Pedagogika, 2018, 68(3), 279–292
http://pages.pedf.cuni.cz/pedagogika/

Unplugged Programming: The future of teaching computational thinking?

George Aranda, Joseph Paul Ferguson

Abstract: *We currently live in digital times, with educators increasingly coming to realise the need to prepare students to productively participate in such a coding-infused society. Computational Thinking (CT) has emerged as an essential skill in this regard. As with any new skill, the ways it is theorised and practiced vary greatly. In this paper, we argue for the importance of Unplugged Programming (UP) as a hands-on and practical approach to teaching and learning, which emphasises embodied and distributed cognition. UP has the potential to open up what it means to enact CT in the classroom when computational devices are put to the side. Preparing for the issues of the future is a matter of reconnecting with the past, in particular with ideas such as epistemological pluralism. By appreciating the diversity of ways that students can undertake CT and teachers can support them in doing so – from coding with digital devices to pencil-and-paper programming – we can work to make the classroom a place in which students can explore and undertake CT in rich and diverse ways.*

Keywords: *computational thinking, unplugged programming, coding, epistemology, distributed cognition, embodied cognition.*

1. Introduction

In response to the unfolding digital age in which we live, there is a renewed focus in schools to prepare students to productively participate in a 21st century society that is increasingly ruled by digital technology. This has seen the recent introduction of Computational Thinking (CT) as an essential skill, comparable to reading and writing, to education curricula around the world. As with any new skill, the ways it is framed, taught and assessed can vary greatly (Lockwood & Mooney, 2018). This paper focuses on a trend of teaching CT known as Unplugged Programming (UP) that changes the way we focus on some of these issues by not requiring the use of computational devices. UP has been used in the teaching of CT as educators start to come to terms with this new part of the curriculum as it provides a hands-on and practical way of teaching concepts central to CT. We focus on one particular version of UP, known as CS Unplugged, in order to explore the educative potential of this epistemological approach.

DOI: 10.14712/23362189.2018.859

2. Computational Thinking

Definitions of CT vary across the literature, but all have their roots in Papert's (1991, p. 1) "constructionism" that he began to explore in the early 1980s with his influential book, *Mindstorms – Children, computers and powerful ideas* (Papert, 1980). Papert was concerned with the way in which students could engage in programming as bricoleurs – tinkering with what they had available to them at the time in order to achieve their computing goals. This approach to computing has a focus on the concrete – students work intimately with the machines almost as other beings. Constructionism is part of the 'constructivist' tradition, and as such, learning is considered a process through which the student constructs their own understandings as opposed to a simple exchange of information from teacher to student (Papert, 1991). But more than this, constructionism highlights the importance of students creating, and publicly sharing, a meaningful product that involves working with materials in a very direct and tangible way (i.e. concretely) (Papert, 1991). For Papert (1980; 1991), engaging in these processes is more than just a way to program a machine; it is a means for students to make meaning of their world – a way of knowing and doing. Within this constructionist context, the student of computer science as bricoleur is welcomed as are other epistemological approaches (Papert, 1991). Constructionism, as such, is intimately tied to an openness to different ways of knowing, what Turkle and Papert (1990, p. 128) refer to as "epistemological pluralism."

The current notion of CT was largely popularised by Janette Wing who in 2006 framed CT within terms of problem-solving. Over time this definition has been expanded to include the role of agents to carry out these solutions. *"Computational thinking is the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer – human or machine – can effectively carry out"* (Wing, 2014, para. 5). Inherent in this definition is the use of computational devices such as computers and robots, but also the fact that humans can enact the role of the computer and execute programs, something important to the ideas of UP.

Selby & Woollard (2014) examined a broad number of CT definitions to develop a list of concepts that were consistent across the literature. They proposed a number of core CT concepts, including: logical thinking, algorithmic thinking, decomposition, generalisation and pattern recognition, modelling, abstraction and evaluation. In contrast to Wing's definition, problem-solving was not included in their definition of CT, indicating that "although there appears to be a consensus that computational thinking is a type of problem solving, the term may not be sufficiently specific to define it (Selby & Wollard, 2014, p. 4). Other disagreements about whether CT should be implemented by a computer have been proposed by Barr and Stephenson (2011) and Bers (2018), the latter stating that it is not enough to solve problems with CT, but that the solution needs to be enacted by a computational device. Together

these controversies highlight the fluid and disputed nature of CT definitions and how they are applied differently across the varying context of education.

3. Unplugged Programming

Unplugged Programming (UP) broadly refers to learning CT and computer science concepts without relying on computational devices. This may be done through role-play, manipulation of real-world objects (e.g. post-it sticky notes, cards, wooden blocks) and the physical actions of the body, among others. Tim Bell (Bell et al., 2009), one of the creators of CS Unplugged, claims that learning this way is not simply about simulating the processes of the computer, but rather it concerns providing students with the opportunity to explore the fundamental ideas of computer science without being encumbered by the technical expertise required to code.

CS Unplugged

A popular example of UP is CS Unplugged (Computer Science Unplugged; http://www.csunplugged.org) that was initially developed in the late 90s by the Computer Science Education Research Group, at the University of Canterbury in New Zealand, and promotes learning computer science concepts without computers using constructivist strategies of learning (ACER, 2016; CSUnplugged, 2015). This includes: drawing, problem-solving, interacting with physical objects, and enacting fundamental aspects of computer science (e.g. conditional statements). CS Unplugged was originally designed by computer science lecturers and school teachers (CSUnplugged, 2015) who subscribed to the pedagogical approach of allowing students to explore computer science ideas before working with a computer.

"*We have found that many important concepts can be taught without using a computer—in fact, sometimes the computer is just a distraction from learning. Often computer science is taught using programming first, but not every student finds this motivating, and it can be a significant barrier to getting into the really interesting ideas in computer science.*" (CSUnplugged, 2015, p. i)

The course was first designed for primary school aged children and has been successfully used by students of all ages, including in higher education and senior citizen contexts, and in formal and informal educational settings (e.g. school camp programmes such as referred to below) (Earp, 2016). The creation of the course was motivated by the desire to involve primary school students in computer science, and provide them access to computer science concepts by undertaking guided 'hands-on' activities with few instructions, making it more straightforward for teachers to run and encouraging students to explore these concepts and to begin to construct their own understandings, all of which is consistent with constructivist approaches to learning (Tytler et al., 2013).

Around the world there has been an upsurge in the teaching of CT as these concepts are incorporated into curricula and teachers are thus expected to teach

these concepts, with which they may be unfamiliar or inexperienced (ACER, 2016; Sentance & Csizmadia, 2017). CS Unplugged provides them with resources which are free, have a clear rationale and are connected with multimedia resources that can extend the learning experience. Earp (2016) highlights that quite often the challenge with teaching these new ideas extends beyond the concepts themselves; it is the intricacies of the software involved and how to use it with students that often undermines the confidence of teachers. When engaging with CT concepts via CS Unplugged, when surveyed, teachers reported increases in confidence when the focus on the computational devices is removed and the ideas can be explored in direct, meaningful ways (Blum & Cortina, 2007).

CT Conceptual Development

Research has demonstrated that UP can facilitate the teaching of CT concepts (AlAmer et al., 2015; Brackman et al., 2017). Brackman et al. (2017) utilised a quasi-experimental design to examine the CT skills developed after an experimental group (years 5-6) engaged in UP activities over five weeks. Quantitative analysis of the scores of their CT test demonstrated a statistically significant larger global effect size of the experimental group when compared to controls. AlAmer et al. (2015) utilised UP in an outreach program and reported increases in the utilisation of computer science concepts taught on the camps as evidence of greater understanding of these concepts. In a comparative study by Wohl, Porter and Clinch (2015), it was reported that teaching using UP activities generated the highest level of understanding of CT concepts when compared to coding in Scratch in an early primary education context. By contrast, a study reported inconclusive learning of CT concepts by the use of UP activities such as the Binary Numbers task (Campos, Cavalheiro, Foss, Pernas, Piana, Aguiar, Du Bois, & Reiser, 2014, cited by Brackman et al., 2017). Together, these studies demonstrate some gains in the learning of CT concepts by students undertaking UP activities, although more research needs to be done to examine the influence of learning context, and how the specifics of different activities facilitate the learning of particular CT concepts. Historically, however, more research has been conducted to examine the benefits of UP for changing students' attitudes to computer science.

Attitudinal Change

A decline in students' interest in the field of computer science (Bell et al., 2009) paralleled by an increased perception of the importance of coding and data manipulation in STEM and non-STEM fields, has led to the recognition that students are not positively experiencing computer science and coding and that this needs to change. CS Unplugged activities often form part of the suite of learning experiences offered at outreach programs and school camps (AlAmer et al., 2015; Ericson & McKlin, 2012; Mano, Allan, & Colley,

2010; Urness & Manley, 2013), which are designed to expose students of different ages to computer science concepts. Additional purposes of these programmes are to encourage positive ideas about careers in computer science, self-identification by participants of relevant skills, and to foster positive representations of people who work in the field. Outreach programmes have utilised CS Unplugged with elementary school (Lambert & Guiffre, 2009), middle school (Mano et al., 2010) and high school (Taub, Ben-Ari, & Armoni, 2009) students. These studies have indicated positive gains in student interest in computer science and their confidence in cognitive and mathematical competence, as well as more positive representations of people working in the field. Similar positive effects of using CS Unplugged have been reported in workshops with high school computer science teachers (Blum & Cortina, 2007). While it is unclear whether these increases are short- or long-term, they do indicate that UP activities can have a positive influence on attitudes to computer science and coding.

Criticisms

While it has been shown that CS Unplugged can lead to gains in content knowledge about CT and attitudinal change to computer science as a career, studies have criticised the programme. It has been criticised for focusing too much on the macro aspects of computer science (e.g. binary numbers, designing algorithms) at the cost of the micro aspects (e.g. local and global variables, conditional statements) (AlAmer et al., 2015). Feaster et al. (2011) reported contradictory findings using a quasi-experimental survey design with high school students. They reported that across the student groups, there were no improvements in the students' understandings of computer science concepts or student attitudes. The authors highlight the importance of the role of kinaesthetic learning in CS Unplugged, and that while it may be an effective technique in earlier years of school, high school students involved in their study (years 9-12) might not have found it as engaging. Additionally, the students in their study were already studying computer science and may have found these introductory activities too simplistic, even though the authors modified the activities for these more experienced students. The lack of change of attitudes might be attributable to the students' potential lack of interest in computer science, compared to earlier studies using outreach groups where participants are self-selecting and presumably interested in the field.

Others, such as Bers (2018), have criticised UP programmes for framing CT as a process of problem-solving, but not allowing a means for students to express their ideas with the creation of an external artefact. Bers (2012) proposed the metaphor of coding in a 'playground' versus a 'playpen'. The former captures the notion of self-expression when coding and learning early CT concepts, while the latter highlights limitations to expression, risk-taking and learning opportunities (Bers, 2018). We would argue that while UP

practices should be built upon by coding at some stage, they have their own benefits such as what they offer as embodied and distributed cognition which should be explored and valued.

4. Embodied and Distributed Cognition

Computers as Extensions of the Mind and Body

Research in the cognitive sciences (Hollan, Hutchins, & Kirsh, 2000; Hutchins, 1995; Zhang, 1996, 1997) has established that the interaction between humans and machines, including in the classroom context, is not simply that between an individual with an isolated brain and an external tool known as a computer. Rather, when humans interact with machines they form a system such that cognition is distributed (Hollan et al., 2000). There is no separate human and machine, but rather a human/machine. We make use of these machines as constructors of representations, or as representations themselves, that can then be used to generate new understandings (Zhang & Patel, 2006; Zhang & Wang, 2009). We make tools afforded by the environment around us (Gibson, 1979; Norman, 1988).

The human involvement in these systems is not simply mental, but also bodily. The body enacts meaning-making as part of this human/machine. The human cannot interact and form a system with the machine without the body; the body here serves an epistemic as well as a haptic role. We make meaning through the mind and the body. By physically interacting with the world around us, whether this is with whatever is at-hand and/or purpose-built devices or making subtle gestures and/or large movements of the body, and perceiving (often through seeing) that which we encounter, we can form a relationship with our environment that enables us to construct an understanding of the world that transcends that which is achievable by any one individual (Gibson, 1979; Hayes & Kraemer, 2017). In this way, "the brain, body, and environment comprise a single, dynamic system" (Hayes & Kramer, 2017, p. 2). Further, we do so in collaboration with others - it is a social process – and in the context of a particular culture (Hollan et al., 2000). For us, the social is the school classroom and the cultural is the computer culture (globally and locally).

The Multimodal Nature of Science and Mathematics

The embodied and distributed nature of learning is well established in the education literature and is receiving increasing attention in STEM education research (Hayes & Kraemer, 2017; Weisberg & Newcombe, 2017). The rise of digital technology and the need for students and teachers to make effective use of such advances has co-occurred with a renewed focus on STEM. Much research has concerned exploring the need to recognise the multimodal nature of mathematics (e.g. Núñez, 2012; Tran, Smith, & Buschkuehl, 2017) and science (e.g. Xu & Clarke,

2012). Research in mathematics education has explored the embodied and distributed nature of meaning-making when it comes to mental arithmetic (Vallee-Tourangeau, Sirota, & Vallee-Tourangeau, 2016), statistics (Rueckert et al., 2017), interpreting graphs (Michal & Franconeri, 2017) and algebra (Marghetis, Landy, & Goldstone, 2016). Research in science education is making significant headway exploring the embodied and distributed nature of meaning making in science, for example in physics (Johnson-Glenberg & Megowen-Romanowicz, 2017), geoscience (Jaeger, Wiley, & Moher, 2016) and earth science (Atit et al., 2016), subdisciplines which conceptually focus on spatial and temporal dimensions and therefore can be more easily related to the movement of the body in its relationship with the environment.

These studies demonstrate that the learning of mathematics and science, and thus the teaching of these disciplines, is multimodal in nature. Students make use of physical and virtual artefacts (including the latest digital technology), as well as using their hands to gesture and their bodies (sometimes their whole bodies, other times just parts) to enact key processes and concepts. Through these processes, the abstract concepts that in many ways define the STEM disciplines are rendered more concrete and thus understandable.

We consider these tools that students make use of, including computers, as representations, such that we can think of a human/representation system (not just a human/machine system as earlier discussed). Anything that is useful for making meaning can function in this role as a representation. In this way, meaning is made by students in mathematics and science through the use of their minds and bodies to interact with these representations, in order to explore in a multimodal way the meaning of particular phenomena.

Considering UP from the Distributed and Embodied Perspective

Research exploring the embodied and distributed nature of UP is starting to emerge, in particular the work of Sung, Ahn and Black (2017) building on the research of Fadjo (2012). This research explores the way in which the body, as part of a larger distributed system, is involved in students' development of key computational skills and concepts. Fadjo (2012) explores the way in which students' engagement with coding software, specifically Scratch, is more productive if students also embody – act out – the relevant code. Students are then more likely to produce meaningful products and to develop their CT. Sung et al. (2017) investigate the way in which the development of students' mathematical skills (addition, subtraction, number line, magnitude comparisons) and computational skills (programming accuracy and proficiency, as reflected in abstraction, sequential thinking and pattern recognition) are influenced by the degree of embodiment required by the UP activities. They discovered that a higher level of embodiment, in the form of students enacting full body movements with a large

number line on the floor, led to better performance on mathematics tests and programming with Scratch Jnr, than activities requiring a lower level of embodiment, in the form of hand movements with a number line on a piece of paper. However, the value of this increased embodiment was dependent on students adopting the perspective of a computer programmer (i.e. a computational perspective that considers CT as a way to problem solve across domains and sometimes independently of computers). Sung et al. (2017, p. 449) concluded "that a greater degree of bodily engagement supports the perceptual experiences of learners by providing concrete experiences."

Therefore, there is evidence emerging that CT – as a key educational construct – as explored through UP is embodied and distributed much like other processes producing knowledge. We suggest that such research is key to developing a better understanding and appreciation of the value of UP for CT as an educational priority. As Sung et al. (2017, p. 447) argue: "little attention has been paid to the design of learning procedures that CT can be exhibited without technology tools." There is a need to explore "interventions…designed to practice CT without the programming application" (Sung et al., 2017, p. 448).

We argue, as researchers committed to supporting the development of CT among teachers and students, that we need to start to explore in detail the embodied and distributed nature of UP as this seems key to valuing it as a part of a multimodal approach to CT. For example, while Sung et al. (2017) question the value of students working with pencil and paper as an embodied process of meanin-making, we suggest that further exploring the value of UP practices that involve writing and drawing, such as pen and paper programming (Kim, Kim, & Kim, 2013), is an important way forward in better understanding the value of UP for CT. The value of drawing and writing as physical processes that contribute in a meaningful way to knowledge production is well established in the literature (e.g. Britton, 1980; Latour, 1986; Magnani, 2013). We plan to extend this notion of "thinking through drawing" (Magnani, 2013, p. 303) – or what Britton (1980, p. 147) calls "shaping at the point of utterance" – to UP. In such cases, the objects that students are close to are not the machines, but rather the code as written/drawn representations.

We propose that UP, by enabling this distancing from machines and closeness to code, can enable students to develop a deeper understanding of programming – a more powerful form of CT – that might then enable them to interact with machines in more productive ways. Students are likely to be able to undertake more effective CT by having this distance from the machines. Our closeness to machines in the digital age means that it is difficult to understand and engage in CT in the desired way. UP offers a possible way out of this but, in order for this to happen, there needs to be a broader theoretical shift in how we conceptualise what it might mean to teach and learn CT.

5. Looking Back to Move Forward: Epistemological Pluralism

Different Ways of Knowing and Doing

Almost three decades ago, Turkle and Papert (1990; 1992) made a claim for the importance of "epistemological pluralism" (Turkle & Papert, 1990, p. 128) in the rapidly-developing world of computer science, particularly in the educational context. Motivated by the desire to make computer science, principally programming, appealing and accessible to as many students as possible, they desired a democratic computer culture to replace what they considered to be the dominant computer culture of the time. This was a culture defined by an insistence on a very formal and abstract approach to programming, the remnants of which are still evident in the now dominant and relatively conservative notions of CT and how it is taught. While many students were inclined to program in this traditional way, not all were happy to do so. As a result, many students felt excluded from the world of computer science because they did not conform to the dominant epistemology.

What was needed, Turkle and Papert (1990; 1992) argued, was a cultural revolution of sorts such that computer science could be undertaken in different ways. In such a culture, all the different ways of programming and relating to machines are not only tolerated but celebrated. There are different ways of knowing – and indeed teaching, we can talk of pedagogical pluralism as well – when it comes to programming, and these need to be fostered and encouraged as part of a democratic computer culture. Constructionism (Papert, 1991) was the alternative pedagogy to the formal and abstract ways dictated by the dominant computer culture of the early 1990s.

A lot has changed since Turkle and Papert (1990; 1992) first proposed epistemological pluralism, and Papert (1991) outlined constructionism as a distinct and legitimate pedagogy. Revolution has indeed taken place, but perhaps in unexpected ways (as tends to be the case with revolutions). We are closer to machines than ever before and they are now a key part of school life – not just computer science, but all disciplines are permeated by their presence. While this permeation of education technology is a complex issue, there is certainly more acceptance these days of different ways of interacting with machines, and conceptualising and enacting programming. Something we argue would make Turkle and Papert happy.

6. Discussion

We argue that in order for UP to be valued as a means for students to develop CT, then we must turn again to epistemological pluralism as a way to frame not just computer science, but all instances – spanning diverse age groups and disciplines – in which students may now undertake CT. And this epistemological pluralism must also extend to the way in which we

research; a multimodal account of UP and other approaches, in particular the embodied nature of these meaning-making processes, is only possible if we are open to different ways of knowing and thus defining these constructs. By embracing a pluralistic approach to ways of knowing and teaching CT, then UP comes to be considered as an important aspect of all classrooms in which CT is valued.

Epistemological pluralism opens up the frontiers of CT as an educational priority. Students can interact with tablets and laptops to code in the digital world, as well as get closer to the underlying code and program through UP, which has the potential to change for the better our relationships with these machines. This is not to say that UP should be the only way or even the preferred way for students and teachers to enact CT. In the tradition of Turkle's and Papert's (1990; 1992) epistemological pluralism, UP must be used alongside other ways of knowing and relating to machines. For example, books such as Linda Luikas' *Hello Ruby* series have provided an imaginative and playful way for young students to connect with abstract concepts, while Martin Erwig's *Once Upon an Algorithm* offers older students ways to re-examine fairy tales through the lens of CT. Making use of literature in this way offers rich connections to CT concepts, while trends such as the rise of 'augmented reality' and 'mixed reality' offer opportunities for connecting interactions between the digital and physical worlds through the use of products such as OSMOs (Cortez, 2017; Wertheim, 2018) that could revolutionise the ways we teach and learn. Similarly, tangible coding technologies, such as Bee Bots, Cubettos and Cubelets, offer another means of coding without screens. Each of these approaches involves different and varied objects – computational and non-computational, but always meaningful – as products and as devices that enable different ways of thinking and doing. It is not a case of using UP just to lead to direct interactions with machines, but rather the focus must be for teachers and students to engage in various ways of knowing in a linked and iterative way that will best support CT.

We need to conceptualise and enact CT as a way of reasoning that students can use to solve the problems they encounter as citizens of this digital age in which we live. But what does this look like pedagogically? How can CT as a key focus of schools be taught in such a way that all students are provided with an opportunity to develop the necessary knowledge and skills to prosper now and in the future?

We propose that we can meaningfully prepare the next generation for a future as responsible, productive and self-aware digital beings by embracing different ways of knowing when it comes to CT. There is more than one way to enact CT, as Turkle and Papert (1990; 1992) so importantly pointed out. This not only welcomes more students to engage in programming and to forge more productive relationships with machines, but it opens up possibilities for teachers – who may be self-conscious about how to go about teaching CT in the classroom – to start to develop their own

ways of teaching their own conceptions of CT. But this is only realisable if the curricula and policy frameworks within which teachers and students operate are supportive of epistemological pluralism, including UP. We must work with teachers, as well as those designing curricula and policy, to develop an appreciation and understanding of the different epistemological approaches, and the power of linking these through carefully considered sequences as part of a coherent and interdisciplinary undertaking of CT. The full value of UP, and the other ways of enacting CT, is only realised when they are considered as part of a multifaceted approach to CT. What we need is a new plurality, one that reflects our current times.

References

ACER (2016). *Unplugging computer science.* Retrieved from https://rd.acer.org

AlAmer, R. A., Al-Doweesh, W. A., Al-Khalifa, H. S., & Al-Razgan, M. S. (2015). Programming unplugged: Bridging CS Unplugged activities gap for learning key programming concepts. *Proceedings of the Fifth International Conference on e-Learning* (97-103). Manama, Bahrain, October 18-20.

Atit, K., Weisberg, S. M., Newcombe, N. S., & Shipley, T. F. (2016). Learning to interpret topographic maps: Understanding layered spatial information. *Cognitive Research: Principles and Implications, 1*(1), 1-18.

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.

Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *New Zealand Journal of Applied Computing and Information Technology*, *13*(1), 20-29.

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

Bers, M. U. (2018). *Coding as a playground: Programming and computational thinking in the early childhood classroom*. New York: Routledge.

Blum, L., & Cortina, T. (2007). *CS4HS:* An outreach program for high school CS teachers. In *Proceedings of the 38th ACM Technical Symposium on Computer Science Education* (pp. 19-23). New York, March 7-11.

Brackman, C. P., Roman-Gonzalez, M., Robles, G., Moreno-Leon, J., Casali, A., & Barone, D. (2017). Development of computational thinking skills through unplugged activities in primary school. In *Proceedings of the 12th Workshop on Primary and Secondary Computer* Education (pp. 65-72). Nijmegen, Netherlands, November 8-10.

Britton, J. (1980). Shaping at the point of utterance. In A. Freedman & I. Pringle (Eds.), *Reinventing the rhetorical tradition* (pp. 61-65). Conway, AR: L & S Books.

Cortez, M. B. (2017). How will AR transform education? [#Infographic]. *EdTech Magazine*. Retrieved from https://edtechmagazine.com

CSUnplugged.org. (2015). *Computer Science Unplugged*. Retrieved from https://csunplugged.org

Earp, J. (2016). The research files special episode: Professor Tim Bell. [on-line] Retrieved from https://www.teachermagazine.com.au

Ericson, B., & McKlin, T. (2012). Effective and sustainable computing summer camps. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (pp. 289-294). Raleigh, North Carolina, February 29 - March 3.

Fadjo, C. L. (2012). *Developing computational thinking through grounded embodied cognition.* Graduate School of Arts and Sciences, Columbia University. Doctoral dissertation. Retrieved from https://academiccommons.columbia.edu

Feaster, Y., Segars, L., Wahba, S., & Hallstrom, J. (2011). Teaching CS unplugged in the high school (with limited success). In *Proceedings of the 16th annual joint conference on Innovation and Technology in Computer Science Education* (pp. 248-252). Darmstadt, Germany, June 27-29.

Gibson, J. (1979). *The ecological approach to visual perception*. Boston: Mifflin.

Hayes, J. C., & Kraemer, D. J. M. (2017). Grounded understanding of abstract concepts: The case of STEM learning. *Cognitive Research: Principles and Implications, 2*(1), 1-7.

Hollan, J., Hutchins, E., & Kirsh, D. (2000). Distributed cognition: Toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction, 7*(2), 174-196.

Hutchins, E. (1995). *Cognition in the wild*. Cambridge, MA: MIT Press.

Jaeger, A. J., Wiley, J., & Moher, T. (2016). Leveling the playing field: Grounding learning with embedded simulations in geoscience. *Cognitive Research: Principles and Implications, 1*(1), 1-23.

Johnson-Glenberg, M. C., & Megowan-Romanowicz, C. (2017). Embodied science and mixed reality: How gesture and motion capture affect physics education. *Cognitive Research: Principles and Implications, 2*(1), 1-24.

Kim, B., Kim, T., & Kim, J. (2013). Paper-and-pencil programming strategy toward computational thinking for non-majors: Design your solution. *Journal of Educational Computing Research, 49*(4), 437-459.

Lambert, L., & Guiffre, H. (2009). Computer science outreach in an elementary school. *Journal of Computing Sciences in Colleges archive, 24*(3), 118-124.

Latour, B. (1986). Visualisation and cognition: Drawing things together. In H. Kuklick (Ed.), *Knowledge and society. Studies in the sociology of culture past and present* (vol. 6, pp. 1-40). Stamford, CT: Jai Press.

Lockwood, J., & Mooney, A. (2018). Computational Thinking in education: Where does it fit? A systematic literary review. *International Journal of Computer Science Education in Schools*, *2*(1), 41-60.

Magnani, L. (2013). Thinking through drawing. *The Knowledge Engineering Review, 28*(3), 303-326.

Mano, C., Allan, V., & Colley, D. (2010). Effective in-class activities for middle school outreach programs. In *Proceedings of the 40th ASEE/IEEE Frontiers in Education Conference* (pp. F2E1-6). Washington, D.C., 27-30 October.

Marghetis, T., Landy, D., & Goldstone, R. L. (2016). Mastering algebra retrains the visual system to perceive hierarchical structure in equations. *Cognitive Research: Principles and Implications, 1*(1), 1-25.

Michal, A. L., & Franconeri, S. L. (2017). Visual routines are associated with specific graph interpretations. *Cognitive Research: Principles and Implications, 2*(1), 1-20.

Norman, D. (1988). *The design of everyday things*. New York: Doubleday.

Núñez, R. (2012). On the science of embodied cognition in the 2010s: Research questions, appropriate reductionism, and testable explanations. *Journal of the Learning Sciences, 21*(2), 324-336.

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

Papert, S. (1991). Situating constructionism. In I. Harel & S. Papert (Eds.), *Constructionism* (pp. 193-206). Norwood, New Jersey: Ablex.

Rueckert, L., Church, R. B., Avila, A., & Trejo, T. (2017). Gesture enhances learning of a complex statistical concept. *Cognitive Research: Principles and Implications, 2*(2), 1-6.

Selby, C., & Woollard, J. (2014). *Computational Thinking: The developing definition*. Paper presented at the 45th ACM Technical Symposium on Computer Science Education, Atlanta, Georgia, March 5-8.

Sentance, S., & Csizmadia, A. (2017) Computing in the curriculum: Challenges and strategies from a teacher's perspective. *Education and Information Technologies, 22*(2), 469-495.

Sung, W., Ahn, J., & Black, J. B. (2017). Introducing computational thinking to young learners: Practicing computational perspectives through embodiment in mathematics education. *Technology, Knowledge and Learning, 22*(3), 443-463.

Taub, R., Ben-Ari, M. & Armoni, M. (2009). The effect of CS Unplugged on middle-school students' view of CS. In *Proceedings of the 14th ACM Annual Conference on Innovation and Technology in Computer Science Education* (pp. 99-103). Paris, July 6-9.

Tran, C., Smith, B., & Buschkuehl, M. (2017). Support of mathematical thinking through embodied cognition: Nondigital and digital approaches. *Cognitive Research: Principles and Implications, 2*(1), 1-16.

Turkle, S., & Papert, S. (1990). Epistemological pluralism: Styles and voices within the computer culture. *Signs, 16*(1), 128-157.

Turkle, S., & Papert, S. (1992). Epistemological pluralism and the revaluation of the concrete. *Journal of Mathematical Behavior, 11*(1), 3-33.

Tytler, R., Prain, V., Hubber, P., & Waldrip, B. (Eds.) (2013). *Constructing representations to learn in science*. Rotterdam: Sense.

Urness, T., & Manley, E. D. (2013). Generating interest in computer science through middle-school Android summer camps. *Journal of Computing Sciences in Colleges*, *28*(5), 211-217.

Vallée-Tourangeau, F., Sirota, M., & Vallée-Tourangeau, G. (2016). Interactivity mitigates the impact of working memory depletion on mental arithmetic performance. *Cognitive Research: Principles and Implications*, *1*(1), 1-26.

Weisberg, S. M., & Newcombe, N. S. (2017). Embodied cognition and STEM learning: Overview of a topical collection in CR:PI. *Cognitive Research: Principles and Implications, 2*(1), 1-6.

Wertheim, M. (2018). *Virtual reality: from Giotto to VRporn*. The Monthly, February 2018, accessed on May 2, 2018. Retrieved from https://www.themonthly.com.au

Wing, J. M. (2014). *Computational thinking benefits society* [blog post]. Retrieved from http://socialissues.cs.toronto.edu

Wohl, B., Porter, B., & Clinch, S. (2015). Teaching computer science to 5-7 year-olds: An initial study with Scratch, Cubelets and unplugged computing. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (pp. 55-60). London, United Kingdom, November 9-11.

Xu, L., & Clarke, D. (2012). What does distributed cognition tell us about student learning of science? *Research in Science Education, 42*(3), 491-510.

Zhang, J. (1996). A representational analysis of relational information displays. *International Journal of Human-Computer Studies, 45*(1), 59-74.

Zhang, J. (1997). Distributed representation as a principle for the analysis of cockpit information displays. *International Journal of Aviation Psychology, 7*(2), 105-121.

Zhang, J., & Patel, V. L. (2006). Distributed cognition, representation, and affordance. *Pragmatics and Cognition, 14*(2), 333-341.

Zhang, J., & Wang, H. (2009). An exploration of the relations between external representations and working memory. *PLoS One, 4*(8), 1-10.

George Aranda (Australia), Deakin University, School of Education;
e-mail: george.aranda@deakin.edu.au

Joseph Paul Ferguson (Australia), Deakin University, School of Education;
e-mail: joe.ferguson@deakin.edu.au