Education Tech Research Dev (2022) 70:2211–2234
https://doi.org/10.1007/s11423-022-10174-0

DEVELOPMENT ARTICLE

Learning programming through robots: the effects of educational robotics on pre-service teachers' programming comprehension and motivation

Alex Fegely[1] · **Hengtao Tang**[2]

Accepted: 19 November 2022 / Published online: 30 November 2022

Abstract
The purpose of this convergent mixed-methods study was to evaluate the effect of educational robotics on pre-service teachers' programming comprehension and motivation. Computer science is increasingly being integrated into K-8 curricula. However, a shortage of teachers trained to teach basic computer science concepts remains unresolved. This study thus utilized educational robotics as "mindtools" to teach programming concepts to pre-service teachers. Data were obtained through a pre-post comprehension assessment, a pre-post motivation survey, field notes, and individual interviews. The findings of this study indicated that pre-service teachers' comprehension of programming concepts and motivation related to programming can be improved through educational robotics to statistically significant levels. Design implications on integrating educational robotics into pre-service teacher programming instruction are discussed.

Keywords Programming · Robotics · Pre-service teachers · Teacher education · STEM

Introduction

Computer science is being increasingly integrated into K-8 curricula in the United States; however, due to the dearth of teachers specializing in teaching computer science (CS), core subject teachers without sufficient CS competence are being asked to integrate CS concepts into their instruction (Burke et al., 2016; Mannila et al., 2014). Compounding the lack of teachers prepared to teach CS concepts, teachers may erroneously feel that CS can only be taught through high-level computer programming languages like C++ or Java (El-Hamamsy et al., 2020). A pervasive impression of intimidation among teachers' vis-a-vis

✉ Alex Fegely
agfegely@coastal.edu

Hengtao Tang
htang@mailbox.sc.edu

1 Instructional Technology, Department of Educational Studies Spadoni College of Education & Social Sciences, Coastal Carolina University, P.O. Box 261954, Conway, SC 29528, USA

2 Department of Educational Studies College of Education, Learning Design & Technologies, University of South Carolina, 820 Main St. 239 Wardlaw, Columbia, SC 29208, USA

learning programming concepts makes them less motivated to implement any programming instruction, denying students the chance to develop their CS competencies from an early age (Rogerson & Scott, 2010; Sentance & Csizmadia, 2020). Research has attributed teachers' intimidation with programming to a lack of opportunities for pre-service teachers (PSTs) to learn effective CS pedagogy, especially given few educator preparation programs prepare PSTs to implement CS concepts in their teaching (Gleasman & Kim, 2020). Efficient curriculum with a focus on PSTs' CS competence and their motivation towards teaching CS is needed to overcome PSTs' intimidation with programming.

Educational robotics kits have gained intrigue for their potential to make learning programming less intimidating for PSTs (Ortiz et al., 2015). Robots are physical manipulatives that can help cut through PSTs' initial apprehension of CS concepts and become engaged in programming (Kim et al., 2015). This research thus examined the effectiveness of robotics in preparing PSTs to teach programming and helping PSTs overcome the challenge of learning programming. A convergent mixed methods study (Creswell & Plano Clark, 2018) was conducted in a required educational technology course for PSTs, by collecting complimentary sources of qualitative and quantitative data to converge the findings about the effectiveness of robotics on the programming comprehension and motivation of PSTs.

Two novel aspects as follows distinguish this study from previous research. First, while PSTs' experiences learning programming on a computer screen have been researched (e.g., Erol & Kurt, 2017; Gleasman & Kim, 2020; Yukselturk & Altiok, 2017), few studies have investigated the use of robotics manipulatives for teaching PSTs programming concepts (Kim et al., 2015; Kucuk & Sisman, 2018). Second, while numerous studies have aimed to measure the impacts of robotics on variables such as PSTs' STEM engagement (Kim et al., 2015), engineering design (Yuan et al., 2022), or debugging (Kim et al., 2022) research evaluating the impact of robotics specifically on PSTs' motivation related to programming is still emerging (Jaipal-Jamani & Angeli, 2017; Sisman & Kucuk, 2019). The findings benefit researchers by adding to the limited literature on PSTs learning programming and provide implications for pre-service teacher educators (PSTEs) in designing effective programming-focused instruction.

Specifically, the research questions for this study were:

(RQ1) What is the effect of educational robotics on PSTs' comprehension of programming concepts?

(RQ2) How and to what extent does educational robotics influence PSTs' motivation related to programming?

Literature review

Programming

Common text-based programming languages have been reported to be challenging to learn because of the specific grammar and syntax requirements for each command (Alkaria & Alhassan, 2017). There are educational block-based versions of programming languages (e.g., Scratch, Alice) that offer varying scaffolds to novice programmers while they learn to write programs (Weintrop & Wilensky, 2017). Such programming languages remove the complex syntax and related errors likely to be encountered by novices by incorporating codes into blocks that have the grammar essential to programming languages built-in (Weintrop & Wilensky, 2017).

Block-based programming languages are therefore often integrated in PST classes to introduce novices to programming. Research has indicated that PSTs' attitudes and motivation to integrate CS concepts into their teaching improved because of block-based programming instruction (Gleasman & Kim, 2020). However, PSTs have experienced issues with programming concepts like identifying variables, defining conditions, and identifying errors (Kim et al., 2015). A study by Ortiz et al., (2015) noted that PST participants felt intimidated by the abstract math concepts required to teach programming. These studies imply that PSTs experience difficulties with programming concepts.

Constructivism and robotics

According to Piaget (1973), constructivism is the building of abstract knowledge structures in one's mind through concrete experiences. In the constructivist view, the mental creation of knowledge necessitates the use of hands-on activities (Bruner, 1996). Robots' abilities to be used as physical manipulatives which can illuminate abstract concepts (Han, 2013), like programming concepts, make them an ideal constructivist mindtool for learning.

Constructivist robotics instruction has been increasingly integrated by educator preparation programs to foster in-service teacher (IST) and PST programming skills. For example, Alimisis et al., (2007) developed a constructivist approach aligned computer and robotics technologies with the construction of meaning through hands-on activities. Further, Jaipal-Jamani and Angeli (2017) found significant increases in elementary PSTs' understanding of science and computational thinking concepts as a result of constructivist robotics activities in a science teaching methods course. PSTs also perceived that a constructivist robotics programming course improved their programming skills in a study by Kucuk and Sisman (2018). Additionally, constructivist robotics instruction can help improve various aspects of teachers' motivation about programming. Kay et al., (2014) found that ISTs' confidence in their programming skills increased significantly after they completed robotics activities, and Sisman and Kucuk (2019) found that constructivist learning methods improved PSTs' motivation related to robotics.

Impact of robotics on PSTs' comprehension of programming concepts

Numerous researchers (e.g., Kaya et al., 2015; Majherová & Králík, 2017) point out that robotics instruction is becoming more common in PST preparation around the world. Jaipal-Jamani and Angeli (2017) found robotics activities were effective for increasing PSTs' abilities to write algorithms and debug programs as Canadian PSTs learned about algorithms, debugging, control structures, and writing sequences of programming. Kucuk and Sisman (2018) studied Turkish PSTs' experiences while learning programming and robotics in a 13-week course. The PSTs learned about composing original programs for the robots in collaborative groups and felt the robotics programming course improved their programming skills (Kucuk & Sisman, 2018). Contrasting these findings, a study by Kim et al. (2018) determined that PSTs did not successfully learn programming concepts or debugging, and they instead relied upon tinkering to successfully program robots. As robotics instruction has been increasingly integrated by PSTEs, the research calls for evidence on its impact on PSTs' comprehension of programming concepts.

Impact of robotics on PSTs' motivation related to programming

Few studies have examined motivation in relation to learning programming (Kelleher et al., 2007). Motivation is the extent to which persistent effort is sustained toward a goal (Ryan & Deci, 2020). Studies have shown participants with high levels of motivation spend more time learning, get more engaged in learning materials, and are more likely to apply new knowledge (Ryan & Deci, 2020). Researchers have presented numerous indicators of motivation like (a) behavioral engagement – behaviors associated with effort in learning (Fredricks et al., 2004), (b) intrinsic motivation – learners' desires to learn about a topic due to their inherent interest (Ryan & Deci, 2020), (c) career motivation – motivation exhibited when learners understand the topic is relevant to their future careers (Arwood, 2004), (d) self-efficacy – learners' confidence in their abilities to achieve learning tasks (Bandura, 1997), and (e) self-determination – the control learners exhibit over their learning (Black & Deci, 2000).

Despite few studies addressing PST's motivation towards programming, research has found that robotics interventions enhance PSTs' intention of integrating robotics into their instruction (Jaipal-Jamani & Angeli, 2017; Kaya et al., 2015). A study by Jaipal-Jamani and Angeli (2017) reported that over 85% of their PST participants were willing to use robotics in their teaching after learning with them. Kaya et al.'s (2015) study reported that all PSTs decided to integrate block-based programming and robotics into their elementary science classes. In addition, a study by Sisman and Kucuk (2019) added that PSTs who decided to integrate robotics were most motivated by the idea that they could learn to teach their future students how to program robots. However, literature on the impact of robotics on PSTs' motivation towards teaching programming remains limited but urgently needed.

Methods

Research design

A convergent parallel mixed-methods (Creswell & Plano Clark, 2018) design was applied to provide holistic evaluation of the impact of robotics on PSTs. This study used a pre-post one-group design for the quantitative investigation and collected qualitative data concurrently. The findings from the two sources were then converged. Convergent parallel mixed-methods design is a technique in which researchers gather quantitative and qualitative data simultaneously then analyze the data separately to see if the triangulation of results "confirm or disconfirm" each other (Creswell, 2014, p. 219). This convergence allows researchers to pinpoint more specific conclusions than single-method research as the qualitative results can provide depth to the quantitative results.

Participants

This study took place at a medium-sized liberal arts university in the southeastern USA and included a purposeful sample of participants from one educational technology class. The inclusion criteria stipulated that participants had to be PSTs in education majors. Out of the 23 students in the class, two non-education majors were excluded. Three education majors dropped the class during the study, so their data were removed prior to analysis. A total of

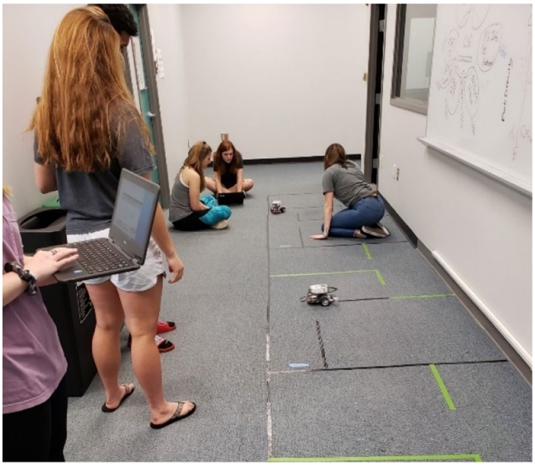
18 PSTs, including 15 females and three males, made up the sample. These participants represented all the education majors offered by the university: early childhood (2), elementary (9), middle level (3), special (2), and physical (2). The participants' ages ranged from 18 to 23 ($M=19$, $SD=1$). The participants included freshmen (6), sophomores (11), and one junior.

Intervention

This study utilized a constructivist robotics intervention that spanned four weeks. As described by Martin et al., (2011), Lego EV3 robots running the EV3-G block-based language were chosen because of Lego robotics' developmental appropriateness for the K-8 learners the PSTs will teach, and the popularity of Lego EV3 robots in K-8 schools. Participants were paired randomly for the intervention as suggested by Marzano (2007) to foster problem-solving and collaboration. The class met twice per week for one hour and fifteen minutes per session. Each lesson was aligned to K-8 state and course standards.

The robotics intervention was divided into four week-long units: (1) Basic Procedures, (2) Advanced Procedures, (3) Control Structures, and (4) Variables. Each unit consisted of demonstrations, activities, and challenges. The Basic Procedures unit focused on the core syntactic programming skills needed to write functional programs. Participants were challenged to program their robots to travel exactly one meter by writing programs based on three different methods: time, revolutions, and degrees. The Advanced Procedures unit focused on semantic and strategic programming skills needed to write programs which navigated the robots around obstacles. Participants were introduced to pseudocode and writing algorithms. Next, participants were presented with step-by-step instructions for writing more advanced programs for turning. Then, participants wrote more advanced programs to have their robots follow paths. Pairs then debugged and modified a given program to move their robots around a box. In the RoboMaze Challenge (Fegely et al., 2021), pairs programmed their robots through a maze made from electrical tape (Fig. 1). Before placing their robot in the maze, partners were required to write their programs based on a provided schematic and their own calculations. The Control Structures unit focused on writing programs utilizing flow control, like if/then statements and loops. The learning activity for this

Fig. 1 Participants test programs in a maze



unit required pairs to program their robots to move in a zigzag motion, making a sound at the end of the program after the required loops. In the challenge, pairs modified the programs they had written to navigate their robots around the box by replacing superfluous recursive programming with succinct loops. The Variables unit focused on integrating variables into the flow control of advanced programs. Participants were presented with step-by-step instructions for writing programs using variables inside if/then statements. Then, pairs wrote programs utilizing the color sensor that scanned different colors, incrementing a variable each time a predetermined color was detected. In the learning activity, pairs programmed their robots to speed up when the color sensor detected blue (increasing the speed variable each time), and stop the robot when the color sensor detected red. For the final challenge, the RoboMaze Challenge utilized in the Advanced Procedures unit was modified with the addition of red (right) and green (left) pieces of tape where the robots needed to turn. The walls of the maze and the finish line were made of black tape, and the robots were programmed to stop if they detected black. The criteria for the Color Maze Challenge stipulated that every time the robots encountered a red line, they turned right and every time they encountered a green line they turned left and incremented a variable by one on the robot's screen using a variable and the formula $(x+1)$. Pairs completed the Color Maze Challenge when they successfully navigated their robots to the finish line of the maze using the programming concepts they learned related to procedures, control structures, and variables.

Data sources

Complementary sources of quantitative and qualitative data were collected: Programming Comprehension Assessment (PCA), Programming Motivation Survey (PMS), field notes, and individual interviews.

Quantitative instruments

The participants completed the researcher-created PCA before (pre-tests) and after (post-tests) the intervention. The PCA included 20 multiple choice questions in four subsections of five questions. Each subsection was aligned to the four units of instruction: Basic Procedures, Advanced Procedures, Control Structures, and Variables. The questions prompted participants to read, debug, differentiate, problem-solve, and arrange portions of programs. Each correct answer was worth one point. The instrument was reviewed by two experts in programming and robotics to establish face validity (Salkind, 2010). Item analysis was conducted on the post-test to ensure the reliability. Cronbach's alpha value was 0.847, suggesting an acceptable level of internal consistency for the instrument (DeVellis, 2003). However, if Q12 was removed, Cronbach's alpha value for the instrument increased to 0.864. This item was thus removed for the analysis. Point Biserial correlation coefficient was calculated to determine item-total correlation for each item (Gupta, 1960). The result showed all the remaining question had a coefficient value higher than a minimum desired score (0.15).

The PMS was given before and after instruction. It was designed using a combination of intentionally and carefully selected statements from an existing valid and reliable instrument in addition to researcher-designed statements. The 25-item Likert scale PMS was adapted from the Science Motivation Questionnaire II (SMQ-II) (Glynn et al., 2011). Five statements from the SMQ-II representing the category Grade Motivation did not fit this study because participants were not being graded on the intervention. These were replaced with researcher-created statements.

The final PMS had five subscales: (1) intrinsic motivation, (2) career motivation, (3), self-determination, (4) self-efficacy, and (5) motivation to integrate programming into teaching (MTIPIT). The instrument was validated by three experts in programming and education. Participants responded to items on a five-point Likert-type scale from (1) strongly disagree to (5) strongly agree. The statements participants responded to were straight-forward in meaning and in random order (DeVellis, 2003). The Cronbach's alpha for the PMS in pre- ($\alpha = 0.96$) and post- ($\alpha = 0.94$) surveys indicated a very good internal consistency (DeVellis, 2003).

Qualitative instruments

Field notes have been described as essential for rigorous qualitative research and offer an extra layer of detail with which to aid in the construction of thick, rich descriptions (Creswell, 2017). Observations related to motivation and behavioral engagement (Fredricks et al., 2004; Kim et al., 2015) were recorded in a composition book by the primary researcher during the teaching of the intervention. Examples of such observations included on-task behavior and teamwork dynamics.

Individual interviews provided descriptive qualitative data of participants' perspectives (Mertler, 2017). Purposeful sampling was used to select participants for the interviews (Creswell, 2017). One third of the participants ($n = 6$) were purposefully selected for individual interviews based on participants' behavioral engagement recorded in the field notes. All participant quotes are attributed with gender-neutral pseudonyms and pronouns as to preserve the participants' confidentiality. Two participants representing high (Skyler, Tracy), medium (Marion, Kerry), and low (Casey, Jessie) behavioral engagement were selected randomly for individual interviews to have a balanced population of interviewees. High behavioral engagement was exhibited as on-task behavior, deep involvement, and active participation (Fredricks et al., 2004). To offset researchers' subjective bias in the selection of the participants, peer debriefing (Lincoln & Guba, 1985) was conducted with two scholars with expertise in robotics instruction for PSTs. Each individual interview followed a semi-structured interview protocol and lasted approximately 30 min. Each interview was audio-recorded and transcribed for analysis.

Data analysis

For the quantitative data from the PCA and the PMS, participants' responses in each of the units or subscales were analyzed to determine the difference between the pre- and post-tests and surveys. Shapiro-Wilk tests were used to evaluate the data's normality. The data were found to violate the normality assumption ($p < .05$) for both the PCA and PMS. Wilcoxon signed-rank tests were used for comparing pre- and post- tests and surveys within the units or subscales. The effect size of the change in these non-parametric data was reflected by the correlation coefficient r (Pallant, 2007).

Inductive analysis (Mertler, 2017) was conducted to process qualitative data from the interview transcripts and field notes. The transcripts and field notes were uploaded into the coding tool Delve. Two cycles of coding were performed. For first cycle coding, two rounds of open coding were used to separate the qualitative data into discrete parts to analyze similarities and differences (Saldaña, 2016). The transcripts and field notes were analyzed sentence-by-sentence. Codes which summarized the experience of the participant in the transcript or observations in the field notes were assigned to the qualitative data (Mertler, 2017).

The second cycle consisted of two rounds of pattern coding. Pattern coding was used to condense large amounts of data into smaller units to develop categories and then themes (Saldaña, 2016). In this cycle, pattern coding was used to filter the first cycle codes down into pattern codes. In a code mapping process described by Saldaña (2016), “categories of categories” in “superordinate and subordinate arrangement” (p. 278) were created by moving around the pieces of paper for each pattern code. Pattern codes were united into categories. The categories were analyzed, and themes were revealed.

Results

The analysis and findings are divided into two parts representing the two research questions of this study.

(RQ1) what is the effect of educational robotics on PSTs’ comprehension of programming concepts?

As demonstrated in Table 1, Wilcoxon signed-rank tests indicated that that there were statistically significant differences between pre-tests and post-tests in the total score and each unit of the PCA. The effect sizes were large ($r < -.50$) for all the units and total (Pallant, 2007).

(RQ2) how and to what extent does educational robotics influence PSTs’ motivation related to programming?

Quantitative findings

As demonstrated in Table 2, Wilcoxon signed-rank tests revealed that the overall increase in motivation from the PMS pre-survey to post-survey as well as the increases in all

Table 1 Results of wilcoxon signed-rank tests for PCA

Units	Pre-test *Mdn.*	Pre-test *SD*	Post-test *Mdn.*	Post-test *SD*	*Z*	*df*	*p*	*r*
Basic Procedures	1.00	0.73	3.50	1.47	− 3.30	17	< 0.001*	−0.78
Advanced Procedures	1.00	0.90	3.50	1.49	− 3.43	17	< 0.001*	−0.81
Control Structures	1.00	0.75	2.00	1.20	− 3.19	17	0.001*	−0.75
Variables	1.00	0.87	3.00	1.58	− 2.85	17	0.004*	−0.67
Total	3.50	1.32	11.50	4.86	− 3.45	17	< 0.001*	−0.81

Control Structures included four items and the other three units had five questions. The total was out of 19

*$p < .05$.

Table 2 Results of wilcoxon signed-rank tests for PMS

	Pre-survey		Post-survey					
Subscales	*Mdn.*	*SD*	*Mdn.*	*SD*	*Z*	*df*	*p*	*r*
Intrinsic Motivation	2.00	0.94	3.30	0.96	− 3.13	17	0.002*	−0.52
Career Motivation	3.00	0.98	3.72	0.59	− 3.58	17	< 0.001*	−0.60
Self-Determination	1.70	0.98	3.40	0.72	− 3.69	17	< 0.001*	−0.62
Self-Efficacy	2.00	0.82	3.69	0.84	− 3.53	17	< 0.001*	−0.59
MTIPIT	2.80	1.04	3.80	0.75	− 3.34	17	0.001*	−0.56
Total	2.20	0.84	3.42	0.64	− 3.64	17	< 0.001*	−0.61

Subscale and total medians contextualized within a five-point Likert scale

*$p < .05$.

subscales were significant. The effect sizes were large ($r < -.50$) for all the subscales and total (Pallant, 2007).

Qualitative findings

Theme 1: participants perceived that a problem-based robotics curriculum improved their intrinsic motivation toward programming Interviewees described their intrinsic motivation (Table 3) through characterizations of the robotics activities by referring to them as being "fun," "cool," or "interesting". Jessie explained, "I've taken technology classes before and if we did something like this it would have been like 10 times cooler." Tracy added, "Honestly, I think the whole experience is really fun and just being able to move the… program things so you could move a robot. I think that's a really cool thing to do."

Half the interviewees (one from each behavioral engagement group) commented that an element they found interesting was the ability of the programming and robotics to take abstract concepts and make them concrete for learning. All interviewees articulated that the authentic activity and challenge elements of the curriculum were intrinsically motivating, especially the problems that utilized mazes.

Theme 2: participants agreed that teachers with programming skills had advantages in their professions This theme describes participants' agreement that knowing programming as a skill had advantages for them professionally as teachers. Interviewees described their career motivation through references to the personal career and teaching advantages of learning programming. Two categories subsumed this theme (Table 4).

Interviewees expressed their perceptions of the value of learning programming in terms of obtaining more advantages on the job market in two aspects: (a) being more marketable

Table 3 Theme 1 categories

Category	Example evidence
Representing abstract concepts in concrete form fostered interests	Skyler: "It's like a physical way, it shows them like visual, like they'll be able to see like you do this you add this and the robot does something." Casey: "I liked when there was a maze and we had to make an equation to figure it out because I think it's interesting how it translates from like a math equation to like actually like seeing it happen in front of your eyes."
Problem solving using programming improved motivation	Marion: "I think the most enjoyable part was we had to do the maze." Kerry: "Probably when we learned to get them to talk… I think it added more of a sense of like depth to it, maybe? Not just in moving around, like they were like moving and talking and it was like really interesting to see like a box do that really."

Table 4 Theme 2 categories

Category	Example evidence
Job seeking advantages for PSTs	Marion: "Especially with how society is going with more technology, so it'll be a plus for you to have that special for employers that you have that [sic.], so I think it's a plus." Jessie: "You walk into a job interview, and you tell them I don't even need training like I know how to do this I think it goes a long way."
Expanded PSTs' teaching skillsets	Casey: "I think it would be like beneficial just like the parallelogram blocks and stuff, like kids play with that they don't even realize that they're learning." Skyler: "I think it's a way to like get them [students] to learn without realizing that they're learning something 'cause they're just like oh cool it's robots like they're not really thinking about the fact that they are learning something through using them."

in interviews, and (b) creating more opportunities for themselves for positions outside their licensure area. Overall, half the interviewees viewed learning programming as a skill that would be valuable in obtaining employment. While some interviewees noted that programming was a skill that would impress employers, others noted that learning programming might provide more options on the job market for positions different from their licensure area. Four interviewees expressed that learning programming through robotics would help them expand their teaching skillsets to benefit their future students. Participants perceived that programming offered new teaching strategies and would allow them to enhance their lesson plans to grab students' attention, engage them, and have students learn through play.

Theme 3: participants experienced self-determination towards programming in the face of robotics challenges In individual interviews, most participants cited using personalized problem-solving techniques and collaborative problem-solving strategies to solve problems. Field note entries highlighted participants' preference for autonomy in their problem-solving solutions. The following section outlines the categories subsumed in support of this theme (Table 5).

Participants described that the open-ended nature of robotics activities and challenges fostered the autonomy to try their own unique options to solve problems. Marion highlights that there was not one correct answer to programming the robot through the maze. Participants had the freedom of decision-making and the autonomy to choose their own programming method and path through the maze. Skyler affirms that they felt they had the independence and autonomy to figure their "own way" to navigate the maze. The field notes also revealed that participants asked if they had to solve a problem

Table 5 Theme 3 categories

Category	Example evidence
Autonomy in trying different programming options to solve problems	Marion: "I guess you could use the same but different program but like I guess use different ways to get to the same result." Skyler: "We didn't really know that much about programming yet and we had to kind of like figure out our own way to like get through the maze."
Actively implementing collaborative problem-solving strategies	Skyler: "I just worked with my partner and like used her insight and used my insight together…I guess I will lean on the people that [I] worked with and ask them questions." Jessie: "We all had somebody or some people either next to us doing it with us and like if you didn't know how to do something like maybe your partner did…but there was always somebody in the class."

in a particular way (using rotations, degrees, or seconds) or if they could choose their own programming method to solve the problem. The participants were excited that they could use their own preferred method. All interviewees commented that actively implementing collaborative problem-solving (CPS) strategies contributed toward learning the programming concepts. They agreed that grouping participants into partners provided a strong collaboration aid. Interviewees also noted they sought help from peers outside of their immediate partner when they were unsure. For example, Jessie described how they picked out peers in other pairs who had completed the activities and challenges to help him. In addition, the field notes affirmed participants' interview descriptions. For example, during the Control Structures unit, one note mentioned "some groups finished quickly while others struggled to keep their robot in a straight line. Groups [are] helping each other."

Theme 4: participants perceived that the gradually increasing level of difficulty in the robotics curriculum improved their self-efficacy about programming from initially low levels. All interviewees described low initial levels of self-efficacy "like a blank slate" (Kerry noted in the interview) due to their perceived low comprehension of programming concepts. Then most participants described that the gradually increasing level of difficulty of the robotics curriculum increased their self-efficacy related to programming. The following section will outline the categories subsumed (Table 6).

All interviewees perceived their initial level of programming comprehension as nonexistent and felt a considerable increase in their programming expertise after the intervention. Five interviewees attributed the gradually increasing level of difficulty of the robotics curriculum as being helpful, especially the introductory concepts. Successes with these basic concepts developed participants' confidence gradually, and the basics that they learned helped them have success with more difficult problems.

Table 6 Theme 4 categories

Category	Example evidence
Overcoming initially low self-efficacy	Jessie: "At the beginning, I didn't really know what to expect. I don't really know but I remember we took the pre-test and like I see all these codes and stuff, and like I even sent the picture to my mom and I was like, 'do you have any idea how to do this?' And she's like, 'what are you talking about?' And I was like I wasn't really sure what to expect." Skyler: "Oh, it [self-efficacy] was definitely at a zero before."
Developing confidence about programming gradually	Kerry: Probably the first couple [lessons helped build confidence]…You're really helping conceptually building the foundations of like the other stuff that we learned. Casey: "I feel like they [instructional units] all were valuable [in building confidence] because they all like built onto each other, and then I feel like each time you did it like you could apply stuff from the last time."

Theme 5: participants perceived programming as a viable fit in their future classrooms This theme describes participants' perceptions of how programming could be applied into their pedagogy. Two categories subsumed this theme (Table 7).

Five interviewees intended to integrate programming into teaching, as evidenced by each of their responses to interview question #9: "Where do you position yourself in the continuum of adding or not adding programming activities to your classes? Why?". Their intentions ranged from reserved responses in which participants affirmed intentions to integrate programming but needed to learn more about programming aforehand (Casey, Jessie), to more decisive intentions. For example, Tracy starkly stated in their interview, "I want to add it." The first-hand experience with programming factored into interviewees' intentions. For example, "I think it's more valuable now and I understand like why it helps students like learning like through math and stuff," explained Casey. Interviewees had multiple ideas for integrating programming into their future instruction, including singular subject and cross-curricular connections. Casey explained they would use programming to teach students the different parts of math equations. Kerry explained that they would input numbers to a program to have a robot demonstrate different lengths for their elementary students to help illustrate math problems. Out of five integration ideas, four interviewees shared strategies for integrating programming with math, with the programming of robots to represent abstract math concepts as a commonality.

Table 7 Theme 5 categories

Category	Example evidence
Developing intentions to integrate programming	Skyler: "When you first proposed the idea that we would be using programming…I didn't really think that it would be useful at all, like I didn't really understand how I can possibly even use it in teaching and how it had anything to do with teaching, but obviously going through it I realized like it is very useful so it's kind of done a complete 180 to be honest." Kerry: "I could really see myself adding this to my lesson plans…I don't know exactly how it would fit in, but I know I could definitely like find a way once I get their curriculum. Like I would love to find a way."
Actively devising strategies to integrate programming	Skyler: "I want to teach second or third grade probably, and I feel like there's a lot of different ways I could incorporate it. Probably with math even like using the algorithms." (singular subject) Marion: "You could do like longitude and latitude…voyages of different explorers…I guess I can go back to the example with um…about colonialization in America. We can talk about…the different British ships that came over and we could talk about…how long they took to travel and as far as like mileage and then we can do like a fun activity with programming." (cross-curricular)

Integrating quantitative and qualitative findings

The qualitative data and findings were used to emphasize and detail the quantitative findings. The integrated quantitative and qualitative findings (Table 8) of this study indicate that PSTs' motivation related to programming can be improved significantly through robotics' influences on (1) intrinsic motivation, (2) career motivation, (3) self-determination, (4) self-efficacy, and (5) motivation to integrate programming into teaching. Integrated findings will be examined in the Discussion.

Table 8 Integrating Quantitative and Qualitative Findings

Finding	Quantitative evidence	Qualitative evidence
Intrinsic motivation improved in PSTs	Intrinsic motivation medians increased between pre-survey (2.00) and post-survey (3.30), ($Z = -3.13$, $p = .002$, $r = -.52$).	Theme 1: Participants perceived that a problem-based robotics curriculum improved their intrinsic motivation toward programming.
Career motivation improved in PSTs	Career motivation medians increased between pre-survey (3.00) and post-survey (3.72), ($Z = -3.58$, $p < .001$, $r = -.60$).	Theme 2: Participants agreed that teachers with programming skills had advantages in their professions
Self-determination improved in PSTs	Self-determination medians increased between pre-survey (1.70) and post-survey (3.40), ($Z = -3.69$, $p < .001$, $r = -.62$).	Theme 3: Participants experienced self-determination towards programming in the face of robotics challenges.
Self-efficacy improved in PSTs	Self-efficacy medians increased between pre-survey (2.00) and post-survey (3.69), ($Z = -3.53$, $p < .001$, $r = -.59$).	Theme 4: Participants perceived that the gradually increasing level of difficulty in the robotics curriculum improved their self-efficacy about programming from initially low levels.
MTIPIT improved in PSTs	MTIPIT medians increased between pre-survey (2.80) and post-survey (3.80), ($Z = -3.34$, $p = .001$, $r = -.61$).	Theme 5: Participants perceived programming as a viable fit in their future classrooms.

Discussion

(RQ1) what is the effect of educational robotics on PSTs' comprehension of programming concepts?

The findings of this study strongly indicate that robotics can be used to significantly improve PSTs' comprehension of programming concepts related to (1) basic procedures, (2) advanced procedures, (3) control structures, and (4) variables. These findings echo prior findings (Jaipal-Jamani & Angeli, 2017; Sullivan & Moriarty, 2009) on how robotics improve teachers' comprehension of programming concepts. Participants entered the study with low levels of programming comprehension, and after the intervention, the participants' scores increased significantly. It is worth noting that two participants' scores stayed the same and another one had a lower score on the post-test. Although possible that these participants did not learn anything over the four weeks of the intervention's instruction, these low scores might also be attributed to factors like assessment apathy (Thompson, 2008). While no participants achieved a perfect score on the PCA, five of the 18 participants scored 80% or higher on the post-test. Altogether, these findings suggest that PSTs' comprehension of programming concepts can be improved through educational robotics. The nearly unanimous positive results in this study confirm previous studies' findings (Jaipal-Jamani & Angeli, 2017; Sullivan & Moriarty, 2009) on the comprehension of programming concepts. Jaipal-Jamani and Angeli (2017) found that their population of elementary PSTs had statistically significant differences in programming knowledge between pre- and post- tests as the result of an educational robotics intervention. This study's results also confirm research by Sullivan & Moriarty (2009), which indicated that in-service teachers' understanding of programming increased from the no proficiency and low proficiency levels to the moderate and strong proficiency levels after robotics workshops.

Basic procedures

The increase in comprehension of basic procedures might be explained best by Ala-Mutka (2004) who suggested that "visualizing the basic programming structures" can be beneficial for novices in building their comprehension of programming (p. 6). The educational robots' actions allowed participants to visualize basic programming concepts through constructivist mental processes. Despite research by Kim et al. (2018), which noted that "participants omitted commands that were necessary for the robot to perform as planned" (p. 772) when PSTs used robots to learn programming, the results of this study were different. This difference might stem from Kim et al. (2018) using a different block-based programming language than the one used by PSTs in this study to demonstrate comprehension of the syntactic aspects of programming. Another possibility is that the robotics activities and challenges in this study improved the proficiency of participants in basic programming procedures beyond the level of comprehension of participants in the Kim et al. (2018) study.

Kim et al. (2018) also noted that debugging was difficult for PSTs as an overarching finding of their study. PSTs in this study also struggled to identify program errors in debugging questions. Kim et al. (2018) theorized that it is difficult for even those who are advanced programmers to debug a program as "it requires mindful, persistent engagement" (p. 769). Similarly, Falloon (2016) has noted that debugging can be a complicated process because it necessitates perseverance and a systemic approach, which is often discounted by students who adopt random, unsystematic, hasty approaches. There is little relevant research on debugging in block-based programming languages (Kim et al., 2018); therefore, it is the researchers' supposition that participants may have struggled because they did not adopt disciplined, systematic debugging approaches.

The findings of this study run counter to those of Kim et al. (2018) as the PSTs in this study did not struggle with basic programming concepts and PSTs increased their comprehension of basic programming procedures significantly from the pre- to post-test. Existing literature (Falloon, 2016; Kim et al., 2018) in combination with this study's results suggest that while educational robotics can be used to increase PSTs' comprehension of basic procedures in programming, debugging remains a difficult skillset for this population.

Advanced procedures

Participants' scores showed the greatest average increase on this unit. The findings on the Advanced Procedures unit echo those by Kay et al., (2014) in which ISTs' and PSTs' correct answers on the movement programming question of their content knowledge assessment that conceptually aligned to this study's Advanced Procedures unit increased from 40 to 100% after three days of robotics workshops. These data might suggest that participants were comfortable with combining syntactic and semantic skills to solve problems because the Advanced Procedures unit exercised the skills participants needed to solve these questions through mazes. In the individual interviews, the RoboMaze Challenge from the Advanced Procedures unit of instruction was the most-noted fun and enjoyable curriculum element. The researchers' supposition is that the highly enjoyed RoboMaze Challenge contributed toward the highest comprehension improvement since it motivated participants to learn the Advanced Programming unit concepts.

Control structures

Participants' scores increased significantly on the Control Structures unit which indicated that educational robotics had a positive effect on PSTs' comprehension of programming concepts. However, this increase was the second lowest of all units. Supporting the quantitative data, the interviewees commented that the Control Structures concepts were difficult and needed more time dedicated to them in the instruction. This research corroborated Kim et al.'s (2018) findings which indicated that PSTs often struggled with "improperly defined conditionals" (p. 772). This study's findings indicated that PSTs specifically struggled with problems that utilized multiple loops. Evaluating scores across this unit, participants excelled with problems featuring a single loop but struggled with tracing multiple loops in an algorithm. Kim et al. (2018) explained that PSTs incorrectly designed their programs, "omitting loop or other commands that had to be included to complete the program" (p. 772). This study's findings build on those findings by indicating that PSTs had trouble with multiple loops in particular, which suggests that PSTs' struggles may increase as more loops are added to a problem.

Variables

The increase in variable comprehension by participants in this study may be best explained by the visualization and concrete modeling of programming through the actions of the robots. According to Ala-Mutka (2004) recursion, or the use of loops with variables to complete smaller tasks that reiterate to complete a larger task, is a programming concept which can be taught through visualizations “on [a] high level” (p. 8). The Variables unit of instruction included the most advanced programming concepts of the intervention, and correspondingly it had lower pre- and post-test unit scores. While scores increased significantly, these data suggest that participants did not have as deep of a comprehension of variables as other programming concepts. This study confirms Kim et al.’s (2018) findings that PSTs commonly demonstrate errors in defining values of variables while programming robots and Govender and Grayson’s (2008) non-robotics findings that ISTs and PSTs found the concept of variables confusing. Interviewees mentioned that the concept of variables was difficult for them. Overall, these findings suggest that educational robotics can be used to increase PSTs’ comprehension of variables but to a lesser extent than other programming concepts due to the difficulty in obtaining a high-level understanding of relevant concepts.

(RQ2) how and to what extent does educational robotics influence PSTs’ motivation related to programming?

Quantitative findings of this study indicate that robotics positively influences PSTs’ motivation related to programming. Qualitative themes further explain how robotics influence PSTs’ motivation related to programming. The following paragraphs discuss explanations for why participants’ motivation related to programming increased by comparing the qualitative themes with quantitative survey findings.

Intrinsic motivation

This study extends previous findings by pinpointing high intrinsic motivation gains by participants in the areas of interest and enjoyment. While the results of this study are consistent with previous research (Kim et al., 2015; Kucuk & Sisman, 2018) that PSTs perceived robotics to be intrinsically motivating while learning to program, Theme 1 explained that participants experienced increased interest and enjoyment due to the problems they solved. In particular, the challenges that utilized mazes were noted in the interviews to be motivating to participants. It can be logically inferred that challenges that prompted participants to write programs to navigate the robots through the mazes increased participants’ intrinsic motivation. This study’s combined findings paralleled those of Kucuk and Sisman (2018), who found that PSTs considered educational robotics activities and learning by doing to be fun. Authentic problems afford learners opportunities to solve content-specific problems through real-life scenarios (Kopcha et al., 2017). Robotics can be used to demonstrate physical representations of abstract concepts, such as equations (Han, 2013). Theme 1 also explained that participants were interested in the representation of abstract concepts in concrete form through the robotics curriculum, which boosted their intrinsic motivation levels. This finding is supported by Bayman and Mayer’s (1983) study that suggested novice programmers should be given concrete models of programs to build their mental models. Because constructivism centers on the building of abstract knowledge structures in one’s mind through concrete experiences (Bruner, 1996; Piaget, 1973), participants were intrinsically motivated by constructivist processes of representing abstract concepts in concrete form via robotics. This study’s findings support those of Kim et al. (2015, 2018) and Kucuk and Sisman (2018) while also extending their findings by pinpointing high intrinsic motivation gains by participants in the areas of interest and enjoyment when programming robots to solve authentic problems.

Career motivation

The findings of this study echo those of Kim et al. (2015) and also provide new insights to the literature because participants' highest combined pre- and post- survey motivation levels were in the Career Motivation subscale. In Theme 2, participants voiced perspectives that schools and the economy were moving toward technology-rich futures. The large increase for this subscale could be attributed to the intervention's use of lectures about new state standards for K-8 CS and videos showcasing how teachers are implementing CS into instruction. While high pre-survey career motivation indicated that participants were cognizant of the economy's trajectory before they participated in the intervention, they may not have been informed about the relevance and imminence of CS standards for their targeted grade level. This indicates that PSTs may already be motivated to learn programming concepts as a way to be more desirable in the job market and more qualified in their professional practice, and this high career motivation level can be even further increased through educational robotics activities. These findings also add to the literature by noting PSTs' perspectives that learning how to program would provide them with career advantages.

Self-determination

Participants demonstrated the largest increase to their motivation in the subscale of Self-Determination. Because self-determination can be improved through confidence-building (Ryan & Deci, 2020), the participants' building of competence via gradually increasingly difficult content likely contributed to their large self-determination increase. The competence of participants may have been most directly impacted by the achievement of completing the different activities and challenges in the intervention.

Kim et al. (2015) found that PSTs put in more effort when they encountered difficulties while programming robots. According to Kim et al. (2015), one of the methods the PSTs used to solve problems was "seeking help from peers" by "exchanging ideas, questioning, and answering questions in collaborative small groups" (p. 26). Qualitative data from Theme 3 indicated that participants used multiple different CPS strategies (Roschelle & Teasley, 1994) when they encountered difficulty. This study's combined quantitative and qualitative findings of effort and CPS strategies between groups confirm Kim et al.'s (2015) findings that PSTs using robots put in extra effort to solve problems through collaboration. Combined quantitative and qualitative evidence suggests that collaborative educational robotics activities can be used to significantly increase PSTs' self-determination related to programming.

Self-efficacy

Evidence from Theme 4 supported the participants' increased quantitative self-efficacy. This finding parallels the literature. For example, research by Jaipal-Jamani and Angeli (2017) indicated that robotics could improve PSTs' self-efficacy pertaining to programming. Further, Kay et al.'s (2014) findings centered on confidence and found that ISTs' self-efficacy related to learning and teaching programming improved with robotics. Research by Rogerson and Scott (2010) explained that students often exhibit apprehension and fear related to programming, which in turn can cause negative perceptions of programming. Participants' initial lack of confidence in learning programming could be attributed to what Rogerson and Scott (2010) described as “the nature of programming that gives rise to [negative] feelings” (p. 147). Once participants experienced programming through the robots, their fears were diminished, and their confidence improved. Most qualitative data that demonstrated participants’ increased confidence came from their explanations of their improved programming comprehension. As described in Theme 4, participants used words such as “zero” or a “blank slate” to define their initial programming comprehension and self-efficacy.

Motivation to integrate programming into teaching

This study’s findings add depth to the literature while supporting prior findings (Jaipal-Jamani & Angeli, 2017; Kaya et al., 2015; Sisman & Kucuk, 2019). While this study’s findings suggested that participants enjoyed the idea of teaching programming to students and mentioned improved confidence that they can teach the topic, quantitative and qualitative data indicated that their motivation is tempered by uncertainty of programming’s fit within their curriculum. Nearly all participants interviewed explained that they wanted to integrate programming into their future teaching. Theme 5 showed that PSTs’ MTIPIT can be improved through robotics from a level of disinterest to where they are motivated and have devised strategies to integrate programming into future instruction. This study offers new insights into the extent to which PSTs can be motivated to integrate programming into their instruction.

Conclusion

Practical implications

This research provides significant practical implications for PSTEs on delivering instruction and encouraging motivation to learn programming.

Based on the results of this study, PSTEs should carefully sequence concepts when designing programming instruction, dedicating focused instructional time to the concepts of debugging, multiple loops, and variables. For example, PSTEs can gradually increase the difficulty of programming concepts within their units. The programming concepts at the start of instruction should focus on foundational syntactic and semantic concepts that can be utilized and built upon in later units (Bucks, 2010; Soloway & Ehrlich, 1984). PSTs should then be afforded time to apply these programming concepts through activities and challenges which test their problem-solving skills. Strategic programming concepts should next be introduced to students (McGill & Volet, 1997). As indicated in this study’s findings, added emphasis should be placed on debugging, utilizing multiple loops, and variables in programs, concepts PSTs struggled with in this study.

As introduced above, PSTEs can embed authentic problem-solving activities and challenges in programming instruction, especially in the form of writing programs to navigate robots through mazes. Adding authentic problems can be used to increase PSTs’ comprehension of advanced programming concepts and intrinsic motivation. As outlined in the findings of this study, unique mazes can be used as an intrinsically motivating way to scale the difficulty of the problems that PSTs are given at each stage of the instruction.

PSTEs can provide PSTs collaborative problem-solving opportunities while programming. Based on the findings of this study, PSTEs may consider harnessing the power of group interaction to support self-determination by creating special challenges where multiple groups must work together to program their robots to interact to achieve a specific task (e.g., passing a baton between robots in a race or one robot pushing a button to allow another robot to advance in a maze). Then, groups working in collaboration could share ideas and help each other, further promoting group to group collaboration and support of building of self-determination.

Finally, PSTEs should explain contemporary trends in CS education and provide specific integration strategies for the different subjects. The results of this study indicate that PST motivation can be significantly improved when they are informed about current expectations for CS standards relative to their future grade level and subject area. While motivation to learn how to program may increase due to the immediacy and potential impact on their careers, results indicate that PSTs may struggle with identifying exactly how they would implement programming within their instruction. Therefore, pre-made lesson plans and resources for the different subject areas (e.g., English, social studies) should be provided in order to guide PSTs' effective integration of programming and ease any hesitancy they may have.

Limitations and future research

Several limitations of this study can be addressed. (1) A lack of control and experimental groups in this study does limit its generalizability. The ethical notion that all participants must receive the same benefits (Creswell, 2017) limits the research design in this context. (2) The novelty effect of new technologies is a limitation for a short-term intervention (Hanus & Fox, 2015). (3) The sample was limited by the course cap of the class. This population is small and largely homogenous. Therefore, results of this study cannot accurately be generalized to the larger population.

Future research may consider experimental studies to validate the findings and provide generalizable insights. Also, future research could use an updated robotics curriculum with a longer duration and refined design. Further, an investigation into the lasting effects of the intervention on PSTs' long-term memory and motivation is needed to confirm whether the positive effect can be retained. Finally, investigation into how to help PSTs identify effective strategies for integrating programming concepts into their grade and subject area (e.g., linking to standards, aligning concepts) within their methods courses is an area for future research.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Research involving human and animal rights Human participants only.

Informed consent Yes.

References

Ala-Mutka, K. (2004). Problems in learning and teaching programming. *Codewitz Needs Analysis*, 1–13. http://www.cs.tut.fi/~edge/literature_study.pdf

Alimisis, D., Moro, M., Arlegui, J., Pina, A., Stassini, F., & Papanikolaou, K. (2007). Robotics & constructivism in education: The TERECoP project. *EuroLogo*, 1–11. http://users.sch.gr/adamopou/docs/syn_eurologo2007_alimisis.pdf

Alkaria, A., & Alhassan, R. (2017). The effect of in-service training of computer science teachers on scratch programming language skills using an electronic learning platform on programming skills and the attitudes towards teaching programming. *Journal of Education and Training Studies*. https://doi.org/10.11114/jets.v5i11.2608

Arwood, L. (2004). Teaching cell biology to nonscience majors through forensics, or how to design a killer course. *Cell Biology Education, 3*, 131–138.

Bandura, A. (1997). Self-efficacy. *Harvard Mental Health Letter, 13*(9), 4–5.

Bayman, P., & Mayer, R. E. (1983). A diagnosis of beginning programmers' misconceptions of BASIC programming statements. *Communications of the ACM, 26*(9), 677–679.

Black, A. E., & Deci, E. L. (2000). The effects of instructors' autonomy support and students' autonomous motivation on learning organic chemistry: A self-determination theory perspective. *Science Education, 84*(6), 740–756.

Bruner, J. (1996). *The culture of education*. Harvard University Press. https://www.hup.harvard.edu/catalog.php?isbn=9780674179530

Bucks, G. W. (2010). *A phenomenographic study of the ways of understanding conditional and repetition structures in computer programming languages.* https://www.proquest.com/docview/858607918

Burke, Q., Schep, M., & Dalton, T. (2016). *CS for SC: A landmark report on K-12 computer science in South Carolina* (pp. 1–19). National Science Foundation. https://doi.org/10.1145/3017680.3022413

Creswell, J. W. (2017). *Qualitative inquiry and research design: choosing among the five traditions*: Sage.

Creswell, J. W., & Plano Clark, V. L. (2018). *Designing and conducting mixed methods research* (3rd ed.): Sage.

DeVellis, R. F. (2003). *Scale development: theory and applications*. Thousand Oaks, CA: Sage.

El-Hamamsy, L., Chessel-Lazzarotto, F., Bruno, B., Roy, D., Cahlikova, T., Chevalier, M., & Mondada, F. (2020). A computer science and robotics integration model for primary school: Evaluation of a large-scale in-service K-4 teacher-training program. *Education and Information Technologies*. https://doi.org/10.1007/s10639-020-10355-5

Erol, O., & Kurt, A. A. (2017). The effects of teaching programming with scratch on pre-service information technology teachers' motivation and achievement. *Computers in Human Behavior, 77*, 11–18. https://doi.org/10.1016/j.chb.2017.08.017

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using scratch Jnr. On the iPad. *Journal of Computer Assisted Learning, 32*(6), 576–593. https://doi.org/10.1111/jcal.12155

Fegely, A., Winslow, J., Lee, C., & Rubbo, L. J. (2021). The effects of robotics professional development on science and mathematics teaching performance and student achievement in underserved middle schools. *Contemporary Issues in Technology and Teacher Education, 21*(4), 655–679.

Fredricks, J. A., Blumenfeld, P. C., & Paris, A. H. (2004). School engagement: potential of the concept, state of the evidence. *Review of Educational Research, 74*(1), 59–109.

Gleasman, C., & Kim, C. (2020). Pre-service teachers' use of block-based programming and computational thinking to teach elementary mathematics. *Digital Experiences in Mathematics Education, 6*(1), 52–90. https://doi.org/10.1007/s40751-019-00056-1

Glynn, S. M., Brickman, P., Armstrong, N., & Taasoobshirazi, G. (2011). Science motivation questionnaire II: Validation with science majors and nonscience majors. *Journal of Research in Science Teaching, 48*(10), 1159–1176. https://doi.org/10.1002/tea.20442

Grover, S., & Pea, R. (2013). Computational thinking in k-12: A review of the state of the field. *Educational Researcher, 42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

Gupta, S. D. (1960). Point biserial correlation coefficient and its generalization. *Psychometrika, 25*(4), 393–408.

Han, I. (2013). Embodiment: A new perspective for evaluating physicality in learning. *Journal of Educational Computing Research, 49*(1), 41–59. https://doi.org/10.2190/EC.49.1.b

Hanus, M. D., & Fox, J. (2015). Assessing the effects of gamification in the classroom: a longitudinal study on intrinsic motivation, social comparison, satisfaction, effort, and academic performance. *Computers & Education, 80*, 152–161.

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary pre-service teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology, 26*(2), 175–192. https://doi.org/10.1007/s10956-016-9663-z

Kay, J. S., Moss, J. G., Engelman, S., & McKlin, T. (2014). Sneaking in through the back door: Introducing K-12 teachers to robot programming. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*. https://doi.org/10.1145/2538862.2538972
Kaya, E., Newley, A., Deniz, H., Yesilyurt, E., & Newley, P. (2015). Introducing engineering design to a science teaching methods course through educational robotics and exploring changes in views of preservice elementary teachers. *Journal of College Science Teaching*, *47*(2), 66–75.
Kelleher, C., Pausch, R., & Kiesler, S. (2007). Storytelling Alice motivates middle school girls to learn computer programming. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, ACM.
Kim, C., Kim, D., Yuan, J., Hill, R. B., Doshi, P., & Thai, C. N. (2015). Robotics to promote elementary education pre-service teachers' STEM engagement, learning, and teaching. *Computers & Education, 91*, 14–31. https://doi.org/10.1016/j.compedu.2015.08.005
Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science*, *46*(5), 767–787. https://doi.org/10.1007/s11251-018-9453-5.
Kopcha, T. J., McGregor, J., Shin, S., Qian, Y., Choi, J., Hill, R., & Choi, I. (2017). Developing an integrative STEM curriculum for robotics education through educational design research. *Journal of Formative Design in Learning*, *1*(1), 31–44. https://doi.org/10.1007/s41686-017-0005-1.
Kucuk, S., & Sisman, B. (2018). Pre-service teachers' experiences in learning robotics design and programming. *Informatics in Education, 17*(2), 301–320. https://doi.org/10.15388/infedu.2018.16
Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. Sage.
Majherová, J., & Králík, V. (2017). Innovative methods in teaching programming for future informatics teachers. *European Journal of Contemporary Education, 6*(3), 390–401. https://doi.org/10.13187/ejced.2017.3.390
Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014). Computational thinking in K-9 education. Proceedings of the Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference - ITiCSE-WGR '14, 1–29. https://doi.org/10.1145/2713609.2713610
Martin, F. G., Scribner-MacLean, M., Christy, S., Rudnicki, I., Londhe, R., Manning, C., & Goodman, I. F. (2011). Reflections on iCODE: using web technology and hands-on projects to engage urban youth in computer science and engineering. *Autonomous Robots*, *30*(3), 265–280. https://doi.org/10.1007/s10514-011-9218-3.
Marzano, R. J. (2007). *The art and science of teaching*. ASCD.
McGill, T. J., & Volet, S. E. (1997). A conceptual framework for analyzing students' knowledge. *Journal of Research on Computing in Education*, *29*(3), 276–298.
Mertler, C. A. (2017). *Action research: improving schools and empowering educators* (5th ed.). Sage.
Ortiz, A., Bos, B., & Smith, S. (2015). The power of educational robotics as an integrated STEM learning experience in teacher preparation programs. *Journal of College Science Teaching*. https://doi.org/10.2505/4/jcst15_044_05_42
Pallant, J. (2007). *SPSS survival manual:A step by step guide to data analysis using SPSS for Windows* (3rd ed.). McGraw Hill Open University Press.
Piaget, J. (1973). *To understand is to invent*. Basic Books.
Rogerson, C., & Scott, E. (2010). The fear factor: How it affects students learning to program in a tertiary environment. *Journal of Information Technology Education: Research, 9*, 147–171. https://doi.org/10.28945/1183
Roschelle, J., & Teasley, S. D. (1994). The construction of shared knowledge in collaborative problem solving. *NATO ASI Series F Computer and Systems Sciences*, *128*, 69–69.
Ryan, R. M., & Deci, E. L. (2020). Intrinsic and extrinsic motivation from a self-determination theory perspective: definitions, theory, practices, and future directions. *Contemporary Educational Psychology*, *61*, 101860. https://doi.org/10.1016/j.cedpsych.2020.101860.
Saldaña, J. (2016). *The coding manual for qualitative researchers* (3rd ed.). Sage.
Salkind, N. J. (2010). *Encyclopedia of research design (Vols. 1 – 0)*. Sage. https://doi.org/10.4135/9781412961288
Sentance, S., & Csizmadia, A. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies*. https://doi.org/10.1007/s10639-016-9482-0
Sisman, B., & Kucuk, S. (2019). An educational robotics course: Examination of educational potentials and pre-service teachers' experiences. *International Journal of Research in Education and Science, 5*(1), 510–531.
Soloway, E., & Ehrlich, K. (1984). Empirical studies of programming knowledge. *IEEE Transactions on Software Engineering*, *10*(5), 595–609.

Sullivan, F., & Moriarty, M. (2009). Robotics and discover learning: Pedagogical beliefs, teacher practice, and technology integration. *Journal of Technology and Teacher Education, 17*, 109–142. http://people.umass.edu/florence/jtate.pdf%5Cnpapers2://publication/uuid/284416E1-4D1B-48FA-8F07-583B7FCCFA47

Thompson, G. (2008). Beneath the apathy. *Educational Leadership*, *65*(6), 50–54.

Weintrop, D., & Wilensky, U. (2017). Comparing block-based and text-based programming in high school computer science classrooms. *ACM Transactions on Computing Education, 18*(1), 1–25. https://doi.org/10.1145/3089799

Yuan, J., Kim, C., Vasconcelos, L., Shin, M. Y., Gleasman, C., & Umutlu, D. (2022). Preservice elementary teachers' engineering design during a robotics project. *Contemporary Issues in Technology and Teacher Education.* https://citejournal.org/volume-22/issue-1-22/science/preservice-elementary-teachers-engineering-design-during-a-robotics-project/

Yukselturk, E., & Altiok, S. (2017). An investigation of the effects of programming with scratch on the preservice IT teachers' self-efficacy perceptions and attitudes towards computer programming. *British Journal of Educational Technology, 48*(3), 789–801. https://doi.org/10.1111/bjet.12453

Alex Fegely Assistant Professor of Instructional Technology at Coastal Carolina University. His research focuses on computer science education, diversity in edtech, and the use of XR in education.

Hengtao Tang Assistant Professor in the Department of Educational Studies at the University of South Carolina. His research interests include learning analytics, self-regulated learning, STEM education, and open educational resources.