Cultivating students' computational thinking through student–robot interactions in robotics education

Jing Ru Qu¹ · Ping Kwan Fok¹

Accepted: 4 June 2021 / Published online: 11 June 2021
© The Author(s) 2021

Abstract

This research focuses on student–robot interaction in the learning environment of robotics education (RE) and attempts to explore how it cultivates students' computational thinking (CT). Different from child–robot interactions as investigated in the social robot field, student–robot (S–R) interactions focus mainly on the process of interaction between learners and programmable robot kits in RE settings. At a four-week robotics summer camp in China, mixed-methods research was conducted. Forty primary school students and one dedicated robotics teacher participated in this research, while 32 students and the teacher completed all the lessons and data collection procedures of the summer camp. Results indicated that students' CT skill increased during the summer camp and that the change in their CT skill was positively correlated to the time spent on S–R interaction. Additionally, how three kinds of S–R interaction—programming-computing, observational investigation, and participatory investigation—cultivated students' CT were found. Moreover, the hierarchy of three S–R interactions and students' role-shifting in the hierarchy were discussed. Previous studies rarely discussed S–R interaction; however, this kind of interaction should be explored because it provides more information about students' natural learning process, which might be meaningful to RE practice.

Keywords Robotics education · Computational thinking · Student–robot interactions · China · Summer camp

Introduction

With the rapid development of science and technology, robotics has become increasingly attainable and visible in our daily lives and even in the field of education (Benitti, 2012; Toh et al., 2016). Children are inevitably exposed to an environment that features numerous robotized devices. Many robotics curricula and activities within and outside of schools have appeared in China in recent years, becoming popular choices among students. Meanwhile, the application of educational robotics to the cultivation of students' attitude toward

✉ Jing Ru Qu
qujingruru@gmail.com

¹ Department of Curriculum and Instruction, The Education University of Hong Kong, Hong Kong, China

STEM (Üçgül & Altıok, 2021), problem-solving ability (Atmatzidou et al., 2018; Barak & Assal, 2018; Barak & Zadok, 2009), spatial ability (Julià & Antolí, 2016), and twenty-first century competencies (Eguchi, 2013) have gained popularity among educational scholars. Several previous studies show abundant evidence regarding students' improvement in computational thinking in robotics education (Atmatzidou & Demetriadis, 2016; Bers, 2010; Bers et al., 2014; Chalmers, 2018; Chen et al., 2017; Eguchi, 2014; Fanchamps et al., 2021; Ioannou & Makridou, 2018; Kim & Lee, 2019; I. Lee et al., 2011; Leonard et al., 2016; Noh & Lee, 2020; Yang et al., 2020). However, few of them have explained *how* students' learning was developed, and they have often fallen into a *technological-determinist paradigm* which "attributed the main cause of learning outcomes to the robotics technologies" (Jung & Won, 2018, p. 13).

With the continuous deepening of research on technology and design education, how to prevent our research from falling into the paradigm of technological determinism and how to truly help students develop their cognitive abilities from their perspectives should receive more discussions. The profound contributions of this study are not limited to interpreting the interaction between students and robots, nor examining the development of students' computational thinking in a robotics learning environment; it also contributes a perspective to technology and design education field, that is, to be aware of students' role played in a technology-enhanced environment, and by doing so, our researchers and educators can shift the focus from the technology itself to the students and their growth.

This research focuses on student–robot (S–R) interaction in the learning environment of robotics education (RE) and attempts to explore how they cultivate students' computational thinking (CT). To do so, mixed-methods research was conducted at a four-week RE summer camp in China, adopting the methods of rubric scoring, classroom observation, and semi-structured interviews. A total of 32 primary school students (i.e., 10 girls and 22 boys) and one dedicated robotics teacher completed all the lessons and data collection procedures in the summer camp.

Literature review

Robotics education (RE)

Although it is important to define RE explicitly, most previous studies have not done so (Jung & Won, 2018). Before defining RE, it is necessary to clarify the position and view of robotics in the current study, as previous studies on RE have "established different educational purposes for employing robotics" (Jung & Won, 2018, p. 4). As summarized by Jung and Won (2018), there are two main established perspectives in positioning robotics within the educational context. The first perspective posits robotics as an effective tool for teaching other subjects, such as mathematics, science, or physics, which should be classified into corresponding subject areas (such as mathematics education, science education, or physics education). The second perspective "view[s] robotics as a tool to teach robotics itself" (p. 5), and this is what RE is talking about. In this context, Eguchi (2013) summarized previous studies in terms of three trends. The first trend posited "robotics as [a] learning objective" (p. 3), which aligns with the second perspective of Jung and Won (2018). The second trend views robotics as an aid to teaching or helping students with special needs (Chang et al., 2010). The third trend regards robotics as a tool for enhancing students' learning in

different subjects (Toh et al., 2016), which echoes the first perspective of Jung and Won (2018).

This study lies in the second perspective of Jung and Won (2018) and positions itself in alignment with the first trend of Eguchi (2013); thus, it defines **Robotics Education (RE)** as: *Using robot kits to form an environment for teaching robotics knowledge and nurturing learners' competencies/skills* (Benitti, 2012; Eguchi, 2012; Jung & Won, 2018). More specifically, this study regards robot kits as a tool in RE, the learning content of RE as robotics-intensified knowledge; and the purpose of RE as nurturing learners' competencies/skills. Additionally, this study aligns with the view of Fortunati et al. (2020), which believes that, comparing to using an already made commercial robot, building a robot with students is better for their cognitive thinking development. Therefore, the "robot kits" herein refers to a set of elements provided for students to build their own innovative robot.

Student-robot (S-R) interactions in RE

Most of the previous studies on RE classroom interactions have probed the interactions between teachers and students (Liu et al., 2013) or among peers (Rowell, 2002; Yuen et al., 2014; Zhong & Wang, 2021). In contrast, in the current study, conducted in a robotics learning environment, robots (learning kits) can be deemed to be a crucial component of interaction for students. As stated by Shin and Kim (2007), younger children tend to interact with robots as their peers. Also, through analyzing participants' explanations on robots' actions, Levy and Mioduser (2008) found that, young children tend to describe a robot's movements from a psychological perspective, such as using the words like "*it wants to...*", especially when the tasks become more difficult. Inspired by this finding, the researchers of this study believe that, it is time for us to switch from the perspective of adults to the perspective of children to investigate classroom interactions in RE. As a start, S-R interaction, the unique and crucial facet of classroom interactions in RE, should be concerned.

S-R interaction in RE is an under-researched point, and previous literature calls for attention to be paid to this interaction (such as Jung & Won, 2018). In some traditional subjects, such as mathematics and physics, students have few opportunities to directly interact with an object. In contrast, students who learn robotics rely on "talking" to the robot (i.e., inputting design ideas to build and program a robot) and "listening" to the robot's responses (i.e., observing the robot's outputs and reactions). Afterward, they will decide how to debug their programs and modify their ideas about building or programming the robot. As a result, students learn robotics from such student-robot interactions. Therefore, Jung and Won (2018) suggested that future RE studies should leverage learners' voices in order to probe "their dynamic and complicated interactions with robotics kits" (p. 15).

What should be kept in mind here is that student-robot interaction in the RE field are a different concept from child-robot interactions in the social robot field. Child-robot interactions focus on interactive behaviors between humanoid social robots (such as tutor robots) and users (children) in many kinds of learning settings (e.g., language learning, special education needs, etc.) (Belpaeme et al., 2013), while student-robot interactions focus mainly on the process between learners and programmable robot kits in RE settings.

Limited studies have probed student-robot interactions in RE. For example, in Levy and Mioduser's (2010) observation, *Participatory Investigations* were found to be one manner in which students interacted bodily with the robot; "in such interactions, the child's role shifts from designer and observer to that of participant" (p. 28). Nevertheless, Levy's study did not probe other possible interactions between students and robot. In Bakala

et al. (2019), a learning environment designed for child-robot interaction and promoting CT is described, however, the influence mechanism from child-robot interactions to CT is under-discussed.

In addition, most of the previous studies have focused only on one-way behaviors of students handling robots. For example, in Yuen et al. (2014), when describing students' tasks in robotics projects, the researchers determined that there are seven kinds of tasks students should carry out with robot kits: "building, programming, testing, debugging, observing, planning, discussing" (p. 41). They found that building and observing were the two most favored parts among students, while programming was students' least favorite part. Their study described the process of how a student made a robot, which was simply a one-way process of *doing*. However, without information about how students interacted with the robot, for instance how they dealt with the unexpected actions of a programmed robot and how they fixed bugs, we can hardly know why students are often not positively disposed toward the programming part—which is a significant concern for the cultivation of future STEM talents.

Computational thinking (CT) in RE

Inspired by Wing's (2006) call for research on CT, many RE studies have attempted to probe students' CT in RE settings in recent years. In the current study, **computational thinking (CT)** refers to "a problem-solving process that includes formulating problems; logical organization of analysis of data; representation of data through abstractions; identifying and automating solutions through algorithmic thinking; analysing and implementing possible solutions; and generalizing and transferring the problem-solving process" (Leonard et al., 2016, p. 868). This definition stems from the operational definition of CT proposed by the International Society for Technology in Education (ISTE), which not only focus on the proficiency of programming and computing concepts but also on general problem-solving skills (Tang et al., 2020).

Lee et al. (2011) described how youth perform CT in RE. Three phases of CT were introduced, namely, abstraction, automation, and analysis. In the context of the RE environment, abstraction occurs when students develop robots with constrained reactions to conditions that could occur in reality. During the abstraction phase, students should brainstorm to consider different ways to make their robots interact with the real world; they should also consider abstracting the inputs and outputs of their program. In the automation phase, the processing unit executes the program compiled by the student. Following this, the students pay attention to investigating the behavior of the robot in response to different reactions and debug the program to locate the bug and, then, return to the abstraction phase.

A robotics learning program was introduced by Bers (2010) and Bers et al. (2014), known as TangibleK and designed to involve kindergarten students in CT. The study considered four variables to understand how students approach CT, namely, debugging, correspondence, sequencing, and control flow. With regards to debugging, the performance of students in four steps of the debugging process was rated using a 5-point Likert scale; debugging skills reflected understanding ranging from partial to good, and when debugging skills were applied, the mean score was above three points. The studies based on TangibleK have provided valuable insight into CT in RE; however, these studies were limited in one of the most important aspects, that is, abstraction was not considered in these studies

(Wing, 2006, 2010). As a result, the framework of these studies is mostly based on the skill of programming design instead of on CT skill.

RoboCupJunior is a globally popular robotics competition aimed at students in primary and middle school. Eguchi (2014) conducted a survey based on RoboCupJunior that revealed that a majority of participants (in excess of 60%) had learned CT prior to participating in the competition. More specifically, across eight domains of CT, 100% of participants thought they had learned debugging, 93% that they had learned problem-solving, 79% that they had learned how to break a problem into subproblems, 79% that they had learned logical thinking, 79% that they had learned analysis skills, 71% that they had learned creation of a step-by-step procedure, 69% that they had learned critical thinking skills, and 64% that they had learned prototyping. The study thereby introduced a CT model with eight domains, but its main limitation was failing to demonstrate the theoretical foundation behind the introduced model.

Atmatzidou and Demetriadis's (2016) study involved 164 students from vocational junior high and high schools. The CT skill of students in RE was measured across five domains, namely, abstraction, generalization, algorithm, modularity, and decomposition. The experimental environment of the study was an after-school environment consisting of 11 two-hour robotics learning sessions. From the pre-post results, significantly positive shifts ($p < 0.001$) in students' CT skill were found, with $t(163) = -5.27$.

The CT skills of 37 primary school students was analyzed by Chen et al. (2017). The measurements were done based on pre-/post-tests. The assessment included two different contexts, namely, everyday scenarios and robotics programming. When these two contexts were compared, it was revealed that students' learning outcomes were diminished in everyday scenarios. With regards to the robotics programming context, the researchers introduced two forms of programming language: a text-based form (similar to a professional programming language) and a drag-drop form (a graphic language, similar to Scratch). When CT skill was compared across these two forms, no significant difference was found. The SDARE model was the adopted CT framework; it includes five CT components, namely, syntax, data, algorithms, representing, and "efficient and effective." A significant increase in the CT in two observed classes (Class 1: $t = -3.14$, $p = 0.002$; Class 2: $t = -3.87$, $p = 0.001$) was observed in the overall pre-post results; however, the analysis of the pencil-paper test results indicated that students encountered difficulties following a given programming syntax, which can be attributed to the fact that students mostly use graphical interfaces, which do not employ a specific syntax. This finding is useful for future RE activities, to demonstrate the importance of syntax to students. The main limitation was that it failed to clearly indicate the manner in which CT skills were developed.

Chalmers (2018) investigated four primary school teachers' experiences of using WeDo 2.0 robot kit for developing students' CT concepts and skills, and a mutually beneficial situation was found. On the one hand, for the students, robot-based learning is helpful for their computational thinking development; and on the other hand, teachers' confidence in teaching STEM knowledges has also strengthened.

Experimental analysis of the CT skills of students in RE have been conducted in a Chinese context. For instance, Li et al. (2019) interviewed 26 primary school RE teachers in Wuhan, China. Four stages of teaching of CT for primary school students were demonstrated, namely stimulating motivation, pattern construction, implementing creative ideas, and communication and reflection. However, there was a major shortcoming: the study failed to provide an explanation of how the CT skills of students were cultivated at each stage. Moreover, a number of other papers have also contributed to the research on the CT skills of students in a Chinese context. However, a majority of these studies have been

presented in the form of reflective reports based only on the individual teaching experiences of the authors, that is, they lack a scientific research design.

Research gaps

After review of relevant studies on RE, we noticed some research gaps that needed to be filled. First, studies on S–R interaction are still sparse, but S–R interaction is an important aspect of classroom interactions in RE. Second, studies on students' CT development in RE are abundant, but they rarely explain students' CT from the view of classroom interaction or even S–R interaction. This research aims to fill these gaps.

Conceptual framework

As reviewed previously, student–robot interaction has rarely been discussed in RE studies, and needs more attention (Jung & Won, 2018). To begin understanding the patterns of S–R interaction, we should first know how students communicate with their robots. In this communicative exchange, first, a student need to “talk” to a robot, and the robot then “listens” to and transfers the student's orders to its own language. In other words, this process involves students programming (i.e., talking and giving instructions to) the robot, and the robot computing (i.e., listening to and transferring students' orders to its own language). In other words, this is a *programming-computing* interaction. Second, the robot “responds” to the student's orders by moving and behaving, and the student observes its movements and behaviors; that is, so *observational investigation* occurs. Thirdly, since some robots need to detect and react to the outside situations (such as obstacles), students physically interact with these robot to investigate their behaviors and reactions, in *participatory investigation* (Levy & Mioduser, 2010). To sum up, three key concepts are included in the S–R interaction: programming-computing, observational investigation, and participatory investigation.

Additionally, this study adopts the CT framework proposed by the ISTE (2011), because of its features of operability, widely-cited, and not limited in domain specific knowledge (Tang et al., 2020). ISTE's CT framework contains six key concepts: formulating problems, abstraction, logical thinking, using algorithms, analyzing and implementing solutions, and generalizing and problem transfer. *Formulating problems* necessitates that the students recognize problems and formulate them in such a way that computers and other tools (such as Intelligent Bricks) can be used to solve them (ISTE, 2011). Along the way, this process tests the ability of students to break down general problems into subproblems (Atmatzidou & Demetriadis, 2016). The primary objective of formulating problems is to make them solvable. *Abstraction* refers to the use of abstract methods such as models and simulations to represent data (ISTE 2011). During the design of robots, students use abstract thinking to simulate the reaction of the robots to given conditions (Lee et al., 2011). An indicator that students have grasped this concept well is that students start to use programming concepts to describe different scenarios. *Logical thinking* refers to the skill of logically organizing and analyzing data (ISTE 2011). Students who tend to think logically are more likely to organize their programs logically, for instance expressing their thoughts by using terms like “because...so...” or “if...then...”. *Using algorithms* assesses the ability of students to follow the logic of the algorithms and use proper commands (ISTE 2011). A good indicator that students have gained this skill is when they use proper programming logics, such as conditional logic, iterative logic, or parallel logic, to order their robots to complete an allocated task. *Analyzing and implementing solutions* refers to the process where students

automate solutions and evaluate their effectiveness (ISTE 2011). This skill can be regarded as similar to debugging skills (Leonard et al., 2016), as debugging is the process of locating the faults in the program (bugs), which are intrinsically related to effectiveness. Finally, *generalizing and problem transfer* is the skill of generalizing or transferring the process used to solve specific problems to a wider range of problems (ISTE 2011). A highly computational thinker is able to encapsulate the problem-solving process and extrapolate solutions to cover similar problems or different problems, as an alternative to solving individual problems in a one-off manner. When students solve different problems using solutions implemented before, it indicates that students have developed this skill.

The current study

Research purpose and questions

This research aims to explore how S–R interactions cultivate students' CT and to answer the following research questions. (1) If any, what is the change in students' CT skills during the course of the robotics summer camp? (2) What is the relationship between S–R interactions and change in students' CT skills? (3) Can S–R interaction cultivate students' CT skills, and if yes, how?

Research design

A mixed methods design was applied in the current study to investigate research questions as stated before. According to Creswell (2014), this study was guided by an explanatory sequential design model, complying with the procedure that “the researcher first conducts quantitative research, analyzes the results and then builds on the results to explain them in more detail with qualitative research” (p. 55). Specifically, this study firstly examined the change in students' CT skills and the relationship between this change and S–T interactions, then explained above results through their learning process from qualitative data. Importantly, the adoption of a mixed-methods strategy should be justified (Creswell, 2014); in this research, while quantitative data provide an insight into the general status of the CT skills of students, qualitative data assumes a much more important role in this study, enhancing the conclusions drawn through the analysis of quantitative data.

The robotics summer camp

This study was carried out at a robotics summer camp in Liuzhou, China. Robotics summer camp has been used widely as an extracurricular activity on previous occasions when studying RE (see, e.g., Larkins et al., 2013; Lee et al., 2013; Yuen et al., 2014). Throughout the four weeks of the summer camp, a total of 12 lessons along with 11 problem-solving tasks were given (the first lesson was an introduction to the robotics hardware and software), thrice weekly. Specifically, students were required to design and create a robot to accomplish the given tasks (see “Appendix 1”), such as making a vehicle which can move forward, turn around, following the blackline; making an off-road car, electric fan, and toy gyro; making a “walking” robot with two legs; making a football robot. In the process of designing and creating, students had to solve various problems of building and programming. Each lesson lasted for 90 min, among which

there were about 70 min for students to accomplish the lesson tasks, 10 min for the teacher to describe the lesson task and introduce the required knowledge and concepts for completing the task, and 10 min for students to display and reflect on their works.

KAZI EV5 is a programmable robot kit—a product of the KAZI Robotics Education Corporation (Shenzhen, China). KAZI EV5 was used in the summer camp organized in the present study. The most desirable feature of KAZI EV5 is that its functionality and quality are identical to those of Mindstorms Education EV3 set from Lego, yet the KAZI EV5 is significantly inexpensive (roughly \$280 less than the Lego set).

The programming language used in the summer camp was Scratch—a visual programming language developed by the Massachusetts Institute of Technology (MIT). Scratch is a popular visual programming language used in the field of RE, and is preferred by young children. It allows users to create simple programs with very little formal training. In contrast, it could also be used to develop complex sophisticated programs (Olabe et al., 2010, p. 359). Students can drag the graphical and colorful blocks to develop their designs (Bers, 2010; Maloney et al., 2010).

Data collection

Methods

Qualitative and quantitative data were collected, analyzed, and interpreted in this research. Using this approach, the researcher reveals implications across quantitative and qualitative databases (Creswell, 2014). More specifically, rubric scoring (closed-ended) is used in the quantitative method, and semi-structured interviews and classroom observation (open-ended) are used as qualitative methods. The aim is to provide a comprehensive understanding of the learning process of students in RE.

Participants

Forty student participants at a summer camp were randomly recruited on a voluntary basis. Consent was obtained from all students' parents. The students were from various local primary schools and aged seven to nine (i.e., Grades 1–3). Based on the preferences of the participants, they were assigned to four classes, each consisting of 10 participants. All classes were identical in terms of the teacher, venue, curriculum, facilities, and duration. All the participants from the four classes were considered as a single group during the analysis of data. Out of the 40 participants, two dropped out during the summer camp, and six were absent for at least one lesson. At the end of the summer camp, a total of 32 participants—10 females and 22 males—had completed all the lessons and thus had been available for data collection throughout the summer camp.

Besides the participating students, one teacher also participated in the study. This dedicated robotics teacher was responsible for conducting teaching activities at the summer camp. The teacher was a local 25-year-old male with a Bachelor of Education degree and three years of experience teaching robotics to primary school students; he had previous experience using the KAZI robot kits and Scratch in his lessons. This was his second time participating in robotics summer camp as a teacher.

Instrument: computational thinking rubric (CTR)

As illustrated in the “Appendix 2”, the CTR, designed by Leonard et al. (2016), was adopted to collect data pertaining to the CT skills of students. The CTR involves six domains: formulating problems, abstraction, logical thinking, using algorithms, analyzing and implementing solutions, and generalizing and problem transfer. Each component was rated according to one of three levels: emerging (1), moderate (2), substantive (3). The original version of CTR was in English, and both raters can read English. Nonetheless, it was translated into the Chinese version by a professional translator and proofread by the authors of this paper to ensure the consistency of the two versions. Both English and translated versions were provided to the teacher before the summer camp. Then, the researcher, the teacher, and one expert in the RE field discussed the criteria and examples of different levels of CT skills based on both Leonard's and ISTE's definition and examples for each CT skill. Through this process, the content validity of CTR was ensured. In accordance with J. Cohen (1960), inter-rater reliability was measured by Cohen's Kappa test. Each domain of the rubric was tested. Kappa coefficients for the six domains suggested a high level of agreement between the two raters ($0.54 < \kappa < 0.92$, $p < 0.001$).

Procedure

With regards to rubric scoring, prior to the summer camp, the researcher and the class teacher discussed the criteria for scoring. During each class, the CT skills of each student were rated by both researcher and teacher, because having an independent rater (the teacher) aside from the researcher could help reduce potential unconscious grading bias. The assessments were based on classroom programming, debugging performance, and the ability to use Scratch programs. To avoid mutual influences, the two raters scored the students independently.

As regards classroom observation, the researcher attended each class held during the summer camp and took field notes on the observation form mentioned previously. Moreover, each class was recorded using five cameras. Videotaping helps mitigate the partialness of the observer's view of a single event, and is also helpful to overcome the tendency to only record events that occur frequently (L. Cohen et al., 2011). In order to clearly capture the entire scene of the classroom, four fixed cameras were used, fixed on the four corners of the classroom, and a movable camera was held by the researcher to record details. Moreover, four audio-recorders were used to clearly record the verbal interactions of the participants.

Each participant was invited to participate in the semi-structured interviews, which were recorded. The interviews were held in 12 groups each consisting of two or three students. The duration of each interview exceeded 30 min. All the interviews were conducted in the last week of the summer camp; by this time, the students could reflect on their learning experiences. Two interviews were conducted with the teacher, the first on the day of lesson 6, marking the end of the first half of the summer camp, and the second at the end of the summer camp.

Consent letters containing statements about the purposes of the research, data collection procedures, potential risks, confidentiality of data, and other related information were delivered to each participant. These letters were signed by the participants and

their guardians when they were collected. The rights of the participants were protected throughout the research. All collected data remained strictly confidential.

Data analysis

For the quantitative data analysis using SPSS 21, data from the first six days (i.e., the first two weeks) were compared with the data obtained in the last six days (i.e., the last two weeks), by the mean of each six days. The two scores for rating were merged by simply counting the mean and applying the Wilcoxon test. Moreover, the correlation between the change in students' CT scores and the time spent on S–R interactions was tested by Spearman correlation analysis. Furthermore, G*Power was utilized in the calculation of effect size and power, because it is a widely used professional program for power analysis (Erdelder et al., 1998).

Thematic analysis was utilized to analyze the qualitative data collected from video-taped/radio-taped classrooms, observation notes, and interviews. All radio tracks were transcribed verbatim and was used as input to the NVivo software. All the videos were divided into clips (10 min for each). A first-round coding scheme was used in the analysis of the collected qualitative data; supplementary themes that emerged from first-round coding were added to second-round coding.

Findings

What Is the change in students' CT skills during the robotics summer camp?

Based on the Kappa coefficients of the six CT domains, the level of agreement between the two raters ranged from moderate to strong ($0.54 < \kappa < 0.92$, $p < 0.001$). This, it is reasonable to merge the two rating scores. As illustrated in Table 1, the results from the Wilcoxon tests indicated that the CT skills of students (i.e., sum score of six domains) significantly increased from the first six days ($M_{\text{first}}=7.33$, $Mdn_{\text{first}}=7$) to the last six days ($M_{\text{last}}=9.39$, $Mdn_{\text{last}}=8.96$), with $z=-4.94$, $T=528$, and $p<0.001$. More specifically, across all six domains of CTR, there were strongly significant increases. With regards to the effect size

Table 1 Results of wilcoxon test for computational thinking skills (N=32)

	First-six-day			Last-six-day			<i>z</i>	<i>T</i>	<i>p</i>	<i>d</i>	Power
	<i>M</i>	<i>SD</i>	<i>Mdn</i>	<i>M</i>	<i>SD</i>	<i>Mdn</i>					
C1 ^a	1.008	.033	1.000	1.164	.233	1.083	−3.638	153	.000	.692	.997
C2 ^a	1.172	.257	1.167	1.674	.367	1.667	−4.846	495	.000	1.980	1
C3 ^a	1.323	.442	1.167	1.844	.539	1.792	−4.756	464	.000	1.364	1
C4 ^a	1.531	.428	1.458	1.781	.452	1.750	−3.298	348	.001	.679	.996
C5 ^a	1.172	.230	1.083	1.661	.359	1.667	−4.669	459	.000	1.392	1
C6 ^a	1.120	.167	1.000	1.268	.350	1.167	−2.794	247	.005	.552	.963
Total	7.326	1.229	7.000	9.393	1.716	8.958	−4.938	528	.000	2.507	1

^aC1 to C6 refer to the six domains of CTR: C1—Formulating problems, C2—Abstraction, C3—Logical thinking, C4—Using algorithms, C5—Analyzing and implementing solutions, C6—Generalizing and problem transfer

and power of the above results, the effect sizes were at moderate to high levels in all the results, and power was high.

What Is the relationship between S–R interactions and change in students' CT skills?

Each student's time spent on S–R interactions was counted; then, the relationship between the increase in students' CT skills and time spent on S–R interactions was examined by Spearman's correlation analysis. The result indicated a positive correlation, with $rs(30) = 0.51$ and $p = 0.003$.

How do S–R interactions cultivate students' CT?

S–R interactions played an important role in students' abstraction—the foundation and essence of computational thinking (Wing, 2006, 2008)—because the robot is a tangible and visible agent functioning between the real world and the abstracted computing world (Bers et al., 2014). Interacting with a robot helps students abstract the real-world situation to the programming process, which can benefit students' CT. In the quantitative analysis, a significant positive correlation was found between students' time spent on S–R interactions and their CT skill development, but this is insufficient to infer the influence of S–R interactions on students' CT. Therefore, the qualitative data are significant for our understanding of whether S–R interactions can cultivate students' CT and how it happens. By analyzing data from classroom observation (the video-taped classroom) and interviews with students and teacher, we found that three S–R interactions (programming-computing, observational investigation, and participatory investigation) played important roles in cultivating students' CT. How the three kinds of S–R interactions cultivated students' CT will be interpreted below.

Programming-computing

Programming-computing is a mutual process involving students “talking” to the robot by “manipulation of variables and computational instructions” (Bers, 2010, p. 3) and the robots “listening” to and transferring student's orders into its own language (i.e., compiling). This process involves students' CT skills of formulating problems, abstraction, logical thinking, and using algorithms. For example, when programming for a line-following robot, students began by formulating problems that can be solved by programming, such as “How to make the robot detect the blackline?” and “In what condition(s) should the robot turn left/right?” After that, when programming, students represent those problems and relevant data through abstractions and organize them logically. Meanwhile, students select and use suitable algorithms, such as loops, conditionals, operators, and more, in order for robots to complete the given task.

Although all students participated in programming-computing interactions, not all of them made significant progress in CT. Some did learn to critically analyze problems and make decisions carefully that is, achieved progress in CT skills. For example, when Kevin, a student whose CT score increased more than those of most other students, was programming, he browsed the pre-set sample programs in Scratch and then simulated the possible consequences of different programs. Meanwhile, he selected and gathered some scripts suitable for solving the problems, and selected suitable conditional expressions and decided on the proper parameters for setting conditions by using the internal-test model of

Intelligent Brick. In contrast, another student, Terry, whose CT score increased less than most other students, tended to program without critically analyzing problems or making decisions carefully. Often, he started to program by referring to his personal preference and habits. For example, during the first week, he used a linear algorithm that was suitable for the lesson tasks; however, during the second week, he still adopted a linear algorithm, even though it was not suitable for the new lesson tasks. When the teacher asked why he used the linear algorithm again, he said, “Because I got used to programming like this.” In other words, he did not critically analyze problems when the situations changed.

To sum up, programming-computing is a kind of S–R interaction showing potential for cultivating students’ CT skills of formulating problems, abstraction, logical thinking, and using algorithms. However, its potential relies on students to critically analyze problems and make decisions carefully.

Observational investigation

Observational investigation is a process of in which students observe robots’ “responses” (i.e., movements and behaviors). Computational abstraction relies on the observation of outputs (Bers, 2010). In pure programming, the outputs are visible on the computer interface but are not tangible, while in robotics programming, the outputs can be observed from the robot’s behaviors, which are quite tangible; therefore, it is easier for students to observe and analyze them. Observational investigation may facilitate students’ CT skills of abstraction, logical thinking, use of algorithms, and solution implementation and analysis. For example, students often observed the robot’s behaviors and compared them to the expected ones. If the robot moved beyond expectations, students had to logically analyze why the mistakes happened. When debugging the program, they used abstract and algorithmic thinking to simulate the running process of the program step by step.

What should be noted is that observational investigation is more than just watching what happens. For example, some students, although also watching their robots’ performances, usually ignored the robot’s misbehaviors and did not seriously consider unexpected outcomes. As a result, they sometimes did not know how to revise their programming until the teacher or peers pointed out the faulty part; other times, they were distracted by laughing at the robot’s strange behaviors.

Participatory investigation

Just as some robots detect and react to outside situations (such as obstacles), so students usually need to interact with the robot to investigate its behaviors and reactions. As noted, participatory investigation is a kind of direct bodily S–R interaction in which “the child’s role shifts from designer and observer to that of participant” (Levy & Mioduser, 2010, p. 28). This S–R interaction may involve students’ CT skills of abstraction, logical thinking, using algorithms, and solution implementation and analysis. For example, one lesson task was to make a robot that can turn back when it touches a wall. When one student was debugging her robot, she held it in her hands and pressed the touch sensor button to see whether her program was logical, that is, whether the robot would follow the command. She did not put the robot on the ground or allow it to really touch a wall, because she abstracted the condition “when it touches a wall” to “when the touch sensor is pressed” and simulated this condition by using her hands. When making a line-following robot, she followed the robot’s routine with her own footsteps; when the robot got lost, she used her

hands to correct the robot's routine and see in which part the robot would get lost and whether the algorithms she used were proper. When making a football robot, this girl moved the lighting ball (the football) by hand to simulate the ball's movements so that, in programming, she could consider more possible conditions when searching for the football. By participatory investigation, as mentioned above, students debugged their programs so that they became more concretely logical; thereby, their CT skills were developed, as the qualitative data will explore further.

From our observations, almost every student was willing to play bodily with the robot, for example by blocking, changing, or following the robot's path. However, not all of them could effectively gather and investigate information from these playful bodily S-R interactions. Some students played with the robot frequently; however, they simply enjoyed the playing process, even when the lesson task had not yet been completed, and did not engage with, complete, or learn from the task.

In conclusion, S-R interactions, including programming-computing, observational investigation, and participatory investigation, showed potential to cultivate students' CT skills. These kinds of S-R interactions allowed students to think by doing, by bridging abstracted program design and debugging to concrete robot behaviors.

Discussion and conclusion

Overview

Previous studies have rarely discussed S-R interaction, which, however, should be explored because it provides more information about students' natural learning process, which might be meaningful to RE practice (Jung & Won, 2018). Some studies regard robots as playing a passive role in RE and regarded building and programming as unidirectional behaviors (such as Mubin et al., 2013). However, either building or programming rely on obtaining information from the robot's responses, so it is actually a mutual interaction process. S-R interactions played an important role in students' abstraction—the foundation and essence of CT (Wing, 2006, 2008)—because the robot is a tangible and visible agent between the real world and the abstracted computing world (Bers et al., 2014). Interacting with a robot helps students abstract the real-world situation to the programming process, which could benefit students' CT skill development.

This research answered three major research questions. The first was “What is the change in students' CT skills during the robotics summer camp?” The results showed that students' CT skill significantly increased from the first six days to the last six days; for all six domains of CTR, the increases were strongly significant. The second question was “What is the relationship between S-R interactions and change in students' CT skills?” Results indicated that students' time spent on S-R interactions was positively correlated with increase in their CT skills: those who spent more time with their robots made greater progress in CT skills. The third research question, which is the main focus of this study, is “Can S-R interactions cultivate students' CT? If yes, how?” By analysing data from (video-recorded) classroom observation and interviews with students and teacher, we found that three kinds of S-R interactions (programming-computing, observational investigation, and participatory investigation) showed potential to cultivate students' CT. Specifically, programming-computing interaction might cultivate students' CT skills of formulating problems, abstraction, logical thinking, and using algorithms;

however, it also relies on students' non-CT skills of critically analyzing problems and carefully making decisions. Moreover, observational investigation may facilitate students' CT skills of abstraction, logical thinking, using algorithms, and solution implementation and analysis. Finally, participatory investigation may involve students' CT skills of abstraction, logical thinking, using algorithms, and solution implementation and analysis.

Three S–R interactions and three roles of the student

In different kinds of S–R interactions, students' roles were different. In programming-computing interaction, students' role is that of *designer*; in observational investigation, they become *observers*; while in participatory investigation, they become *participants* (Levy & Mioduser, 2010).

Programming-computing interaction has usually been deemed a one-way, inputting behavior (such as in Yuen et al., 2014). However, when programming for a robot, people inevitably consider how to make the robot understand what it should do. If we regard the robot as an inanimate machine executing whatever we tell it to do, we shut down the gateway to communicate with it. Therefore, in this research, we regard the programming-computing interaction as a process of student's "talking" to the robot and the robot's "listening" to and transferring student's orders to its own language (i.e., compiling). In this way, students are designers who accurately formulate problems, interpret problems and data in abstractions, logically think about the problems and solutions, and decide on proper algorithms; otherwise, the robot might be unable to understand what it should do. In the programming-computing interaction, this research found that students' critical analysis of problems and careful decision-making were crucial, and that therefore, RE teachers should guide students to analyze problems more critically and make decisions more carefully when programming to better promote students' CT skills.

Observational investigation is a mutual process in which students observe robots' "responses" (i.e., movements and behaviors). Computational abstraction relies on the observation of outputs (Bers, 2010). In robotics programming, the outputs can be directly observed from the robot's behaviors. When the robot is moving, it tries to tell us what it thought it should do, so we can compare its movements/behaviors to what we expected. In fact, programming is usually not a one-off matter, because what we want the robot to do is often different from what it actually does, and needs to be iteratively amended. Observational investigation then allows us to know where the difference is and to probe what the problem(s) might be. Importantly, observational investigation is more than just watching what happens; instead, in this interaction, students become observers adopting a range of thinking skills, such as abstraction, logical thinking, use of algorithms, and solution implementation and analysis. Importantly, RE teachers should remember that observing is not just watching: properly guiding students to "observe" is necessary.

Participatory investigation is a direct body S–R interaction in which children's roles change from designer, observer to participant (Levy & Mioduser, 2010). Commonly, a robot obtains outside information from its sensors and reacts to that information according to some conditions, so that interacting bodily with the robot means seeing whether it can detect outside situations and react as per the programmer's expectations. In this interaction, students participate in the robot's performance, which helps students think and act like a robot, which in turn might also benefit their CT skills.

The hierarchy of S–R interactions

The three roles discussed above represent three levels of students' engagement in S–R interactions. As a designer in the programming-computing interaction, student's engagement is at the lowest level, because a designer is only indirectly interacting with the robot through programming. After shifting to an observer in observational investigation, the student's engagement is at the second level, since an observer can directly interact with the robot but is not part of the robot's world. When the student's role then shifts to that of a participant in participatory investigation, he/she directly participates in the robot's world, so the engagement in S–R interactions is at the highest level (Fig. 1).

Limitations and suggestions

Several limitations of this research should be acknowledged. First, only 32 valid samples were analyzed. In quantitative analysis, a small sample size might lead to statistical problems, although this research tried to provide information about effect size and power analysis to address this limitation. Due to the small sample size, some statistical analyses, such as regression analysis, path analysis, confirmatory factor analysis, and so forth, could not be conducted. Second, the summer camp conducted for this research lasted only four weeks. Although short-term research is common in the RE field (Toh et al., 2016), long-term research is still necessary to understand students' learning development. Third, this research was conducted in an informal extracurricular environment, because RE is not yet included in the formal curriculum in mainland China. Finally, the current study focused only on the influence of S–R interactions on students' CT; influences of other factors, such as gender, age, level of learning interest, and cultural and pedagogical context, which are also important for students' CT, were not included.

For future studies, long-term research with a larger sample size, especially in formal learning environments, should be conducted. Also, other lifelong-learning skills should be assessed and probed. Additionally, future studies could consider other factors which might play important roles in students' CT skills development.

Fig. 1 The Hierarchy of S–R Interactions

Appendix 1: Course outline

Lesson 1. Introduction to the Intelligent Brick and Scratch

Using Scratch to write a simple program for controlling the motors; download to the Intelligent Brick

Lesson 2. Make your vehicle 1 (Moving forward)

Build your own vehicle and make it move forward (or as you wish)

Lesson 3. Make your vehicle 2 (Go and return)

Build your own vehicle and make it move forward for several seconds (depends on you) and return back

Lesson 4. Make your vehicle 3 (Touch and return)

Make a vehicle which can move and return when it touches a wall

Lesson 5. Make your vehicle 4 (Line following)

Make a vehicle which can move following the black line

Lesson 6. Make your vehicle 5 (An off-road vehicle)

Make an off-road vehicle which can turn to the left/right

Lesson 7. Make your electronic fan

Make an electronic fan which can be turned on/off by pressing the button

Lesson 8. Make your toy gyro

Build your own gyro and make it spin

Lesson 9. A “walking” robot

Make a “walking” robot which can walk like a human

Lesson 10. A Dagao machine (i.e., a machine for making a Chinese rice cake)

Make a Dagao machine with one motor and several gears

Lesson 11. A robot after you

Make a robot after you which can follow your steps

Lesson 12. A simple football robot

Using only one lighting sensor to make a simple football robot which can detect and hit the light football

Appendix 2: Computational thinking rubric

	Emerging (1)	Moderate (2)	Substantive (3)
Formulating problems: Formulating problems in a way that enables us to use a computer and other tools to help solve them. It may involve problem decomposition	Problems that too general to be solved by computer programming OR unrelated to task goal. e.g.: “How to make a robot?” “Can pigs fly?”	Problems that are to task goal BUT too general to be solved by computer programming. e.g.: “How to make a football robot?”	Problems that are to task goal AND specific enough so could be possibly solved by computer programming. e.g.: “How to order the robot to detect whether it touches the wall?”

	Emerging (1)	Moderate (2)	Substantive (3)
Abstraction: Representing data through abstractions such as models and simulations	No evidence of interpreting problems in an abstractive way	Limited evidences of interpreting problems in an abstractive way. e.g.: "If the robot touches the wall, then it should return back."	Can properly interpret problems in an abstractive way. e.g.: "If the robot receives a '1' sign from the touch sensor, then the parameters of motor speed should be negative."
Logical thinking: Logically organizing and analysing data	Statements do not follow logical path. e.g.: The robot is out of control	Statements follow logical path with some complexity. e.g.: The robot basically completes the given task	Statements follow logical path with more complexity. e.g.: The robot can complete challenging task(s)
Using algorithm: Automating solutions through algorithmic thinking (a series of ordered steps)	No evidence of using proper algorithm. e.g.: Fails to use proper programming logics (i.e., conditional logic, iterative logic, or parallel logic)	Some evidence of using proper algorithm. e.g.: Properly uses only one programming logic	More evidence of using proper algorithm. e.g.: Properly uses two or more programming logics
Analysing and implementing solutions: Identifying, analysing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources	No evidence of the ability to debug the program	Some evidence of debugging	Strong evidence of debugging
Generalizing and problem transfer: Generalizing and transferring this problem-solving process to a wide variety of problems	Always get stuck by similar problems	Stuck by novel problems but can transfer problem-solving process to similar problems	Can transfer problem-solving process to a wide variety of problems

Adapted from ISTE (2011) and Leonard et al. (2016)

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, 75(B), 661–670. <https://doi.org/10.1016/j.robot.2015.10.008>

- 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>
- Bakala, E., Visca, J., Tejera, G., Seré, A., Amorin, G., & Gómez-Sena, L. (2019). Designing child-robot interaction with Robotito. In *2019 28th IEEE international conference on robot and human interactive communication* (pp. 1–6).
- Barak, M., & Assal, M. (2018). Robotics and STEM learning: Students' achievements in assignments according to the P3 Task Taxonomy—Practice, problem solving, and projects. *International Journal of Technology and Design Education*, 28(1), 121–144. <https://doi.org/10.1007/s10798-016-9385-9>
- Barak, M., & Zadok, Y. (2009). Robotics projects and learning concepts in science, technology and problem solving. *International Journal of Technology and Design Education*, 19(3), 289–307. <https://doi.org/10.1007/s10798-007-9043-3>
- Belpaeme, T., Baxter, P., Greeff, J., Kennedy, J., Read, R., Looije, R., et al. (2013). Child-Robot interaction: Perspectives and challenges. In G. Herrmann, M. J. Pearson, A. Lenz, P. Bremner, A. Spiers, & U. Leonards (Eds.), *International conference on social robotics* (pp. 452–459). Springer. https://doi.org/10.1007/978-3-319-02675-6_45
- 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>
- Bers, M. U. (2010). The TangibleK robotics program: Applied computational thinking for young children. *Early Childhood Research and Practice*, 12(2), 1–20.
- Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers and Education*, 72, 145–157. <https://doi.org/10.1016/j.compedu.2013.10.020>
- Chalmers, C. (2018). Robotics and computational thinking in primary school. *International Journal of Child-Computer Interaction*, 17, 93–100.
- Chang, C.-W., Lee, J.-H., Chao, P.-Y., Wang, C.-Y., & Chen, G.-D. (2010). Exploring the possibility of using humanoid robots as instructional tools for teaching a second language in primary school. *Educational Technology & Society*, 13(2), 13–24.
- Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., & Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computers & Education*, 109, 162–175. <https://doi.org/10.1016/j.compedu.2017.03.001>
- Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1), 37–46.
- Cohen, L., Manion, L., & Morrison, K. (2011). *Research methods in education* (7th ed.). Routledge.
- Creswell, J. W. (2014). *Research design: Qualitative, quantitative and mixed methods approaches* (4th ed.). Sage Publications.
- Eguchi, A. (2012). Educational robotics theories and practice: Tips for how to do it right. In B. Barker, G. Nugent, N. Grandgenett, & V. Adamchuk (Eds.), *Robots in K-12 education: A new technology for learning* (pp. 1–30). IGI Global. <https://doi.org/10.4018/978-1-4666-0182-6..>
- Eguchi, A. (2013). Educational robotics for promoting 21st century skills. *Journal of Automation, Mobile Robotics & Intelligent Systems*, 8(1), 5–11. https://doi.org/10.14313/JAMRIS_1-2014/1
- Eguchi, A. (2014). Learning experience through robocupjunior: Promoting engineering and computational thinking skills through robotics competition. In *121st ASEE annual conference & exposition* (pp. 24–41).
- Erdfelder, E., Faul, F., & Buchner, A. (1998). GPOWER: A general power analysis program. *Behavior Research Methods, Instruments, & Computers*, 28(1), 1–11. <https://doi.org/10.3758/BF03203630>
- Fanchamps, N. L., Slagen, L., Hennissen, P., & Specht, M. (2021). The influence of SRA programming on algorithmic thinking and self-efficacy using Lego robotics in two types of instruction. *International Journal of Technology and Design Education*, 31, 203–222.
- Fortunati, L., Manganelli, A. M., & Ferrin, G. (2020). Arts and crafts robots or LEGO® MINDSTORMS robots? A comparative study in educational robotics. *International Journal of Technology and Design Education*. <https://doi.org/10.1007/s10798-020-09609-7>.
- International Society for Technology in Education (ISTE). (2011). Operational definition of computational thinking for K-12 education. Retrieved January 17, 2020, from <https://id.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf?sfvrsn=2>.
- 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. <https://doi.org/10.1007/s10639-018-9729-z>
- Julià, C., & Antolí, J. Ö. (2016). Spatial ability learning through educational robotics. *International Journal of Technology and Design Education*, 26, 185–203. <https://doi.org/10.1007/s10798-015-9307-2>

- Jung, S.-E., & Won, E.-S. (2018). Systematic review of research trends in robotics education for young children. *Sustainability*, 10(905), 1–24. <https://doi.org/10.3390/su10040905>
- Kim, S. W., & Lee, Y. (2019). Development of project-based robot education program for enhancing interest toward robots and computational thinking of elementary school students. *Journal of the Korea Society of Computer and Information*, 24(1), 247–255.
- Larkins, B. D., Moore, C. J., Rubbo, L. J., & Covington, L. R. (2013). Application of the cognitive apprenticeship framework to a middle school robotics camp. In *Proceeding of the 44th ACM technical symposium on computer science education* (pp. 89–94). <https://doi.org/10.1145/2445196.2445226>.
- Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., et al. (2011). Computational thinking for youth in practice. *ACM Inroads*, 2(1), 32–37. <https://doi.org/10.1145/1929887.1929902>
- Lee, K. T., 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. <https://doi.org/10.1080/07380569.2013.805676>
- Leonard, J., Buss, A., Gamboa, R., Mitchell, M., Fashola, O. S., Hubert, T., & Almughyirah, S. (2016). Using robotics and game design to enhance children's self-efficacy, STEM attitudes, and computational thinking skills. *Journal of Science Education and Technology*, 25(6), 860–876. <https://doi.org/10.1007/s10956-016-9628-2>
- Levy, S. T., & Mioduser, D. (2008). Does it “want” or “was it programmed to...”? Kindergarten children's explanations of an autonomous robot's adaptive functioning. *International Journal of Technology and Design Education*, 18(4), 337–359.
- Levy, S. T., & Mioduser, D. (2010). Approaching complexity through playful play: Kindergarten children's strategies in constructing an autonomous robot's behavior. *Journal of Computers for Mathematical Learning*, 15(1), 21–43.
- Li, Y., Yang, B., & Chen, Z. (2019). 小學生計算思維培養的過程和策略研究—基於對武漢市從事機器人教育的 26 位教師的深度訪談[Learning powered by technology from perspective of cost-effectiveness analysis: Based on survey of minors' new media use in 2018] (in Chinese). *E-Education Research*, 40(12), 115–121.
- Liu, E. Z. F., Lin, C. H., Liou, P. Y., Feng, H. C., & Hou, H. T. (2013). An analysis of teacher-student interaction patterns in a robotics course for kindergarten children: A pilot study. *Turkish Online Journal of Educational Technology*, 12(1), 9–18.
- Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education*, 10(4), 1–15. <https://doi.org/10.1145/1868358.1868363>
- Mubin, O., Stevens, C. J., Shahid, S., Mahmud, A. A., & Dong, J.-J. (2013). A review of the applicability of robots in education. *Technology for Education and Learning*. <https://doi.org/10.2316/Journal.209.2013.1.209-0015>
- Noh, J., & Lee, J. (2020). Effects of robotics programming on the computational thinking and creativity of elementary school students. *Educational Technology Research and Development*, 68(1), 463–484.
- Olabe, J. C., Olabe, M. A., Basogain, X., & Castaño, C. (2010). Programming and robotics with Scratch in primary education. In A. Méndez-Vilas (Ed.), *Education in a technological world: Communicating current and emerging research and technological efforts* (pp. 356–363). Oxford University Press.
- Rowell, P. M. (2002). Peer interactions in shared technological activity: A study of participation. *International Journal of Technology and Design Education*, 12, 1–12.
- Shin, N., & Kim, S. (2007). Learning about, from, and with robots: Students' perspectives. In *RO-MAN 2007—The 16th IEEE international symposium on robot and human interactive communication* (pp. 1040–1045). <https://doi.org/10.1109/ROMAN.2007.4415235>.
- Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education*, 148, 1–22. <https://doi.org/10.1016/j.compedu.2019.103798>
- Toh, E. L. P., Causo, A., Tzuo, P. W., Chen, I. M., & Yeo, S. H. (2016). A review on the use of robots in education and young children. *Educational Technology & Society*, 19(2), 148–163. <https://doi.org/10.2307/jeduc techsoci.19.2.148>
- Üçgül, M., & Altıok, S. (2021). You are an astronaut: The effects of robotics camps on secondary school students' perceptions and attitudes towards STEM. *International Journal of Technology and Design Education*. <https://doi.org/10.1007/s10798-021-09673-7>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35. <https://doi.org/10.1145/1118178.1118215>
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717–3725.
- Wing, J. M. (2010). *Computational thinking: What and why?* <https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf>. Accessed 10 June 2021.

- Yang, K., Liu, X., & Chen, G. (2020). The influence of robots on students' computational thinking: A literature review. *International Journal of Information and Education Technology*, 10(8), 5.
- Yuen, T. T., Boecking, M., Tiger, E. P., Gomez, A., Guillen, A., Arreguin, A., & Stone, J. (2014). Group tasks, activities, dynamics, and interactions in collaborative robotics projects with elementary and middle school children. *Journal of STEM Education*, 15(1), 39–46.
- Zhong, B., & Wang, Y. (2021). Effects of roles assignment and learning styles on pair learning in robotics education. *International Journal of Technology and Design Education*, 31, 41–59. <https://doi.org/10.1007/s10798-019-09536-2>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.