In D. Mentor (Ed.) *Handbook of Research in Mobile Learning in Contemporary Classrooms*. Hershey, PA: IGI Global, 2016

Incorporating touch-based tablets into classroom

Woonhee Sung, Ahram Choi and John Black

INTRODUCTION

activities: Fostering children's computational T\thinking through iPad integrated instruction.

The advancement of communication technology and the emergence of technology-rich tools continuously transforms the way in which we live, work and learn (Voogt, Erstad, Dede, & Mishra, 2013). This rapid development of technology imposed new challenges to educators because of "fundamental changes in both what has to be learned and how this learning is to happen" (Voogt et al., 2013, p. 403), and increased attention to learning in Science, Technology, Engineering, and Mathematics (STEM) domains. President Obama identified STEM learning as a key for innovation and global leadership in the 21st century and established initiatives for STEM learning (White House Press Release, 2010). Individual subjects in STEM domains are not new and have been taught in schools (Sanders, 2009). However, they were not taught as interconnected practices as they are in the real world. Therefore, the growing attention to STEM learning focuses on the interconnected nature of each domain in STEM. The *Next Generation Science Standards* (NGSS) and *Common Core State Standards* (CCSS) address the interdependence of individual practice (Bybee, 2014). However, despite the growing demands for STEM education, studies indicate a shortage of student interest and an undersupply of a trained workforce (Hossain & Robinson, 2012; Voogt et al., 2013). The National Science Board (2010) has emphasized the significance of early exposure to STEM concepts in childhood education, but appropriate curriculum is still lacking pedagogically and developmentally.

In response to growing demands for STEM learning in elementary classrooms, this chapter seeks to make STEM learning attractive to young students by proposing effective instructional strategies that develop computational thinking ability, as a core component of STEM learning. Computational thinking is often identified as part of a set of 21st century skills, which provides the underlying framework for the CCSS and NGSS (Dede, Mishra, & Voogt, 2013; Chris Dede, 2010; Voogt et al., 2013). "21st century skills" often refer to the competencies necessary for living and working in the 21st century (Voogt et al., 2013). Although the existing 21st century skill frameworks vary in details, reviews of such frameworks showed consensus on the importance of digital technology literacy (Dede et al., 2013; Dede, 2010; Einhorn, 2012). Across the frameworks, digital literacy, problem solving, critical thinking and creativity are identified as skills essential in 21st century societies (Voogt et al., 2013). These skills combined consist of the core of "computational thinking" as defined by Wing (2006). This term was used to describe a set of thinking skills, habits, and approaches that are integral to solving problems and designing systems from a scientist's perspective. Computational thinking gives learners a framework to visualize and analyze problems (Einhorn, 2012). Wing (2006) defined computational thinking as "thinking involving solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (p. 33). The gist of computational thinking is to think like a computer scientist when being confronted with a problem (Grover & Pea, 2013). Many scholars have argued that computational thinking skill is not limited to computer science but also a vital analytical skill in other forms of STEM learning (Wing, 2006). Moreover, early exposure to computational thinking helps students become more conscious about how to apply computational thinking to problem solving in the

STEM domain (Yadav, Zhou, Mayfield, Hambrusch, & Korb, 2011). Therefore, educational efforts should be made to deepen students' understanding in core concepts of computational thinking for successful integrated STEM learning.

This chapter elaborates on a classroom study that focuses on young children's acquisition of several types of computational thinking skills, by incorporating a touch-based tablet, iPads, in the physical activities in classrooms to teach mathematics content as well as programming perspectives that are fundamental to computational thinking skills. The authors suggest instructional strategies that are unique to mathematics classrooms and beneficial for children's development of programming skills as well as general cognitive abilities.

Technology Integration in Classrooms

With growing interest in incorporating technology into the classroom within the last two decades, a range of initiatives have taken place to provide teachers and schools with the tools and skills required for a seamless integration of learning technologies into teaching and learning. Some of these initiatives include software-focused technology integration, sample technology-based teaching and learning resources, as well as structured professional development courses (Harris, Mishra & Koehler, 2009). Unfortunately, the majority of earlier initiatives tended to be "technocentric" (Papert, 1987) as how these technological tools were implemented in the classroom was dependent on their specific affordances and constraints. The recent mass production of affordable touch-based mobile devices and tangible user interfaces (TUI), however, have created new conditions for learning and knowledge building (Scardamalia & Bereiter, 2006), and provided improvements to the ways in which technology may be integrated more seamlessly into classroom curriculum and pedagogy. The fact that most of these new tools with tangible user interfaces (TUI) are portable and thus more mobile also helps to increase access to a wide variety of educational resources, including applications and games, for educational purposes.

The adoption of mobile devices has changed the way of content production, which now emphasizes and enables creative meaning-making processes through multimodal interactions rather than being limited to passive consumption or learning. These affordances support users' internal knowledge construction while using iPads in open-ended ways such as creating virtual artifacts or projects (Resnick, Martin, Sargent, & Silverman, 1996). This type of learning originated constructionism (Papert, 1980), which pays particular attention to the ways that internal knowledge is constructed in the world through the use of digital tools. Moreover, a wide variety of available applications on iPads allows students to practice different types of cognitive strategies. For instance, programming applications (e.g., Scratch, Hopscotch) provide an opportunity to master metacognition and computational thinking (Clements & Gullo, 1984). A core practice of computational thinking is tackling complex tasks by decomposing problems into step-by-step sub-goals (Wing, 2006). The key point of computational thinking is not about making people automatically think like computers, but rather about developing a set of mental models necessary to solve complex human problems effectively.

The introduction of the iPad brought into practice a refined form of learning with increasing awareness about technology as a learning medium. Meaningful learning occurs in technology-based learning environments where there is an interplay of complex interactions among cognitive, motivational, affective, and social processes (Anderson & Labiere, 1998; Collins, Brown, & Newman, 1989; Derry & Lajoie, 1993; Jonassen & Land, 2000; Jonassen & Reeves, 1996; Lajoie, 2000; Pea, 1985; Shute & Psotka, 1996; Solomon, Perkins & Globerson, 1991; Wenger, 1987). The open-ended programming applications such as Scratch or Hopscotch enable constructive problem-solving practices through manipulation and interacting with virtual sprites.

This chapter focuses on developing iPad-integrated curricula targeting underserved and underachieving young learners. Regarding low-level curricula and support (Hossain & Robinson, 2012; Voogt, Erstad,

Dede, & Mishra, 2013) among underrepresented groups in early educational settings, this chapter pays special attention to the "learn-by-doing" approach to support concrete ways of thinking about abstract concepts, mathematics and programming. The iPad is a powerful carrier for new ways of thinking; however adding physical activities prior to iPad use helps learners reflect about their own thinking process in the real world by manipulating external objects and act-out by themselves. Therefore, the combination of pre-activities in the mathematics domain followed by programming on the iPads can allow active construction of computational thinking skills throughout the proposed instruction. Following reviews of the mobile infused initiatives, we discuss the integration of computational thinking in the mathematics and programming domains, and the use of the embodied approach in the design and delivery of classroom instruction in this regard.

Computational Thinking

Traditional technology-rich classrooms have not been accompanied by curricula emphasizing higher-order computational thinking skills, but rather placed more emphasis on basic how-to skills lacking in creativity and relevance to everyday life (Jona, Wilensky, Trouille, Horn, Orton, Weintrop & Beheshti, 2014). To address these challenges, computational thinking needs to be embedded and integrated into traditional STEM courses in a way that promotes students' application of computational thinking strategies across multiple domains. In our approach, computational thinking is grounded in reality within familiar and age-appropriate contexts. Computational thinking that is introduced in this study includes set of cognitive skills applicable to a broad range of problem solving tasks.

What Is It And Why Is It Important?

Technology has provided us with a medium to develop solutions to problems we face in 21st century and changed our ways of approaching problems. Thinking computationally not only includes thinking processes fundamental to computer science, but more importantly involves systematic and efficient processing of information and tasks (Lee, Martin, Denner, Coulter, Allan, Erickson & Werner, 2011). Wing (2011) underscored the role of an information-processing agent, which carries out computational thinking thought processes so that the solutions are represented in effective ways. Researchers consistently define computational thinking as thought processes involved in formulating problems and decomposing goals, as well as designing solutions as computational steps and algorithms (Aho, 2012; Lee et al., 2011; Wing, 2008, 2011). These core thought processes share traits with mathematical thinking and engineering thinking in many ways when approaching problems (Barr & Stephenson, 2011; Chang & Biswas, 2011). Mathematics and engineering concepts also overlap in the forms of algorithmic thinking, sequential thinking and design thinking skills that are fundamental to analytical thinking skills and that are fundamental to computational thinking. Bers (2010) employed Tangible K Robotics for teaching computer programming-algorithms or sequences of instructions that move robots by sensing and responding to their environments. Bers (2010) employed Tangible K Robotics for teaching computer programming-algorithms or sequences of instructions that move robots by sensing and responding to their environment. By engaging in the engineering design process within programming projects, students learn engineering thinking and scientific thinking. Additionally, there have been studies emphasizing the use of "computing as a medium" (diSessa, 2000) for exploring mathematics and science via programming environments (Bers, 2010; Clements & Battista, 1989; Feurzeig & Papert, 2011). Clements and Battista (1989) investigated the positive effects of computer programming in Logo in conceptualizing geometric objects among primary grade children. This work was rooted in Papert's research around Logo programming (Papert, 1980); which pioneered the idea of the computer being "a medium" or "a machine", where children can develop procedural thinking through programming. Feurzeig and Papert (2011) revisited the first published paper on the Logo programming language and underscored the design of Logo that promotes a constructive vision on learning mathematics through the expression of solutions to problems via Logo programmable robot turtle. The extensive literature over the last three decades exploring computational thinking and its application to domain learning revealed essential elements of computational thinking. The following is a set of thinking skills selected and adapted from Barr and

Stephenson (2011) and Grover and Pea (2013, p.39) to operationally define computational thinking in the present study.

- Abstraction and pattern generalization
- Problem decomposition
- Systematic processing of information
- Procedural and sequential thinking
- Efficiency and performance constraints
- Constructive thinking

The characteristics discussed above are the skills required for formulating, identifying, implementing and testing possible solutions to problems combining knowledge and technology. As there is no common definition of computational thinking, it is even more challenging to develop a suitable one specific for K-12 educational settings. Barr and Stephenson (2011) highlighted the necessity of developing a definition of computational thinking from a practical approach that can be embedded in K-12 classrooms (p50). In other words, focusing on what students would actually *do* to achieve defined criteria of computational thinking is important for designing classroom activities. One common factor among successful classroom activities is the constructive environments, in which learners are required to design solutions iteratively, simulate and reflect on the processes with feedback (Resnick, 2007). Barr and Stephenson (2011) also pointed that in solving an interesting problem, abstraction of thinking is one heuristic that can help students to attack the problem. Thus, activities should allow students to construct solutions to problems using different levels of abstraction and to design diverse ways of tackling problems.

The next step is to choose appropriate curriculum and contexts where these activities can be embedded. There is extensive research conducted on the relationships between programming, computational thinking and mathematics (Fadjo, Hallman, Harris, & Black, 2009; Feurzeig & Papert, 2011; Jona et al., 2014; Voskoglou & Buckley, 2012). Carnegie Mellon Center for Computational Thinking[1] described computational thinking as "thinking algorithmically and with the ability to apply mathematical concepts such as induction to develop more efficient, fair, and secure solutions". Therefore, the authors aim at implementing computational thinking and its applications within existing mathematics curricula in the K-12 classroom. As mentioned earlier, skills taught within traditional mathematics classrooms were found to overlap well with several important computational thinking skills, and seemed the most appropriate subject domain to be used in our study. Programming applications were adopted as tools to help students practice computational perspectives in the context of learning mathematical concepts through programming. The following sections review the relationships between computational thinking particularly in mathematics and programming domain.

Computational Perspectives in Programming: Thinking About Thinking

Programming is not simply about using computers. Grover and Pea (2013, p.40) highlighted that "programming is not only a fundamental skill of computer science and a key tool for supporting the cognitive tasks involved in computational thinking but a demonstration of computational competencies as well". Programming enables the application of both explicit and tacit computational thinking skills by incorporating challenges that require higher order thinking (Einhorn, 2012). Programming is concerned with answering 'How would I get a computer to solve this problem?', where the computer is a machine, and a human provides commands to the machine recursively. Answering the question: 'How would I solve it?' fosters the identification of appropriate abstractions that lead to solutions for the computer (Wing, 2008). This is the essence of *computational perspectives* that is developed by Berland and Wilensky (2015) as a unique element within computational thinking.

In other words, programming requires problem solving processes by the 'necessarily explicit nature of programming', which make people to 'articulate assumptions' and 'precisely specify steps to their problem solving approach' (Papert, 1980; Pea & Kurland, 1984, p. 142). This process of solution

construction requires analytic perspectives for solving problems that are unique and fundamental to computer programmers or scientists. The next concern is then "how to separate the cognitive activity of computational thinking from the action of merely working on a computer" (Dede, Mishra, & Voogt, 2013, p. 4). As Dede, Mishra and Voogt (2013) argued, distinctive computational thinking skills is not about how to program a computer, but rather a cognitive approach to problem solving that uses skills as abstraction, decomposition, algorithms, and iterative processes (Yadav et al., 2011). Thus, this chapter seeks to explore computational thinking in terms of thinking like a computer programmer or a computer artist for domains that are not necessarily related to computer science. This is called *computational perspectives*, a concept coined by Berland and Wilensky (2015). Berland and Wilensky (2015) used the term *computational perspective* to separate it from broader definition of computational thinking and to highlight that the perspective of thinking with the computer-as-a-tool is contextualized and constrained. Computational thinking is not limited to learning technical skills in computer science; rather it is the perspective that can be applied in a domain that is not necessarily computer science. The central argument of the chapter is to investigate design features of a learning environment that alters students' perspectives within a nontechnical domain to improve mathematics learning as well as programming knowledge in transfer.

Continuing the line of argument of computational thinking, Clements and Gullo (1984) indicated that computer programming can make abstract concepts concrete by making children's thinking process more conscious and explicit and this leads to more effective learning. This 'explicit nature of programming' is the key point that needs to be adopted in instructional design priming *computational perspective*. In other words, if the practice of *computational perspective* can be employed in other domains of learning outside of computer science, it could provide an effective setting for cognitive process instruction that focuses on *how* to think rather than *what* to think, which is fundamental to computational thinking (Clements & Gullo, 1984). Designing a learning environment that promotes explicit thinking processes is a key concern in deciding how to introduce programming within the context of K-3 curriculum and assist students in learning to program. Consequently, the next issue is to figure out what thinking practices or skills are particularly useful in the mathematics domain that can further benefit programming ability.

Computational Thinking in Mathematics

Mathematical ability is often discussed as a core factor predicting students' ability to learn computer programming (Pea & Kurland, 1984). Despite the theoretical relationship between mathematical ability and computer programming, early reviews often concluded that effects of programming on overall mathematics performance were not consistently strong (Clements, 1985). The negative result is attributable to the mere "exposure" to programming that failed to provide significant improvement (Clements, 1999). Efforts have been made to propose effective instruction for mathematics within the context of computer programming (Barr & Stephenson, 2011; Clements & Battista, 1989; Clements & Gullo, 1984; Feurzeig & Papert, 2011; Liu & Wang, 2010; Matarić, Koenig, & Feil-Seifer, 2007). These studies showed the relevance among mathematical, spatial problem solving abilities and computer programming which is supported by theoretical argument as well. By adding "concrete" instances of reasoning that is inherent in programming activities benefit learning abstract mathematical concepts and procedural thinking for problem solving (Rittle-Johnson, Siegler, & Alibali, 2001; Settle & Perkovic, 2010; Voskoglou & Buckley, 2012). In order to explore the overlap in programming and mathematical concepts, the next section investigated the role of embodied approach as learning paradigm.

The Embodied Approach

Grounded cognition has been discussed as a variety of forms in cognitive science in its venerable history and continued evolving by taking new forms in robotics, cognitive ecology, cognitive neuroscience, and developmental psychology (Barsalou, 2008, 2010; Black, 2010). Grounded cognition reflects the assumption that the mental representations are shaped by multiple ways of grounding such as mental

simulations, situated action, bodily states, or modality specific factors (Barsalou, 2008). In particular, embodied cognition emphasizes the connection between knowledge representation and the bodily states (Barsalou, 2008; Glenberg, 2008, 2010; Wilson & Golonka, 2013; Wilson, 2002). The theory of embodied cognition is rooted in the premise that body movement and how bodily activities are placed in a richly perceived physical world affect on cognitive processes deeply. This premise supports the findings that the embodied approach plays a critical role in learning higher-order and abstract concepts (Croft & Cruse, 2004; Gallese & Lakoff, 2005; Johnson & Lakoff, 2002; Lindgren, 2014; Lindgren & Johnson-Glenberg, 2013).

The following research studies provide evidence that the body serves as a significant resource for people's understanding of abstract concepts and language. The kinesthetic action grounded in concept leads to conscious thinking, perception, experiences, and deliberate uses of language for shaping the learning. The embodiment claim for language emphasized the contribution of action to cognition and meaning. Glenberg and Kaschak (2002) tested that sentence understanding worked faster for the participant who was asked to select congruent action to the literal direction of the target meaning. When participants were asked to judge the sensibility of a sentence for implied direction (e.g., toward or away) that are matched with the literal meaning (e.g., when you give vs. when the other gives you), they performed faster since it was not only perceptual qualities but also action that affected their decisions. Gibbs (2005) echoed the importance of metaphor in "mapping experiences of the body to help structure abstract ideas that are fundamental to how people speak and think" (p.12). The kinesthetic action grounded in concept leads to conscious thinking, perception, experiences, and deliberate uses of language for shaping learning. In a similar manner, the use of iPads allows students to apply acquired skills to virtual programmable manipulatives, which provides multimodal feedback and perceptual experiences (Paek, 2012). Concrete learning experiences from physical act-out and manipulation of real-world objects in pre-stages are then transferred into virtual environments on the iPads where students utilize different sensory modalities from offline activities.

Another line of research by Glenberg, Gutierrez, Levin, Japuntich, and Kaschak (2004) demonstrated the effect of action as a facilitator for reading comprehension among second grade students. This was in line with Glenberg & Kaschak's work (2002) on the use of the embodied instructional method to improve memory for a better comprehension of the text material. Students manipulating toy objects to act out stories demonstrated increased understanding and memory of the stories they read. Further, imagined manipulation of objects after acting out also showed increased acquisition and memory of the story (Glenberg et al., 2004). Glenberg et al (2004) concluded that both actual manipulation of toy objects and imagined manipulation resulted in better memory and comprehension compared to non-embodied students. Black (2007) also pointed out that imagining actions for another related story after acting out with toys enhances skill development in forming the imaginary world of the story. In summary, Black (2010) specified three steps in a grounded cognition approach involved in these studies:

1. Have a perceptually grounded experience
2. Learn to imagine the perceptually grounded experience
3. Imagine the experience when learning from symbolic materials (p. 46)

Outside of the cognitive linguistics field, evidence has also been provided in the field of cognitive and developmental psychology that supports the embodied cognition approach. Research studies in cognitive psychology have provided support for the effects of sensory-motor engagement in diverse tasks with perception, memory, knowledge, thought and language (Black, 2010; Glenberg, 2008; Glenberg et al., 2004; A. D. Wilson & Golonka, 2013; M. Wilson, 2002). Similar evidence can be found in Barsalou's research (2010), where he found that manipulating different levels of bodily engagement could causally affect higher cognitive processes such as evaluation, decision-making and attribution (Barsalou, Niedenthal, Barbey, & Ruppert, 2003; Niedenthal, Barsalou, Winkielman, Krauth-Gruber, & Ric, 2005). In developmental psychology, Smith and Gasser (2005) proposed the idea that intelligence emerges

within a physically, socially and linguistically grounded environment where sensorimotor activities play central roles in the development.

Most importantly, these research studies prove the case that cognition is grounded in the sensorimotor activity of our bodies, thus expanding the body's physical interaction within conceptually grounded environments that could become "fertile soil onto which we can lay the seeds of new learning" (Lindgren, 2014, p. 40). Therefore, the embodied approach provides valuable resources for learning symbolic concepts or new literacies that are unfamiliar to young learners, such as mathematics, science and technology literacies. An embodied perspective promotes a conceptualization of mathematics, and in fact all STEM content, as grounded and situated in the spatial–dynamical and somatic experiences of the person who is engaging in well-designed activities (Abrahamson & Lindgren, 2014). Thus, the current study design reflects previous research findings on the effectiveness of embodied instruction, and employs act-outs and a manipulation of action within the learning environment.

Connection Between Embodiment And Mathematics

In elementary education, experiential hands-on education has been in use to promote learning new materials or even abstract concepts by forming real-world meaning and increasing motivation. Hands-on experience can bridge the disconnection between subject areas and contextual experience within learners. Especially for younger learners, mathematical practice tends towards about learning and applying arbitrary symbolic inscriptions that are not necessarily meaningful to them. Even during practice with mathematical formulae, students often fail to solve same conceptual concept in a different context or apply it in other contexts. The problem stems from the absence of perceptually grounded or embodied cognition that fosters a deeper level of understanding.

Fischer, Moeller, Bientzle, Cress and Nuerk (2013) employed the concept of embodiment in learning number line sense by 'moving along the number line' using a sensorimotor training concept. Fischer et al. (2013) incorporated systematic full-body movement allowing for an embodied experience of the trained numerical concept on a physical number line. Their results support theoretical arguments of the embodied approach and other research studies by showing pronounced training effects on children's number line estimation following the embodied training. Moreover, the embodied group demonstrated transfer effects in counting ability as compared to the control group. Incorporating a physical number line analogous to students' mental number line and a systematic full-body experience corresponding to their mental processes resulted in significant improvements in number sense among young learners (Link, Moeller, Huber, Fischer, & Nuerk, 2013).

Therefore, what appears important for effective mobile learning is the structural and analogical relationships between the subject matter and the physical context of where learning is taking place. Segal (2011) had emphasized that the conceptual mapping of actions congruent with learned concepts support thinking and learning. The current learning design is informed by the studies discussed above, and incorporates these ideas through physical layouts on the floor to help students learn geometric shapes. The conceptually congruent environment for learning geometric shapes is realized by providing a spatial grid layout on the floor that allows the incorporation of bodily-movement within the learning activities.

Computational Perspectives in Embodied Activities

Conceptual Congruency In Embodied Simulation

STEM learning in elementary educational settings ideally focuses on learner-centered and constructionist approaches for designing motivating and engaging activities. Embodied cognition recognizes the importance of learners' participation through active interactions with a physical world, where abstract concepts may be constructed through concrete sensorimotor experiences (Abrahamson & Howison, 2010;

Abrahamson & Trninic, 2015; Bamberger & DiSessa, 2003). When physical interactions are grounded in conceptual reasoning, the conceptual metaphor becomes cognitive substrate for learning abstract concepts (Lakoff & Johnson, 1999; Segal, 2011).

The goal of this study is to improve students' problem-solving skills in programming tasks - a primary measure of computational thinking skills (Perkovic & Settle, 2010; Wing, 2008) - by designing embodied unplugged activities in the domain of mathematics. The following sections discuss how these pre-activities are grounded in the embodied approach for promoting successful use of tablets. How programming concepts are intertwined with embodied activities in mathematics is discussed in the next section, together with explanation of how the core concepts of programming are incorporated into the learning activities.

A number of studies have shown the necessity of physical interactions grounded in conceptual reasoning, which will become internalized as simulated actions. Abrahamson and Lindgren (2014) quoted Piaget's (1968) argument that action-oriented mental processes delivered in concrete situations supports the development of mathematical or scientific ideas. Similarly, Lakoff and Johnson (1999) explained how conceptual metaphors emerge from concrete sensorimotor experience that is grounded in image schemas. Therefore, providing physical mathematical representations where learners can act out solutions is key to promoting physical interaction and concrete simulation. The math content topic targeted in this study is on 2-dimensional geometric shapes, their features and other related concepts. Employing the embodied perspective in mathematics learning design, students are moved their bodies to draw 2-dimensional geometric shapes based on their knowledge of these shapes' unique characteristics. However, this still lacks the integration of the computational perspective, a core concept in computational thinking. The next question is how to combine computational perspectives into the embodied activities on2-dimensional geometric learning.

In order to address the question, we reviewed the various types of embodiment studied in the context of programming and computational literacy. Researchers in recent studies, such as Fadjo (2012), proposed a conceptual framework of embodiment in formal educational settings, called Instructional Embodiment, within the STEM domain (Black, Segal, Vitale, & Fadjo, 2012; Fadjo, Lu, & Black, 2009), which integrates both physical and imagined movement of pre-defined content (Black et al., 2012; Fadjo et al., 2009). The main categories in Instructional Embodiment are physical and imagined. Within physical embodiment, there are four forms of Instructional Embodiment: Direct, Surrogate, Augmented and Gestural embodiment (Fadjo, 2012).

Direct Embodiment is the physical enactment of pre-defined scenarios or sequences that contains explicit and implicit cues for movement (Fadjo, 2012). The application of direct embodiment in our study is the form of bodily movement involved in completing the given geometry problem solving. Surrogate Embodiment is a type of physical enactment where the movement of an external surrogate is controlled and manipulated by the learner (Fadjo, 2012). Manipulating a surrogate resembles the concept of computer programming as the cognitive process is analogous to the processes involved when programming virtual sprites. Direct embodiment has shown significant effects on developing programming skills, especially conditional sequences, while surrogate embodiment has been shown to provide a unique opportunity for the instruction of arithmetic topics during video game design (Fadjo et al., 2009). There is thus a conceptual comparison that could be made between surrogate embodiment and computational perspectives, since manipulating a surrogate during the surrogate embodiment process resembles the thought process of a programmer during programming.

In our study, direct embodiment was employed as a form of physical role-play that involved students' own bodily movement. Direct embodiment does not increase the likelihood of displaying each step to reach solutions, but rather allows students to implement solutions by moving their own bodies. Operating

a surrogate, on the other hand, requires students to display explicit knowledge in the form of articulation in order to provide commands to move a surrogate. *Computational perspective,* which reflects conceptual congruency during embodiment, is thus adopted when a surrogate is employed within an embodied activity, whereas direct body movement does not provide opportunities to practice the perspectives from programmer or computer scientist. Based on embodied cognition theories and research in learning, this study investigates the effects of conceptually congruent embodiment on young children's learning. Moreover, learning concepts are extended into thinking skills, specifically computational thinking and problem-solving skills in geometry learning and programming domain.

INCORPORATING TOUCH-BASED TABLET INTO ELEMENTARY CURRICULUM

Addressing Issues in Tablet Integrated Learning

Mobile learning has been researched to some extent in recent years, with an emphasis on the mobility of handheld devices such as mobile phones and PDAs. It was conceived as extension of e-learning, which provides access to networks when computer access was restricted. (Motiwalla, 2007). Although in aforementioned cases, mobile learning tend to be used as a replacement for a physical learning environment delivering standalone instruction, several studies noted advantages of integrating the device into physical environments for school subjects. Other research showed how virtual information delivered via handheld devices provides authentic learning experiences. For instance, students conducted scientific investigations on local environmental problems (Squire & Klopfer, 2007) and evaluated regional historical information (Schrier, 2006) using location-aware technology (e.g. use of RFID chips, GPS) and handheld devices. Although such studies were implemented outside classrooms, they demonstrated the potentials of using mobile devices for academic subjects. By incorporating information provided by handheld devices into their physical surroundings and activities, students can relate the information to the real world problems in meaningful way. Given their developmental stage, incorporating the use of technology as a tool into physical activities are particularly effective for children's learning (Price & Rogers, 2004).

Young children in the early learning stages from K-3 have yet to fully develop fine motor skills necessary to manipulate small objects to operate technology, and thus may experience issues operating and controlling conventional input devices such as mouse and keyboards for computers or keypads of a mobile phone, such that they would not be able to benefit fully from the technology. A study by Lauricella, Barr, and Calvert (2009), for example, showed that young children lack proficiency in pointing and clicking with a computer mouse. However, touch-based tablets such as iPads allow children to perform various tasks using its touch interface, which is intuitive and easy to use. Moreover, Paek (2012) reported that children using a touch-interface had better learning outcomes than children using desktop computers when playing a math game. Therefore, touch-based portable devices such as iPads have greater potential for classroom applications with young children in early elementary settings.

While programming skills have been taught traditionally to older students at the middle and high school levels, there has been a shift to open this domain to young children. To better customize the learning of basic programming skills for younger children, the DevTech research group at Tufts University and MIT's Lifelong Kindergarten developed Scratch Jr., an open-ended block-based version of the popular Scratch programming app. Through this app, the researchers attempted to address the challenges encountered by young children when navigating text-heavy programming environments, by creating a more intuitive programming environment based on graphics instead of text. In our study, we make use of Hopscotch, a similar iPad application that teaches foundational programming skills to young children aged five to seven using a graphical block-based programming language. In a similar manner to Scratch Jr., Hopscotch attempts to address the needs of children in the younger age group by creating a simpler and graphically intuitive version of a programming application. The Hopscotch application provides a

combination of tangible and graphical programming language tools that would appeal more to young children. In the Hopscotch application, children snap a collection of graphical "programming blocks" to create a stackable programming structure.

When the iPad, a touch-based smart device, first appeared in the market, it received a lot of attention as 'next-generational educational technology' (Murray & Olcese, 2011). Several states such as New York and Virginia equipped classrooms with iPads across different grade levels, for various uses such as e-textbooks (Hu, 2012). However, equipping students with 21st century devices do not necessarily promote the development of 21st century skills in the classroom. Despite the promises of digital tools for learning, children are only on the receiving end of operating media; they do not necessarily know how to optimally use media creatively or critically (Rideout, Foehr & Roberts, 2010). Even in the 20th century, innovative technology (e.g. Skinner's Programmed Instruction, Papert's Mindstorms) had been introduced into educational settings with an expectation that this new technology would change the educational system (Sawyer, 2006). However, these technological software and equipment alone were not enough to prepare students for the 21st century. The number of classrooms with computers and networks has been increasing in the last decade, but equipment alone does not lead to educational change (Scardamalia, 2001; Wenglinsky, 2005). Changes at the classroom level occur around curriculum – through "curriculum materials, teaching practices, and beliefs or understandings about the curriculum and learning practices" (Fullan, 2007, p.85). Given that the *Common Core Standards* (CCS) and *Next Generation Science Standards* (NGSS) emphasize the interdisciplinary nature of the STEM domains, this chapter seeks to weave fundamental 21st century cognitive skills into existing math curricula, using an iPad application as a scaffolding tool.

STUDY DESIGN TO TACKLE CHALLENGES IN ADOPTING TABLETS IN EDUCATION

Goal of the Study

The study proposes an interdisciplinary STEM curriculum incorporating touch-based tablets to prepare young students for the 21st century society. A gap still exists between conceptual definitions of computational thinking in STEM areas and how it is applied in classrooms. Several studies confirmed that technology integration in the classroom has failed to take into account the new conditions for the development of unique skills required for computer literacy, in what is referred to as computational thinking ability (Hatlevik, Ottestad, Skaug, Kløvstad, & Berge, 2009; Scardamalia & Bereiter, 2006). Lankshear and Knobel (2006) emphasized the importance of designing educational contexts in which technology literacy and fluency can be successfully embedded. This study is thus designed to embed computational perspectives into existing math curriculum to develop computational thinking skills with the use of a mobile programming application. In order to examine how the new instructional approach incorporating an iPad as a cognitive tool affects children's STEM learning, an experimental study was conducted in the second grade elementary classrooms. Study design, procedures and results are discussed in the following sections.

Study Design and Procedures

The purpose of this study is to examine the effects of an instructional method that incorporates iPad applications in the development of children's early computational thinking skills, within the domains of mathematics and programming. Children were assigned to three different conditions: the Surrogate Embodiment (SE) condition, the Direct Embodiment (DE) condition, and the Control condition. Before they build programming code to solve given math problems via Hopscotch, children were engaged in

three different pre-activities depending on the condition as described in Table 1. Children in the SE condition gave commands to their partner to solve the problem, and partners moved their bodies around a grid layout on the floor in order to create a solution. Children in the DE condition moved their own bodies to demonstrate the step-by-step solution. Children in the control condition were given a worksheet on which they had to draw their solution. All children received a handout that describes the differentiating features of various 2D geometric shapes (e.g. degree of angles and length of side) as a reference for the problem they were trying to solve.

Given the various perspectives and evolving definitions of computational thinking, a set of skills is included in defining computational thinking that are applicable to this study. They were adapted from the definition of the International Society for Technology in Education and Computer Science Teacher Association[2] to fit our study objectives:

1. Formulating problems in a way to use a computer to solve: Commanding
2. Logically organizing the data: Procedural and sequential steps
3. Implementing possible solutions: Programming fluency and efficiency
4. Transferring problem solving process to a wide variety of problems: transfer mathematics learning into programming task

This study proposes embodied activities as a way to take *computational perspectives* in the domain of mathematics and programming. Students in the SE condition solve mathematics problems from a computer programmer's perspective by taking the role of a *commander* who commands the surrogate by giving procedural and sequential steps to reach solutions. This is what the SE group executed during the intervention when given mathematics problems. While students in SE group embody themselves as programmers, the DE group engages their own body movement to enact steps to solve problems. Therefore, the critical difference between SE and DE is the presence of CPP engaged in bodily activities. The intervention design is described in table 1 and the procedure of the study is described in table 2.

Table 1. Details of conditions

Intervention	Details of each group
Surrogate Embodiment Group (SE) -Presence of CPP and embodied simulation (Enact the role of commander to practice computational perspectives)	Students manipulate a surrogate to perform solutions for given geometry problems. Students provide step-by-step commands to the surrogate to draw a given 2D shape based on its basic features.
Direct Embodiment Group (DE) -Absence of CPP and direct embodied activity (Directly engaged bodily movement without a commander role)	Students move their body to draw given geometric shapes. Students do not need to command a surrogate, but perform a solution by moving their own body.
Control (None of above)	Students study and draw solutions on paper handouts.

Table 2. Study procedures

	Week 1 Pre-activity stage-1	**Week 2 Coding-1**	**Week 3 Pre-activity stage -2**	**Week 4 Coding-2**	**Week 5-6 Advanced Coding**
Task	Drawing a rectangle and a square	Programming a virtual character to move and draw a rectangle/square	Drawing an equilateral triangle	Programming a virtual character to move for drawing an equilateral triangle	Programming a virtual character to draw complicated shapes such as pentagon or hexagon or other shapes
SE group	Students command a surrogate (teacher) to move and leave a trail for drawing a rectangle/square on the grid drawn on the floor	All 3 groups are asked to do programming on Hopscotch to make virtual character draw a rectangle/square.	Students command a surrogate to move and create a triangle on the floor	All 3 groups are asked to do programming on Hopscotch to draw an equilateral triangle by commanding a virtual character	Students create programming to draw complicated shapes or multiple shapes in one screen. All 3 groups are asked to create programming for either hexagon or pentagon
DE group	Students move their own body to draw a rectangle/square		Students move their own body to create a triangle on the floor		
Control group	Students read hand-out and draw shapes on the paper		Students read hand-out and draw shapes on the paper		

Programming Activity Using Hopscotch

After the pre-activity, students were engaged in programming activities to create the shapes they learned in the pre-activity, using the Hopscotch app on the iPad, (http://www.gethopscotch.com/) which is a visual block-based programming application for iPads. This app was used to help encourage and identify whether there had been a transfer of what students learned through pre-activities into programming tasks. The Hopscotch app is similar to Scratch (http://scratch.mit.edu), which was built based on the Logo programming framework, and designed to support simple programming (Papert, 1980). In Hopscotch, characters can be created and manipulated through a certain algorithm, which is created using coding blocks (Amer & Ibrahim, 2014). These characters act according to either specific built-in scenarios or in reaction to the algorithm created by users. In order to code in Hopscotch, a user drags a block from a palette of command blocks onto the stage, where a series of blocks can be snapped together. By hitting a

play button, the screen changes to show the movement of characters based on the code, or algorithm, built with sequenced graphical blocks.

In the programming activity, children were asked to program a character move along a path based on the geometric shapes they studied in the pre-activity. The children did not receive any specific instruction about meaning of each command block, but they were taught how to move command blocks into sequences of instruction, and how to execute their codes. They revised their code until they were able to create the given shapes without any explicit help or feedback from the experimenters.

Figure 1. A screenshot of Hopscotch programming stage for creating a square.

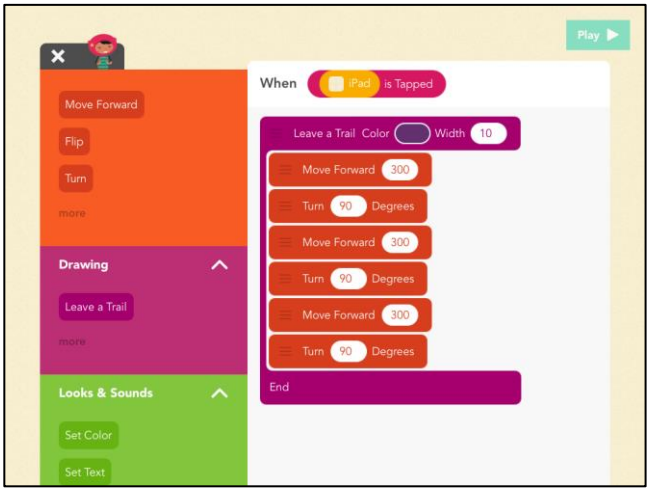
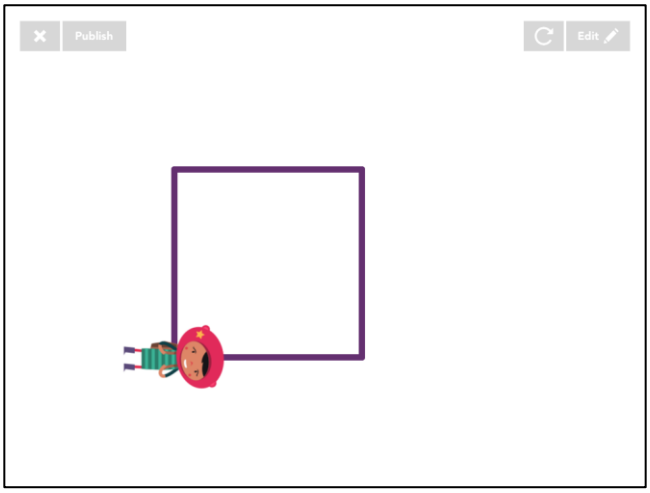


Figure 2. A screenshot of Hopscotch result page after creating programming blocks.



After the children learned basic geometric shapes (e.g. triangles and rectangles), they proceeded to an advanced programming session where they had to code the paths using more complex geometric shapes (e.g. pentagons and hexagons). The children did not act on these shapes, because those were not used in the pre-activity. Upon completion of all the programming activities, their cognitive skills, learning gains in geometry, and programming fluency were measured.

Participants

The study was conducted with 39 2nd grade students (25 males, 14 females) during a coding afterschool program in a New York City public school. The data obtained from five students were excluded from analysis due to frequent absences. The intervention was conducted for 10 weeks from March to May in 2015. This ethnically diverse school consists of 25% Hispanic, 39% Black, 20% White, and 6% Asian students

Measures

Cognitive Skills

To assess students' abilities to reason about and generalize a solution for realistic situations, the authors administered the Test of Problem Solving-3 (TOPS-3: Elementary; Bowers, Huisingh, & LoGiudice, 2005), a commonly used test to measure students' thinking and reasoning skills. The study utilized three pictured situations from the TOPS-3 battery, with a total of 14 questions. Among the cognitive skills measured by the TOPS-3 test, we selected problem solving, inference, predicting, determination of causes, and sequencing skills as skills that are most relevant to this study. Children verbally responded to the standard set of questions that were read aloud by the experimenter. Tests were scored using scale norms from the scoring manual. Responses categorized under each thinking skill were summed to form subscales. Higher scores indicated stronger abilities to solve problems, infer, predict, determine causes, and understanding sequence.

Learning in Mathematics and Programming

The authors developed tests to measure learning outcomes in geometry and programming. Geometry test was developed based on the geometric thinking level theory (Van Hiele, 1986), which consists of four levels: recognition, visual association, description/analysis, and abstraction/relation. The test items were selected from Chang, Sung, and Lin (2006) measurement, which originally had 20 multiple-choice questions. 10 items were selected for 2nd graders based on the their school curriculum.

To measure students' programming fluency, a paper-based programming skill test was developed. Testing items were developed based on three assessment criteria in computer science education that Meerbaum-Salant, Armoni, and Ben-Ari (2013) developed based on the Bloom's taxonomy.

1. Understanding: The ability to summarize, explain, exemplify, classify, and compare CS concepts, including programming constructs;
2. Applying: The ability to execute programs or algorithms, to track them, and to recognize their goals;
3. Creating: The ability to plan and produce programs or algorithms. (p. 245)

The above criteria were evaluated by asking students to choose correct programming results, create effective codes, detect patterns, and apply the algorithms in the paper-based programming skill test.

Results

The results of the study measures were analyzed to identify the effects of different types of iPad-integrated instructional methods: Surrogate Embodied instruction, Direct Embodied instruction, and non-embodied instruction. Learning outcomes were measured in two domains: mathematics (geometric knowledge) and programming.

Cognitive Skills: Result from TOPS-3

The scores from the TOPS-3 measure were compared between the Control and Embodiment groups (EM; Combination of SE and DE), and further analysis was conducted within the embodiment groups. Given the cognitive and behavioral nature of the embodied instructional methods, we expected improvements in participants' abilities on critical thinking skills. We analyzed the TOPS-3 test measure results using the Mann-Whitney test, and found that the EM group does not show significant gains compared to Control group. Nevertheless, the SE group scored slightly higher on problem solving (EM *Mdn*=6.0, Control *Mdn*=5.5), sequential skills (EM *Mdn*=5.0, Control *Mdn*=3.5), and determination of causes (EM *Mdn*=4.0, Control *Mdn*=3.5).

To better understand the effects of different types of embodied instruction in the development and practice of computational thinking skills, further analysis between the SE group and the DE group were conducted. Examination of the TOPS-3 test revealed that the SE group performed better than DE group in problem solving (SE *Mdn*=6.0, DE *Mdn*=5.0), $U = 13.0$, $p = .027$, $r = .57$. sequence (SE *Mdn*=5.5, DE *Mdn*=4.0), $U = 13.5$, $p = .027$, $r = .53$., determine causes (SE *Mdn*=5.0, DE *Mdn*=3.0), $U = 13.0$, $p = .027$, $r = .55$. On the other hand, the DE group showed more significant improvement in skills associated with creating solutions for problems by drawing logical reason for a given aspect, evaluating alternative solutions, and predicting anticipated future.

Learning in Mathematics and Programming

The relationships between the three groups of instructional methods were analyzed using regression. A one way ANOVA conducted for the geometry test and programming test measures shows significant differences between groups. Table 2 reports the mean scores for each group.

Table 3. Mean of groups for geometry and programming test score

	Direct Embodiment group		Surrogate Embodiment group		Control		Total	
	M	SD	M	SD	M	SD	M	SD
Geometry	7.33	1.58	7.43	1.51	5.38	1.98	6.48	1.97
Programming	7.90	3.45	10.14	6.18	5.35	2.83	7.60	4.87

For the geometry test measure, the total mean score differences are significant, $F(2, 26) = 4.578$, $p=.02$. A post-hoc test confirms that the mean difference between the DE and Control groups is significant with $p=.044$. On the other hand, the differences between the SE and Control groups is only marginally significant with $p=.051$. However, when analyzing the scores of higher level testing items for abstraction within the geometry test measure, the DE group shows significantly higher mean scores ($M=2.57$, $SD=0.53$) as compared to the Control group ($M=1.46$, $SD=0.776$) with $p=.017$. When examining the lower level testing items in the geometric knowledge test, which assessed recognition and visual association, the DE group ($M=5.55$, $SD=0.881$) was shown to be significantly different from the Control group ($M=3.92$, $SD=1.382$) with $p= .017$.

When it comes to the coding test, significant differences are found, $F(2, 26) = 3.383$, $p=.04$. A post-hoc test shows significantly higher mean scores for the SE group ($M=7.90$, $SD=3.45$), as compared to the Control group ($M=5.35$, $SD=2.83$) with $p= .039$. On the other hand, the DE group failed to show significant differences from Control group.

DISCUSSION

Many cognitive scientists have studied how bodily action affects conceptual development and proposed models to explain how learning abstract concepts benefits from concrete embodied experiences. As discussed earlier, it has been argued that STEM disciplines requires analytical and computational perspectives, and the embodied approach to help create learning environments that encourage learners toward these perspectives (Abrahamson & Lindgren, 2014). This study design incorporates embodied activities into the practice of computational perspectives in learning geometric concepts. Through the implementation of various embodied intervention activities followed by programming practice on the iPads, students are motivated to execute problem solving skills specific to computer programming contexts. Students in the surrogate-embodiment condition practiced computational perspectives through the use of a surrogate during the pre-activities. The presence of a surrogate in the SE group promotes analytical thinking processes that are unique to computational perspectives such as commanding, procedural and sequential thinking strategies. This study examines if such explicit efforts at the pre-activity stage maximizes the learning process using Hopscotch to improve student understanding of both mathematics and programming concepts. The overall results demonstrated promising effects of instructional methods involving surrogate embodiment over methods that include direct or no embodiment at all.

Overall, student groups engaged in embodied activity report higher mean scores in geometric learning, programming and the TOPS-3 test measures as compared to the control group. The TOPS-3 test fails to show significant differences between the embodied and control groups; however when comparing between the SE and DE groups, the group that experienced surrogate embodiment activities (thus involving CPP) shows significantly higher performance in the TOPS-3 test measure, that includes problem-solving, sequential thinking and determining causes. This finding implies that when iPad use is accompanied by embodied activity with CPP (as in the SE group), students gain opportunities to practice skills that are fundamental to computational thinking. As the core idea of computational thinking is to tackle complex tasks by decomposing the task into sequential steps to reach a solution (Wing, 2006), students who practice computational perspectives in an embodied activity followed by practicing on the iPad seemed to have attained computational skills such as problem-solving, sequential thinking and error correction by determining causes. This finding is consistent with the programming test results which reported a significantly higher mean score among students in the SE group who experienced CPP, as compared to the control. These results indicates that students in the SE group may have been able to facilitate their knowledge construction process, through the utilization of skills specific to computational thinking during the pre-activity and when programming with the iPads. Even without providing explicit coding language instruction, participants in this study appeared to have acquired some mastery of certain computational thinking skills, such as problem solving, sequential thinking and determining causes.

In addition, the implementation of embodiment activities that encourage the use of computational perspectives may also serve to improve student learning in geometrical shape concepts. The embodied CPP activity within the SE student group helped the students to not only recognize the unique features for each basic geometric shape, but also to gain a deeper conceptual understanding of these features through the embodied and programming activities. . Creating geometric shapes through a surrogate encourages students' use of analytical thinking skills by explicitly articulating the steps for drawing each geometric shape based on their features. This process resembles programming activity in terms of the analytical and explicit nature of coding (Papert, 1980; Pea & Kurland, 1984, p. 142). In this way, students successively deepen their understanding of programming concepts, and develop mastery in certain computational thinking skills. Given that young children are novice learners in the programming or computer science domain, the embodied activity instruction in this study hence serves to anchor the introduction of computational perspectives within familiar physical and mathematical contexts, which in turn helps to lower students' barrier towards learning programming concepts. In this project, we therefore find that it is possible to design coherent classroom activities that effectively embed mobile technology into the

existing mathematics and programming curricula in an interdisciplinary manner to improve student learning within both mathematics and programming domains.

Programming Education

Several research studies regarding programming education for children has demonstrated interactions among the design features of coding applications, content of coding activities, and emotional development (Bers, 2009; Burke, 2012; Kazakoff, 2014). However, not much attention has been given to understanding the cognitive processes behind coding activities. To encourage the development of problem solving skills among younger learners, students may participate in programming projects that are situated in environments where exploration and interactions with virtual artifacts are supported.

To encourage learners to become active participants, the learning environment needs to be based on meaningful contexts interwoven with familiar subject areas related to learners' everyday experiences. In reality, when it comes to designing educational technologies in the classroom, understanding school contexts is a complicated issue regardless of the various learning theories. The goal of programming education is hence to provide younger children a basic understanding of programming concepts, making them more discerning technology users and, potentially, innovative creators themselves (Scaffidi, Shaw, & Myers, 2005).

IMPLICATIONS

This study proposes a way to bring computational thinking to classroom activities within math curriculum. The results of this study indicate that a curriculum emphasizing computational thinking applied to math classrooms and strengthened through the use of technology can be an example of successful technology integration in classrooms. Given that students learned the basic concepts of programming without coding lessons, the curriculum discussed in this chapter has a potential to promote technology literacy for students with limited access to technology. Bringing qualified curricula into the school with under-served groups, teachers and leaders can minimize squandering resources while maximizing the effects of tablet uses.

To improve current design, future research can include: 1) attention to the social context of learning that predicts how and when peer group may be beneficial for learning and 2) embedding different types of computational thinking practices, such as engineering thinking, debugging, and error detection skills. Future studies can provide insights to leaders, policy makers, and teachers to apply computational thinking in multiple domains and recognize the utility of tablets in a range of applications.

First, tablet-implemented instructions can set the place for productive peer interaction. Much of the use of technological tools in the classroom is social as they are recognized as symbolic artifacts and mediated by social groups and cultural values (Moll, 2014). Theoretical and empirical work on social cognition reports that peer interaction may promote cognitive restructuring in significant ways (Mayer, 1988). Furthermore, Kafai and Quinn (2013) argued that building and remixing published artifacts in a programming classroom encourages collaboration and participation in a larger community. Group settings to provide a diversity of interaction while programming can provide young students with alternative perspectives promoting reshaping ides or understandings programming knowledge as they become exposed to multiple solutions in a community.

Secondly, the classroom implements various types of instructional approaches to practice computational skill. One of the strategies to encourage computational thinking skill that may be appropriately integrated into existing subject domains and curricula is debugging, or systematic error detection. This form of computational thinking skill involves a highly complex and dynamic process to search for issues within an imperfect solution, and to achieve an overall task goal (Carver & Klahr, 1986; Law, 1998). In

education, debugging presents a valuable learning opportunity for cognitive skills development, including problem solving, metacognitive skills, logical reasoning, and persistence (National Research Council, 2005; Holbert, & Wilensky, 2011; Goulet & Slater, 2009). This research study has focused on students' development of debugging skills as one of the computational perspectives involved, of which mechanisms are heavily dealt with in the field of computer science but ignored in education sector. Future research could therefore extend on this study and target the specific development of debugging skills within the educational domain, particularly in the mathematics and science content areas. Young students could construct their knowledge of computational thinking in the form of error correction by engaging in a step-by-step decomposed problem-solving procedure (e.g., debugging code) within coding activities.

CONCLUSION

Although novel technology can attract student attention, the way novel technology is integrated into teaching and learning determines the success of educational technology. Prior research has already noted the importance of teachers to design appropriate curriculum and implement it within appropriate technology for student learning (Looi, Wong, So, Seow, Toh, Chen, Zhang, Norris, & Soloway, 2009). For instance, a $1.3 billion project by Los Angeles Unified School District to distribute iPads with preloaded digital curriculum to all students ended up demanding a refund from Apple without significant improvement in student outcomes (Blume, 2015, Jan 12). A follow-up study (American Institutes for Research, 2014) revealed that the iPads were used as a newer version of whiteboards without new instructional methods being created, and thus, failed to improve student learning. This example shows that pedagogy should come first before technology to ensure learning gains for successful technology integration. As discussed in this chapter, incorporating physical activities that are familiar to children to embody computational perspectives as a kernel of STEM learning, in one way in which curriculum instruction may be changed while helping children to learn better with technology.

One obstacle in adopting tablets to classrooms is that the curriculum accompanying tablets tends to be limited to mere delivery of information via technology. Moreover, this content focuses on abstract concepts that are not grounded on meaningful contexts, which students can easily relate with. Therefore, developing curriculum based on where students can simulate their cognitive process with actions bound to social, cultural, and physical contexts can overcome the hurdles by facilitating young students to learn abstract concepts. Findings from the current study are promising because students appeared to have acquired necessary skills for programming and improved general cognitive skills without requiring specific instruction specific to programming. In addition, coherent curriculum design that integrates both mathematics and programming concepts not only reduced students' barriers to learn new concepts, but also allowed students to actively construct knowledge with the given programming application.

Another challenge is that administrators and leaders are not ready to help teachers establish a culture that values risk taking, promotes exploration, and celebrates innovation when adopting technology in classrooms (Schrum, Galizio, & Ledesma, 2011). Rather than solely relying on teachers to adopt technology, collaboration between school districts and teachers to establish rigorous student-centered activities can overcome these challenges. This chapter suggests physical activities through which students can practice computational perspectives that may help them improve content learning as well as develop computational thinking strategies when accompanied with iPad use in the classroom.

REFERENCES

Abrahamson, D., & Howison, M. (2010). *Embodied artifacts: coordinated action as an object-to-think-with.* Paper presented at the annual meeting of the American Educational Research Association, Denver, CO.

Abrahamson, D, & Lindgren, R. (2014). Embodiment and embodied design. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (2 ed., pp. 358-376). Cambridge: UK: Cambridge University Press.

Abrahamson, D., & Trninic, D. (2015). Bringing forth mathematical concepts: signifying sensorimotor enactment in fields of promoted action. *ZDM, 47*(2), 295-306.

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832-835.

Amer, H., & Ibrahim, W. (2014). *Using the iPad as a pedagogical tool to enhance the learning experince for novice programing students.* Paper presented at the Global Engineering Education Conference (EDUCON), 2014 IEEE.

American Institutes for Research (2014). Evaluation of the common core technology project: Interim report. Washington: DC: Margolin, J., Haynes, E., Heppen, J., Reudel, K., Meakin, J., Hauser, A., … Hubbard, A.

Anderson, J. R., & Lebiere, C. (1998) *The atomic components of thought*. Mahwah, NJ: Erlbaum

Aspen Institute Task Force on Learning and the Internet. (2014). *Learner at the center of a networked world*. Washington, DC: The Aspen Institute.

Bamberger, J., & DiSessa, A. (2003). Music as embodied mathematics: A study of a mutually informing affinity. . *International Journal of Computers for Mathematical Learning*(8), 123–160.

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.

Barsalou, L. W. (2008). Grounded cognition. *Annual Review of Psychology, 59*, 617-645. doi:10.1146/annurev.psych.59.103006.093639

Barsalou, L. W. (2010). Grounded Cognition: Past, Present, and Future. *Topics in Cognitive Science, 2*(4), 716-724. doi:10.1111/j.1756-8765.2010.01115.x

Barsalou, L. W., Niedenthal, P. M., Barbey, A. K., & Ruppert, J. A. (2003). Social embodiment. *Psychology of Learning and Motivation, 43*, 43-92.

Berland, M., & Wilensky, U. (2015). Comparing Virtual and Physical Robotics Environments for Supporting Complex Systems and Computational Thinking.*Journal of Science Education and Technology*, *24*(5), 628-647.

Bers, M. U. (2007). Project InterActions: A multigenerational robotic learning environment. *Journal of Science Education and Technology*, *16*(6), 537-552.

Bers, M. U. (2008). *Blocks to robots: Learning with technology in the early childhood classroom* (p. 154). New York: Teachers College Press.

Bers, M. U. (2008). Civic identities, online technologies: From designing civics curriculum to supporting civic experiences. *Civic life online: Learning how digital media can engage youth*, 139-160.

Beals, L., & Bers, M. U. (2009). A developmental lens for designing virtual worlds for children and youth. *International Journal of Learning and Media, 1*(1), 51-65.

Bers, M. U. (2010). Beyond computer literacy: supporting youth's positive development through technology. *New directions for youth development, 128*, 13-23.

Bers, M. U. (2010). The TangibleK Robotics program: Applied computational thinking for young children. *Early Childgood Research & Practice, 12*(2).

Bers, M. U. (2012). *Designing digital experiences for positive youth development: From playpen to playground*. Oxford University Press.

Black, J. B. (2007). Imaginary Worlds. In. MA Gluck, JR Anderson & SM Kosslyn (Eds.), Memory and mind (pp. 195-208): Mahwah, NJ: Lawrence Erlbaum Associates.

Black, J. B. (2010). An embodied/grounded cognition perspective on educational technology *New Science of Learning* (pp. 45-52): Springer.

Black, J. B., Segal, A., Vitale, J., & Fadjo, C. L. (2012). Embodied cognition and learning environment design. *Theoretical foundations of learning environments, 2*, 198-223.

Blume, H (2015, Jan 12). L.A. Unified's iPad program plagued by problems early, review says. Retrieved from http://www.latimes.com/local/education/la-me-ipad-report-20150113-story.html

Bowers, L., Huisingh, R., & LoGiudice, C. (2005). TOPS 3 elementary. Linguisystems: East Moline, IL.

Brown, J. S., Collins, A., & Duguid, P. (1989). Situated cognition and the culture of Learning. *Educational Researcher, 18*(1), 32-42.

Burke, Q. (2012) The markings of a new pencil: Introducing programming-as-writing in the middle school classroom. *Journal of Media Literacy Education, 4*(2), 121-135.

Bybee, R. W. (2014). NGSS and the Next Generation of Science Teachers. *Journal of Science Teacher Education, 25*(2), 211-221. doi:10.1007/s10972-014-9381-4

Carver, S. M., & Klahr, D. (1986). Assessing children's Logo debugging skills with a formal model. *Journal of Educational Computing Research, 2*(4),487-525.

Chang, C., & Biswas, G. (2011). *Design engaging environment to foster computational thinking*. Paper presented at the World Conference on Educational Media and Technology 2011, Lisbon, Portugal. http://www.editlib.org/p/38274

Chang, K. E., Sung, Y. T., & Lin, S. F. (2006). Computer-assisted learning for mathematical problem solving. *Computers & Education, 46*(2), 140-151.

Clements, G. N. (1985). The geometry of phonological features. *Phonology, 2*(01), 225-2

Clements, D. H. (1999). The future of educational computing research: The case of computer programming. *Information Technology in Childhood Education Annual, 1999*(1), 147-179.

Clements, D. H., & Battista, M. T. (1989). Learning of geometric concepts in a Logo environment. *Journal for Research in Mathematics Education, 20*(5), 450-467. Retrieved from http://www.jstor.org/stable/749420 .

Clements, D. H, & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology, 76*(6), 1051-1058.

Cognition and Technology Group at Vanderbilt (1990). Anchored instruction and its relationship to situated cognition. *Educational Researcher*, *19* (6), 2-10.

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In Resnick L. B. (Ed.), *Knowing, learning, and instruction: essays in honor of Robert Glaser* (pp. 453-494). Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers.

Croft, W., & Cruse, A. (2004). *Cognitive linguistics*: Cambridge University Press.

Dede, C., Mishra, P., & Voogt, J. (2013). *Working Group 6: Advancing computational thinking in 21st century learning.* Paper presented at the EDUsummIT.

Dede, C. (2010). Comparing frameworks for 21st century skills. *21st century skills: Rethinking how students learn, 20*, 51-76.

Derry, S.J., & LaJoie, S.P. (1993). A middle camp for (un)intelligent instructional computing:An introduction. In S.P. LaJoie & S.J. Derry (Eds.), *Computers as cognitive tools*. Hillsdale, NJ: Lawrence Erlbaum Associates.

DiSessa, A. (2000). Changing minds. *Computers, learning, and liter acy*.

Einhorn, S. (2012). *Micro-Worlds, Computational Thinking, and 21st Century Learning*. white paper. Logo Computer Systems Inc.

Fadjo, C. L. (2012). *Developing Computational Thinking Through Grounded Embodied Cognition.* Columbia University. Retrieved from http://www.editlib.org/p/117391

Fadjo, C. L., Lu, M., & Black, J. B.. (2009). *Instructional Embodiment and Video Game Programming in an After School Program*. Paper presented at the World Conference on Educational Media and Technology 2009, Honolulu, HI, USA. http://www.editlib.org/p/32064

Fadjo, C. L., Hallman Jr, G., Harris, R., & Black, J. (2009, June). Surrogate embodiment, mathematics instruction and video game programming. In*Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2009* (pp. 2787-2792).

Felleisen, M., & Krishnamurthi, S. (2009). Viewpoint Why computer science doesn't matter. *Communications of the ACM, 52*(7), 37-40.

Feurzeig, W, & Papert, S. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments, 19*(5), 487-501.

Feurzeig, W., Papert, S. A., & Lawler, B. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments, 19*(5), 487-501.

Fischer, U., Moeller, K., Cress, U., & Nuerk, H.-C. (2013). Interventions supporting children's mathematics school success: A meta-analytic review. *European Psychologist, 18*(2), 89.

Fryer, W. A. (2014). Hopscotch Challenges: Learn to Code on an iPad! *Publications Archive of Wesley Fryer, 1*(1).

Fullan, M. (2007). *The new meaning of educational change*. New York: Teachers College Press.

Gallese, V., & Lakoff, G. (2005). The brain's concepts: The role of the sensory-motor system in conceptual knowledge. *Cognitive neuropsychology, 22*(3-4), 455-479.

Gibbs, R. W. (2005). *Embodiment and cognitive science*: Cambridge University Press.

Glenberg, A. M. (2008). *Toward the integration of bodily states, language, and action.* Paper presented at the Embodied grounding: social, cognitive, affective, and neuroscientific approaches.

Glenberg, Arthur M. (2010). Embodiment as a unifying perspective for psychology. *Wiley Interdisciplinary Reviews: Cognitive Science, 1*(4), 586-596.

Glenberg, A. M., Gutierrez, T., Levin, J. R., Japuntich, S., & Kaschak, M. P. (2004). Activity and Imagined Activity Can Enhance Young Children's Reading Comprehension. *Journal of Educational Psychology, 96*(3), 424.

Glenberg, A. M., & Kaschak, M. P. (2002). Grounding language in action. *Psychonomic Bulletin & Review, 9*(3), 558-565.

Goulet, D. V. & Slater, D. (2009). Alice and the introductory programming course: An Invitation to Dialogue. *Information Systems Education Journal, 7*(50).

Grover, S., & Pea, R. (2013). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher, 42*(1), 38-43. doi:10.3102/0013189x12463051

Harris, J., Mishra, P., & Koehler, M. (2009). Teachers' technological pedagogical content knowledge and learning activity types: Curriculum-based technology integration reframed. *Journal of Research on Technology in Education, 41*(4), 393-416.

Holbert, N.R, & Wilensky, U. (2011). Racing Games for Exploring Kinematics: A Computatoinal Thinking Approach. *Paper presented at AERA 2011.*

Hossain, M, & Robinson, M. (2012). How to Motivate US Students to Pursue STEM (Science, Technology, Engineering and Mathematics) Careers. *Online Submission.*

Hu, W. (2012). 'Math that Moves: Schools Embrace the iPad'. The New York Times. 4 Jan 2011. Available online: http://www.nytimes.com/2011/01/05/education/05tablets.html

Johnson, M., & Lakoff, G. (2002). Why cognitive linguistics requires embodied realism. *Cognitive linguistics, 13*(3), 245-264.

Jona, K., Wilensky, U., Trouille, L., Horn, M, Orton, K., Weintrop, D., & Beheshti, E. (2014). *Embedding computational thinking in science, technology, engineering, and math (CT-STEM).* Paper presented at the future directions in computer science education summit meeting, Orlando, FL.

Jonassen, D. H., & Reeves, T. C. (1996). Learning with technology: Using computers as cognitive tools. In D. H. Jonassen (Ed.), *Handbook of research for educational communications and technology* (1st ed.).

Jonassen, D.H. & Land S.M. (2000). *Theoretical foundations of learning environments.* Mahwah, NJ: Lawrence Erlbaum Associates.

Kafai, Y.B. & Burke, Q. (2013). Computer programming goes back to school: Why and what K-12 schools need to know. *Phi Delta Kappan, 95*(1), 61-65.

Kazakoff, E.R. (2014). Toward a theory-predicated definition of digital literacy for earlychildhood. *Journal of Youth Development – Bridging Research and Practic*e (Special Issue: Media and Youth), 9(1), 41-58.

Lakoff, G., & Johnson, M. (1999). *Philosophy in the flesh: The embodied mind and its challenge to western thought*: Basic books.

Lankshear, C. & Knobel, M. (2006) *New literacies: Everyday practices and classroom learning*, 2nd edition. Maidenhead: Open University Press.

Lauricella, A., Barr, R., & Calvert, S. (2008). Emerging computer skills: Influences of young children's executive functioning abilities and parental scaffolding techniques. *Paper presented at International Communication Association Conference*, Montreal, Canada.

Law, L. C. (1998). A situated cognition view about the effects of planning and authorship on computer program debugging. *Behaviour & Information Technology, 17*(6), 325-337.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., . . . Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads, 2*(1), 32-37. doi:10.1145/1929887.1929902

Lindgren, R. (2014). Getting into the cue:Embracing technology-facilitated body movements as a starting point for learning. In Victor. R. Lee (Ed.), *Learning Technologies and the Body:Integration and Implementation in Formal and Informal Environment*: Routledge.

Lindgren, R, & Johnson-Glenberg, M. (2013). Emboldened by embodiment: Six precepts for research on embodied learning and mixed reality. *Educational Researcher, 42*(8), 445-452. doi:DOI: 10.3102/0013189X13511661

Link, T., Moeller, K., Huber, S., Fischer, U., & Nuerk, H. (2013). Walk the number line – An embodied training of numerical concepts. *Trends in Neuroscience and Education, 2*(2), 74-84. doi:http://dx.doi.org/10.1016/j.tine.2013.06.005

Liu, J., & Wang, L. (2010). *Computational Thinking in Discrete Mathematics.* Paper presented at the 2010 Second International Workshop on Education Technology and Computer Science.

Looi, C., Wong, L., So, H., Seow, P., Toh, Y., Chen, W., Zhang, B., Norris, C., & Soloway, E. (2009). Anatomy of a mobilized lesson: Learning my way. Computers and Education, 53, 1120-1132.

Matarić, M. J., Koenig, N., & Feil-Seifer, D. (2007). Materials for enabling hands-on robotics and STEM education. *AAAI spring symposium on robots and robot venues: resources for AI education*.

Mayer, R. E. (1988). *Teaching and learning computer programming: Multiple research perspectives*. Routledge.

Mayer, R. E., Dow, G., & Mayer, S. (2003). Multimedia learning in an interactive self-explaining environment: What works in the design of agent-based microworlds? *Journal of Educational Psychology, 95*, 806–813.

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

Moll, L. C. (2014). *L. S. Vygotsky and education*. London: Routledge

Motiwalla, L. F. (2007). Mobile learning: A framework and evaluation. *Computers & Education, 49*(3), 581-596.

Murray, O., & Olcese, N. (2011). Teaching and Learning with iPads, Ready or Not?" *TechTrends: Linking Research and Practice to Improve Learning.* 55(6), 42-48.

National Research Council. (2005). *Systems for state science assessment*. Committee on Test Design for K-12 Science Achievement. Wilson, M. R., & Berthenthal, M. W. (Eds.). Center for Education, Division of Behavioral and Social Sciences and Education. Washington DC: The National Academics Press.

National Science Board. (2010). *Expert panel discussion on preparing the next generation of STEM innovators.* Retrieved from https://www.nsf.gov/nsb/publications/2010/nsb1033.pdf

Niedenthal, Paula M, Barsalou, Lawrence W, Winkielman, Piotr, Krauth-Gruber, Silvia, & Ric, François. (2005). Embodiment in attitudes, social perception, and emotion. *Personality and social psychology review, 9*(3), 184-211.

Paek, S. (2012). *The impact of multimodal virtual manipulatives on young children's mathematics learning.* (Doctoral dissertation) Retrieved from ProQuest Dissertations & Theses Global database (3554708 Ed.D.).

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*: Basic Books, Inc.

Papert, S. (1987). Computer criticism vs. technocentric thinking. *Educational researcher, 16*(1), 22-30.

Pea, R. D., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology, 2*(2), 137-168. doi:http://dx.doi.org/10.1016/0732-118X(84)90018-7

Pea, R. D. (1985). Integrating human and computer intelligence. In E. L. Klein (Ed.), *New directions for child developmen*t: No. 28, Children and computers (pp. 75–96). San Francisco: Jossey Bass.

Perkovic, L., & Settle, A. (2010). Computational Thinking across the Curriculum: A Conceptual Framework. *College of Computing and Digital Media Technical Report*, 10-001.

Piaget, J. (1968). Quantification, conservation, and nativism. *Science, 162*(3857), 976-979. doi:10.1126/science.162.3857.976

Price, S., & Rogers, Y. (2004). Let's get physical: The learning benefits of interacting in digitally augmented physical spaces. *Computers & Education*,*43*(1), 137-151.

Resnick, M., Martin, F., Sargent, R., & Silverman, B. (1996). Programmable Bricks: Toys to Think With. *IBM Systems Journal, 35*(3-4), 443-452.

Resnick, M. (2007, June). All I really need to know (about creative thinking) I learned (by studying how children learn) in kindergarten. In *Proceedings of the 6th ACM SIGCHI conference on Creativity & cognition* (pp. 1-6). ACM.

Resnick, M., Rusk, N., & Cooke, S. (1999). Technological fluency and the representation of knowledge. *High technology and low-income communities: Prospects for the positive use of advanced information technology*, 263-286.

Rideout, V. J., Foehr, U. G., & Roberts, D. F. (2010). Generation M 2: Media in the lives of 8-18 year olds. Menlo Park, CA: Kaiser Family Foundation.

Rittle-Johnson, B., Siegler, R. S., & Alibali, M. W. (2001). Developing conceptual understanding and procedural skill in mathematics: An iterative process. *Journal of educational psychology*, *93*(2), 346.

Rogers, Y. and Price, S. (2004). Extending and augmenting scientific enquiry through pervasive learning environments. *Children Youth and* Environments. 14(2), 67-83.

Rogoff, B., Goodman Turkanis, C., & Bartlett, L. (2001). *Learning together: Children and adults in a school community*. New York: Oxford University Press.

Salomon, G., Perkins, D. N., & Globerson, T (1991). Partners in cognition: Extending human intelligence with intelligent technologies. *Educational Researcher, 20*, 2-9.

Sawyer, R. K. (2006). Introduction: The new science of learning. In R. K. Sawyer (Ed.) *Cambridge handbook of the learning sciences* (pp.1-16). New York: Cambridge University Press.

Scaffidi, C., Shaw, M., & Myers, B. (2005). Estimating the Numbers of End Users and End User Programmers. *Proceedings of the IEEE Symposium on Visual Languages and HumanCentric Computing*, pp. 207-214.

Scardamalia, M., & Bereiter, C. (2006). Knowledge building. *The Cambridge*.

Scardamalia, M. (2001). Getting real about 21st century education. *The Journal of Educational Change, 2*, 171-176.

Scardamalia, M., & Bereiter, C. (2006). Knowledge building. *The Cambridge*.

Schrier, K. (2006). Using augmented reality games to teach 21st century skills. *Conference proceedings, ACM Siggraph Educators Program*, Boston, MA.

Schrum, L., Galizio, L. M., & Ledesma, P. (2011). Educational leadership and technology integration: An investigation into preparation, experiences, and roles. *Journal of School Leadership Journal of School Leadership, 21*, 241-261.

Segal, Ayelet. (2011). *Do gestural interfaces promote thinking? Embodied interaction: Congruent gestures and direct touch promote performance in math.* Columbia University.

Shute, V. J., & Psotka, J. (1996). Intelligent tutoring systems: Past, present, and future. In D. Jonassen (Ed.), *Handbook of research for educational communications and technology* (pp. 570-600). New York, NY: Macmillan.

Smith, Linda, & Gasser, Michael. (2005). The development of embodied cognition: Six lessons from babies. *Artificial life, 11*(1-2), 13-29.

Squire K., & Klopfer E (2007) Augmented reality simulations on handheld computers. *Journal of Learning Science. 16*(3), 371–413.

The White House, Office of the Press Secretary. (2009). *Educate to innovate* [Press release]. Retrieved from https://www.whitehouse.gov/the-press-office/president-obama-launches-educate-innovate-campaign-excellence-science-technology-en

Van Hiele, P. M. (1986). *Structure and insight: A theory of mathematics education*.

Voogt, J., Erstad, O., Dede, C., & Mishra, P. (2013). Challenges to learning and schooling in the digital networked world of the 21st century. *Journal of Computer Assisted Learning, 29*(5), 403-413. doi:10.1111/jcal.12029

Voskoglou, M. G., & Buckley, S. (2012). Problem solving and computational thinking in a learning environment. *arXiv preprint arXiv:1212.0750*.

Wilson, A. D., & Golonka, S. (2013). Embodied Cognition is Not What You Think It Is. *Frontiers in Psychology, 4*. doi:10.3389/fpsyg.2013.00058

Wilson, M. (2002). Six views of embodied cognition. *Psychonomic Bulletin & Review, 9*(4), 625-636. doi:10.3758/BF03196322

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-36.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717-3725.

Wing, J. M. (2011). *Computational thinking.* Paper presented at the VL/HCC.

Wenglinsky, H. (2005). *Using technology wisely: The keys to success in schools*. New York: Teachers College Press.

Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*. Los Altos, CA: Morgan Kaufmann Publishers, Inc.

Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T. (2011). *Introducing computational thinking in education courses.* Paper presented at the ACM Special Interest Group on Computer Science Education, Dallas, TX.

ENDNOTES

[1] http://www.cs.cmu.edu/~CompThink/

[2] news article published on 1.31.2012 on Spotlight on Digital Media and Learning. http://spotlight.macfound.org/featured-stories/entry/programming-with-scratch-jr-when-it-comes-to-screen-time-and-young-kids/