



A mixed-method cluster analysis of physical computing and robotics integration in middle-grade math lesson plans

Shenghua Zha^{a,*}, Yi Jin^b, Rebecca Wheeler^a, Erin Bosarge^a

^a Department of Counseling and Instructional Sciences, University of South Alabama, Mobile, AL, 36688, USA

^b School of Instructional Technology and Innovation, Kennesaw State University, Kennesaw, 30144, Georgia

ARTICLE INFO

Keywords:

Applications in subject areas
Improving classroom teaching
Secondary education

ABSTRACT

This study analyzed 281 lesson plans collected from the producers' websites of 12 educational physical computing and robotics (ePCR) devices. We extracted and coded five variables from each lesson. They were ePCR functionality, coding skills, computational thinking skills, math knowledge, and activity design. First, a two-step cluster analysis was administered to find how three ePCR-related knowledge: ePCR functionality, coding skills, and computational thinking skills, were integrated to teach students ePCR technology in middle-grade math lessons. Results showed three types of lesson plans, including lessons to use basic ePCR functionality to teach students lower-level CT skills, lessons to teach students basic to intermediate coding skills, and lessons to use the technology at the advanced level. Next, we applied the Technological Pedagogical Content Knowledge (TPACK) framework and conducted a second two-step cluster analysis to identify how the technology (ePCR technology), content (math knowledge), and pedagogy (activity design) were integrated into those lesson plans. Results suggested ten clusters of lesson plans with distinct features. We summarized those ten lesson clusters into five categories: 1) ePCR technology lessons, 2) transdisciplinary problem-based learning lessons, 3) technology-assisted lessons, 4) lessons without real-world connections, and 5) lessons integrating middle-grade math learning into ePCR projects. Implications for educators and researchers were discussed at the end of the article.

1. Introduction

Physical computing and robotics devices (PCR) are interactive systems that sense and respond to the real world through computer programming (Blikstein, 2013). The past decades observed new PCRs being developed and adopted in K-12 classrooms. These educational PCRs (ePCRs) have demonstrated their power of helping young learners to improve problem-solving skills and knowledge in computer science and engineering (Kopcha et al., 2020; O'Sullivan & Igoe, 2004; Psycharis & Kallia, 2017). According to a recent market research report, ePCRs will continue to grow at a double-digit percentage through 2028 (Emergen Research, 2022).

Despite the positive impact shown in studies, there is a lack of ePCR education opportunities for middle-school students in the U.S. In 2021, only seventeen states offered computer science (CS) education for middle-school students, among which only 3.9% of students were enrolled in one foundational CS course (Hendrickson et al., 2021) Code.org. Therefore, CS researchers and educators proposed infusing ePCR in the middle-school core curriculum to broaden participation (Malyn-Smith et al., 2018; Ntemngwa & Oliver, 2018; Pollock, Mouza, Guidry, & Pusecker, 2019, February).

* Corresponding author.

E-mail address: shzha@southalabama.edu (S. Zha).

Prior studies have summarized ePCR or CS integration in the K-12 core curriculum from multiple perspectives (Waterman et al., 2020; Zhong & Xia, 2020). Results of these studies indicate that effective ePCR-integration requires an understanding of different types of knowledge. In addition, how these different types of knowledge work together is critical for successful integration (Cabrerá, 2019; Ketelhut et al., 2019).

The need for a high-quality ePCR-integrated curriculum warrants research on exploring existing ePCR lessons. Thus, we decided to focus on analyzing some existing ePCR lessons designed for middle school mathematics integration because math classes observed more ePCR integration than other subjects (Francis & Davis, 2018; Sáez-López et al., 2019; Saqr et al., 2021). In this study, we collected and analyzed 281 publicly available lesson plans from the major ePCR websites. The lesson plans on those websites were created mainly by the key stakeholders of ePCR education: teachers and ePCR technology providers. Our study aimed to identify types of ePCR integration in middle-grade math lessons. The research questions were:

1. How were different aspects of ePCR-related content and technology used together in middle-grade math lessons?
2. How were ePCR technology, pedagogy, and content used together in middle-grade computing-integrated lessons?

1.1. Types of knowledge in CS or ePCR integration

The Technological Pedagogical Content Knowledge (TPACK) framework was initially proposed in 2006 to depict three types of knowledge in effective technology-integrated learning: technology, pedagogy, and content knowledge (Mishra & Koehler, 2006). Since then, several TPACK-based frameworks have been developed to explain the different types of knowledge in CS or ePCR integration in K-16 curriculum. For example, Angeli et al. (2016) suggested that CS-related knowledge, such as computational thinking (CT), should be considered the content knowledge and used together with the pedagogy and technology knowledge to develop a quality CS stand-alone course. CT refers to a problem-solving skill set that not only computer scientists typically use in their jobs but also the general public use to solve real-world problems (Denning, 2017; Wing, 2008). However, Mouza et al. (2017) thought that CS-related knowledge, like CT concepts, tools, and applications, should be considered technology knowledge. Teachers shall be competent in using the CS technology, pedagogy, and content knowledge to develop CS-integrated courses. Zha et al. (2020) conducted a proof-of-concept study in a high-school Spanish class and proposed a Technological Pedagogical Content CT (TPC²T) framework. This framework considered content knowledge and CT skills as two types of content knowledge that teachers should consider in designing CT-integrated lessons. In addition, the authors suggested that tools used to help students apply CT skills should be considered technology. Thus, teachers need to master the CS-related content and technology knowledge, the K-12 core curriculum knowledge, pedagogy knowledge, and their intersectional knowledge to design and deliver a quality CS-integrated course.

The deviations of the TPACK framework in K-12 CS education and teacher education suggest a need to update the types of knowledge to better illustrate the CS-integrated instruction where CS concepts and technology may not be appropriate to be considered either technology or content knowledge. In the following part of this section, we summarized five types of TPACK knowledge found in the literature review of ePCR K-12 education: ePCR hardware functionality, coding skills, CT skills, math as another content knowledge, and pedagogy.

It is essential to establish students' technology efficacy at the beginning of an ePCR-integrated curriculum (Rich et al., 2021b). Technology-related skills in ePCR classes typically include understanding hardware functionality, programming skills, and CT skills. Slangen et al. (2011) explained that robots functioned on the sensor-reasoning-action (SRA) loop if they were coded (reasoning) to use the sensors (sensor) to interact with real-world situations (action). If a robot ran on the reasoning-action (RA) loop without using sensors or other input, it was in the automatic mode and irresponsive to the real-time changes of the physical world. Researchers found that novice middle-school students had fewer problems reasoning along an ePCR's RA loop than the SRA loop (Slangen et al., 2011; Sullivan & Heffernan, 2016). According to Piaget's cognitive development theory (Piaget, 1952), it could be because middle-school students were in the process of transitioning to or just starting the formal operational stage, in which they began to develop abstract reasoning skills. Therefore, they had fewer issues when reasoning concrete actions performed by ePCRs than reasoning and associating the abstract sensor input with ePCR actions.

Coding and programming skills were a direct way for students to use ePCR functionality. According to the K-12 CS standards by the national Computer Science Teachers Association (CSTA), students are expected to achieve a set of coding skills at each grade level, ranging from simple coding (e.g., sequence, loops, and conditionals) to advanced coding (e.g., compound conditionals, variables, and functions) (Computer Science Teachers Association, 2017). Various coding tools have been developed in consideration of young students' cognitive development (Ching et al., 2018). For example, tangible coding tools, such as code-a-pillar, were adopted in PreK to lower elementary classrooms (Kazakoff et al., 2013; Taylor, 2018). Visual block-based coding programs, such as Scratch, have been widely used in elementary and middle-school classrooms (Grover et al., 2016; Sáez-López et al., 2016). High-school students began to use text-based programming, like Python (Weintrop & Wilensky, 2017).

A common misconception in K-12 CS education was to equate CT with coding skills (Webb et al., 2017). In fact, CT skills used in coding, e.g., abstraction, algorithmic thinking, and pattern recognition, were only one subset of CT skills as identified in national and state standards and frameworks. A broad definition of CT also included the skills of working with data, understanding systems, and creating computer models and simulations, which required students to learn beyond coding (Lee et al., 2014; Malyn-Smith et al., 2018).

Math was closely related to the CS discipline. For example, algorithmic thinking, one CT skill, was frequently used by computer scientists to solve problems step by step. Mathematicians confirmed that they often used this skill set to solve math problems,

communicate math, and identify math mistakes (Lockwood et al., 2016). As a result, math classes observed more ePCR integration than other subject courses. Studies have shown that students' math knowledge, such as understanding of coordinates, integers, and negative numbers, improved after applying the math concepts to design, code, and debug robotics tasks (Francis & Davis, 2018; Sáez-López et al., 2019). Zhong and Xia (2020) reviewed twenty studies of robotics integration in K-16 math classes and summarized them into three categories based on how robotics technology was used in math learning. The most popular integration was that students learned math by interacting with the robots. Students maneuvered the robots and played the games. The math learning occurred when students

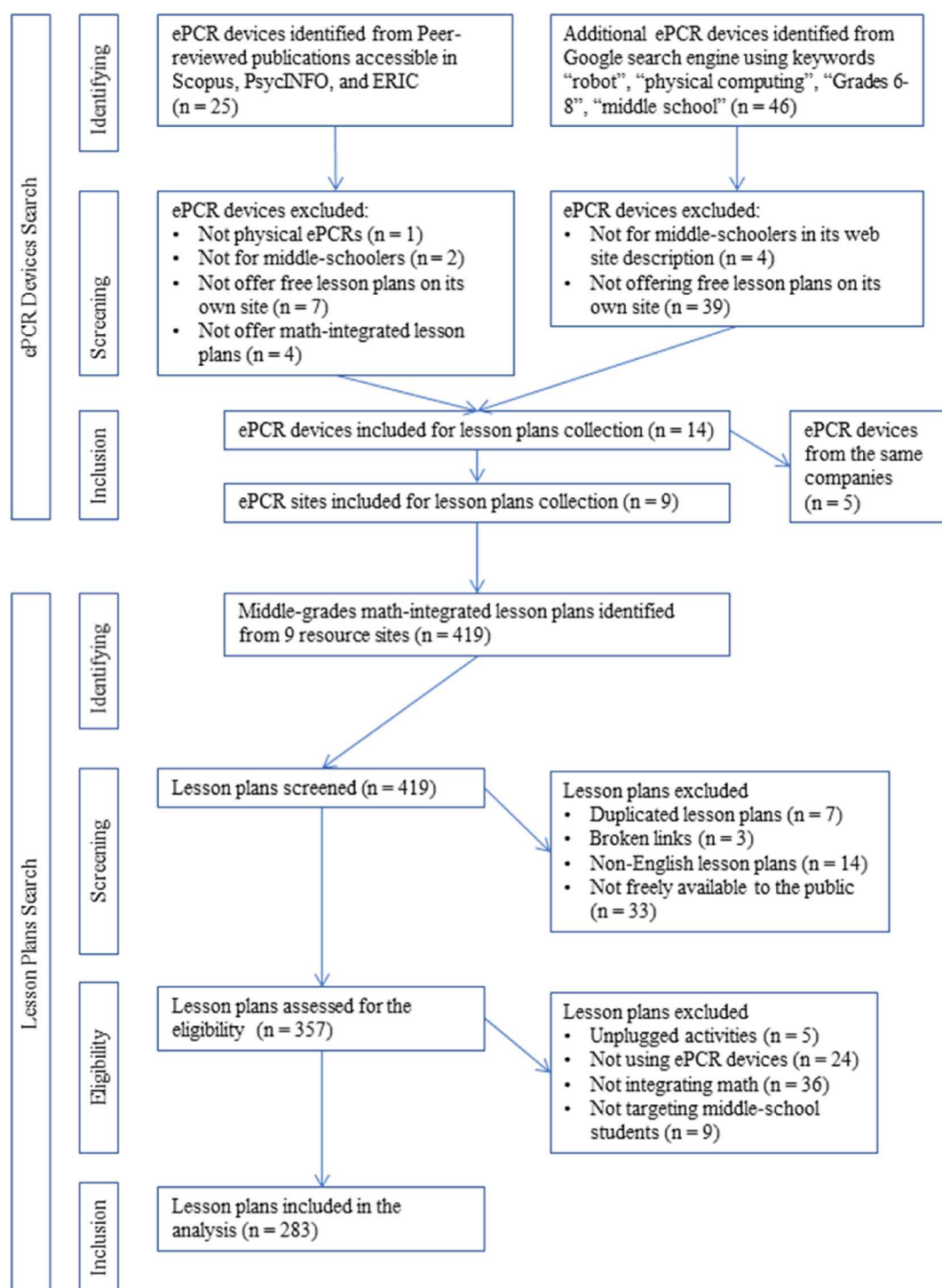


Fig. 1. Flowchart of lesson plan selection process.

followed the rules and executed the gaming tasks. In the second category, students learned the math concepts by building and coding the robots. In the third category, students learned math concepts through coding.

Pedagogical methods were examined in prior studies of CT or ePCR integration. They included learning progression methods, such as the use-modify-create method (Lee et al., 2011) and pair programming (Kanika et al., 2020), and design methods, such as game design (Leonard et al., 2016), project-based learning (Barak & Assal, 2016; Gutierrez et al., 2018), and transdisciplinary problem-based learning approach (Century et al., 2020). In recent years, the culturally responsive design was introduced in CT or ePCR classes to engage students from diverse backgrounds (Leonard et al., 2019; Morales-Chicas et al., 2019). Instructions and activities were designed to connect students' learning with their lived experiences, heritage culture, social culture, or communities. Results showed that the culturally-responsive design effectively enhanced students' self-efficacy, interest, and knowledge of CT (Leonard et al., 2016, 2019).

2. Methods

2.1. Lesson plan search and inclusion

We adopted the identification-screening-inclusion procedure in the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) design guidelines (Page et al., 2021) to find ePCR lesson plans (Fig. 1). At first, we administered a comprehensive search to identify ePCR devices commonly used in secondary education settings from two strands of resources: 1) peer-reviewed publications published between 2015 and 2021 and accessible in the databases of SCOPUS, PsychINFO, and ERIC and 2) popular K-12 education websites, such as eduporium.com and weareteachers.com. We used the search terms "robot", "robotics", and "physical computing" in the device search. In addition, we used the ePCR device information collected from Coding is Elementary, a site listing current tools used in K-12 CS education (Rich et al., 2021a). At the end of the device search, we identified 12 ePCR devices for lesson plan search.

Next, we conducted another search to locate the lesson plans written in English and publicly available before July 2021. We mainly searched the ePCR device producer websites. We also searched popular coding education websites, such as code.org and csfirst.withgoogle.com, though we did not find ePCR-integrated math lesson plans.

The initial search rendered 419 lesson plans. At the end of the screening and eligibility assessment, 283 lesson plans were included in the further analysis. They all met the following criteria:

1. The lesson plans were complete, not excerpts.
2. The target students were middle-school students.
3. They explicitly and intentionally required students to learn or use specific math and ePCR knowledge and skills.
4. They included the use of free and publicly available coding languages or environments.
5. They included the use of physical computing devices and/or robotics.
6. Each lesson plan included a complete learning practice or task lasting at least 30 min. Many lessons took more than one class session to complete. If a curriculum idea had one consistent learning goal but required several lessons to complete, we counted them as one lesson plan.

2.2. Data preparation

Four researchers participated in extracting and coding variables from each lesson plan. The two leading researchers extracted and coded the five variables to be used in the further analysis: ePCR functionality, coding skills, CT skills, math knowledge, and activity design. We used the *a priori* coding method, coding each variable based on our literature review (Stemler, 2000). The student researchers were responsible for coding the background information, such as the producer names and the duration of lesson plans.

The coding process was iterative (Strauss & Corbin, 1998). We started with sample coding and randomly selected a list of lessons for the initial discussion and coding. It resulted in a coding book for future reference. After that, the two leading researchers split and selected lesson plans for the individual coding. In the next round, we reviewed each other's coding of each lesson plan independently. Then, the whole team met several times to discuss and resolve any disagreements until our inter-rater reliability was 100%. The coding book was updated accordingly. We also reviewed and corrected previously coded lesson plans together if any changes occurred in the relevant codes.

2.3. Variables

We used each lesson plan as a case and extracted and coded the following variables: ePCR functionality, coding skills, CT skills, math knowledge, and activity design. For lesson plans that did not provide information on those variables, we treated them as missing information. Thus, two lesson plans were excluded from further analysis.

ePCR functionality refers to an ePCR's input-reasoning-action (IRA) functions used in a lesson plan. We adapted Slangen et al.'s (2011) interpretation of the ePCR's SRA functionality. We replaced the word *sensor* with *input* as some ePCR lessons used human-controlled input, like keyboards and buttons, instead of sensors. Thus, the full SRA loop became the IRA loop in our study. A lesson plan was coded as 1 if students were only required to reason an ePCR automatic process along the RA loop; 2 if students needed to maneuver an ePCR along the IRA loop but the action was displayed on the computer screen instead of through the device; and 3 if students needed to reason the full IRA loop and the action was executed by the ePCR devices.

Coding skills refer to the highest level of coding algorithms used in a lesson plan. We followed the progression map in the K-12 Computer Science Standards (Computer Science Teachers Association, 2017). A lesson plan was coded as 0 if the coding blocks were pre-loaded onto ePCRs or if the coding samples or cheat sheets were not provided on their lesson plans; 1 if students only used the sequence and simple loop coding blocks; 2 if students used conditional logic in the coding; and 3 if students used advanced algorithms, such as function and arrays.

CT skills refer to the highest level of CT skills that students were expected to demonstrate in a lesson plan. We followed the definitions of traditional CT skills (Malyn-Smith et al., 2018) and coded a lesson plan as 1 if students only used basic skills typically done in coding, such as abstraction, decomposition, pattern recognition, testing, and debugging; 2 if students were required to work with data; 3 if students needed to understand systems or create computational models; and 4 if students were likely to use all of those CT skills.

Math knowledge focuses on how math knowledge was used in a lesson. This variable was coded based on the Common Core math standards (Common core state standards initiative, 2010) and types of ePCR integration identified in Zhong and Xia's (2020) systematic review. A lesson was coded as 0 if no middle-grade math knowledge was used. In this case, students may use math knowledge learned before the middle grades, such as addition and subtraction. A lesson was coded as 1 if middle-grade math knowledge was used outside ePCR planning, coding, assembling, or testing. For example, students may measure and calculate data generated by a working ePCR. A lesson was coded as 2 if middle-grade math knowledge was used in the ePCR planning, coding, assembling, or testing. A lesson was coded as 3 if middle-grade math knowledge may be used at any stage of the projects. For example, students were given a real-world scenario or problem. They needed to develop solutions using ePCR and there was no single correct solution.

Activity design includes two types: instructional practice and projects (Barak & Assal, 2016; Leonard et al., 2019; Morales-Chicas et al., 2019). Instructional practices were used for students to practice math and/or ePCR knowledge. This type of activity was not bound to students' real-world experiences. An example was to require students to code and run ePCRs in geometric shapes. Lesson plans only including instructional practices were coded as 0. On the other hand, projects were culturally and/or socially related to students, such as games, movies, or their family culture (Brown, 2017; Eglash et al., 2013). Students were expected to complete culturally or socially related tasks or solve those problems using math and/or ePCR knowledge. Examples of projects included Harry Potter finding the rest of Horcrux and autonomous wheeled robots in warehouses. Lesson plans with projects were coded as 1.

2.4. Data analysis

After the five variables were coded with numerical values, the dataset was imported into the IBM SPSS version 27 for cluster analysis. We used the two-step cluster analysis as it accepted categorical data and identified the optimal number of clusters automatically (Bacher et al., 2004; Chiu et al., 2001). This analysis began with the pre-clustering by using a sequential approach to develop sub-clusters. Then the sub-clusters were categorized into major clusters using the hierarchical clustering approach. The distance in the two-step cluster analysis was measured by the Log-Likelihood ratio.

We conducted two cluster analyses. The first analysis focused on categorizing the ePCR technology. The cluster numbers generated from the first analysis were included in the variable of ePCR technology, which was used together with the variables of math knowledge and activities in the second analysis. The second analysis aimed to find the types of ePCR-integrated lesson designs for middle-grade math classes.

Table 1
Background information of lesson plans.

ePCR Devices	Number of Lesson Plans
Cue	5
Finch	13
Hummingbird	11
LEGO Education SPIKE Prime Set	27
LEGO MINDSTORMS Education EV3 Core Set	42
LittleBit Core Set	19
LittleBits RVR Topper Kit	4
LittleBits STEAM Student Set	7
MakeyMakey	17
Micro:bit	13
Ozobot Bit + Evo	49
Sphero	74
Coding Languages used by ePCR Devices	Number of Lesson Plans
Designed only for specific devices	227
Used by multiple devices	54
Duration of Lessons	Number of Lesson Plans
Within 60 min	136
Between 61 and 120 min	79
More than 120 min	14
Not specified	52
Authors of Lesson Plans	Number of Lesson Plans
Individual authors identified	119
Individual authors not identified	162

We used two methods to validate the results of the cluster analyses. One was the Silhouette measure of cohesion and separation (Rousseeuw, 1987). A Silhouette value ranged from -1 to 1 , with a high value representing an object's cohesion to its own cluster and dissimilarity to other clusters. The other validation method was Pearson's Chi-square test (Norusis, 2008). It was used to verify whether the clusters varied significantly across the variables.

3. Results

Overall, 281 lesson plans were included in the cluster analyses as two lesson plans had missing information on our interested variables. The selected lesson plans discussed using 12 ePCR devices in middle-grade math lessons (Table 1). Among them, 81% of lessons used the coding languages developed by the ePCR companies. Nineteen percent of lessons used free coding languages or environments. Forty-eight percent of lesson plans could be completed in one 60-min class session, and 33% took more than one class session to finish. We also found that 19% of lesson plans did not clarify the duration. Forty-two percent of lessons showed individual contributors' names or usernames, which showed evidence of real classroom implementations with videos in the lesson plans. The remaining 58% of lesson plans did not show the individual contributors' names.

3.1. Clusters of ePCR technology

We used three ePCR-related variables: ePCR functionality, coding skills, and CT skills in the first cluster analysis to identify the types of combinations of ePCR technology in middle-grade math lessons. Results suggested that three clusters were the optimal number to classify ePCR technology (Fig. 2). The Silhouette measure of cohesion and separation was .4, which suggested the cluster quality was acceptable (Rousseeuw, 1987). The clustering was also supported by the results of the Pearson Chi-square tests. The three clusters varied significantly by ePCR functionality ($\chi^2(4, N = 281) = 278.58, p < .001$), coding skills ($\chi^2(6, N = 281) = 259.23, p < .001$), and CT skills ($\chi^2(6, N = 281) = 90.07, p < .001$) used in the lessons.

Cluster 1 ($N = 95$) lesson plans were featured using the basic RA functions of ePCR to develop students' lower-level CT skills (e.g., the Navigating Around Town with Coordinates lesson at <https://classroom.ozobot.com/lessons/lnz5kGvZ4vQcK3kjAXCOMaKwy7>). Ninety-four percent of the lesson plans explored ePCR's RA loop. Only 6% required students to use the IRA loop, whose purpose was to collect and analyze data. Twenty-nine percent of lesson plans did not require students to code, and 26% taught students advanced coding skills. Forty-five percent of lesson plans did not provide coding samples or cheat sheets though coding was a part of the learning activities. The CT skills varied in this cluster, with 42% on basic CT skills, 47% on working with data, 6% on systems and computational models, and 5% possibly using CT skills at any level.

Cluster 2 ($N = 79$) lesson plans were featured using the technology at the advanced level (e.g., the Training Camp 1: Driving Around lesson at <https://education.lego.com/en-us/lessons/prime-competition-ready/training-camp-1-driving-around#lesson-plan>). Students explored ePCR's functionality using the full IRA loop in all lessons. Sixty-two percent of lessons taught students advanced CT skills, such as creating computational models. Thirty-five percent taught basic CT skills, while 3% taught working with data. Most of the lessons focused on intermediate (56%) and advanced (15%) coding skills, while 16% taught students fundamental coding. About 13% of lesson plans did not share the coding samples or cheat sheets though coding was still required.

Cluster 3 ($N = 107$) lesson plans featured developing novice students' basic-to-intermediate coding skills (e.g., the Fitness Friend lesson at <https://microbit.org/lessons/fitness-friend/>). Eighty-two percent of lessons taught students to explore ePCR's RA loop, while 18% of lessons required students to code along the IRA loop with actions displayed on computer screens. Coding skills in most lessons

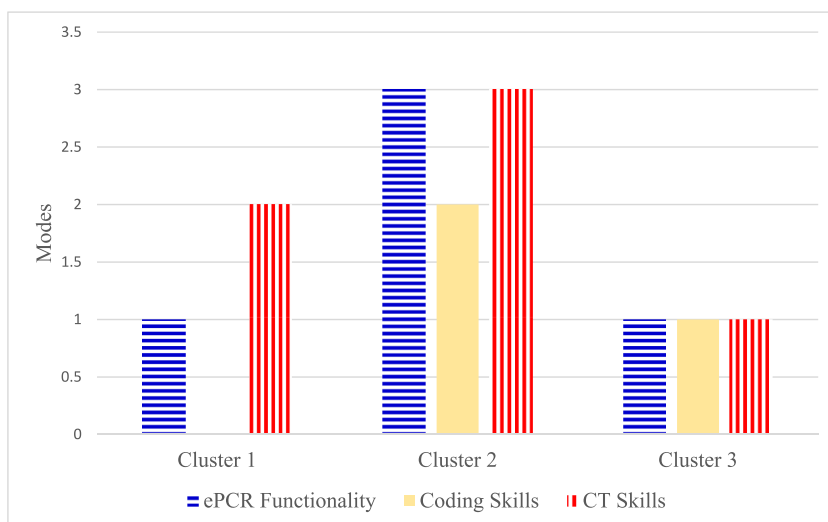


Fig. 2. Clusters of ePCR technology.

ranged from fundamental to intermediate skills, with 67% on sequence and simple loop coding and 31% on conditional logic coding. Only 2% of lesson plans used advanced coding. CT skills varied in the lessons, with 54% focusing on basic CT skills, 16% on working with data, and 30% on creating computational models.

3.2. Clusters of ePCR-integrated lesson design

The second cluster analysis was conducted to identify the types of ePCR-integrated math lesson design. The cluster variable of ePCR technology was included together with the variables of math knowledge and activity design. Results showed that ten was the optimal cluster number (Fig. 3). The Silhouette measure of cohesion and separation was .8, which indicated the clustering quality was good (Rousseeuw, 1987). Results of the Pearson Chi-square tests also showed the good quality of the clusters. The ten clusters differed significantly by their ePCR technology ($\chi^2(18, N = 280) = 481.49, p < .001$), math knowledge ($\chi^2(27, N = 280) = 524.79, p < .001$), and activity design ($\chi^2(9, N = 280) = 280.00, p < .001$) used in the lessons.

In the following part of this section, we grouped nine clusters into superior categories based on their commonalities and discussed the clusters within each category. We also discussed two strands in Clusters 3 based on our review of the lesson plans.

3.3. ePCR technology lessons

Clusters 4 ($N = 33$), 7 ($N = 18$), and 9 ($N = 33$) lessons aimed to improve students' competence in ePCR technology by engaging them in projects with real-world connections. Students did use math knowledge in those lessons. However, the knowledge was at the elementary grade levels.

The differences between the three clusters were as follows: Ninety-seven percent of the lessons in Cluster 4 taught students introductory or intermediate coding skills (e.g., the Curved Move lesson at <https://education.lego.com/en-us/lessons/ev3-tutorials/stop-at-line>), with only 1 lesson on advanced coding. All lessons in Cluster 7 used ePCR's RA functionality (e.g., the Code a Tree lesson at <https://edu.sphero.com/cwists/preview/7070x>). In this cluster, 78% of lessons did not provide coding samples, and 22% used advanced coding skills. All lessons in Cluster 9 used ePCR's IRA functionality (e.g., the Micro:bit Obstacle lesson at <https://classroom.littlebits.com/lessons/microbit-obstacles>). More than half (67%) of lessons used the conditional blocks, with 8% on basic coding blocks and 15% on advanced coding blocks. Unlike Clusters 4 and 7, which taught CT skills at various levels, lessons in Cluster 9 did not teach students to work with data (e.g., 48% on algorithms, 52% on creating computational models).

3.3.1. Transdisciplinary problem-based learning lessons & technology-assisted lessons

Cluster 3 ($N = 26$) lessons included two strands of lessons worthy of further division. One strand focused on transdisciplinary problem-solving. It had 54% of lesson plans where their projects mimicked real-world problems and required students to design, develop, and assess math solutions (e.g., the Super Smiles with Sphero Challenge lesson at <https://edu.sphero.com/cwists/preview/42811x>). ePCR technology and content were parts of the solution. Students often needed to use other disciplinary knowledge, such as science and economy, to find a solution. There were no single solutions to the technology or math problems, and hence, no coding samples or cheat sheets were provided in the lessons.

ePCR technology was used to assist students' math learning in the other strand of this cluster. It contained 46% of lessons where math learning occurred outside ePCR planning, coding, assembling, or testing (e.g., A Simplified Epidemic Experiment lesson at <https://classroom.ozobot.com/lessons/IndqqfKgixTES10NALAa5bCAOB>). In some lessons ($n = 6$), students did not code, and the code

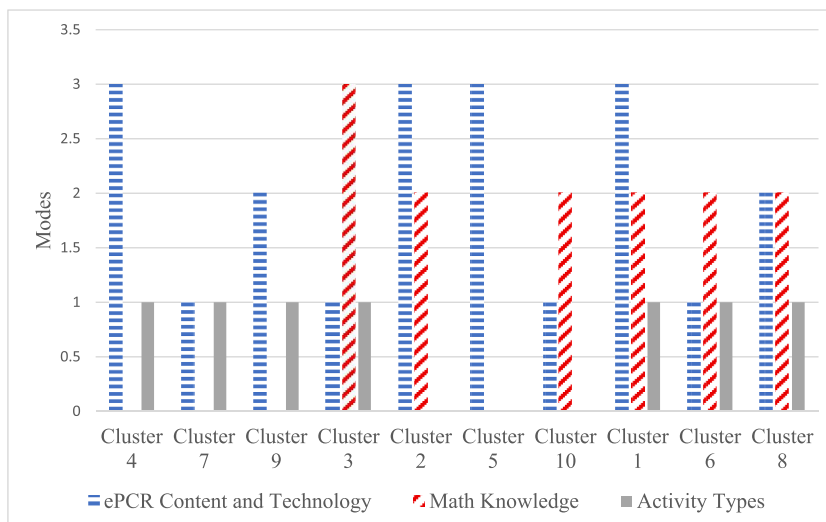


Fig. 3. Clusters of ePCR-integrated lesson design.

was pre-loaded onto ePCRs. In this strand, 92% of lessons used lower-level CT skills, such as working with coding algorithms (25%) and data (67%), with only 1 lesson on creating computational models.

3.3.2. Lessons without real-world connections

Lessons in Clusters 2 ($N = 23$), 5 ($N = 17$), and 10 ($N = 31$) only had instructional practices for students to use their math and ePCR knowledge. Those instructional practices were not embedded in students' cultural or social backgrounds. Meanwhile, most of the lessons in the three clusters taught students lower-level CT skills, such as working with algorithms and data, with only 1 lesson from each cluster on advanced CT skills.

All Cluster 2 (e.g., the Measuring Distance lesson at <https://www.birdbraintechnologies.com/finch/projects/measuring-distance>) and 58% of Cluster-10 lessons embedded math practices in the planning, coding, assembling, and testing ePCR. However, Cluster 5 lessons did not. Fifty-nine percent of cluster 5 lessons used elementary-level math knowledge during the ePCR planning, coding, assembling, or testing, while 41% of the lessons integrated middle-grades math knowledge outside these processes (e.g., the Coding Multiplication Fact Families lesson at <https://edu.sphero.com/cwists/preview/20222x>). Clusters 2 and 5 lessons used basic-to-intermediate coding skills. However, 32% of Cluster 10 lessons did not require students to code, 35% did not provide coding samples, and 33% taught students to use advanced coding skills (e.g., The Euclidean Algorithm lesson at <https://classroom.ozobot.com/lessons/lnhmCghQrvTDe0z7b1tCMd4gPD>).

3.3.3. Lessons integrating middle-grade math learning into ePCR projects

Clusters 1 ($N = 38$), 6 ($N = 29$), and 8 ($N = 32$) lessons engaged students in middle-grade math learning projects while they planned, coded, assembled, or tested ePCRs. This category included the largest number of collected lesson plans.

The difference between the three clusters fell in the ePCR technology. Cluster 1 lessons were represented using entry-level technology (e.g., the Mission Ready lesson at <https://education.lego.com/en-us/lessons/prime-competition-ready/mission-ready>). They taught students the fundamental to intermediate coding skills. About 93% of lessons required students to reason an ePCR's RA loop, while only 7% of lessons required students to code along the IRA loop with actions displayed on computer screens. Cluster 6 lessons were featured using ePCR's RA functionality, with only 1 lesson on the IRA functionality (e.g., the Sketching Algorithmic Artwork lesson at <https://portal.makewonder.com/>). Contrary to the other two clusters, Cluster 8 lessons featured advanced ePCR technology (e.g., the Creating Waves with a Gear Motor lesson at <https://learn.birdbraintechnologies.com/hummingbirdduo/projects/creating-waves-with-a-gear-motor/>). All lessons used ePCR's IRA functionality. Eighty-one percent of the lessons taught students advanced CT skills, and 19% taught basic CT skills.

4. Limitations

Our clusters of lesson plans are not inclusive for several reasons. First, many ePCR producers did not provide free and public lesson plans. Thus, our selection of lesson plans was limited to those offering public and open access. Second, our study is based on the affordance of current technology and devices, which may change when new functionality emerges. Third, the TPACK knowledge in our study does not cover all perspectives on ePCR integration. For example, the pedagogy in our study only focused on the design of the learning activities. It did not include the delivery or classroom management strategies.

The existing cluster analysis was not intelligent enough to handle the qualitative data automatically and accurately. We coded the five categorical variables based on the literature review before administering the cluster analyses. Although most of the results were self-explanatory, Cluster 3 in the lesson design analysis needed further human intervention to be meaningful. Therefore, we call for more studies with rigorous methods to investigate the types of ePCR integration in K-12 core subject areas.

5. Discussion & implications

The results of this study prompt researchers and teachers to revisit the TPACK framework for CS integration in the K-12 core curriculum. First, boundaries between the content and technology become blurred, as suggested by Archambault and Crippen (2009). Only a small percentage (4%) of our collected lesson plans used CS technology as a medium to facilitate students' learning in the content area concepts. Students were technology users in those lessons. Codes were pre-loaded onto ePCR devices. Devices moved around or executed other pre-programmed actions to help students understand mathematical concepts or record numerical data. Meanwhile, 96% of lesson plans in our study enabled students to be creators. They planned, coded, and tested ePCRs to generate solutions. CS technology played an equally important role as math knowledge in the lessons, which blurred the border between content and technology. Second, teachers' technology knowledge, TK, of how different parts of technology (e.g., hardware and software) influence and constraint each other is critical in successful CS integration. For example, results of the ePCR technology clusters showed that lessons utilizing ePCR's full IRA functionality generally required students to use advanced coding skills. A few lessons taught students using the basic coding to perform ePCR's full IRA functionality. It was because the coding blocks in these lessons were simplified. Students did not need to use advanced algorithms to reason for sensor-based actions. Thus, teachers need to understand the interplay of different technology before they are able to design a quality CS-integrated lesson. Overall, there is a great need for teachers to develop various domains of their TPACK to design and implement a quality CS-integrated lesson. Scholarships on cultivating teachers' TPACK and best practices are needed.

The clusters of ePCR technology debunk the myth that equates coding skills with CT skills (Webb et al., 2017). First, we do not observe a directional relationship between coding and CT skills. For example, lesson plans in Cluster 3 teach students to use simple

sequence and loop coding blocks to create computational models, like simulating a solar-powered Mars rover. We also find lesson plans in Cluster 2 that teach advanced coding skills, like functions, to move ePCRs in geometric patterns. Second, CT skills can be acquired outside coding. The strand of technology-assisted lessons augments students' math learning experiences without them coding. In those lessons, coding is pre-loaded onto the ePCR devices. ePCR technology works like learning technology and assists students in working on real data, such as calculating the speed of ePCRs.

The highest level of math integration is embedded in learning activities with real-world connections, as described in the first strand of section 3.2.2. We adopt the name suggested by [Century et al. \(2020\)](#) and call them transdisciplinary problem-based learning lessons. These lessons ask students to work in small groups and solve real-world problems, which requires the applications of math and ePCR knowledge, possibly at any problem-solving stages. An example lesson is designing and building a city and then having a coded ePCR move to a specific part of the city to accomplish tasks. This lesson requires students to use their rulers and protractors to design a city with facilities, calculate the distance and angles, and code and maneuver the ePCR to complete tasks. On the other hand, the highest level of integration is not identified in the lesson plans without real-world connections. Likewise, we do not find the latter lesson plans developing students' advanced CT skills.

5.1. Implications for teachers

The clusters of lesson designs indicate a need to share with teachers different types of integration and their benefits to students at various learning stages. The transdisciplinary problem-based learning lessons, at the highest level of integration, only occupy 5% of the lesson plans. The largest category of the lesson design focuses on integrating students' current math knowledge in ePCR planning, coding, assembling, or testing projects. Although the latter integration provides a great opportunity for students to apply and transfer their math knowledge to other disciplines, teachers need to understand that it is not the only way to integrate ePCR.

Results of our study suggest a learning trajectory of ePCR integration in middle-grade math lessons. The lesson plans in the category of ePCR technology can be used at the beginning of an ePCR integration curriculum. They focus on improving students' competence and efficacy in ePCR technology and content. The math knowledge that students apply is mostly at lower-elementary levels, like addition or subtraction, which would not burden their cognitive load in technology learning. Teaching these lessons would help to improve students' technology efficacy at the beginning of an ePCR-integrated curriculum. Then students move into the stage when they apply math knowledge in ePCR planning, coding, assembling, and testing in small group projects. Students will be able to practice their math and ePCR knowledge in those projects. In the end, they may work in transdisciplinary problem-based learning activities with teachers' facilitation. The last stage may aim to develop students' comprehensive problem-solving skills.

We encourage K-12 teachers and teacher educators to design ePCR-integrated projects relevant to students' personal, family, cultural, and social lives. As suggested in motivation and instructional design theories ([Keller, 2010](#); [Merrill, 2002](#)), these learning activities are likely to improve students' interest and engagement when their prior knowledge or experience is activated. In addition, these lesson plans have the capacity to further develop students' CT skills based on the findings of our study.

5.2. Implications for researchers

The clusters of lesson plans in our study differed from the integration categories that [Zhong and Xia \(2020\)](#) identified in their review of studies. The largest category in their study is featured math learning through playing with the robots, while our analysis of existing lesson plans shows that the largest category is the math learning in ePCR planning, coding, assembling, or testing projects. It is possible that more teachers have received the training and grown their knowledge of ePCR planning, coding, assembling, or testing since 2018. Another possibility is that [Zhong and Xia's \(2020\)](#) analysis is based on empirical studies, while ours are based on current practice. Nonetheless, either assumption suggests a need for more research on the largest category of the lesson plans of our study. It will offer teachers suggestions to improve their design and delivery of ePCR planning, coding, assembling, or testing projects in math lessons.

We call for empirical studies on the impact of different clusters of lessons on students' learning. The cluster analyses of this study generated different types of ePCR integration in middle-grade math classes. We then further grouped them into large categories based on their commonalities. However, clusters under the same categories differ in other aspects. Empirical evidence is needed on how different lesson clusters under the same categories impact students' learning. Results from these studies will provide details, such as the sequence of teaching CT skills or the staged integration into math learning, and support educators in developing a quality ePCR-integrated math curriculum.

Results of our study indicate a need to discuss the activity design and ePCR technology, such as their functionality, device design, and even the related coding environment, in the empirical studies and systematic reviews. The lesson plan clusters identified in our study suggest that it is challenging to compare the ePCR lessons and their impact without considering the technology usage and activity design, as the latter factors could vary a lot in different lessons. Even with the same device, lessons varied across spectrums of math and ePCR knowledge development. Our literature review shows a lack of systematic reviews examining these variables ([Anwar et al., 2019](#); [Benitti, 2012](#); [Xia & Zhong, 2018](#); [Zhang et al., 2021](#)). It could be due to the missing descriptions in primary studies. Therefore, we suggest more comparison studies discussing the different technology and activity designs as well as their impact on students' learning.

6. Conclusions

This study aimed to identify how different knowledge was used together in middle-grade ePCR-integrated math lessons. We

collected 281 lesson plans available on 12 ePCR producers' websites. We administered a collaborative and iterative coding process and extracted five variables for cluster analyses: ePCR functionality, coding skills, CT skills, math knowledge, and activity design. Results showed three types of combinations of ePCR technology in the lessons. They focused on teaching students different technology and content at different levels. This finding indicates the need to view ePCR technology from different perspectives, such as ePCR design and functionality. We call for future studies to expand the research scope of ePCR and identify optimal uses of technology for different student groups.

We also identified five categories of ePCR integration in middle-grade math lesson plans based on the TPACK framework. They showed the mainstream of ePCR integration as well as other possibilities for improving students' technology competence, math knowledge, and problem-solving skills. However, the TPACK knowledge discussed in our study is not inclusive. More research is needed to examine and compare the ePCR-integrated lessons in K-12 classrooms.

Credit author statement

Shenghua Zha Conceptualization, Methodology, Investigation, Validation, Formal analysis, Writing, Supervision, Yi Jin Conceptualization, Methodology, Investigation, Validation, Formal analysis, Writing, Rebecca Wheeler Investigation, Validation, Formal analysis, Erin Bosarge Investigation, Validation, Formal analysis.

Data availability

Data will be made available on request.

Acknowledgements

This work is supported by the National Science Foundation under Grants DUE-2121417 and CNS-1953544. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

- Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Journal of Technology and Teacher Education*, 19(3), 47–57.
- Anwar, S., Bascou, N. A., Menekse, M., & Kardgar, A. (2019). A systematic review of studies on educational robotics. *Journal of Pre-College Engineering Education Research (J-PEER)*, 9(2). <https://doi.org/10.7771/2157-9288.1223>
- Archambault, L., & Crippen, K. (2009). Examining TPACK among K-12 online distance educators in the United States. *Contemporary Issues in Technology and Teacher Education*, 9(1), 71–88.
- Bacher, J., Wenzig, K., & Vogler, M. (2004). *SPSS twostep cluster-a first evaluation*.
- Barak, M., & Assal, M. (2016). 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>
- Benitti, F. B. V. (2012). Exploring the educational potential of robotics in schools: A systematic review. *Computers & Education*, 58(3), 978–988. <https://doi.org/10.1016/j.compedu.2011.10.006>
- Blikstein, P. (2013). *Gears of our childhood: Constructionist toolkits, robotics, and physical computing, past and future proceedings of the 12th international conference on interaction design and children*. <https://doi.org/10.1145/2485760.2485786>. New York, New York, USA.
- Brown, J. C. (2017). A metasynthesis of the complementarity of culturally responsive and inquiry-based science education in K-12 settings: Implications for advancing equitable science teaching and learning. *Journal of Research in Science Teaching*, 54(9), 1143–1173. <https://doi.org/10.1002/tea.21401>
- Cabrera, L. (2019). Teacher preconceptions of computational thinking: A systematic literature review. *Journal of Technology and Teacher Education*, 27(3), 305–333. <https://www.learntechlib.org/primary/p/210234/>.
- Century, J., Ferris, K. A., & Zuo, H. (2020). Finding time for computer science in the elementary school day: A quasi-experimental study of a transdisciplinary problem-based learning approach. *International Journal of STEM Education*, 7(1). <https://doi.org/10.1186/s40594-020-00218-3>
- Ching, Y.-H., Hsu, Y.-C., & Baldwin, S. (2018). Developing computational thinking with educational technologies for young learners. *TechTrends*, 62(6), 563–573. <https://doi.org/10.1007/s11528-018-0292-7>
- Chiu, T., Fang, D., Chen, J., Wang, Y., & Jeris, C. (2001). *A robust and scalable clustering algorithm for mixed type attributes in large database environment Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. San Francisco: California. <https://doi.org/10.1145/502512.502549>
- Common core State standards initiative. (2010). <http://www.corestandards.org/>.
- Computer Science Teachers Association. (2017). *K-12 computer science standards*. <https://drive.google.com/file/d/1-dPTAI1yk2HYPKUWZ6DqaM6aVUDa9iby/view>.
- Denning, P. J. (2017). Remaining trouble spots with computational thinking. *Communications of the ACM*, 60(6), 33–39. <https://doi.org/10.1145/299843>
- Eglash, R., Scott, K. A., Clark, K., Gilbert, J. E., Taylor, V., & Geier, S. R. (2013). Culturally responsive computing in urban, after-school contexts. *Urban Education*, 48(5), 629–656. <https://doi.org/10.1177/0042085913499211>
- Emergen Research. (2022). *Educational robot market by constituents (software, hardware, controllers, sensor and actuators), by type (pre-programmed, humanoid, autonomous robot, tele-operated, augmenting), by education level, and by region forecast to 2028* <https://www.emergenresearch.com/industry-report/educational-robot-market>.
- Francis, K., & Davis, B. (2018). Coding robots as a source of instantiations for arithmetic. *Digital Experiences in Mathematics Education*, 4(2–3), 71–86. <https://doi.org/10.1007/s40751-018-0042-7>
- Grover, S., Pea, R., & Cooper, S. (2016). *Factors influencing computer science learning in middle school proceedings of the 47th ACM technical symposium on computing science education*. USA: Memphis, Tennessee. <https://doi.org/10.1145/2839509.2844564>
- Gutierrez, F. J., Simmonds, J., Hitschfeld, N., Casanova, C., Sotomayor, C., & Peña-Araya, V. (2018). Assessing software development skills among K-6 learners in a project-based workshop with Scratch proceedings of the 40th international conference on software engineering: Software engineering education and training. <https://doi.org/10.1145/3183377.3183396>. Gothenburg, Sweden.
- Hendrickson, Katie, Gauthier, Liz, Glennon, Maggie, Harrigan, Alexis, Weissman, Hannah, Fletcher, Carol, ... Mak, Janice (2021). *2021 state of computer science education: Accelerating action through advocacy*. Code.org, CSTA, & ECEP Alliance.
- Kanika, Chakraverty, S., & Chakraborty, P. (2020). Tools and techniques for teaching computer programming: A review. *Journal of Educational Technology Systems*, 49(2), 170–198. <https://doi.org/10.1177/0047239520926971>

- Kazakoff, E. R., Sullivan, A., & Bers, M. U. (2013). The effect of a classroom-based intensive robotics and programming workshop on sequencing ability in early childhood. *Early Childhood Education Journal*, 41(4), 245–255. <https://doi.org/10.1007/s10643-012-0554-5>
- Keller, J. M. (2010). *Motivational design for learning and performance: The ARCS model approach*. Springer.
- Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2019). Teacher change following a professional development experience in integrating computational thinking into elementary science. *Journal of Science Education and Technology*, 29(1), 174–188. <https://doi.org/10.1007/s10956-019-09798-4>
- Kopcha, T. J., Ocak, C., & Qian, Y. (2020). Analyzing children's computational thinking through embodied interaction with technology: A multimodal perspective. *Educational Technology Research & Development*, 69(4), 1987–2012. <https://doi.org/10.1007/s11423-020-09832-y>
- Lee, I., Martin, F., & Apone, K. (2014). Integrating computational thinking across the K-8 curriculum. *ACM Inroads*, 5(4), 64–71. <https://doi.org/10.1145/2684721.2684736>
- Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*, 2(1). <https://doi.org/10.1145/1929887.1929902>
- Leonard, J., Barnes-Johnson, J., & Evans, B. R. (2019). Using computer simulations and culturally responsive instruction to broaden urban students' participation in STEM. *Digital Experiences in Mathematics Education*, 5(2), 101–123. <https://doi.org/10.1007/s40751-018-0048-1>
- Leonard, J., Buss, A., Gamboa, R., Mitchell, M., Fashola, O. S., Hubert, T., & Almughayrah, 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>
- Lockwood, E., DeJarnette, A. F., Asay, A., & Thomas, M. (2016). *Algorithmic thinking: An initial characterization of computational thinking in mathematics*. 38th. Tucson, AZ: Annual Meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education.
- Malyn-Smith, J., Lee, I. A., Martin, F., Grover, S., Evans, M. A., & Pillai, S. (2018). Developing a framework for computational thinking from a disciplinary perspective. In S. C. Kong, D. Andone, G. Biswas, T. Crick, H. U. Hoppe, T. C. Hsu, R. H. Huang, K. Y. Li, C. K. Looi, M. Milrad, J. Sheldon, J. L. Shih, K. F. Sin, M. Tissenbaum, & J. Vahrenhold (Eds.), *International conference on computational thinking education* (pp. 182–186). Education University of Hong Kong.
- Merrill, M. D. (2002). First principles of instruction. *Educational Technology Research & Development*, 50(3), 43–59. <https://doi.org/10.1007/BF02505024>
- Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for integrating technology in teacher knowledge. *Teachers College Record*, 108(6), 1017–1054. <https://doi.org/10.1111/j.14679620.2006.00684.x>
- Morales-Chicas, J., Castillo, M., Bernal, I., Ramos, P., & Guzman, B. (2019). Computing with relevance and purpose: A review of culturally relevant education in computing. *International Journal of Multicultural Education*, 21(1). <https://doi.org/10.18251/ijme.v21i1.1745>
- Mouza, C., Yang, H., Pan, Y.-C., Yilmaz Ozden, S., & Pollock, L. (2017). Resetting educational technology coursework for pre-service teachers: A computational thinking approach to the development of technological pedagogical content knowledge (TPACK). *Australasian Journal of Educational Technology*, 33(3), 61–76. <https://doi.org/10.14742/ajet.3521>
- Norusis, M. J. (2008). *SPSS 16.0 guide to data analysis* (2nd. ed.). Prentice Hall.
- Ntemngwa, C., & Oliver, J. S. (2018). The implementation of integrated science technology, engineering and mathematics (STEM) instruction using robotics in the middle school science classroom. *International Journal of Education in Mathematics, Science and Technology*, 6(1), 12–40. <https://doi.org/10.18404/ijemst.380617>
- O'Sullivan, D., & Igoe, T. (2004). *Physical computing: Sensing and controlling the physical world with computers*. Thomson.
- Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hrobjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., ... Moher, D. (2021). Mar 29). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ*, 372, n71. <https://doi.org/10.1136/bmj.n71>
- Piaget, J. (1952). *The origins of intelligence in children*. International Universities Press.
- Pollock, L., Mouza, C., Guidry, K. R., & Pusecker, K. (2019, February). *Infusing computational thinking across disciplines: Reflections & lessons learned*. Proceedings of the 50th ACM technical symposium on computer science education. <https://doi.org/10.1145/3287324.3287469>. Minneapolis, MN.
- Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science*, 45(5), 583–602. <https://doi.org/10.1007/s11251-017-9421-5>
- Rich, P., Bartholomew, S., Daniel, D., Dinsmoor, K., Nielsen, M., Reynolds, C., ... Yaune, J. (2021a). Trends in tools used to teach computational thinking through elementary coding. <https://doi.org/10.13140/RG.2.2.33628.10887>
- Rich, P. J., Mason, S. L., & O'Leary, J. (2021b). Measuring the effect of continuous professional development on elementary teachers' self-efficacy to teach coding and computational thinking. *Computers & Education*, 168. <https://doi.org/10.1016/j.compedu.2021.104196>
- Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20, 53–65. [https://doi.org/10.1016/0377-0427\(87\)90125-7](https://doi.org/10.1016/0377-0427(87)90125-7)
- Sáez-López, J.-M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "Scratch" in five schools. *Computers & Education*, 97, 129–141. <https://doi.org/10.1016/j.compedu.2016.03.003>
- Sáez-López, J.-M., Sevillano-García, M.-L., & Vázquez-Cano, E. (2019). The effect of programming on primary school students' mathematical and scientific understanding: Educational use of mBot. *Educational Technology Research & Development*, 67(6), 1405–1425. <https://doi.org/10.1007/s11423-019-09648-4>
- Saqr, M., Ng, K., Oyelere, S., & Tedre, M. (2021). People, ideas, milestones: A scientometric study of computational thinking. *ACM Transactions on Computing Education*, 21(3), 1–17. <https://doi.org/10.1145/3445984>
- Slangen, L., van Keulen, H., & Gravemeijer, K. (2011). What pupils can learn from working with robotic direct manipulation environments. *International Journal of Technology and Design Education*, 21(4), 449–469. <https://doi.org/10.1007/s10798-010-9130-8>
- Stemler, S. (2000). An overview of content analysis. *Practical Assessment, Research and Evaluation*, 7(1). <https://doi.org/10.7275/z6fm-2e34>
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research* (2nd. ed.). SAGE Publications.
- Sullivan, F. R., & Heffernan, J. (2016). Robotic construction kits as computational manipulatives for learning in the STEM disciplines. *Journal of Research on Technology in Education*, 48(2), 105–128. <https://doi.org/10.1080/15391523.2016.1146563>
- Taylor, M. S. (2018). Computer programming with pre-k through first-grade students with intellectual disabilities. *The Journal of Special Education*, 52(2), 78–88. <https://doi.org/10.1177/0022466918761120>
- Waterman, K. P., Goldsmith, L., & Pasquale, M. (2020). Integrating computational thinking into elementary science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning. *Journal of Science Education and Technology*, 29(1), 53–64. <https://doi.org/10.1007/s10956-019-09801-y>
- Webb, M., Davis, N., Bell, T., Katz, Y. J., Reynolds, N., Chambers, D. P., & Syslo, M. M. (2017). Computer science in K-12 school curricula of the 21st century: Why, what and when? *Education and Information Technologies*, 22(2), 445–468. <https://doi.org/10.1007/s10639-016-9493-x>
- 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>
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Trans. Royal Socie. A*, 366(1881), 3717–3725. <https://doi.org/10.1098/rsta.2008.0118>
- Xia, L., & Zhong, B. (2018). A systematic review on teaching and learning robotics content knowledge in K-12. *Computers & Education*, 127, 267–282. <https://doi.org/10.1016/j.compedu.2018.09.007>
- Zha, S., Morrow, D. A. L., Curtis, J., & Mitchell, S. (2020). Learning culture and computational thinking in a Spanish course: A development model. *Journal of Educational Computing Research*, 59(5), 844–869. <https://doi.org/10.1177/0735633120978530>
- Zhang, Y., Luo, R., Zhu, Y., & Yin, Y. (2021). Educational robots improve K-12 students' computational thinking and STEM attitudes: Systematic review. *Journal of Educational Computing Research*, 59(7), 1450–1481. <https://doi.org/10.1177/0735633121994070>
- Zhong, B., & Xia, L. (2020). A systematic review on exploring the potential of educational robotics in mathematics education. *International Journal of Science and Mathematics Education*, 18(1), 79–101. <https://doi.org/10.1007/s10763-018-09939-y>