TEACHER PROFESSIONAL DEVELOPMENT FOR COMPUTATIONAL THINKING

Stacie L. Mason

Design & Development Project Report
Instructional Psychology & Technology, Brigham Young University

Product Description

The purpose of this project was to design effective professional development to provide teachers the skills and confidence to teach computational thinking using robots. The product, Teacher Professional Development for Computational Thinking (TPD4CT), includes four professional development (PD) trainings and resources.

1. In-school training: I designed the in-school training for individual teachers and grade-level groups. The two main goals of training are to help teachers (1) feel comfortable using robots and (2) plan how to teach with robots. The training is brief, providing just enough information and resources to get started, with the idea that teachers will continue learning on their own through experience and personal research. Each training session follows the following framework:
    a. Model: The trainer models how to use robots.
    b. Practice: The teachers play with and program the robots.
    c. Instruction: The trainer explains computational thinking concepts and shares relevant state, ISTE, and CSTA learning standards.
    d. Plan: The trainer and teacher discuss plans for teaching with the robots.
    e. Implement: Teachers have 1-3 weeks to teach their students using robots.
    f. Reflect: Teachers are asked to reflect on their experience teaching with robots.

Although all training sessions follow the same basic format, content can be individualized based on grade level and technology to be used. I designed three modules, one for each robot available for training: Ozobot Bit, Sphero SPRK+, and Dash robots.

2. Lesson plan library: In conjunction with the in-school training, I developed a small library of lesson plans for teaching computational thinking using robots. The library, currently housed in Google Drive, includes plans for teaching computational thinking concepts and basic robot programming skills. I designed plans for use with grades K-6 on four topics related to computational thinking:
    a. Decomposition
    b. Pattern recognition
    c. Abstraction
    d. Algorithms/automation

3. District-level training: I developed a two-hour training session for Provo district K-12 teachers. I aligned the training with the Computational Thinking and Coding Level 1 badge in Badge School: http://badgeschool.org/library/search/coding. The three primary objectives of the training are to (1) practice using robots and online coding programs, (2) gain a basic understanding of computational thinking, and (3) make a plan to teach computational thinking skills.

4. Train-the-trainers workshop: I designed this training for a large group (10-30 educators). The training includes four two-hour sessions. The first hour of each session includes instruction and

guided practice; the second hour includes collaborative lesson planning, practice, feedback, and reflection. The four sessions focus on the following content:

a. Computational thinking and unplugged activities
b. Ozobots and learning standards
c. Dash robots and learning standards
d. Sphero robots and learning standards

The purpose of the training is to prepare participants to train and support other teachers in their schools. If a school were using only one type of robots, the trainer from that school would need to attend only two of the four training sessions.

Needs Analysis

Learning computer science skills can benefit students economically and academically. In the United States, job opportunities in computer and information technology were projected to increase 13 percent in 10 years, compared to 7 percent projected job growth overall (Bureau of Labor Statistics, 2018, Computer and Information Technology; Bureau of Labor Statistics, 2018, Employment Projections). Studies have indicated a host of benefits from learning robotics or coding, including improvement in student engagement, motivation, confidence, problem-solving, communication, STEM learning and performance (Rich, Jones, Belikov, Yoshikawa, & Perkins, 2017; Kim et al., 2015). Recognizing the benefits of learning computing and coding, school districts and state governments are increasingly adopting policies that require computer science instruction (Rich et al., 2017).

One barrier to providing effective computer science instruction is a shortage of trained teachers (Rich et al., 2017). In a report by Google Inc and Gallup Inc (2016), 63% of surveyed K-12 principals in schools that did not offer computer science instruction said that they lacked qualified teachers. One solution is to train practicing teachers to teach computer science.

At the time that I began this project, Provo School District (PSD) did not have a program in place to provide K-6 instruction in computational thinking. Teachers, principals, and district personnel expressed interest in the training. Teachers said that they would like to teach using robots, but lacked knowledge and confidence. Principals were willing to buy robots and other technology for classroom use, but only if teachers would implement the technology effectively. Karen Brock, the Director of Professional Development for Provo School District was interested in providing district teachers with training, but lacked personal expertise in teaching computing, coding, and robotics.

Learner Analysis

For this project, the learners were K-12 teachers in the Provo School District. Most learners were females aged 22-55 years old, with 0-30 years teaching experience. During the design and development phases of TPD4CT, learners self-selected to participate in the TPD4CT project; therefore, most were comfortable using technology and interested in improving their teaching. However, most lacked previous training in computational thinking and robotics. Eventually the

training and resources may be used with teachers who are required to participate. Therefore, I have tried to design the training and resources such that they could appeal to both novice and experienced teachers, and to both reluctant and enthusiastic technology users.

Two teachers at Spring Creek Elementary School and three teachers at Lakeview Elementary School participated in the in-school training and pilot study (Table 1). The teachers taught grades K-4 and used at least one of three types of robots with their students for a period of 1-3 weeks.

Table 1

Participants by Grade Level, Technology, and Period of Implementation

Teacher Pseudonym	**Grade level**	**Robot**	**Period of Implementation**
Annie	3	Ozobots	3 weeks
Becca	4	Spheros & Ozobots	2 weeks
Cathy	1	Dash	1 week
Deb	2	Ozobots	10 days
Evelyn	K	Dash	1 week

Environmental Analysis

Located in a mid-sized city, Provo School District includes 18,836 students who attend 13 elementary schools, two middle schools, two traditional high schools, and one alternative high school (Provo School District, 2017, Annual Statistical Report). The district's high school graduation rate is 71% (Provo School District, 2017, 2015-2016 Progress Report).

Lakeview is a large elementary school in northwest Provo (Table 2). Over the past three school years, enrollment has remained approximately steady, while test scores have increased. Students' test proficiency rates are significantly higher than district and state proficiency rates (Table 3). Approximately 20% of Lakeview students participate in the school's Portuguese immersion program. Principal Drew Daniels has served as a district principal since 1992 and has led Lakeview since its inception in 2008.

Spring Creek Elementary is a Title I school in southeast Provo. As shown in Table 2, most Spring Creek students are economically disadvantaged, and 44% are English Language Learners (ELL). Despite the school's challenges, student test scores have increased steadily over the past four

school years, and the school was awarded the 2017 National Title I Distinguished School Award (Franklin, 2017). The principal, Jill Franklin, taught for 18 years and has served as an instructional facilitator, Title I coordinator, CFA, and assistant principal.

The demographics of each school may affect teachers' experience implementing their TPD4CT training. Students learning English may have a harder time than other students following instructions or learning the computational thinking vocabulary. Therefore, in the training and resources, I may want to talk with teachers about adapting their instruction for ELLs.

Table 2

School Demographics, 2016-2017

	Lakeview	**Spring Creek**
October Enrollment	754	475
Ethnic Minority	29%	64%
Economically Disadvantaged	40%	80%
Special Education	12%	15%
English Language Learner (ELL)	18%	44%
Chronic Absenteeism	13%	18%
Mobility	<10%	26%

(Provo School District, 2017)

Table 3

SAGE Test Proficiency, 2017

	Lakeview	**Spring Creek**	**Provo S. D.**	**Utah**
Language Arts	63.4%	47.3%	50.9%	43.6%
Mathematics	64.3%	55.9%	50.2%	45.7%
Science	69.5%	47.2%	51.7%	47.5%

(Provo School District, 2017)

Consulting Products

The purpose of this project was to develop training to prepare teachers to teach computational thinking using robots. The three robots being used were Ozobot Bit, Sphero SPRK+, and Dash. The websites for each of the robots include instructional materials for educators, described

below.

Ozobot.com provides several supports for educators:

1. Educator's Guide: The eight-page Educator's Guide describes what Ozobot can do, how it works, and where to find additional resources including links to lessons, games, and the ozoblockly editor.
2. Webinars: As of this writing, there were 12 online webinars on various topics, including "How to introduce your students to Ozobot." Most of the webinars run about 30 minutes, followed by Q&A.
3. Tip sheets including "Ozobot Tips," "Color Code Reference," "Calibration Tips," and "OzoBlockly Getting Started."
4. Ozobot Lesson Library. Lessons are organized into four content areas: (a) color codes basic training, (b) ozoblockly basic training, (c) a computer science series, and (d) STEAM integrated lessons. The lessons are further organized by grade level. Many of the lessons were authored by Ozobot, while others were contributed by teachers. The library is searchable and growing.

Among the three websites, sphero.com provides the least support for teachers. The website has two links for educators: the "Education Page" and the "Educators Page," both of which describe their products. The latter also has useful links to an Educator's Guide and to Sphero Edu. The 16-page Educator's Guide describes Sphero, its uses, Sphero Edu, and classroom management. Most of the teacher materials are found at Sphero Edu, available online and as an app. Sphero Edu includes sample programs (under the tab "Programs") with accompanying videos, and activity lesson plans (under the tab "Activities"), organized by subject matter and grade level.

Wonder Workshop, the maker of Dash robots, provides extensive online support, including an online professional development course (teachwonder.com), a Learn to Code Curriculum, and a Code to Learn Lesson Library. However, the professional development course is not free—as of this writing, it cost $350—and some of the lesson plans are accessed only through a paid subscription or incorporate materials that have an additional cost. The Learn to Code Curriculum is aligned to CSTA and ISTE standards and Code.org curriculum and has six levels involving six lessons per level. The Code to Learn Lesson Library includes dozens of lesson plans searchable by subject, grade level, and robot/accessory. Some of the lessons were provided by Wonder Workshop; others were shared by teachers .

Besides the training and resources provided by the makers of Ozobot, Sphero, and Dash, there are several other websites that provide training and resources for computational thinking.

Google for Education offers both training and resources:

1. Computational Thinking for Educators is a free, online course to help humanities, math, science, and computing teachers integrate computational thinking into their instruction. It is divided into five units and is aimed at high school teachers and students.
2. Exploring Computational Thinking provides lesson plans, videos, and resources to help educators understand and teach students about computational thinking. The materials can be searched by subject, type, and age (8-18). I use at least one video from this website in my trainings.

Code.org provides free, online courses for grades K-12, professional learning programs, and lesson plans. The resources for grades K-5 are as follows:

1. Computer Science Fundamentals is Code.org's curriculum for teaching K-5 students computational thinking and programming concepts. There are eight courses, one per grade K-5, plus two express courses. Each course has about 15 lessons and takes 15-20 hours to complete. The curriculum is aligned to CSTA, Next Generation Science Standards (NGSS), and Common Core standards.
2. Professional learning programs:
    a. Code.org offers free, one-day workshops
    b. Teaching Computer Science Fundamentals is a free, online course that introduces computer science fundamentals and Code.org resources. The course is divided into 11 stages, each of which takes 10-60 minutes to complete.
3. Unplugged lessons can be taught on their own or as part of the Computer Science Fundamentals curriculum.

The ISTE website provides links to several resources:

1. ISTE Standards for Students includes seven areas of digital literacy, including computational thinking.
2. Computational Thinking for All links to presentations, handouts, and booklets including
    a. Computational Thinking Teacher Resources provides definitions, a progression chart, and several lesson plans.
    b. Computational Thinking Leadership Toolkit includes definitions, a progression chart, and an implementation strategies guide.

In my training sessions with teachers, I shared online resources described above. Mine isn't so much a competing product as a supporting product. The above resources provide more information than I can provide in a single training session. What my training provides that these other options do not is in-person, personalized, hands-on training. The in-school training is designed for individuals or small groups, for 30-60 minutes. The format allowed me to adjust to teachers' needs and wants. The more help teachers wanted, the more help I gave. Online training shows how to use a robot, but seeing it done is not the same as doing it. In my training, teachers learn to use the robots by using the robots.

Design Process & Evolution

Planning Phase

During the planning phase of the design process, I first determined what training and resources were needed. To do so, I consulted with Rick West from BYU and Karen Brock from Provo School District. I also researched professional development for teaching computational thinking, to help me understand what kinds of training had been tested. Based on the research and needs, I determined a basic framework for the trainings. Once I had an idea of the product I hoped to design and develop, I contacted principals at two elementary schools, who agreed to allow me to work with their teachers. I then applied for and received approval from the IRBs at BYU and Provo School District to conduct training and collect data. In writing the project proposal, I completed a learner and environment analysis, analysis of competing products,

content analysis, design specifications, and implementation and evaluation plan. Rick West provided feedback for the proposal.

In-School Training

Having completed the planning stage, I began designing in-school training, in conjunction with the lesson plan library. The content I needed to develop included instruction in computational thinking, learning standards, and robot use. For the instruction in computational thinking I designed a handout that includes an explanation of computational thinking. Initially, the handout included a brief definition of computational thinking:

> Computational thinking is an approach to problem solving that includes five main principles:
>
> 1. Decomposition: breaking down problems
> 2. Pattern recognition: organizing data according to patterns
> 3. Abstraction: representing data in models
> 4. Algorithms: automating solutions in a series of steps
> 5. Analysis: analyzing and implementing solutions

In the second iteration I added further description:

> Computational thinking also involves the following activities:
>
> - Tinkering
> - Creating
> - Debugging
> - Persevering
> - Collaborating

When I shared these definitions in training sessions, I could tell that teachers still lacked a clear understanding computational thinking. I wanted a definition that was clear, concise, and memorable. Based on something I read (Wing, 2006), I added, “Computational thinking is thinking like a computer scientist.”

For the planning portion of the training, I added ISTE and CSTA standards to the handout. I initially did not include Utah Core Standards because they are not very specific about teaching computational thinking or robotics, but I later added relevant Utah technology standards since teachers are expected to teach them.

For the robotics instruction I designed three modules, one for each type of robot in the McKay School Technology Lab: Ozobots, Spheros, and Dash robots. To develop the instruction, I first needed to understand how to use the robots. I checked out each of the robots from the tec lab, explored the robot websites, spent time playing with and programming the robots, and then designed lessons to train teachers to instruct their students.

To help me refine the design, I conducted a pilot study with teachers at two elementary schools. In my first training session I trained a teacher to use Ozobots. Before the training, I had designed inquiry-based activities that would guide the teacher through basic operating skills for Ozobots.

These activities worked well. Other than editing for clarity, I have made few changes since the first iteration.

My next training session was with Dash robots. Make Wonder, the company that makes Dash robots, has developed their apps and curriculum with a strong scope and sequence. They start with a single instruction, and the user must complete each task in order to unlock the next task. Because the apps are so easy to use, I did not write out a detailed lesson plan as I had for Ozobots. Instead, I determined which apps to use with the first grade teachers I was training, and then provided guidance as necessary during the training. Later, I wrote instructions for using the apps so that other trainers could provide consistent training.

The third module I designed was the Sphero lesson. Based on my success with the Ozobot and Dash lessons, I underprepared for my first Sphero training. My plan was to guide the teacher through the Blocks 1 lesson in Sphero edu, which is divided into steps and includes instructional videos. However, because our time was short and we had to set up student accounts, we did not have time to go through the entire lesson. I wished I had broken it down myself so that I could present a shortened version of the instruction. Later, I wrote instructions that could be used with or without the Blocks 1 lesson.

Lesson Plan Library

As noted above, the lesson plan library includes activities for learning to use Ozobot, Sphero, and Dash robots, which activities are incorporated into the in-school training modules. In addition, the lesson plan library includes plans for unplugged activities that teach computational thinking concepts. I wrote three of the plans—Getting Started Algorithm, Program a Partner, and Train Conducting—based on activities that I observed in training sessions led by Peter Rich and Rick West. These activities are also incorporated into the district training and train-the-trainers. Several plans in the library include activities I adapted from online sources, including Scratch Jr. and BBC Bitesize.

District Training

In January, 2018, I attended a district training session led by Rick West. Later in the semester, he asked me to lead a similar, 2-hour district training session. The session advertisement written by Rick said that participants could earn a badge, so I aligned the training with the Computational Thinking and Coding Level 1 badge in Badge School: http://badgeschool.org/library/search/coding. Badge completion required a person to use an online programming tool and engage in either unplugged activities or robot use. I designed the training to include all three media: online programming tools, unplugged activities, and codable robots. I chose Ozobots since they are the smallest, least expensive, and easiest of the three robots to use. For the Ozobot and unplugged activities I incorporated lesson plans from my lesson plan library. I also used the Computational Thinking and Standards handout I had developed for in-school training. I developed a PowerPoint presentation for use in the training, but was not able to use it due to technical difficulties. However, I had planned for such a

contingency, having observed similar difficulties at the January training session. I had also planned to do three unplugged activities, but we had time for only two.

Train-the-Trainers

The last training that I designed incorporates elements of the previous three, as well as new resources. Train-the-trainers consists of four sessions. Session 1 includes the definition of computational thinking used in in-school and district training, and unplugged activities developed for the lesson library and district training. Session 2 includes Ozobot activities used in the in-school and district trainings. Session 3 includes Dash robot activities developed for the lesson library and in-school training. Session 4 includes Sphero activities developed for the lesson library and in-school training. Plans for the training sessions also include new material: activities and links to articles, videos, and websites not referenced in the lesson library or previous trainings.

Because the train-the-trainers workshop is designed to prepare participants to train others, I devoted the second hour of each training session to lesson planning, practice training, feedback, and reflection. As with the in-school teacher training, implementation is an essential feature of the training; between sessions, participants are expected to train other teachers in what they have learned.

Product Implementation

To ensure that the product was used as intended, I was the sole trainer to pilot the product. I piloted the in-school training with teachers at Lakeview and Spring Creek elementary schools during March and April, 2018. Principals invited teachers to sign up to receive training and borrow robots. I trained four teachers at Lakeview and two teachers at Spring Creek. I met with five of the teachers in their classrooms and the sixth teacher in the teachers' lounge. Participants needed no background knowledge to participate. I provided robots, iPads, and other needed materials. A detailed description of in-school training sessions is found in Appendix I.

I also piloted the district training in April 2018 at the Provo District Office. Approximately 25 K-12 teachers attended. Teachers chose to and received credit for attending the session. Participants needed no background knowledge to participate. I provided the materials required for training: iPads, Ozobots, Spheros, markers, and paper. Most teachers also used their phones or laptops to explore online programming tools during the training.

Both the in-school and district trainings include lessons from the lesson library. However, the library also includes lesson plans that have not yet been implemented. The train-the-trainers workshop has yet to be implemented. Karen Brock proposed that I develop a train-the-trainers workshop, but we have not scheduled said workshop.

Evaluation

I evaluated in-school training for effectiveness. I used feedback from the in-school training sessions to inform the design and development of the other three training resources.

Criteria

I assessed in-school training effectiveness in terms of changes in self-efficacy, attitudes toward using robots, feedback about training, and reported implementation of training. Provo School District did not specify any other evaluation criteria.
The questions guiding the evaluation were as follows:
For a sample of K-6 teachers preparing to teach with robots,

1. What learning objectives did teachers use the robots to teach?
2. What obstacles and challenges to implementation did teachers face?
3. What influence did training have on teachers' technology self-efficacy?
4. Before and after training, what did teachers believe were the benefits of using robots in the classroom?
5. What types of professional development did teachers value?

Procedures

To assess training outcomes, I asked participants to complete self-efficacy and attitude surveys before and after the training (Appendix H). I also engaged participants in informal interviews (Appendix H) when I delivered and retrieved robot kits. I analyzed data from the surveys as descriptive statistics. I coded and thematically categorized data from open-ended survey responses and interview notes. Table 4 shows the survey items and their alignment to research topics.

Table 4
Survey Items by Topic

Topic	Survey I Items	Survey II Items
Objectives	What is your reason for borrowing the robots?	--
	What objectives do you hope to achieve b y using robots in your classroom?	What learning objectives did you achieve by using robots in your classroom?
Challenges	What obstacles are there to using robots in your classroom?	What challenges did you experience using robots in your classroom?
Teacher self-efficacy	I am an effective teacher.	Same

Technology self-efficacy	I know how to operate _____ (Dash robots, Ozobots, or Spheros).	Same
Teacher technology self-efficacy	I am confident in my ability to use robots in the classroom to meet specific learning objectives.	Same
Beliefs about benefits of teaching with robots	It is important for students to learn computational thinking skills (e.g., decomposition, pattern-finding, algorithms)	Same
	Using robots in the classroom will help students learn computational thinking skills.	Using robots in the classroom helped students learn computational thinking skills.
	Helping K-6 students gain digital literacy skills is important.	Same
	Using robots in the classroom will help students gain digital literacy skills.	Using robots in the classroom helped students gain digital literacy skills.
Professional development preferences	--	What sort of future training or support would best help you to use robots to meet specific learning objectives?
	--	Which activities have best helped you prepare to use robots to meet specific learning objectives?
Other	--	I am planning to use the robot kits again to teach.

IRB approval covered the pilot test for the 30-60 minute, one-on-one, in-school training sessions. No official data were collected from the pilot of the two-hour, large-group district training. However, teachers at the training provided positive feedback during the session, and Renae Deighton shared selected feedback from participants.

Evaluation Findings

In this section, I share the findings from the evaluation. The five issues discussed include (1) learning objectives, (2) obstacles and challenges to implementation, (3) teachers' technology

self-efficacy, (4) beliefs about technology, and (5) professional development preferences.

Objectives

In responses to Survey I, all five teachers said they borrowed the robots for the purpose of teaching students; four of five teachers said they borrowed the robots to learn to use them personally; and only two teachers said they wanted to share the robots with other teachers. When asked what learning objectives they hoped to achieve, most teachers gave responses related to computational thinking concepts and approaches:

- "Learning computational thinking skills"
- "Teaching students patterns"
- “Critical thinking"
- “Think ahead”
- "To put things in order”
- “Organize”
- “Teach students about trial and error.”
- "Teach students to follow directions and explore using an invention (to go along w/ our inventions unit).

Only one response was not directly related to computational thinking:

- "To give students a hands-on opportunity to learn digital literacy skills."

Considering that the survey mentioned “computational thinking” and “digital literacy,” the two teachers who mentioned those terms in their responses were likely influenced by the survey and may not have had specific learning objectives in mind. If we do not consider those two responses, the remaining responses include recognizing patterns, critical thinking, organizing information, trial and error, and exploring, all of which are related to computational thinking, and none of which was mentioned in the survey. However, the sample size is too small to draw any lasting conclusions.

When asked in Survey II what learning objectives students had achieved, four of the teachers mentioned computational thinking concepts or approaches:

- “We talked about how inventors use observations, trial and error, and debugging to make their inventions better.”
- “Following directions; being careful and specific when programming”
- "Making patterns; Analyzing problems & creating solutions"
- “We learned about algorithms, procedures”

The response by the fifth teacher was related to computing and coding, but was less specific than the other four: "Talked about computers, robots, coding." The teacher who mentioned algorithms also mentioned ordinal numbers, taking turns, and interactive writing. These responses suggest that (a) simple robots were used for teaching a variety of concepts; (b) robots were used to teach computational thinking; (c) the teachers in the study recalled the computational thinking concepts that I had shared with them weeks earlier.

Challenges

When asked on Survey I what obstacles they faced to using robots in their classrooms, teachers noted multiple barriers to use. Three teachers cited “time,” two mentioned classroom management, one mentioned alignment to standards, and one said, "not knowing how to use them." When asked in Survey II what challenges they had faced while using robots, two teachers noted technical difficulties (robots not working), two mentioned logistical difficulties (giving the robots a clear path; remembering to charge robots and iPads), and one noted user difficulty

(students had a hard time drawing lines thick enough for the robot to read).

Teacher Technology Self-Efficacy

In pre and post surveys, teachers were asked to indicate their level of agreement with statements about their teacher self-efficacy, technology self-efficacy, and teacher technology self-efficacy. On both surveys, all five teachers agreed or strongly agreed with the statement, "I am an effective teacher," which indicates all five teachers viewed themselves as effective teachers and trusted in their ability to teach well. In Survey I responses, three teachers disagreed with the statement, "I know how to use _____ (Dash robots, Ozobots, or Spheros)"; one teacher was neutral, and one teacher agreed with the statement. In Survey II responses, all five teachers agreed with the statement, which indicates positive change for four of five teachers in their technology self-efficacy. Responses also indicated positive change for three of the teachers in teacher technology self-efficacy. In Survey I responses, two teachers disagreed with the statement, "I am confident in my ability to use robots in the classroom to meet specific learning objectives"; one teacher was neutral, and two agreed with the statement. After the intervention, all five teachers agreed with the statement.

Beliefs about Technology

In their responses to both Surveys I & II, all respondents agreed or strongly agreed with the statements that computational thinking and digital literacy are important and that using robots would help students gain computational thinking skills and digital literacy. For the five teachers, there was little or no change between pre-training and post-implementation responses.

Professional Development Preferences

On Survey II, I asked teachers to rank training activities in order from most useful to least useful. The activities that ranked highest were (1) in-school training, (2) teaching students using robots, (3) using the robots on their own, and (4) personal research and planning. Four of the five teachers ranked in-school training as the most helpful activity. The activities that ranked lowest were PLC collaboration and planning, other school-sponsored training, and district training. For most of the teachers in the study, these three low-ranked activities were not applicable to their use of robots, despite being recognized in the literature as important to teacher learning. However, this may indicate that more individualized methods are most helpful in learning computer science skills. Only one of the five teachers had invited her PLC group to the in-school training session, and only one of the five teachers had attended a recent, relevant district training session. When asked what sort of future training or support would be most useful, two teachers said more time to explore with the robots, two requested lesson plan ideas, and one wrote a question mark. The in-school training session consisted largely of time to use the robots and lesson plan ideas, and responses suggest that time and lesson ideas were what teachers wanted more of.

Implications and Conclusion

Despite being small in scope, the intervention seems to have been a success, by multiple measures. First, participants had positive experiences. The teachers in the study reported that their students had fun and the teachers planned to use robots again. Second, teachers gained content knowledge. In surveys taken 1-3 weeks after training, teachers mentioned computational thinking concepts. Third, every teacher who participated in the training implemented the training—they used the robots with the students. Two factors that could explain the high rate of implementation are agency and immediacy. The five teachers who participated in the study

elected to borrow the robots and receive training. Because the teachers borrowed the robots for 1-3 weeks, they were motivated to use them while they could. If the teachers had been required to participate, or if the teachers had permanent, easy access to robots, the rate of implementation may have been lower.

Most significantly, the purpose of the PD was to give teachers the skills and confidence to use robots in the classroom, and that goal was achieved. I believe this short-term intervention was successful because it included content and pedagogical instruction, guided practice with the technology, and the expectation of implementation. It was engaging, hands-on, targeted, and practical. This study supports assertions by Somekh (2008) and Mueller et al. (2008) that teachers gain technology self-efficacy by playing with and having positive experiences with technology.

Despite the success of this intervention, it can be improved. The 30-minute intervention was enough to support a 3-week robot unit but to support deeper learning the training would need to be expanded. Furthermore, although the rate of implementation in the current study was high, the number of teachers who elected to participate in the training was low—only two or three per school. The low rate of participation could be attributed to lack of time and motivation. Teachers were busy teaching required curriculum, and I was asking them to spend time learning and teaching new, non-required skills. Next fall, I plan to collaborate with Lakeview Elementary School teachers and administrators to design training that will prepare all teachers at the school to teach computational thinking. Based on what I have learned from this study, we will design training that is engaging, hands-on, targeted, and practical. Additionally, we will need to expand training to support deeper learning and find ways to motivate reluctant teachers to participate in and implement training.

Appendix

Guide to the Appendix

Appendix	Description
A: In-School Training	Plans and handouts developed for three, in-school training modules.
B: Lesson Plan Library	Plans for robot activities and unplugged activities to teach computational thinking concepts.
C: District Training	Plan and slides for 2-hour, K-12 district training in computational thinking using robots.
D: Train-the-Trainers	Plans for four, 2-hour training sessions.
E: Content Analysis	Plan for breaking down content to be learned into component parts.
F: Insights from Research	Synthesis of literature relevant to the project and lends support for design decisions.
G: Project Management	Project timeline and budget.
H: Assessment Instruments	Pre and post surveys and interview questions used to evaluate the product.
I: Implementation Report	Narrative describing pilot study implementation.
J: Reflection and Critique	Lessons learned from completing the project.

Appendix A: In-School Training

Ozobot training (30-60 minutes)

1. Model: Demonstrate how to turn on the Ozobot Bit (press button on the side).
2. Guided Practice:

Activity 1—Surfaces

1. Place the Ozobot various surfaces::
    a. Dark surfaces
    b. Light surfaces
    c. Sloped surfaces
    d. Carpet
    e. A white piece of paper
2. What did you observe?

What do you know about Ozobot from what you observed?

Activity 2—Lines

1. Draw a couple of lines on a white piece of paper.
2. Place the bot on a line.
3. See if the Ozobot follows the line.
   If not, troubleshoot (debug)—why didn't it work?
    a. Is the Ozobot lit up?
    b. Is the line thick enough?
    c. Is the line unbroken and fairly straight?
    d. Experiment with different lines to figure out what lines work best.
    e. Tips are available here: https://files.Ozobot.com/stem-education/Ozobot-tips.pdf

Activity 3—Adding codes

1. Draw a short, straight, black line.
2. Choose a code (a pattern) from the code sheet: https://play.Ozobot.com/print/guides/Ozobot-color-codes-reference.pdf
3. Copy it onto your paper.
    a. Start where the black line ends.
    b. Make your code look as much like the code sheet as possible.
    c. The colored squares should be touching, with no white space between.
    d. The colored squares should be about ¼" by ¼".
        i. Teachers may need to demonstrate.
        ii. Teachers could draw the codes.
        iii. Use boo-boo tape as needed.
        iv. Could use code stickers (come with starter pack).

4. At the end of the code, draw another black line (if using stickers, draw one long line and put the sticker in the middle of it).
5. Turn on Ozobot, place it at the start, see if it follows the code.
6. Start the Ozobot at the other end.
    a. What happened?
    b. Why? (notice that some codes are the same forward and backwards; other patterns might mean one thing from left to right and another thing when going from right to left)
7. Draw more lines and codes and test them.
8. When code doesn't work, troubleshoot—try to figure out why it's not working
    a. Is the bot on?
    b. Try calibrating the bot.
    c. Make sure the line is an appropriate width (3-5 mm).
    d. Does the code look like the codes on the code sheet?
    e. Is there a line for the bot to follow after the code? (don't end a line with a code)
    f. Do the lines allow the bot to follow the instructions? (e.g., if you told the bot to turn left, did you provide a line going left, along with at least one other option?)

3. Instruction—Computational Thinking

Question: What is computational thinking?

Computational thinking is thinking like a computer scientist. It is an approach to problem solving that includes five main principles:

1. Decomposition: breaking down problems
2. Pattern recognition: organizing data according to patterns
3. Abstraction: representing data in models
4. Algorithms: automating solutions in a series of steps
5. Analysis: analyzing and implementing solutions

Computational thinking also involves the following activities:

- Tinkering
- Creating
- Debugging
- Persevering
- Collaborating

Question: Which of the above activities do you think your students will engage in while using robots?

4. Go over other Ozobot know-how

a. How to charge (charging station included in kit)
b. Show the Ozobot website and resources
    i. Home https://Ozobot.com/
    ii. Education links https://Ozobot.com/stem-education
    iii. Getting started https://Ozobot.com/stem-education/education-getting-started
    iv. Lesson plans https://portal.Ozobot.com/lessons

1. Color codes https://portal.Ozobot.com/lessons/compilation/color-codes-basic-training
2. Ozoblockly (more advanced programming, for older students) https://portal.Ozobot.com/lessons/compilation/ozoblockly-basic-training
3. Deconstruction (for use with Evo) https://portal.Ozobot.com/lessons/compilation/evo-deconstruction-series

v. Games https://Ozobot.com/play/print-games

5. Plan, using ISTE and Utah Core standards.
   a. Choose a relevant Utah Core Standard, ISTE Standard, or CSTA Standard that you would like students to learn using robots.
   b. Collaboratively plan a lesson or activity using robots to meet the standard.

Utah Core Standards--Educational Technology (Grades 3-5)
https://www.uen.org/core/core.do?courseNum=2030
Standard 8. Use technology resources (e.g., calculators, data collection probes, videos, educational software) for problem-solving, self-directed learning, and extended learning activities.
Standard 9. Determine when technology is useful and select the appropriate tool(s) and technology resources to address a variety of tasks and problems.

ISTE Standards for Students https://www.iste.org/standards/for-students
1c Students use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.
1d Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies.
5c Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.
5d Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

CSTA Standards https://www.csteachers.org/page/standards
1A-AP-08 Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
1A-AP-10 Develop programs with sequences and simple loops, to express ideas or address a problem.

6. Implement: Teachers will have 1-3 weeks to teach their students using robots.
7. Reflect: After implementation, ask the teacher about their experience teaching with robots.

Dash Robot Training

(30-60 minutes)

1. Model:
    a. Turn on the Dash robot.
    b. Demonstrate how to find a Dash app on an iPad.
        i. Path, K-2
        ii. Blockly, grades 2-6
    c. The app will connect with a Dash robot.
2. Guided Practice:
    Providing assistance as needed, let the teacher play around on either Path or Blockly.

Path

Overview: Draw a path with your finger, then drag and drop icons onto the path. Each icon represents an action or sound. Press the picture of the Dash robot, and Dash will follow the path and complete the actions. Complete tasks to unlock new icons and screens. All instructions are visual; no reading skills are necessary.

a. When you first use Path, you'll see a screen with a triangle in the middle; press the triangle.
b. If you see the + icon in the top right corner, then press it to connect to your Dash robot. You'll need to turn the robot on by pressing the power button on the side.
c. You will then see a screen with a lit-up square showing Dash. Tap the box.
d. With your finger, draw a path for Dash to follow.
e. Tap the picture of Dash on the screen, and the robot will follow the path. Drag and drop icons along the path, and Dash will do the actions.
f. After you practice drawing paths and adding actions, you will unlock the second screen: a racetrack.
g. Draw a path. The instructions at the top center tell you which icons to add to the path and in what order. The icons appear at the bottom of the screen. Drag icons from the bottom of the screen to the path.
h. Complete several challenges to unlock the next screen: a farmyard.
i. There are four screens in Path; all four work the same but have different icons (sounds and actions).

Blockly

Overview: Learn to program Dash using Blockly language. Complete 13 puzzles, each of which is broken down into several steps. Also create your own programs.

a. When you first use Blockly, you'll see a screen with a triangle in the middle; press the triangle.
b. If you see the + icon in the top right corner, then press it to connect to your Dash robot. You'll need to turn the robot on by pressing the power button on the side.
c. You will then see a screen asking you to choose Dash or Dot.
d. You will be taken to the Puzzles page. Select the first puzzle.

e. Note: if you are the first to use the program on your iPad, the first puzzle will be the only one that is unlocked. However, if a previous user has unlocked other puzzles, you should still start with the first.
f. The first puzzle, Driving School, will teach you the basics of programming with Blockly.
g. You will be given instructions that become increasingly more challenging. The first task is simply to press START.
h. After the instructions are given, the next screen will look something like this, with the instructions along the bottom. The tabs needed to complete the task (left column) will be highlighted. Select the appropriate blocks and snap them into place. After completing each task, press START (lower left corner).
i. If the task is incomplete or incorrect, you will be told that there's a problem.
j. When you complete all 12 tasks, you have completed the first puzzle.
k. After completing Puzzle 1, you may choose to create your own project or go to the next puzzle.

3. Instruction—Computational Thinking

Question: What is computational thinking?

Computational thinking is thinking like a computer scientist. It is an approach to problem solving that includes five main principles:

1. Decomposition: breaking down problems
2. Pattern recognition: organizing data according to patterns
3. Abstraction: representing data in models
4. Algorithms: automating solutions in a series of steps
5. Analysis: analyzing and implementing solutions

Computational thinking also involves the following activities:

- Tinkering
- Creating
- Debugging
- Persevering
- Collaborating

Question: Which of the above activities do you think your students will engage in while using robots?

4. Go over other Dash know-how

a. How to charge (the charging cord should be in the box)
b. Show the Dash website and resources:
    i. Home https://www.makewonder.com/dash
    ii. Education links https://education.makewonder.com/
    iii. Curriculum https://education.makewonder.com/curriculum
        1. Learn to Code https://education.makewonder.com/curriculum
            a. 6 levels (A-F)
            b. 6 lessons per level
        2. Code to Learn https://education.makewonder.com/curriculum/code-to-learn
            a. Lesson plans submitted by teachers

b. Select by subject, grade level, robot (Dash or Dot)

5. Plan, using ISTE and Utah Core standards.
   a. Choose a relevant Utah Core Standard, ISTE Standard, or CSTA Standard that you would like students to learn using robots.
   b. Collaboratively plan a lesson or activity using robots to meet the standard.

Utah Core Standards--Educational Technology (Grades 3-5)
https://www.uen.org/core/core.do?courseNum=2030
Standard 8. Use technology resources (e.g., calculators, data collection probes, videos, educational software) for problem-solving, self-directed learning, and extended learning activities.
Standard 9. Determine when technology is useful and select the appropriate tool(s) and technology resources to address a variety of tasks and problems.

ISTE Standards for Students https://www.iste.org/standards/for-students
1c Students use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.
1d Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies.
5c Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.
5d Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

CSTA Standards https://www.csteachers.org/page/standards
1A-AP-08 Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
1A-AP-10 Develop programs with sequences and simple loops, to express ideas or address a problem.

6. Implement: Teachers will have 1-3 weeks to teach their students using robots.
7. Reflect: After implementation, ask the teacher about their experience teaching with robots.

Sphero SPRK+ Training (30-60 minutes)

1. Model:
    a. Demonstrate how to find Sphero edu app on an iPad or Chromebook.
    b. Teacher should set up an account (students will also need accounts, which the teacher can set up later).
    c. The app will connect with a Sphero SPRK+

2. Guided Practice:
    a. Press the + icon to start a new program.
    b. Program Sphero to move in a square.
        i. Click “Movement”
        ii. Choose “Roll”
        iii. Determine degrees, speed, duration (e.g., 0, 60, 1)
        iv. Program 4 rolls to make a square
        v. Add delay or stop before changing direction to make corners sharp
    c. Use loops (found under “Control”) to combine instructions
    Replace “0 degrees” with “Heading + 90”
    d. A more detailed version of this activity with instructional videos is found in Sphero edu under “Activities” Blocks 1.
    e. See also https://docs.google.com/document/d/12lqSTZsYQwG-NoaFttnI7KvHECnQigjJntsdL253I9I/edit?usp=sharing

3. Instruction—Computational Thinking

Question: What is computational thinking?
Computational thinking is thinking like a computer scientist. It is an approach to problem solving that includes five main principles:

1. Decomposition: breaking down problems
2. Pattern recognition: organizing data according to patterns
3. Abstraction: representing data in models
4. Algorithms: automating solutions in a series of steps
5. Analysis: analyzing and implementing solutions

Computational thinking also involves the following activities:

- Tinkering
- Creating
- Debugging
- Persevering
- Collaborating

Question: Which of the above activities do you think your students will engage in while using robots?

4. Go over other Sphero know-how
    a. How to charge (the kit is a charging station)

   b. Show the Sphero website and resources
      i. Home https://www.sphero.com/
      ii. Education links https://www.sphero.com/educators
      iii. Educator Guide https://s3.amazonaws.com/static.gosphero.com/downloads/education/SpheroEdu-EducatorGuide2018_FINAL.pdf

5. Plan, using ISTE and Utah Core standards.
   c. Choose a relevant Utah Core Standard, ISTE Standard, or CSTA Standard that you would like students to learn using robots.
   d. Collaboratively plan a lesson or activity using robots to meet the standard.

Utah Core Standards--Educational Technology (Grades 3-5)
https://www.uen.org/core/core.do?courseNum=2030
Standard 8. Use technology resources (e.g., calculators, data collection probes, videos, educational software) for problem-solving, self-directed learning, and extended learning activities.
Standard 9. Determine when technology is useful and select the appropriate tool(s) and technology resources to address a variety of tasks and problems.

ISTE Standards for Students https://www.iste.org/standards/for-students
1c Students use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.
1d Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies.
5c Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.
5d Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

CSTA Standards https://www.csteachers.org/page/standards
1A-AP-08 Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
1A-AP-10 Develop programs with sequences and simple loops, to express ideas or address a problem.

6. Implement: Teachers will have 1-3 weeks to teach their students using robots.

7. Reflect: After implementation, ask the teacher about their experience teaching with robots.

Appendix B: Lesson Plan Library

1. Bots
    a. Ozobots
        i. Getting Started
        ii. Activity 1--Surfaces
        iii. Activity 2--Single Line
        iv. Activity 3--Multiple Lines
        v. Activity 4--Add Codes
    b. Dash
        i. Getting Started
        ii. Path
        iii. Blockly
    c. Sphero
        i. Getting Started
        ii. Draw
        iii. Blocks
2. Unplugged
    a. Decomposition
        i. Program a Partner
        ii. Learn to Dance
        iii. Make a Sandwich
    b. Pattern-finding
        i. Pattern-finding and building
        ii. Patterns in nature
        iii. Patterns in poetry
    c. Abstraction
        i. Phonemes
        ii. Share Fairly
        iii. Drawing
    d. Automation/Algorithms
        i. Getting Started Algorithm
        ii. Train Conducting
        iii. Dance
        iv. Simon Says

Appendix C: District Training

District PD, computational thinking and robots
Badge: http://badgeschool.org/library/search/coding
Bring: Ozobots, markers, code sheets, ipads, spheros, CT handout, laptop, white paper

0:00 introductions

0:10 Ozobot activities (15 minutes)
Activity 1--Surfaces
Activity 2--Single Line
Activity 3--Multiple Lines
Activity 4--Add Codes

0:25 Computational Thinking & Unplugged Activities (20 minutes)
Define computational thinking
Getting Started Algorithm
Program a Partner

0:45 Explore (from badge) (15 minutes)
Explore and consider pros and cons of a few (at least 3-4) different websites or tools for teaching computational thinking and/or coding to student in the age group you plan to teach. Choose one that you feel you would be most likely to try in your classroom. On your submission form, write a few sentences . . .

a. Suggested tools for completing this badge include Code.org (including Hour of Code), Kodable, Scratch, Scratch, Jr., Dash, and Lego.

Using an online computational thinking/coding game platform of your choice, complete enough games to use each of the following concepts: functions (or procedures), loops, and conditions (or conditionals). Take a screenshot demonstrating that you have reached a level where you are using all three, and insert it onto your submission form.

1:00 Break (10 minutes)

1:10 Standards (go over handout--5 minutes)

1:15 Plan (25 minutes)
In grade-level groups, write lesson plans using one standard, one technology

1:40 Share lesson plans

2:00 Time to go--share code:
Text "@TCTES4" to 39242

To report attendance follow prompts
Badge: http://badgeschool.org/library/search/coding

1. Explore and consider pros and cons of a few (at least 3-4) different websites or tools for teaching computational thinking and/or coding to student in the age group you plan to teach. Choose one that you feel you would be most likely to try in your classroom. On your submission form, write a few sentences . . .
    a. Suggested tools for completing this badge include Code.org (including Hour of Code), Kodable, Scratch, Scratch, Jr., Dash, and Lego.
1. Using an online computational thinking/coding game platform of your choice, complete enough games to use each of the following concepts: functions (or procedures), loops, and conditions (or conditionals). Take a screenshot demonstrating that you have reached a level where you are using all three, and insert it onto your submission form.
2. Spend at least 20 minutes engaging as a learner in an unplugged activity that teaches computational thinking concepts. This may be as part of a class or something you do on your own.
    a. Or Spend at least 20 minutes engaging in an activity in which coding principles (such as functions, loops, and conditions) have effects in the real world (outside a computer)--for example, with a codable robot, codable legos, etc.

Teaching Computational Thinking to Elementary School Students

...

Internet code:

RSBNK-YUWHU

Goals:

1. Practice using coding tools
2. Have a basic understanding of computational thinking
3. Make a plan to teach

Today's Plan:

1:15 Introductions

1:25 Ozobots

1:40 Unplugged activities and computational thinking

1:55 Coding online

2:15 Break

2:20 Standards and computational thinking

2:30 Plan lessons

2:50 Share lessons

Badge:

http://badgeschool.org/library/search/coding

Code.org
Kodable
Scratch
Scratch, Jr.

Utah Core Standards--Educational Technology (Grades 3-5)
https://www.uen.org/core/core.do?courseNum=2030

Standard 8. Use technology resources (e.g., calculators, data collection probes, videos, educational software) for problem-solving, self-directed learning, and extended learning activities.

Standard 9. Determine when technology is useful and select the appropriate tool(s) and technology resources to address a variety of tasks and problems.

ISTE Standards for Students
https://www.iste.org/standards/for-students

1c Students use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.

1d Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies.

5c Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.

5d Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

CSTA Standards
https://www.csteachers.org/page/standards

1A-AP-08 Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
1A-AP-10 Develop programs with sequences and simple loops, to express ideas or address a problem.

Text “@TCTES4” to 39242

To report attendance, follow prompts

Appendix D: Train-the-Trainers

Session 1: Computational Thinking and Unplugged Activities (2 hours)

0:00 Introductions (10 minutes)

- Name
- School
- Position
- Goals for training (why you're here)

0:10 Computational Thinking (15 minutes)
Ask class to define Computational Thinking:
Computational thinking is thinking like a computer scientist. It is an approach to problem solving that includes five main principles:

1. Decomposition: breaking down problems
2. Pattern recognition: organizing data according to patterns
3. Abstraction: representing data in models
4. Algorithms: automating solutions in a series of steps
5. Analysis: analyzing and implementing solutions

Computational thinking also involves the following activities:

- Tinkering
- Creating
- Debugging
- Persevering
- Collaborating

Video (3:43): Solving Problems at Google Using Computational Thinking

Discuss: Why teach computational thinking and computing?

- Jobs
- 21st-century skills
- Problem solving skills
- Persistence
- Teamwork
- Soon to be required by state
- Engaging

0:25 Unplugged Activities (30 minutes)
Algorithms

1. Q: What is an algorithm?
2. A: An algorithm is a sequence of instructions or a set of rules to get something done.
3. Can you think examples of algorithms?
4. Examples: Recipes, spelling rules, class routines

5. In groups, make a list of procedures for preparing to use robots.
6. For example,
    a. When you hear the bell ring three times,
    b. Put away everything on your desk.
    c. One partner quietly gets the ozobot.
    d. The other partner quietly gets the markers and paper.
    e. Both of you quietly return to your seats and wait for instructions.
7. Write an algorithm for putting the robots away.

Program a Partner (programs, commands, loops)

1. Q: What is a program?
2. A: A program is a series of algorithms.
3. We have three commands: ⬆ ↱ ↷
4. In teams/partnerships, using only these three commands, as many times as you want, write instructions for someone to start at an agreed upon starting point and end sitting on a designated chair.
5. After writing the instructions, exchange instructions with a different group.
6. Discuss: Did the actual outcome match the expected outcome?
7. If not, debug. Where do you think the program went wrong?
8. You'll probably need to define what each command means.
9. Discuss: How many commands did you give?
    a. Q: How might we write the program more efficiently?
    b. A: Look for patterns
    E.g. ⬆ x 5 (repeat loop)
    c. Q: Where have you seen loops?
        i. Music
        ii. Routines
        iii. Planting a seed
    d. Types of Loops
        i. While a condition is true . . .
        ii. Repeat . . .
        iii. If/then
        iv. If/else

Train Conducting (Conditionals)

1. The teacher thinks up a conditional statement, but does not tell the students what it is.
    - E.g., If the number is a multiple of 3, the train car goes to the left; else, the train car goes to the right.
2. Each student chooses a number and stands in a line to form a train.
3. As each student (i.e., train car) approaches the teacher, they say their number, and
4. the teacher directs the student to the right or the left, based on the condition.
    - E.g., The first student says, "5"; the teacher directs her to the right.
    - The second student says, "9"; the teacher directs him to the left.

5. After all the students have been directed to the right or the left, they try to guess the conditional statement.

0:55 Recap (5 minutes)
With a partner, take turns explaining

- Computational thinking
- Algorithms
- Loops
- Conditionals

1:00 Plan (20 minutes)
In pairs or small groups, plan a lesson to teach one aspect of computational thinking to other teachers. The plan should include the following:

1. Learning objectives for students
2. Learning objectives for teachers
3. Activities for reaching learning objectives
4. Implementation plan
5. Assessment plan
6. Follow-up/support plan

1:20 Practice & feedback (30 minutes)

- Each group teaches their lesson to another group or to the whole class.
- Class/groups provide feedback (e.g., hearts and wishes)

1:50 Reflect (10 minutes)

- What went well with your lesson?
- What would you do differently?
- How might teachers adapt the lesson
    - For different grade levels
    - For ELLs
    - For special needs students
- Which lessons by other groups would you like to use or share?

1:58 Conclude (2 minutes)
Homework: Train teachers at your school in computational thinking and unplugged activities.
Distribute handouts:
Getting Started Algorithm
Program a Partner
Train conducting
Computational Thinking

Additional Resources:
Videos and presentations:

Computational Thinking for Elementary
Computer Science is for Everyone
Robotics
Robotics in Educational Technology

Articles:
The 5th ‘C’ of 21st Century Skills? Try Computational Thinking (Not Coding)
Using Robots to Teach Computer Science and Computational Thinking
Educational Robots and Computational Thinking (Click “View full-text”)
Computational Thinking in Schools
Computational Thinking Glossary
What is Computational Thinking?

Websites
Code.org
iste.org/computational-thinking
Exploring computational thinking
CS unplugged

Session 2 Ozobots (2 hours)

Materials needed:

- Ozobots
- Markers (one set for each Ozobot)
- White paper (one sheet per person)
- Coding sheets

0:00 Review

- Computational thinking
- Algorithms
- Loops
- Conditionals

0:05 Share & Reflect
Have trainers share their experiences training teachers:

- What went well?
- What would trainers do differently?
- What questions, issues, challenges came up?

0:15 Model

- Following a “Getting Started” algorithm written in Session 1,
- Distribute Ozobots, markers, and paper to each table.
- Demonstrate how to turn on the Ozobot Bit (press button on the side).

Activity 1—Surfaces

1. Place the Ozobot various surfaces::
    a. Dark surfaces
    b. Light surfaces
    c. Sloped surfaces
    d. Carpet
    e. A white piece of paper
2. What did you observe?
3. What do you know about Ozobot from what you observed?

Activity 2—Lines

1. Draw a couple of lines on a white piece of paper.
2. Place the bot on a line.
3. See if the Ozobot follows the line.
    a. If not, troubleshoot (debug)—why didn’t it work?
    b. Is the Ozobot lit up?
    c. Is the line thick enough?
    d. Is the line unbroken and fairly straight?

4. Experiment with different lines to figure out what lines work best.
5. Tips are available here: https://files.Ozobot.com/stem-education/Ozobot-tips.pdf

Activity 3—Adding codes

1. Draw a short, straight, black line.
2. Choose a code (a pattern) from the code sheet: https://play.Ozobot.com/print/guides/Ozobot-color-codes-reference.pdf
3. Copy it onto your paper.
    a. Start where the black line ends.
    b. Make your code look as much like the code sheet as possible.
    c. The colored squares should be touching, with no white space between.
    d. The colored squares should be about ¼" by ¼".
    e. At the end of the code, draw another black line.
4. Turn on Ozobot, place it at the start, see if it follows the code.
5. Start the Ozobot at the other end.
    a. What happened?
    b. Why? (notice that some codes are the same forward and backwards; other patterns might mean one thing from left to right and another thing when going from right to left)
6. Draw more lines and codes and test them.
7. When code doesn't work, troubleshoot—try to figure out why it's not working
    a. Is the bot on?
    b. Try calibrating the bot.
    c. Make sure the line is an appropriate width (3-5 mm).
    d. Does the code look like the codes on the code sheet?
    e. Is there a line for the bot to follow after the code? (don't end a line with a code)
    f. Do the lines allow the bot to follow the instructions? (e.g., if you told the bot to turn left, did you provide a line going left, along with at least one other option?)

0:40 Ozobot know-how and resources (15 minutes)

a. How to charge (charging station included in kit)
b. How to calibrate (turn on, set on large, black circle)
c. Show the Ozobot website and resources
    i. Home https://Ozobot.com/
    ii. Education links https://Ozobot.com/stem-education
    iii. Getting started https://Ozobot.com/stem-education/education-getting-started
    iv. Lesson plans https://portal.Ozobot.com/lessons
        1. Color codes https://portal.Ozobot.com/lessons/compilation/color-codes-basic-training
        2. Ozoblockly (more advanced programming, for older students) https://portal.Ozobot.com/lessons/compilation/ozoblockly-basic-training
        3. Deconstruction (for use with Evo) https://portal.Ozobot.com/lessons/compilation/evo-deconstruction-series

vi. Games https://Ozobot.com/play/print-games

0:55 Share standards related to computational thinking, coding, and robots
Utah Core Standards--Educational Technology (Grades 3-5)
https://www.uen.org/core/core.do?courseNum=2030
Standard 8. Use technology resources (e.g., calculators, data collection probes, videos, educational software) for problem-solving, self-directed learning, and extended learning activities.
Standard 9. Determine when technology is useful and select the appropriate tool(s) and technology resources to address a variety of tasks and problems.

ISTE Standards for Students https://www.iste.org/standards/for-students
1c Students use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.
1d Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies.
5c Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.
5d Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

CSTA Standards https://www.csteachers.org/page/standards
1A-AP-08 Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
1A-AP-10 Develop programs with sequences and simple loops, to express ideas or address a problem.

1:00 Plan (20 minutes)

a. Choose a relevant Utah Core Standard, ISTE Standard, or CSTA Standard that you would like students to learn using Ozobots.
b. Collaboratively plan a lesson or activity using Ozobots to meet the standard. The plan should include the following:
    i. Learning objectives for students
    ii. Learning objectives for teachers
    iii. Activities for reaching learning objectives
    iv. Implementation plan
    v. Assessment plan
    vi. Follow-up/support plan

1:20 Practice & feedback (30 minutes)

- Each group teaches their lesson to another group or to the whole class.
- Class/groups provide feedback (e.g., hearts and wishes)

1:50 Reflect (10 minutes)

- What went well with your lesson?

- What would you do differently?
- How might teachers adapt the lesson
    - For different grade levels
    - For ELLs
    - For special needs students
- Which lessons by other groups would you like to use or share?

1:58 Conclude (2 minutes)
Homework: Train teachers at your school in computational thinking and unplugged activities.
Distribute handouts:
Ozobots activities
Standards

Additional Resources
Ozobots overview (kyte)
How to Teach with Ozobots (webinar)

Session 3: Dash robots (2 hours)
Materials needed:

- Dash robots
- iPads with Path and Blockly apps

0:00 Share & Reflect (10 minutes)
Have trainers share their experiences training teachers:

- What went well?
- What would trainers do differently?
- What questions, issues, challenges came up?

0:10 Model (5 minutes)

- Following a “Getting Started” algorithm written in Session 1,
    - distribute Dash robots and iPads to each table.
- Demonstrate how to turn on the Dash robot (press power button on the side).
- Demonstrate how to find a Dash app on an iPad.
    - Path, K-2
    - Blockly, grades 2-6
- The app will connect to a Dash robot.

0:15 Guided Practice (20 minutes)
Providing assistance as needed, let trainers play with Path and Blockly (10 minutes each).

Path
Overview: Draw a path with your finger, then drag and drop icons onto the path. Each icon represents an action or sound. Press the picture of the Dash robot, and Dash will follow the path and complete the actions. Complete tasks to unlock new icons and screens. All instructions are visual; no reading skills are necessary.

a. When you first use Path, you’ll see a screen with a triangle in the middle; press the triangle.
b. If you see the + icon in the top right corner, then press it to connect to your Dash robot. You’ll need to turn the robot on by pressing the power button on the side.
c. You will then see a screen with a lit-up square showing Dash. Tap the box.
d. With your finger, draw a path for Dash to follow.
e. Tap the picture of Dash on the screen, and the robot will follow the path. Drag and drop icons along the path, and Dash will do the actions.
f. After you practice drawing paths and adding actions, you will unlock the second screen: a racetrack.
g. Draw a path. The instructions at the top center tell you which icons to add to the path and in what order. The icons appear at the bottom of the screen. Drag icons from the bottom of the screen to the path.
h. Complete several challenges to unlock the next screen: a farmyard.

i. There are four screens in Path; all four work the same but have different icons (sounds and actions).

Blockly

Overview: Learn to program Dash using Blockly language. Complete 13 puzzles, each of which is broken down into several steps. Also create your own programs.

a. When you first use Blockly, you'll see a screen with a triangle in the middle; press the triangle.
b. If you see the + icon in the top right corner, then press it to connect to your Dash robot. You'll need to turn the robot on by pressing the power button on the side.
c. You will then see a screen asking you to choose Dash or Dot.
d. You will be taken to the Puzzles page. Select the first puzzle.
e. Note: if you are the first to use the program on your iPad, the first puzzle will be the only one that is unlocked. However, if a previous user has unlocked other puzzles, you should still start with the first.
f. The first puzzle, Driving School, will teach you the basics of programming with Blockly.
g. You will be given instructions that become increasingly more challenging. The first task is simply to press START.
h. After the instructions are given, the next screen will look something like this, with the instructions along the bottom. The tabs needed to complete the task (left column) will be highlighted. Select the appropriate blocks and snap them into place. After completing each task, press START (lower left corner).
i. If the task is incomplete or incorrect, you will be told that there's a problem.
j. When you complete all 12 tasks, you have completed the first puzzle.
k. After completing Puzzle 1, you may choose to create your own project or go to the next puzzle.

0:35 Dash know-how and resources (20 minutes)

a. How to charge (the charging cord should be in the box)
b. Show the Dash website and resources:
    i. Home https://www.makewonder.com/dash
    ii. Education links https://education.makewonder.com/
    iii. Curriculum https://education.makewonder.com/curriculum
        1. Learn to Code https://education.makewonder.com/curriculum
            a. 6 levels (A-F)
            b. 6 lessons per level
        2. Code to Learn https://education.makewonder.com/curriculum/code-to-learn
            a. Lesson plans submitted by teachers
            b. Select by subject, grade level, robot (Dash or Dot)

0:55 Review Standards related to computational thinking, coding, or robots (5 minutes)

1:00 Plan (20 minutes)

a. Choose a relevant Utah Core Standard, ISTE Standard, or CSTA Standard that you would like students to learn using Ozobots.
b. Collaboratively plan a lesson or activity using Ozobots to meet the standard. The plan should include the following:
    i. Learning objectives for students
    ii. Learning objectives for teachers
    iii. Activities for reaching learning objectives
    iv. Implementation plan
    v. Assessment plan
    vi. Follow-up/support plan

1:20 Practice & feedback (30 minutes)

- Each group teaches their lesson to another group or to the whole class.
- Class/groups provide feedback (e.g., hearts and wishes)

1:50 Reflect (10 minutes)

- What went well with your lesson?
- What would you do differently?
- How might teachers adapt the lesson
    - For different grade levels
    - For ELLs
    - For special needs students
- Which lessons by other groups would you like to use or share?

1:58 Conclude (2 minutes)

- Homework: Train teachers at your school in using Dash robots to teach computational thinking and meet standards.

Share handouts/links:
Path lesson
Blockly lesson
Standards

Additional Resources
Dash robots (kyte)

Session 4: Sphero SPRK+ (2 hours)

Materials needed:

- Sphero SPRK+ robots
- iPads or Chromebooks with Sphero edu
- Wifi
- Bluetooth

0:00 Share & Reflect (10 minutes)
Have trainers share their experiences training teachers:

- What went well?
- What would trainers do differently?
- What questions, issues, challenges came up?

0:10 Model (5 minutes)

- Following a "Getting Started" algorithm written in Session 1,
    - distribute Spheros and iPads to each table.
- Demonstrate how to find Sphero edu app on an iPad or Chromebook.
- Trainers should set up an account (teachers and students will also need accounts).
- The app will connect with a Sphero SPRK+

0:15 Guided Practice (20 minutes)

a. Press the + icon to start a new program.
b. Program Sphero to move in a square.
    i. Click "Movement"
    ii. Choose "Roll"
    iii. Determine degrees, speed, duration (e.g., 0, 60, 1)
    iv. Program 4 rolls to make a square
    v. Add delay or stop before changing direction to make corners sharp
c. Use loops (found under "Control") to combine instructions
   Replace "0 degrees" with "Heading + 90"
d. A more detailed version of this activity with instructional videos is found in Sphero edu under "Activities" Blocks 1.
e. See also https://docs.google.com/document/d/12lqSTZsYQwG-NoaFttnI7KvHECnQigjJntsdL253I9I/edit?usp=sharing

0:35 Sphero Know-how (20 minutes)

a. How to charge (the kit is a charging station)
b. Spend time exploring Sphero website and resources
    i. Home
    ii. Education links
    iii. Educator Guide

0:55 Review Standards related to computational thinking, coding, or robots (5 minutes)

1:00 Plan (20 minutes)

a. Choose a relevant Utah Core Standard, ISTE Standard, or CSTA Standard that you would like students to learn using Ozobots.
b. Collaboratively plan a lesson or activity using Ozobots to meet the standard. The plan should include the following:
    i. Learning objectives for students
    ii. Learning objectives for teachers
    iii. Activities for reaching learning objectives
    iv. Implementation plan
    v. Assessment plan
    vi. Follow-up/support plan

1:20 Practice & feedback (30 minutes)

- Each group teaches their lesson to another group or to the whole class.
- Class/groups provide feedback (e.g., hearts and wishes)

1:50 Reflect (10 minutes)

- What went well with your lesson?
- What would you do differently?
- How might teachers adapt the lesson
    - For different grade levels
    - For ELLs
    - For special needs students
- Which lessons by other groups would you like to use or share?

1:58 Conclude (2 minutes)
Homework: Train teachers at your school in Sphero robots and standards.
Share handouts/links:
Sphero Blocks Lesson
Standards

Additional Resources
Sphero (kyte)
Sphero Activities
Sphero Programs
Sphero Youtube
Sphero Educator Guide

Appendix E: Content Analysis

The three main goals of in-school and district trainings were to help teachers (1) feel comfortable using robots and online tools (2) gain a basic understanding of computational thinking, and (3) plan how to teach computational thinking using robots or online tools. The in-school training focuses entirely on robots; the district training includes both robots and websites such as Code.org.

To feel comfortable using robots, teachers must know how to do each of the following:

1. Turn the robot on and off.
2. Charge the robot.
3. Program the robot.
4. Access appropriate apps.
5. Use the accompanying apps.

Teachers will be given time to play with robots and will be guided through the above tasks.

Similarly, to feel comfortable using online tools such as code.org, teachers should know how to do each of the following:

1. Access the website.
2. Create a teacher account.
3. Create student accounts.
4. Access student activities.
5. Use the online tool to code or program.

To meet these objectives, teachers will be given time to use online tools, and will be supported through the above tasks as needed.

To meet the second goal of training, teachers should have a working definition of computational thinking. For this project, I have defined computational thinking as an approach to problem solving that involves decomposition, pattern recognition, abstraction, algorithms, and analysis (CAS, 2014; Google. n.d.; ISTE, 2014). Computational thinking often incorporates tinkering, creating, debugging, persevering, and collaborating (CAS, 2014), and can be developed through coding, computing, and using robots. In in-school and district training, I share this definition with teachers, share examples and activities, and ask teachers to share examples.

To meet the third goal of training, “plan how to teach computational thinking using robots or online tools,” teachers were given time and support in planning lessons. Teachers also need to be familiar with relevant learning standards. To that end, I shared a handout listing relevant Utah, ISTE, and CSTA learning standards. To show that they have met this objective, teachers were asked to share their plans. I also shared lesson plans with teachers.

The train-the-trainers workshop has additional objectives. Besides being comfortable with the technology, understanding computational thinking, and having a plan to teach computational thinking, trainers also need a plan to train and support other teachers. Their plans should include the following:

1. Learning objectives for students
2. Learning objectives for teachers
3. Activities for reaching learning objectives
4. Implementation plan

5. Assessment plan
6. Follow-up/support plan

In the train-the-trainers workshop, I will guide participants through the planning process. Participants will be asked to share their plans. We will also schedule follow-up interviews where I ask trainers how the training has gone and provide support as needed.

Appendix F: Insights from Research

Because my project is designing professional development to provide teachers the skills and confidence to teach computational thinking using robots, I will discuss insights from the literature on three topics: computational thinking, self-efficacy, and professional development for technology education.

Computational Thinking

A primary goal of TPD4CT is to help teachers understand computational thinking so that they can teach their students computational thinking skills. According to Grover and Pea (2013), computational thinking is "viewed as at the core of all STEM disciplines" (p. 38). Being able to think computationally helps students perform well in other STEM disciplines (Rich et al., 2017). Put simply, computational thinking is "thinking like a computer scientist" (Wing, 2006, p.35). Despite the simple definition, computational thinking is a broad term, with multiple definitions (Barr & Stephenson, 2011; Jaipal-Jamani, Angeli, 2017; Sadik, Ottenbreit-Leftwich, and Nadiruzzaman, 2017). Wing (2006) further explained, CT is "thinking recursively," "using and abstraction and decomposition," and "reformulating a seemingly difficult problem into one we know how to solve," and "involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (p. 33). The Computer Science Teachers Association (CSTA), defines computational thinking as "The human ability to formulate problems so that their solutions can be represented as computational steps or algorithms to be executed by a computer." (CSTA Glossary; Lee, 2016, p. 3). For Lee, CT is using a computer to solve problems, whereas other definitions allow practitioners to apply components of CT (e.g., algorithms or pattern-finding) without using computers or solving problems.

For this project, I have defined computational thinking as an approach to problem solving that involves decomposition, pattern recognition, abstraction, algorithms, and analysis (Appendix A; CAS, 2014; Google. n.d.; ISTE, 2014). Computational thinking often incorporates tinkering, creating, debugging, persevering, and collaborating (CAS, 2014), and can be developed through coding, computing, and using robots.

Self-Efficacy

As teachers learn about computational thinking and practice using robots, I expect that their technology self-efficacy will increase. Perceived self-efficacy is a judgement of one's ability to perform (Bandura, 1977). Teacher self-efficacy is a teacher's judgment of her ability to teach. Technology self-efficacy (TSE) is a person's belief that "he/she will be successful in using the technology" (Holden & Rada, 2011, p. 347). Teacher technology self-efficacy, then, is a teacher's judgment of her ability to use technology to teach.

Self-efficacy is distinct from what Bandura (1977) called outcomes expectations, the belief that specific behavior will lead to specific outcomes. Both efficacy beliefs and outcome expectations influence a person's likelihood to do things. For example, a teacher who is confident in their abilities to use computers and teach effectively might choose not to use robots in instruction if they don't expect that using robots will help students to learn what they need to learn. Similarly, if the teacher thinks that using robots will help students gain important skills, but she lacks confidence in her ability to use robots to teach, she again might choose not to teach using

robots.

According to Ertmer and Ottenbreit-Leftwich (2010), “self-efficacy may be more important than skills and knowledge among teachers who implement technology in their classrooms” (p. 261). Teachers with low technology self-efficacy are less likely to use technology than are confident teachers (Holden & Rada, 2011; Vannatta & Fordham, 2004). Experience is key to developing technology self-efficacy. Teacher trainers help teachers gain self-efficacy by providing time to play (Somekh, 2008) and facilitating positive experiences with technology in the classroom (Mueller, Wood, Willoughby, Ross, & Specht, 2008). Effective teacher training must provide learning experiences that build self-efficacy.

Professional Development

Experts have described effective PD as school-based, active, long-term, with follow-up, collective participation of grade-level teams, and a focus on content knowledge or pedagogic practice (Guskey and Yoon, 2009; Avalos, 2011; Odden & Picus, 2014). In their discussion of the literature, Ertmer and Ottenbreit-Leftwich (2010) have suggested that to use technology as “meaningful pedagogical tools,” teachers need to expect that technology will be integral to instruction, and they need to know how to use technology to facilitate student-centered instruction (p. 255). Preservice training should include numerous and varied examples and models, and opportunities for practice. Professional development programs should build on teachers’ pedagogical content knowledge (PCK) and “include information about how they can use these tools in very specific ways, within specific content domains, to increase student content learning outcomes” (p. 272).

Few studies have focused on PD for teaching computing. In a review of 15 studies related to teacher training for teaching elementary computing (Mason, working paper), eight studies showed improvements in self-efficacy or attitudes toward teaching computer science (Bers, Seddighin, & Sullivan 2013; Carter et al., 2014; Jaipal-Jamani & Angeli, 2017; Jeon & Kim, 2017; Kim et al., 2015; Ma, Lai, Williams, Prejean, & Ford, 2008; Rich et al., 2017; Toikkanen & Leinonen, 2017). Nine studies showed evidence of increased knowledge, understanding, or performance (Bers et al, 2013; Cetin, 2016; Cetin and Andrews-Larson, 2016; Coleman et al., 2016; Jaipal-Jamani & Angeli, 2017; Ma et al., 2008; Ng, 2017; Sadik et al., 2017; Yadav, Mayfield, Zhou, Hambrusch, & Korb, 2014). Both limited and extensive trainings yielded results. For example, Jaipal-Jamani and Angeli (2017) found that after completing only six hours of robotics instruction and activities, preservice elementary school teachers’ STEM engagement increased significantly. And in a study by Rich et al. (2017), elementary teachers who participated in year-long professional development reported significantly higher self-efficacy and more positive beliefs toward computing and engineering compared to teachers who had not participated in PD, though self-efficacy varied widely by teacher background, level of implementation, and willingness to experiment.

Appendix G: Project Management

Timeline

Milestone	**Completion Date**
Complete Lit Review & Task Analysis	March 25, 2018
Complete Needs Analysis & Prospectus Write-Up	May 1, 2018
Design in-school training	May 10, 2018
Implement in-school training	May 15, 2018
Design training workshop	June 5, 2018
Finish Project Write-Up (draft to Rick West)	June 10, 2018

Budget

Graduate Student Wages	$20 x 125 hours	$2500
Teacher Pay	$25 x 14 hours x 20 participants	$7000
Robots		$5000
Sphero Power Pack	$1800	
Dash kit	$2000	
Ozobots kit	$1200	
iPads	$500 x 10	$5000
Total		$19,500

Robots, iPads, and graduate student wages were paid for through a private donation to BYU. Teacher pay was provided by Provo School District as per teacher contracts.

Appendix H: Assessment Instruments

The purpose of the surveys was to help me understand the learning objectives teachers taught using the robots, the obstacles and challenges to implementation teachers faced, the influence training had on teachers' technology self-efficacy, the perceived benefits of using robots in the classroom, and the types of professional development teachers valued. Survey I, administered before training, included one multiple-choice question, two open-ended questions, and seven five-point Likert items. Survey II, administered after implementation, included one ordering question, three open-ended questions, and eight five-point Likert items. Five items on pre and posttests were identical and were used to indicate changes in attitude or self-efficacy. Four items were almost identical on pre and posttests, but differed in tense (e.g., "What do you expect to happen" vs. "What actually happened?"). Survey I included one question about objectives that was most relevant before training and implementation, and therefore not included on Survey II. Survey II included two questions about professional development preferences and one question about future plans that would have been less relevant before implementation, and therefore were not included on Survey I.

Informal interviews were conducted during training sessions and after implementation, when I retrieved the technology sets from teachers. Similar to the purpose of the surveys, the purpose of initial informal interviews was to help me understand what teachers needed and wanted from the training session.

Survey I

1. What is your reason for borrowing the robots? (Mark all that apply)

a. To learn how to use them myself

b. To teach students using the robots

c. To share the robots with other teachers

For questions 2-8, please indicate how strongly you agree or disagree with each statement:

2. I am an effective teacher.

Strongly disagree Disagree Neutral Agree Strongly Agree

3. I know how to operate _____ (Ozobots, Spheros, or Dash robots).

Strongly disagree Disagree Neutral Agree Strongly Agree

4. I am confident in my ability to use robots in the classroom to meet specific learning objectives.

Strongly disagree Disagree Neutral Agree Strongly Agree

5. It is important for students to learn computational thinking skills (e.g., pattern-finding, algorithms, debugging).

Strongly disagree Disagree Neutral Agree Strongly Agree

6. Using robots in the classroom will help students learn computational thinking skills.

Strongly disagree Disagree Neutral Agree Strongly Agree

7. Helping K-6 students gain digital literacy skills is important.

Strongly disagree Disagree Neutral Agree Strongly Agree

8. Using robots in the classroom will help students gain digital literacy skills.

Strongly disagree Disagree Neutral Agree Strongly Agree

Open-ended questions

9. What objectives do you hope to achieve by using robots in your classroom?

10. What obstacles are there to using robots in your classroom?

Survey II

For questions 1-8, please indicate how strongly you agree or disagree with each of the following statements:

1. I am an effective teacher.

Strongly disagree Disagree Neutral Agree Strongly Agree

2. I know how to operate _____ (Ozobots, Spheros, or Dash robots).

Strongly disagree Disagree Neutral Agree Strongly Agree

3. I am confident in my ability to use robots in the classroom to meet specific learning objectives.

Strongly disagree Disagree Neutral Agree Strongly Agree

4. It is important for students to learn computational thinking skills (e.g., decomposition, pattern-finding, algorithms).

Strongly disagree Disagree Neutral Agree Strongly Agree

5. Using robots in the classroom helped students learn computational thinking skills.

Strongly disagree Disagree Neutral Agree Strongly Agree

6. Helping K-6 students gain digital literacy skills is important.

Strongly disagree Disagree Neutral Agree Strongly Agree

7. Using robots in the classroom helped students gain digital literacy skills.

Strongly disagree Disagree Neutral Agree Strongly Agree

8. I am planning to use the robots again to teach.

Strongly disagree Disagree Neutral Agree Strongly Agree

Open-ended questions

9. What learning objectives did you achieve by using robots in your classroom?

10. What challenges did you experience using robots in your classroom?

11. What sort of future training or support would best help you to use robots to meet specific learning objectives?

Ordering question

12. Which activities have best helped you prepare to use robots to meet specific learning objectives? Rank the following activities from most helpful (1) to least helpful (7):

____ Provo School District training

____ In-school training with a BYU graduate student (Stacie Mason) when the robots were delivered

____ Other school-sponsored training

____ Personal research and planning

____ PLC collaboration and planning

____ Teaching my students using robots

____ Using the robots on my own

Sample Informal Interview Questions

Before Training:

1. What are your plans for using the technology kits in your classroom?
2. What are your learning objectives?
3. How have you taught these learning objectives before?
4. How do you think using the technology kits will help in students learning?
5. Have you used robots before?
6. How have you taught computational thinking before?
7. How can I help?

After Training:

1. What went well?
2. What didn't go well?
3. What would you do differently?
4. What are your plans for using the technology kits in the future?
5. What additional training do you wish you had had before using the technology in your classroom?
6. What additional training would you like to have before using technology kits again?

Appendix I: Implementation Report

For the pilot study of the in-school training, I met with five teachers to provide training. Before training, each teacher was asked to sign an informed consent form and complete Survey I. Below I describe the five training sessions.

Annie

The first teacher I worked with was Annie, a third-grade teacher in her third year at Lakeview. Our training session was 30 minutes. I demonstrated how to turn on the Ozobot, gave Annie an Ozobot, and guided her through a series of inquiry-based activities. First, I encouraged Annie to try the Ozobot on various surfaces. She noted that the Ozobot moved around on the floor, table, and cardboard, but stood still on a plain, white paper. Second, I gave Annie a piece of paper and markers and suggested she draw a line or two. She drew green and blue lines, put the Ozobot on a line, and saw that the Ozobot followed the lines and changed color to match the color of the line. Third, I gave Annie a coding sheet, encouraged her to draw lines with code, and instructed her to make her codes look as much like the coding sheet as she could. Annie drew a code that made the Ozobot slow down. I suggested she try the Ozobot going the opposite direction; this time after reading the code the Ozobot sped up. Annie looked at the code sheet and saw that going the opposite direction, the code she had drawn was “turbo.”

After this practice portion of our training, I summarized the principles of computational thinking and asked Annie for examples of CT principles in other parts of the curriculum. We discussed algorithms as sets of instructions, and I suggested that the class could write a “getting started” algorithm describing what to do to get ready to use Ozobot s. I suggested that this activity could help facilitate classroom management, and that using terms like “algorithm” would help students learn CT terminology.

For the planning portion of training, I shared written plans for the practice activities we had done and online lesson plans available at Ozobot.com. After a few minutes of lesson planning, Annie asked if she could draw more. She drew a zigzag code, but the bot stopped after the code, so we practiced debugging. Annie drew a line coming off the main line, and the bot followed the line. I asked Annie if she felt comfortable using the robots had enough ideas for teaching with the robots, and she said she did. We agreed to be in touch in a couple of weeks when it was time for me to pick up the robots, and I encouraged her to email if she had questions. Over three weeks, her students spent about two hours using the bots. Annie would have liked to do more of the online lessons but lacked time.

Becca

Becca, an experienced fourth grade teacher at Spring Creek Elementary, planned to use robots for one hour with each of the three classes of fourth graders at her school. Two months before our meeting, Becca had attended a two-hour district-sponsored training session in which she learned about computational thinking, did a few unplugged activities, and spent 30 minutes using Dash and Sphero robots (about 15 minutes each). She requested Dash robots for her classroom, but since they were not available, she agreed to use Sphero robots. When I delivered the robots, she admitted that she had had a hard time controlling them at the district training session and was a little worried. Unfortunately, our training session did not assuage her worries.

Becca had scheduled our meeting for 8:00 am on a school day. The students arrived at 8:35, and Becca was planning to use the robots during her STEAM hour, starting at 9:00. Sphero requires the use of an app—we were using Sphero edu. Although the app connects to the robot using Bluetooth, it also requires Wi-fi to set up and sign into accounts. With 12 university-owned iPads that were not district iPads, logging into Wifi was complicated. We decided to use the school's Chromebooks instead. Once we had made that decision, I spent much of our remaining hour setting up accounts and making sure they worked. Becca went through an online Sphero lesson ("Blocks 1"), approved the accounts I set up, and arranged with another teacher to take her students until 9:00. We logged into the app to see what the students would experience when they logged in and talked through Becca's plan for teaching the students. I suggested that the class could make a list of procedures (i.e., an algorithm) before distributing Chromebooks and robots. Becca was planning to have students work with partners, so we used a paired programming strategy where one student gives instructions while the other follows instructions.

Becca reported that all but one of the robots ran out of batteries almost immediately. The next week, again several quit working, after which she switched to Ozobots and met a second time to spend 10 minutes going through the Ozobot activities. Despite the setbacks, Becca remained positive, enjoyed the Ozobots, and indicated she would like to use them again.

Cathy

Cathy, an experienced first grade teacher, planned to use Dash robots with her students during their invention unit. She had invited two other first grade teachers to our 40-minute training session, though Cathy was the only one of the three planning to use the robots. Giving each teacher a robot and an iPad and some instruction on how to find the "Path" app, they played with the robots, with minimal help available as needed. As they played with the robots, one teacher commented that they were pretty noisy. Cathy had planned to use eight robots in the classroom but was rethinking her plan—maybe she would have her 17 students work in groups instead of pairs. After about 15 minutes of practice, we discussed the teachers' goals and objectives for using the robots. Cathy said that she liked that robots teach persistence—that the students would try and fail and try again. The teachers also thought the robots were fun. In the Path app, there are a few different options for users: Cathy had chosen the second option (a racetrack), while the other two teachers had chosen the first. We turned the robots back on, everyone tried the racetrack option, and we discussed pros and cons for instructing the students to use one or the other versus letting them choose. Then we tried a second app, Blockly, and agreed that it required too much reading for first graders.

After our practice session and some instruction on computational thinking, the teachers discussed algorithms and learned about resources from makewonder.com: a planning page, a reflection page, and arrows that could be used for an unplugged activity. Cathy and her students had fun with the robots and would have liked to have more time with them.

Deb

Deb was a second-grade teacher in her first year of teaching. During a practicum, she had seen a class using Ozobots, so when she had the chance to use them she signed up but lacked a plan for teaching with them. When she first tried the Ozobot, she noticed it stopped on the white paper and that it always faced the same direction, remarking, "That's the front of it."

Next, Deb drew two connected lines, both with the green marker, and put the robot on one line. The robot followed the first line but stopped at the turn. She then drew three, short, thin, unconnected lines. The Ozobot followed one of the lines well but did not follow the thinnest at all and stopped in the middle of the third. Deb remarked that the thinnest was probably too thin. The lead author, leading the workshop, then suggested that depending on whether she preferred direct instruction or inquiry, she could demonstrate to her students how to draw a line the Ozobot could follow or have them experiment as she had done.

Next, Deb attempted to draw a line with code using some provided coding sheets. Deb struggled with getting her robot to both follow the line and her code without assistance. However, when asked what sorts of lesson objectives she might use the Ozobots to teach, she said she noticed a lot of patterns. After some instruction about computational thinking as an approach to problem solving that includes abstraction, algorithms, pattern finding, etc., and involves debugging and encourages persistence, the workshop facilitator shared lesson plans for the activities that we had done and had her look at the Ozobot website resources, including color coding lessons, blockly lessons, and getting started resources. In their 10 days with the bots, Deb's students experimented with lines and codes, made a race track, and played a space game she found online.

Evelyn

The fifth teacher was Evelyn, an experienced kindergarten teacher at LV. She had demonstrated Dash robots for other teachers at her school. Evelyn explained that she had shown her students a Dash robot and they were eager to use them. I helped her unpack robots and showed her how to find the Path app on the iPad. She turned on a robot and opened the app but did not know what to do next. I told her to draw a path, and she asked what to do next. I told her to press the picture of the robot, and the robot followed the path. Then I suggested she return to the home page and choose the race car picture. I told her to draw a path. She added a sound but had not followed the visual instructions as to which sound should come first. I explained that the pictures were instructions, and the program wanted her to follow them in order.

After Evelyn practiced with the robot, I shared the ISTE and CSTA standards and explained computational thinking. I suggested that she and her class could create a "getting started" algorithm and a "putting robots away" algorithm. Evelyn seemed to like the idea and took notes. I described a couple of other unplugged activities: programming a partner and train conducting. After sharing the unplugged activities, I returned to the home page and said that Evelyn may want to instruct her students on where to start. Evelyn said that she would demonstrate on the projector what to do. I also showed Evelyn a more basic app, Go, in which users drive the robot as they would a remote-control car. Evelyn said she preferred Path.

Evelyn mentioned that the school owns four Dash robots, which they initially bought for use with gifted and talented students, but that when she demonstrated Dash robots for other faculty, they thought they were for young students only. Instead, she learned about Blockly, which provided better extension for older students. When asked what additional training she would like, she said she had some ideas of how to use the robots with her class, and that she would have some older students come in during lunch to play with them so they could tell their teachers about them. After using the robots for a week, Evelyn reported that her students had had a lot of fun.

Appendix J: Reflection and Critique

One thing I learned about the field is that it moves quickly. While relatively few studies have been published to date, that is changing. I don't know that I can keep up. I want to do work that makes a difference and with this project I thought I could. When I started, the district was not teaching computational thinking on a broad scale. But while I was planning my small project, someone employed by the district was planning a similar, bigger, better-funded project. I'm glad that the district is moving toward district-wide instruction in computational thinking, and this project was certainly a learning experience for me and I think for the participants, but I don't know that the project will have a big, long-term impact.

One thing I have learned about the design process is the importance of good planning. I tend to want to jump in and design a product that is interesting to me, but without proper planning, that product is unlikely to be worthwhile and get used. From the beginning of the planning stage to the submission of a proposal, the scope of the projected shifted slightly to meet the needs of the district. It's impossible to plan everything, but planning a basic framework for the in-school training saved time during the design process. I wished I had planned the format more thoroughly so I would not have wasted time repeatedly reformatting documents. Planning in the form of researching what works and what has been done allowed me to design something useful and worthwhile.

A final lesson I learned is that good design involves collaboration. In this project, I worked mostly alone. My funding had very few strings attached, which allowed a good deal of freedom. However, I could not have done the project without help from other people. I had a connection to Spring Creek—my older children attended the school—which may have helped me work with teachers there. I met with Karen Brock, who connected me with the Lakeview principal. The Lakeview principal contacted Ron Twitchell, who almost immediately approved the pilot project and collection of data in Provo schools. Without the teachers who signed up and attended trainings I would have had no pilot test nor data. Rick West hired me for an internship, without which I never would have done this project; helped me determine the products; and gave me the opportunity to do the district training. Peter Rich shared information and resources, and allowed me to observe an in-school training. Connections help us get things done, and done well.

References

Avalos, B. (2011). Teacher professional development in teaching and teacher education over ten years. *Teaching and Teacher Education*, *27*(1), 10-20.

Bandura, A. (1977). Self-efficacy: Toward a unifying theory of behavioral change. *Psychological Review*, *84*(2), 191.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is Involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48-54.

Benitti, F. B. V. (2012). Exploring the educational potential of robotics in schools: A systematic review. *Computers & Education*, *58*(3), 978-988.

Bers, M. U., Seddighin, S., & Sullivan, A. (2013). Ready for robotics: Bringing together the T and E of STEM in early childhood teacher education. *Journal of Technology and Teacher Education, 21*(3), 355-377.

Bureau of Labor Statistics (2018, Jan. 30). Computer and information technology. Occupational Outlook Handbook. Retrieved from https://www.bls.gov/ooh/computer-and-information-technology/home.htm

Bureau of Labor Statistics (2018, April 11). Employment Projections. https://www.bls.gov/emp/ep_table_104.htm

Carter, A., Cotten, S. R., Gibson, P., O'Neal, L. J., Simoni, Z., Stringer, K., & Watkins, L. S. (2014). Integrating Computing Across the Curriculum: Incorporating technology into STEM education. In Z. Yang, H. H. Yange, D. Wu & S. Liu (Eds.), Transforming K-12 classrooms with digital technology (pp. 165–192). Hershey, PA: IGI Global.

CAS Barefoot (2014). *Computational thinking.* Retrieved from https://barefootcas.org.uk/barefoot-primary-computing-resources/concepts/computational-thinking/.

Cetin, I., & Andrews-Larson, C. (2016). Learning sorting algorithms through visualization construction. *ComputerScience Education*, *26*(1), 27-43. doi:10.1080/08993408.2016.1160664

Cetin, I. (2016). Preservice teachers' introduction to computing: Exploring utilization of scratch. *Journal of Educational Computing Research, 54*(7), 997-1021. 10.1177/0735633116642774

Coleman, L. O., Gibson, P., Cotten, S. R., Howell-Moroney, M., & Stringer, K. (2016). Integrating computing across the curriculum: The impact of internal barriers and training intensity on computer integration in the elementary school classroom. *Journal of Educational Computing Research*, *54*(2), 275-294.

EdSurge. (2016). *Computer Science for all*. Retrieved from https://www.edsurge.com/research/special-reports/state-of-edtech-2016/k12_edtech_trends/computer_science

Ertmer, P. A., & Ottenbreit-Leftwich, A. T. (2010). Teacher technology change: How knowledge, confidence, beliefs, and culture intersect. *Journal of Research on Technology in Education*, *42*(3), 255-284.

Franklin, J. (2017). Principal's Message. Retrieved from http://www.springcreek.provo.edu/Site_School/0000/index0000.html.

Google Inc. & Gallup Inc. (2016). Trends in the State of Computer Science in U.S. K-12 Schools. Retrieved from http://goo.gl/j291E0

Google (n.d.). *What is computational thinking?* Retrieved from https://computationalthinkingcourse.withgoogle.com/unit?lesson=8&unit=1

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38-43.

Guskey, T. R., & Yoon, K. S. (2009). What works in professional development?. *Phi Delta Kappan*, *90*(7), 495-500.

Holden, H., & Rada, R. (2011). Understanding the influence of perceived usability and technology self-efficacy on teachers' technology acceptance. *Journal of Research on Technology in Education*, *43*(4), 343-367.

ISTE (2014, September 11). *Computational thinking for all.* Retrieved from https://www.iste.org/explore/articledetail?articleid=152

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology, 26*(2), 175-192.

Jeon, Y., & Kim, T. (2017). The effects of the computational thinking-based programming class on the computer learning attitude of non-major students in the teacher training college. *Journal of Theoretical and Applied Information Technology, 95*(17), 4330-4339.

K–12 Computer Science Framework. (2016). Glossary. Retrieved from http://www.k12cs.org.

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.

Kundukulam, V. (2018). 4 ways to truly expand #CSforAll. *EdSurge*. Retrieved from https://www.edsurge.com/news/2018-01-23-4-ways-to-truly-expand-csforall

Lawless, K. A., & Pellegrino, J. W. (2007). Professional development in integrating technology into teaching and learning: Knowns, unknowns, and ways to pursue better questions and answers. *Review of Educational Research*, *77*(4), 575-614.

Lee, I. (2016). Reclaiming the roots of CT. *CSTA Voice: The Voice of K–12 Computer Science Education and Its Educators*, 12(1), 3-4. Retrieved from http://www.csteachers.org/resource/resmgr/Voice/csta_voice_03_2016.pdf

Ma, Y., Lai, G., Williams, D., Prejean, L., & Ford, M. J. (2008). Exploring the effectiveness of a field experience program in a pedagogical laboratory: The experience of teacher candidates. *Journal of Technology and Teacher Education, 16*(4), 411-432.

Mason, S. (working paper). Preparing elementary school teachers to teach computing, coding, and computational thinking. Brigham Young University.

McKenney, S., & Reeves, T. C. (2013). *Conducting educational design research*. New York, NY: Routledge.

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.

Mueller, J., Wood, E., Willoughby, T., Ross, C., & Specht, J. (2008). Identifying discriminating variables between teachers who fully integrate computers and teachers with limited integration. *Computers and Education, 51,* 1523–1537.

Ng, W. S. (2017). Coding education for kids: What to learn? How to prepare teachers? ICICTE 2017 Proceedings. Retrieved from http://www.icicte.org/ICICTE_2017_Proceedings/6.2_Ng%202017.pdf

Odden, A. R., & Picus, L. O. (2014) *School finance: A policy perspective* (5th ed.). New York, NY: McGraw Hill.

Partovi, H. (2017). Should computer science be a mandatory class in U.S. high schools? *Quora*. Retrieved from https://www.quora.com/Should-Computer-Science-be-a-mandatory-part-of-a-high-school-curriculum/answer/Hadi-Partovi

Provo School District (2017). 2016-2017 Provo City School District School Data Profile, Lakeview Elementary. Retrieved from http://www.lakeview.provo.edu/Site_PDF/Lakeview.pdf

Provo School District (2017). 2016-2017 Provo City School District School Data Profile, Spring Creek Elementary. Retrieved from http://www.springcreek.provo.edu/Site_PDF/Spring%20Creek.pdf

Provo School District (2017). Annual Statistical Report. Retrieved from https://provo.edu/wp-content/uploads/2017/01/08162017-s3-Annual-Statistical-Report.pdf

Provo School District (2017). 2015-2016 Progress Report. Retrieved from https://provo.edu/wp-content/uploads/2017/07/07312017DistrictProgressReport.pdf

Rich, P. J., Jones, B., Belikov, O., Yoshikawa, E., & Perkins, M. (2017). Computing and engineering in elementary School: The effect of year-long training on elementary teacher self-efficacy and beliefs about teaching computing and engineering. *International Journal of Computer Science Education in Schools*, *1*(1), 1-20.

Sadik, O., Ottenbreit-Leftwich, A., & Nadiruzzaman, H. (2017). Computational thinking conceptions and misconceptions: Progression of preservice teacher thinking during computer science lesson planning. In P. J. Rich & C. Hodges (Eds.) *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 221-238). Springer.

Stanton, J., Goldsmith, L., Adrion, W. R., Dunton, S., Hendrickson, K., Peterfreund, A., Yongpradit, P., Zarch, R., & Zinth, J. (2017). *State of the states landscape report: State-level policies supporting equitable K–12 computer science education*. Education Development Center. Retrieved from https://www.edc.org/sites/default/files/uploads/State-States-Landscape-Report.pdf

Somekh, B. (2008). Factors affecting teachers' pedagogical adoption of ICT. In J. Voogt & G. Knezek (Eds.), *International handbook of information technology in primary and secondary education* (pp. 449–460). New York: Springer.

Toikkanen, T., & Leinonen, T. (2017). The Code ABC MOOC: Experiences from a Coding and Computational Thinking MOOC for Finnish Primary School Teachers. In *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 239-248). Springer

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35.