didn't move forward.

During the ninth lesson, the teacher solved questions together with the students. This facilitated bringing most of the students to a uniform level of understanding and, furthermore, helped clear up the difficulty of multiple actions associated with one event. They readily volunteered to come to the board to solve questions and were proud when they gave correct answers.

During the tenth lesson, the students learned about tap detection, clap detection and the music event blocks. They composed their own tunes and were enthusiastic about the feature of tapping and the clapping, which this facilitated practicing more advanced event handling.

3.7. The third questionnaire

The third questionnaire (administrated in lessons 14 and 15) involved questions regarding multiple actions per event. The success rates are shown in Table 8.

Table 8. Third questionnaire, % of students who gave correct answers for each question

Question no.	Success Rates
1	**75**
2	**61**
3	**68**
4	34
5	**61**
6	48

The success rates of questionnaire 3 were mixed: while the students were relatively successful on questions 1, 2, 3 and 5, they were less successful on questions 4 and 6, which required the students to complete a partial program and to create a program from scratch. Question 4 was hard since it required them to seek both an event and an action of a pair appropriate for the behavior that was given. The distractors made it difficult because they involved an incorrect ordering of an action before an event, incorrect direction of movement and irrelevant blocks. The fifth question investigated whether the students can perform near transfer of their existing knowledge.

The fifth question investigated whether the students can perform near transfer of their existing knowledge. The success rates in this question were not uniform for the four classes. The α obtained from the chi-squared test was 0.0467, which is greater than the likelihood ratio chi-Squared 0.0383; therefore, the null hypothesis could not be rejected.

The success rates of the four classes on question 5 of questionnaire 3 are presented in Table 9. The students achieved high success rates (>50%) in three of the four classes.

Table 9. The success rates of questionnaire number 3, question number 5 in the different classes

	Class A	Class B	Class C	Class D
Success Rates	**59**	**81**	**59**	44

The table shows that class B obtained very good success rates, while the success rate of classes A and D was much less though still greater than 50%. Class D did not achieve the desired success rate.

3.8. The bottom sensors and the line following algorithm

During lesson 11, the students explored the bottom sensors of the robot. The exploration activity included painting stripes of different colors and experimenting to see how the robot reacted to the different colors. The elementary physical principles of light sensitivity were explained. The exploration activity prompted some creativity on the part of the students, but it would have been preferable to supply the students with stripes in different colors, so that the students could readily identify the differences between the robot behavior to bright colors and dark colors. They kept asking: "Is it dark enough?" or "Is it bright enough?"

During lesson 12, the students received a detailed explanation of the bottom sensors and the differences between them and the front and back sensors. The students were a bit confused on this issue. The task required them to build a white track and a black track, and to execute different line-following algorithms. While the students enjoyed constructing lines using white and black tapes, they were confused by the execution of the algorithms. They kept asking questions regarding the bottom sensors block such as: "Should it be black?" or "What does the red frame mean?" The students invested much effort into building the tracks instead of attempting to understand the meaning of the bottom sensors event.

Lesson 13 started with a review on how the bottom sensors operate and how to use the bottom sensors event block in VPL. The students used the tracks they created during the previous lesson and once again implemented the line-following algorithms. Additionally, the students executed a program in which the robot had to stop at the edge of the table by sensing the edge with the bottom sensor. Again, the students had difficulties executing these algorithms and they needed more assistance about adjusting the bottom sensors.

During lesson 14 the students repeated the exercises they solved in the previous lessons; the repetition was helpful because it summarized the topic and tied up loose ends.

The learning difficulties might have been avoided if each event-action pair had been written and executed as a separate program before integrating them into a single concurrent algorithm.

3.9. The fourth questionnaire

The fourth questionnaire (administrated in lessons 17 and 18) contained questions on the line following algorithm and the bottom sensors. The success rates are shown in Table 10.

Table 10. Fourth questionnaire, % of students who gave correct answers for each question

Question no.	Success Rates
1	**51**
2	48
3	33
4	43
5	50
6	38

The success rates are significantly lower than the success rates of the the other questionnaires. The questions asked about advanced concepts: concurrent execution of event action pairs, implementing a relatively complex algorithm, and using the bottom sensors, which were confusing because of their similarities and differences with the front and back sensors.

In this questionnaire, there were no differences in the success rates on all question among the four classes. Therefore, the null hypothesis is rejected and we could accept the alternate hypothesis that the classes were similar.

3.10. The advanced mode lessons

The students were enthusiastic about learning and understanding the new blocks in the advanced mode and they were curious regarding the new possibilities that the blocks provided.

During lesson 16 the students studied the accelerometer events. In order to explain the functionality of the accelerometers, the teacher showed them a short video that illustrated how the robot was able to maintain its balance on a moving ball. The initial lesson on the accelerometers took place in their normal classroom, not in the computer lab. During the lesson they learned about left/right tilt and forward/backward tilt, and explored how to detection falling by using the accelerometers. Moreover, they explored how different angles can be identified by the accelerometers. The students loved to guess the number of the angles that the robot can detect. The robot was programmed to display a different color for color for each angle and the students grasped the idea rapidly.

In lesson 17 the students practiced exercises with the accelerometers. One problem we encountered was that in advanced mode the events are associated with states, a topic that was not taught, and this caused them to ask a lot of questions and to make mistakes. They solved the exercises quite easily and understood the functionality of the accelerometers.

During the lesson 18, the students learned about the timer event and the timer action. They found the concept exciting and learned how to use the timer to cause the robot to change its color and the direction of its motion after a period of time.

During the next two lessons, the students practiced with these blocks. They encountered some problems and were a little confused on how to write programs that used the timer event and action. It was unnatural for them to put the blocks in different event-action pairs and they kept asking where to put the timer blocks.

During the last class they received an exercise that summarized the entire content of the syllabus. The exercise was not easy for the students and they didn't always volunteer to come to the board to help find a solution.

3.11. The teachers' focus group

The principal and the four teachers of the classes were very receptive to our suggested activities, because the school had been established only a few years previously and the staff was open to new initiatives. The teachers were cooperative and helped us understand the cognitive abilities and the affective aspects of the students. The focus group was held after the classes ended, so that the teachers would have a perspective on the entire course.

The first question was: What do they think about the content of the syllabus? The teachers agreed that the students understood the content of the lessons; they succeeded in their tasks and were able to answer the questionnaires. Moreover, they mentioned that the subject was "fun and cool for the students, and helped the students to bond and to perform the tasks." The teachers believed that the level of understanding students increased throughout the course and that they fully cooperated with us. As in every class, the students showed different levels of understanding, but they helped each other overcome the obstacles they encountered.

The second question was: How did the students profit from the course beyond learning the CS concepts? The teachers indicated that the students cooperated while working in groups. The students had to know how to share their work. The teachers indicated that while most of students were enthusiastic about the class, there were some who were not so excited. Unexpectedly, the teachers noted that most of the students who didn’t like the robotics class were boys, while the girls were more enthusiastic and curious.

The teachers added that the classes helped the students become more motivated about CS and robotics. The opportunity to work by trial and error helped them overcome difficulties and study the subject in a "fun way."

3.12. The project lesson

During the project lesson the students had to come up with their own ideas for programs they wanted to create. The first step was for them to think about the purpose of the program they wanted to create. The second step was to specify the program's steps, which required the students to analyze the purpose of the program and then to decompose it small steps of event-action pairs. We provided a worksheet with several ideas; however, they did not use these ideas and insisted on creating something of their own. The problem was that they wanted to create a program from scratch, but they didn't look at it as an opportunity to make a program with a purpose; instead they looked at it as a just fun activity. The result was that they created programs that showcased the VPL constructs, for example by using a large variety of blocks, but that had no purpose.

Most of the groups managed to create event-action pairs and to explain what the program performs when asked, but some used the advanced blocks incorrectly. For example, they used the states incorrectly, or used the timer event and action in the same pair, which is not meaningful. Most of the students included in their programs "cool" blocks (in their words), such as music, tapping and clapping. They tended to use buttons more than sensors, which are more fundamental in robotics. Most students managed to create a program that functioned correctly.

4 Discussion

The research goal was to characterize the learning outcomes of young second-grade students who took part in a CS through robotics course. We first discuss the findings from the observations and then discuss the achievement of the students in terms of the new taxonomy. Finally, we discuss the three conjectures and present the students' capabilities as measured by the questionnaires.

4.1. Observations

After the 21 lessons that included both frontal instruction and computer labs, the students appeared to like their experience with robotics. Most of the students seemed engaged throughout the lessons and were motivated to succeed in their assignments. They willingly participated in the class demonstrations and worked on the lab assignments, creating meaningful programs.

The robot was perceived by the students as an integral part of the learning, making their first experience with CS a positive one. The robot made CS more tangible, allowing the students to have hands-on interaction with the abstract concepts that they learned. The students tended to imagine that the robot had human qualities. They bonded with the robot, which helped them to overcome difficulties and to be even more engaged during the lessons. In particular, the Braitenberg creatures formed a bridge between the abstract and the tangible, allowing the students to implement complex event handling even with multiple actions.

4.2 Learning of CS concepts

The students had difficulties with the transitions between one event-action pair and several event-action pairs, and between one action per event and multiple actions per event. These difficulties were resolved up with more practice. As they learned more blocks, confusion arose as to which block should be used for which purpose. Furthermore, there was confusion between the bottom sensors and the horizontal sensors, which impaired the students' ability to understand the line following algorithm.

A partial explanation of these difficulties could be that they arose from the sharing of a robot by groups of three students, which resulted in friction within the groups. The students were not allowed to take the robots home, so they were not able to practice creating and executing programs on their own. The latest version of the VPL environment can be run in a software-only simulation, so that programming can be practiced even in the absence of a physical robot.

When compared with previous work such as Bers et al. (2014), which only checked the success of the assignments that the students worked on in class, our work investigated their ability to go beyond what was presented within the framework of the class in order to demarcate what the students of this age can

learn.

4.2. The taxonomy

Question 1 showed that the students are capable of correctly predicting the output of a given program. In all of the questionnaires most of the students answered this question correctly, in particular, the success rates were very high in the first and third questionnaires. This shows that even young students are able to analyze programs.

Question 2, which required the students to choose the program that gives rise to a given output, showed that the students are also capable of identifying a program that can give rise to a given output. Most of the students correctly answered this question in questionnaires 1 and 3, but had difficulties with questionnaire 2. This was probably due to premature testing of the students' capabilities, before they had enough experience with the blocks and with multiple event-action pairs. The low success rates for questionnaire 4 will be discussed in section 4.3.

The students encountered difficulties coping with questions of type 3, which required the students to watch two videos and to choose the statement that correctly described the difference between them. The low success rates were significant for questionnaires 1, 2 and 4. They found it difficult to identify behaviors shown in the videos and memorize them in order to answer the question. It is possible that if the two programs were given in addition to the two videos, the students would not have had to memorize the content of the videos and their success rates would have been higher.

The success rates of the students for questions of type 4 were high in both the first and second questionnaires, but were low in the third and the fourth questionnaires. This question required the students to complete a partial program in order to achieve a given output; this is the first type of question in the producing category. The results indicated that the students knew how to use the different blocks and they understood the functionality of the blocks, in addition to being able to analyze the given code, understand the functionality of the missing parts and how to use them to complete a program.

The success rates of the students were high for questions of type 5 except in the fourth questionnaire. This result is interesting because it means that the students were capable of using the knowledge they gained during the lessons in order to build new knowledge. This is consistent with near transfer of knowledge (Gick & Holyoak, 1980).

Question 6, which required the students to create a program from scratch when the goal of the program is given, showed low success rates in all of the questionnaires. While the students were able to write programs during class using trial and error, and with the help of the teaching staff, they found it difficult to write programs outside the context of the robot and the VPL environment. (Recall that the questionnaires were administered offline.) Moreover, when asking the students to build a program of their own during the project lesson, the students were not able to give a purpose to the programs; instead, they just paired events and actions.

4.3. Interpreting the results in terms of the conjectures

The students' success in the class assignments where they wrote programs for the robot, together with the results of the first, second and third questionnaires, showed that the students are reaching relatively high levels of the taxonomy. This provides evidence that the students are capable of understanding and execute simple programs. In addition, the teaching staff observed the success of the students on the assignments. The students showed proficiency in using the relatively advanced construct of multiple actions per event. Given that event handling is considered to be a relatively advanced core concept of computer science, we conclude that the students were capable of understanding CS constructs.

On the other hand, when the students were asked about these concepts in the questionnaires or the worksheets (in particular in the fourth questionnaire), they encountered many difficulties and struggled to answer the questions and to describe the goal of a program. These difficulties also appeared during the project phase. The contrast between the performance in class and the difficulties with the questionnaires leads me to conclude that the students demonstrated the Jourdain effect: performance

does not necessarily imply understanding.

The inability of the students to specify and implement their own projects showed that while the students were able to understand constructs, they were not capable of creating plans as defined by Soloway and Sphorer (1989).

5 Conclusions

Robotics activities enable even young students to learn basic CS concepts and they are capable of writing and running programs. Students performed well in answering questions on basic programming constructs and the VPL environment enabled the students to create programs graphically, thus overcoming the linguistic barriers to programming. Robotics activities can be successfully used with very young students to increase their interest and possibly motivation to become engaged with STEM in general and CS in particular.

However, our results showed that young students find it difficult to go from learning concepts and individual programming constructs to being able to create programs of more than a few lines. Another important conclusion is that students only functioned well when using the physical robot and with the help of the teaching staff.

Further research is needed in order to map CS concepts to the cognitive capabilities of students at various ages, in order to guide age-appropriate curriculum development for elementary schools.

Acknowledgements

We would like to thank Stella Khazina for assisting in the classroom throughout the entire course, and the teachers and principal of the school where the research was carried out.

References

Anderson, L. W., Krathwohl, D. R., Airasian, P., Cruikshank, K., Mayer, R., Pintrich, P., ... & Wittrock, M. (2001). *A taxonomy for learning, teaching and assessing: A revision of Bloom's taxonomy*. New York. Longman Publishing.

Artz, A. F., & Armour-Thomas, E.(1992). Development of a cognitive-metacognitive framework for protocol analysis of mathematical problem solving in small groups. *Cognition and Instruction*, 9(2), 137-175.

Anderson, M., McKenzie, A., Wellman, B., Brown, M., & Vrbsky, S. (2011). Affecting attitudes in first-year computer science using syntax-free robotics programming. *ACM Inroads*, 2(3), 51-57.

Armoni, M. (2012). Teaching CS in kindergarten: How early can the pipeline begin? *ACM Inroads*, 3(4), 18-19.

Barker, B. S., & Ansorge, J. (2007). Robotics as means to increase achievement scores in an informal learning environment. *Journal of Research on Technology in Education*, 39(3), 229-243.

Ben-Ari, M. (2013). *First Steps in Robotics with the Thymio Robot and the Aseba/VPL Environment*. https://aseba.wdfiles.com/local--files/en:visualprogramming/thymio-vpl-tutorial-en.pdf (last accessed 11 January 2018).

Ben-Bassat Levy, R., & Ben-Ari, M. (2015). Robotics Activities: Is the Investment Worthwhile? In *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives*, Ljubljana, Slovenia. (pp. 22-31).

Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education* 72, (pp. 145-157).

Biggs, J. B., & Collis, K. F. (2014). *Evaluating the quality of learning: The SOLO taxonomy (Structure of the Observed Learning Outcome)*. Academic Press.

Bloom, B. S., Engelhart, M. D., Furst, E. J., Hill, W. H., & Krathwohl, D. R. (1956). *Taxonomy of educational objectives, handbook I: The cognitive domain* (Vol. 19). New York: David McKay.

Braitenberg, V. (1984). *Vehicles: Experiments in synthetic psychology*. Cambridge, MA: MIT Press.

Bruce, K. B., Danyluk, A. P., & Murtagh, T. P. (2006). *Java: An Eventful Approach*. Pearson.

Brousseau, G. (2006). *Theory of didactical situations in mathematics: Didactique des mathématiques, 1970–1990 (Vol. 19)*. Springer.

Clements, D. 2002. Computers in early childhood mathematics. *Contemporary Issues in Early Childhood*, 3(2), 160-181. Available from:

http://www-tc.pbskids.org/read/brochure/powerpoint/Clements_Computers_Math.pdf. Accessed October 25, 2017.

Clements, D. H., & Sarama, J. (1997). Research on LOGO: A decade of progress. *Computers in the Schools*, 14(1-2), 9-46.

Clements, D., & Sarama, J. 2003. Strip mining for gold: Research and policy in educational technology—a response to "Fool's Gold". *Educational Technology Review*, 11(1), 7-69.

Druin, A., & Hendler, J. A. (Eds.). (2000). *Robots for kids: Exploring new technologies for learning*. Morgan Kaufmann.

Duncan, C., & Bell, T. (2015). A pilot computer science and programming course for primary school students. In *Proceedings of the 10th Workshop in Primary and Secondary Computing Education*, London, UK (39-48).

Duncan, C., Bell, T., & Tanimoto, S. (2014). Should your 8-year-old learn coding?. In *Proceedings of the 9th Workshop in Primary and Secondary Computing Education*, Berlin, Germany (60-69).

Fagin, B., & Merkle, L. (2003). Measuring the effectiveness of robots in teaching computer science. In *ACM SIGCSE Bulletin* 35(1), 307-311).

Fuller, U., Johnson, C. G., Ahoniemi, T., Cukierman, D., Hernán-Losada, I., Jackova, J., ... & Thompson, E. (2007). Developing a computer science-specific learning taxonomy. In *ACM SIGCSE Bulletin* 39(4), 152-170.

Gick, M. L., & Holyoak, K. J. (1980). Analogical problem solving. *Cognitive Psychology* 12(3), 306-355.

Glaser, B. (2017). *Discovery of grounded theory: Strategies for qualitative research*. Routledge.

Guzdial M., (2015) *Learner-Centered Design of Computing Education: Research on Computing for Everyone*, San Rafael, CA: Morgan & Claypool.

Hogg, D. W., Martin, F., & Resnick, M. (1991). *Braitenberg creatures. Cambridge: Epistemology and Learning Group*, MIT Media Laboratory.

Liao, Y. C., & Bright, G. W. 1991. Effects of computer programming on cognitive outcomes: a meta-analysis. *Journal of Educational Computing Research*, 7(3), 251-266.

Lister, R., Adams, E. S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., ... & Simon, B. (2004). A multi-national study of reading and tracing skills in novice programmers. In *ACM SIGCSE Bulletin* 36(4), 119-150.

Kaloti-Hallak, F., Armoni, M., & Ben-Ari, M. (2015). Students' attitudes and motivation during robotics activities. In *Proceedings of the 10th Workshop in Primary and Secondary Computing Education*, London, UK, 102-110.

Kay, J. S. (2011). Contextualized approaches to introductory computer science: the key to making computer science relevant or simply bait and switch? In *Proceedings of the 42nd ACM technical symposium on Computer science education*, Dallas, TX, 177-182.

Magnenat, S., Riedo, F., Bonani, M., & Mondada, F. (2012). A programming workshop using the robot "Thymio II": The effect on the understanding by children. In *Advanced Robotics and its Social Impact*, Munich, Germany, 24-29.

Magnenat, S., Shin, J., Riedo, F., Siegwart, R., & Ben-Ari, M. (2014). Teaching a core CS concept through robotics. In *Proceedings of the 19th Conference on Innovation & Technology in Computer Science Education*, Uppsala, Sweden, 315-320.

Markham, S. A., & King, K. N. (2010). Using personal robots in CS1: experiences, outcomes, and attitudinal influences. In *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education*, Bilkent, Turkey, 204-208.

Martinez, C., Gomez, M. J., & Benotti, L. (2015). A comparison of preschool and elementary school children learning computer science concepts through a multilanguage robot programming platform. In *Proceedings of the 15th ACM Conference on Innovation and Technology in Computer Science Education*, Vilnius, Lithuania, (pp. 159-164).

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with Scratch. *Computer Science Education*, 23(3), 239-264.

Papert, S. 1980. Mindstorms: *Children, Computers, and Powerful Ideas, 2nd ed*. New York: Basic Books.

Pattis, R. E. (1981). *Karel the robot: A gentle introduction to the art of programming*. John Wiley & Sons.

Seiter, L., & Foreman, B. (2013). Modeling the learning progressions of computational thinking of primary grade students. In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research*, 59-66.

J. Shin, R. Siegwart, and S. Magnenat. Visual Programming Language for Thymio II Robot. *Interaction Design and Children* (IDC), 2014.

Sierpinska, A. (2003). *Lectures on the Theory of Didactic Situations in Mathematics, Lecture 4*. http://annasierpinska.rowebca.org/pdf/TDSLecture%204.pdf (last accessed 11 January 2018).

Spohrer, J.C. & Soloway, E. (1986). Novice mistakes: Are the folk wisdoms correct?. *Communincations ACM* 29(7), 624-632.

Sullivan, A., & Bers, M. U. (2016). Robotics in the early childhood classroom: learning outcomes from an 8-week robotics curriculum in pre-kindergarten through second grade. *International Journal of Technology and Design Education*, 26(1), 3-20.

Tedre, M. & Denning, P.J. (2016). The long quest for computational thinking. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, 120-129.

Wyeth, P. (2008). How young children learn to program with sensor, action, and logic blocks. *The Journal of the Learning Sciences*, 17(4), 517-550.