Supporting Physically Active CS-Ed for Children: Exploring the Design of Physical Play Friendly Coding Blocks

Noah Q. Cowit
Department of Information Science
University of Colorado Boulder
Boulder, CO, USA
Noah.Cowit@colorado.edu

Junnan Yu
School of Design
The Hong Kong Polytechnic University
Hong Kong SAR
junnan.yu@polyu.edu.hk

ABSTRACT

In this work, we share our design exploration of coding blocks to incorporate physical play into programming kits for children's computing education. First, we cover the tradition of experiential learning in computing education, with descriptions of where physical play fits into that practice. Next, we describe children's programming workshops to explore how physical play could be incorporated into coding kits. From these workshops, we recommend a set of coding blocks for physical play divided into four categories: (1) motion sensing, (2) sound sensing, (3) proximity sensing, and (4) gameplay information. These coding blocks for the first time systematically present physical play friendly programming commands, supplementing previous works on developing coding tools to combine physical play and coding for children. Finally, we describe our implementation of these coding blocks on the micro:bit—a low-cost widely distributed computer science educational kit—and describe the results of a functionality test with nine graduate design students.

CCS CONCEPTS

• **Social and professional topics** → K-12 education; Children; • **Human-centered computing** → Interaction design; • **Software and its engineering** → Visual languages.

KEYWORDS

Computational learning, Physical play, Children, Coding blocks

ACM Reference Format:
Noah Q. Cowit and Junnan Yu. 2024. Supporting Physically Active CS-Ed for Children: Exploring the Design of Physical Play Friendly Coding Blocks. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '24), May 11–16, 2024, Honolulu, HI, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3613905.3650973

1 INTRODUCTION

In the contemporary world computational literacy is a universally required skill, and thus it is of great importance that children are provided engaging opportunities to learn with and gain experience using computers [6]. The typical model of computer science education predominantly follows a sedentary approach in front of a screen, which can decrease engagement and enforce the stereotype of computing as a socially isolated discipline only of interest to those with a singular focus on computers [8]. This may discourage participation from those who don't fit into this stereotype, for instance children who prefer to spend free time outdoors playing with others [8]. There have been a variety of attempts to make children's computing education more physical and less focused on sterile computer lab environments, including integrating computer science in sports [5, 38], incorporating gestures into programming environments [1, 16], and programming stepping games through innovative educational kits [33]. However, to support physically active computational learning, more design and development work on physical-play-friendly programming environments is needed. In this paper, we outline our design exploration of coding blocks needed to accommodate physical play in four phases: 1. In-person workshops with young children (Section 3.1) to gather broad insights on how programming education and physical play could be incorporated together, 2. Analyzing the mechanisms and movements of representative physical play activities (Section 3.2), 3. Recommended coding blocks for physical play (Section 3.3), 4. Implementation (Section 4.1) and testing (Section 4.2) of custom code blocks for general physical play in the micro:bit Makecode interface, and 5. Design reflections for future work.

2 BACKGROUND AND RELATED WORK

2.1 Experiential Computing Education

Debate between instruction-based and experience-based education is as old as modern education theory, exemplified in work of Edward Thorndike/B. F. Skinner [28, 32] and John Dewey [9] respectively. While instructionist models of education dominated American school systems for most of the 20th century [18], more recently the learning sciences have explored an experiential model of education to encourage deep conceptual knowledge, aiming to create comprehensive learning environments with opportunities for real-world inquiry and practice, conversations about learning, and the creation of learning artifacts [26]. Few places are the learning sciences more applicable than in the field of computer science education, with many key elements of the computer science (e.g., solving authentic problems, working collaboratively) only possible in experiential environments [23]. This constructivist understanding of computer science education requires an understanding that the schoolchild learning computer science is engaged in the same kind of intellectual process as a professional computer scientist [7]. The schoolchild learning computer science is a computer scientist.

While some strands of computational thinking research have asserted that programming is unnecessary for introductory computer science education and instead highlight general interdisciplinary problem-solving skills [3, 21], other computer science education researchers have consistently highlighted the disciplinary importance of programming as a practical avenue to learning in computer science [4]. Programming has long been conceptualized as a fundamental competency of computer science and as an essential means of expressing and consuming ideas using computers [14, 15].

2.2 Embodied Learning through Physical Play in Children's CS Education

Embodied learning asserts that physicality and cognition are inherently connected [4, 25], and that incorporating physicality and social action in otherwise stagnant learning environments can serve to enhance learning [31]. Embodied education has been tested with positive results in a variety of empirical work [2, 17, 22], with a small but growing research body in computing disciplines. Computing studies have explored the interpretations of gestures of introductory students [29], included embodied elements into debugging and design projects [19], and incorporated embodied elements into computing pedagogy to test the impact on conceptual understanding and the creation of block-based code artifacts [11]. Physical systems have been piloted to teach the intellectually and language disabled about computing [10], to merge elementary school physical education with coding on scratch [38], and to incorporate computing artifacts with traditional sports camp [5]. Still, embodied computing education is a relatively new subdomain, and major ideas remain to be fully developed.

Physical play has long been identified as one of the main ways children develop socially and intellectually [35, 37], and has been identified as an opportunity for learning more specialized abilities as well as to promote general academic performance [36, 40]. Since 2017, researchers have developed a prototype device and interface called the Scratch Node to incorporate coding education into general outdoor physical play by capturing gesture and body movement [1, 16]. Unruly Splats takes a different approach, with pressure-sensitive tiles that can be programmed to incorporate physical gameplay, and has seen success as a commercial product [24]. However, these tools only support specific types of physical movements and play. Scratch Nodes is used for hand movements like shaking and throwing and Unruly Splats is for stepping. Many other play activities (e.g., Hopscotch and Tag) and the associated movements (e.g., jumping, running, and tagging) remain unsupported. Moreover, it is unclear how to better accommodate different types of physical play in coding tool design, impeding the creation of more play friendly tools to support the adoption of embodied CS learning for children [39]. To mitigate this gap, our work conducts a fundamental design exploration to investigate the necessary coding blocks to accommodate physical play and implementing these code blocks on the micro:bit interface.

3 DESIGN EXPLORATION FOR PHYSICAL PLAY FRIENDLY CODING BLOCKS

In order to identify coding blocks that can accommodate physical play, we conducted two design explorations: (1) In-person workshops with young children in which they created physical play activities using the micro:bit kit; (2) Analyzing the mechanisms and movements of representative physical play activities.

3.1 Initial In-Person Workshops with Young Children

Initial coding-based physical play workshops were conducted to develop broad ideas on how computer programming could be integrated with physical play. These workshops made use of the micro:bit programming kit, a low-cost computer science educational kit with 39 million users in over 60 countries [30]. Workshops were conducted with 20 children aged 7 to 11. Seven children were female and 11 were male. All children had coding experience from previous contexts (schools, public libraries, or homes), using coding kits designed for children (e.g., Scratch, Sphereo, Hour of Code). Each child participated in two four hour long in-person weekend workshops, divided into six segments (1) Meet: Participants were told about workshop activities and split into focus groups to glean their perceptions about coding. Focus groups were given a facilitator to assist project work. (2) Play: Participants grew familiar with the materials, ideated a "physical play activity", and conducted the activity. (3) Code: With the help of the facilitator, participants learned to program the micro:bits (4) Eat: A group lunch break (5) Create: Participants thought about how to incorporate the micro:bit into their physical play exercise, then did so (6) Share: Participants shared their ideas with the entire group, and in post-workshop focus groups.

Data collected included (1) demographic info, (2) audio recordings of pre-workshop focus group on perceptions of computer science and coding, (3) screen capture of coding interface and audio recordings of making process, (4) project descriptions fieldnotes and photos, and (5) audio recordings on post-workshop focus groups on creating experiences. To identify the kind of coding blocks needed for physical play, we followed thematic analysis [34] to analyze participants' making processes and reflections, with a focus on what they were and were not able to do due to the affordances and limitations of the micro:bit kit as well as what new coding blocks would be needed to better support physical play (e.g., sensing the distance between two players, different play movements).

3.2 Analyzing Representative Physical Play Activities

Our design analysis from children's workshops suggests development of custom code blocks to augment physical play, both in sensing proximity to other players and in sensing specific physical movements. We further analyzed a variety of children's games to understand what play elements may need to be captured. We based our investigation on the possibilities and limitations of the micro:bit coding kit, due to its wide distribution, low cost, and internal validity with the data collected from our initial workshops.

In total, we collected eleven different children's representative physical games: Hide and Seek, Tag, Red Light Green Light, Bocce/Cornhole/Horseshoes, Duck Duck Goose, Musical Chairs, Hopscotch, Capture the Flag, Dodgeball, Four Square, and Red Rover. We then analyzed each game (e.g., Hide and Seek) by dividing them up into component portions of gameplay (e.g., counting down, finding hider, changing seeker). Then we considered how to either (1) directly implement that component of gameplay with a code block (e.g., counting down), and/or (2) supplement the component of gameplay with a complimentary code block (e.g., getting near to hider). Next—using micro:bit as the imagined coding platform—we translated those play elements into generalizable situations (e.g., countdown clock, being near another player, touching another player), the input and output sensors needed to read and communicate these play elements (LED display, button, speaker, radio sensor), the variety of code block needed (e.g., Event, Boolean), and potential challenges of implementation (e.g., Could more than two devices close together cause confusion?). It is important to mention that the goal of this ideation was not to create code blocks for specific games, but to create code blocks for general physical play by analyzing the play elements of enough different widely played games. Thus, we aimed to balance our competing goals of recommending only coding blocks with a general use scenario—that is, coding blocks not specific to one or two games—and recommending a kit of coding blocks universally enriching to physical play scenarios.

3.3 Analysis Results: Coding Blocks for Physical Play

Combining the analysis findings from the coding workshops and physical games, we recommend four categories of coding blocks for children's coding kits to accommodate physical play: motion sensing blocks, sound sensing blocks, proximity sensing blocks, and gameplay information blocks. Situating the four categories in the design of micro:bit, we further came up with 18 pieces of coding blocks based on the sensing capabilities of the coding kit as we aim to develop a micro:bit coding interface extension specifically for physical play. Note that the 18 pieces are by no means comprehensive. As new sensors are developed with and incorporated into coding kits, new coding blocks under these four categories will emerge.

- *Motion Sensing Blocks*: Boolean motion sensing blocks, focusing on detecting specific body movements in physical play, are the most straightforward design recommendation we elicited from our analysis. Many playground games have distinctive physical states which can be differentiated in gameplay (e.g., Red Light Green Light, Duck Duck Goose), and open physical play could also benefit from motion sensing blocks. We recommend the following blocks for physical play: (1) *Is standing,* (2) *Is sitting,* (3) *Is moving,* (4) *Is walking*, (5) *Is running*, (6) *Is jumping*, with each block returning true if the corresponding physical movement is sensed.
- *Sound Sensing Blocks*: We determined many games and play situations would benefit from Boolean blocks that could sense sound levels from players (e.g., if a lead player is calling out game commands to the other players). We recommend three blocks, corresponding to three different sound levels that can be sensitively captured by micro:bit's sound sensors: (1) *quiet sound,* (2) *medium sound,* (3) *loud sound.*
- *Proximity Sensing Blocks:* For many games, sensing the distance between two players provides another opportunity for coding to mediate play. Boolean blocks would be helpful to sense if two players are near each other. Using the micro:bit's radio sensing function we recommend two blocks to return true (1) if *players are close to each other,* and (2) if *players are touching each other.* For a coding kit with more robust proximity sensors (e.g., GPS), a suite of blocks to sense the exact location and distance between players is also recommended.
- *Gameplay Information Blocks*: A coding kit could be used as a helpful information storage and communication system during gameplay, particularly as some games are less physical (e.g., Bocce, Cornhole) and would not find as much use for movement sensor blocks. Gameplay information blocks should both store relevant gameplay information while also communicating that information to the player with visual, acoustic, or haptic effects. We recommend the following blocks for a coding kit: (1) A *countdown clock* Event block to take a int as a parameter, score modification Event blocks to (2) *set a win score*, (3) *increase score*, (4) *decrease score*, and (5) *set score to zero*, (6) a Variable block such that the player can *access the current score* in their code, and (7) a Boolean block to *check for a win* based on if the current score is the win score or greater.

4 CASE STUDY: IMPLEMENTING PHYSICAL PLAY CODE BLOCKS ON MIRCO:BIT

We implemented a programming extension to the Makecode micro:bit coding interface that includes some of the above identified coding blocks for physical play. We chose micro:bit as the platform for our physical play code blocks, due to its wide distribution, low cost, and physical play friendly design, such as its light weight and motion sensing features. However, our coding blocks were developed with universality in mind, and thus can be a reference point for the development of comparable physical play coding blocks in other platforms or contexts.

4.1 Development of Physical Play Friendly Programming Interface Extension on Micro:bit

We implemented the four coding categories above with identified coding blocks (Figure 1): (1) Boolean blocks to capture physical movements using the micro:bit's internal three-dimensional accelerometer, (2) Boolean blocks to capture sound using the micro:bit's internal sound sensor, (3) Boolean blocks which capture proximity of micro:bit's to each other making use of the micro:bit's radio sensor, and (4) Event blocks which regulate gameplay information (e.g., score, victory) and display that game information to the user with the micro:bit's speaker and LED display, along with a mutable Variable block which keeps track of game score for programmer use and a Boolean block which checks for a win condition. Below are brief descriptions of their technical implementation.

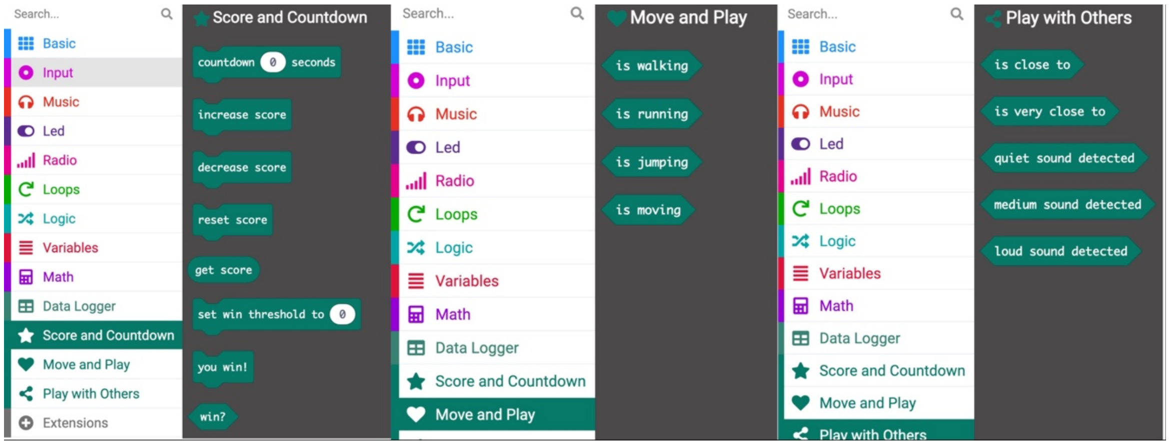


Figure 1: Custom Buttons Developed for the Physical Play Extension to the Micro:bit Makecode Interface

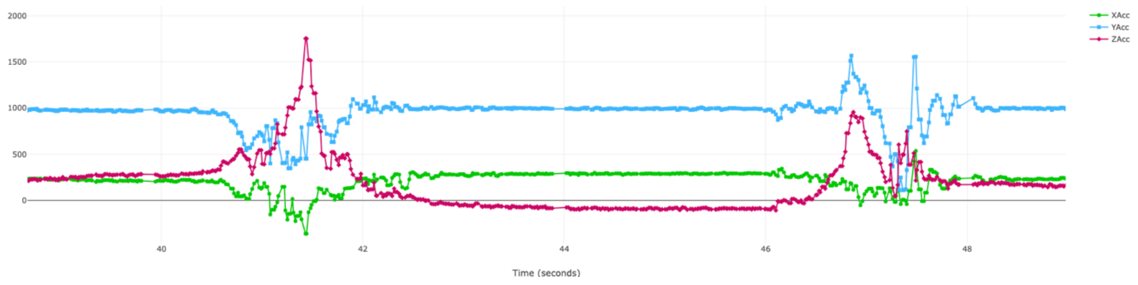
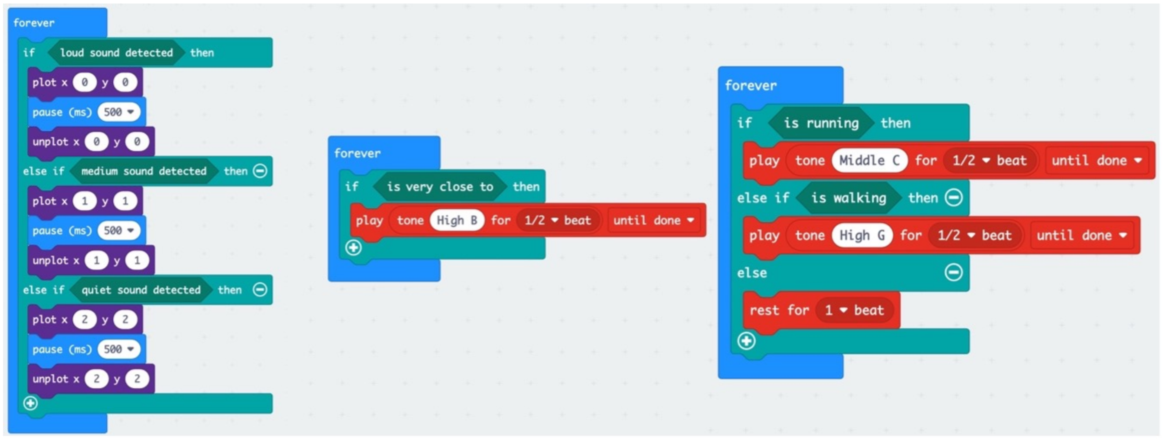


Figure 2: Acceleration Data Gathered on Micro:bit Corresponding to Sitting Down, Pausing, and Standing Up Again

Physical Movement Blocks (*Right Column,* 1*):* As mentioned above, we aimed to capture patterns of movements that corresponded to six actions common in physical gameplay (standing up, sitting down, walking, running, jumping, moving). This was done by tracking change in acceleration in three directions within a time interval, and setting threshold values on one or more of these changes which would ideally correspond to common physical movements. Custom Boolean blocks were then developed to return true if certain physical movements were sensed, but to otherwise return false. Code blocks were design to minimize false negatives (physical movement does not trigger code block), and false positives (code block is triggered without the identified physical movement). Of great value in early playtesting was the micro:bit extension Datalogger, which allowed us to capture sensor data gathered by the micro:bit and visually display it after computer download (Figure 2). We found it to be important that micro:bits maintain similar orientation and that changes in orientation would lead to dramatically different accelerometer readings even though the player may not be moving. It was determined that the most non-intrusive way to attach a micro:bit to a player such that it could get consistent acceleration data would be to use a waistbelt with a pouch for the micro:bit and battery.

While we had success with other blocks, we were unable to consistently capture the actions of standing up or sitting down on the micro:bit. The time it takes to stand up or sit down (usually .5-1 second) is too long for an accelerometer to continuously scan for a unique movement signature, with smaller sub-patterns too gentle to prevent false positives during other physical movements. Unique movement signatures were difficult to identify even for an individual tester, due to the degree of natural variation in the speed and orientation of standing up or sitting down, with small differences in orientation and movement at the beginning and end of these movements making a large difference in overall accelerometer readings. Regardless, we decided to include a block ("is standing") for participant testing.

Proximity Blocks (*Center Column,* 1): As micro:bit does not include GPS that can detect location, we turn to its radio function to detect the proximity between two players. Micro:bit can project radio signals at 7 different strengths, which can be sensed by other micro:bits. For collaborative aspects of micro:bit gameplay, we developed custom blocks which made use of the micro:bits radio sensors to send pings between micro:bit's. The only two radio strengths with any practical utility for gameplay were the lowest (1) and second lowest (2), which we qualitatively renamed to "is close to" and "is very close to" for our Boolean blocks. Micro:bit's set to the lowest setting first sensed other devices at around 1 meter distant, and consistently sensed other devices at about .3 meters distant, while the second weakest setting first sensed other devices



```
forever
  if loud sound detected then
    plot x 0 y 0
    pause (ms) 500
    unplot x 0 y 0
  else if medium sound detected then
    plot x 1 y 1
    pause (ms) 500
    unplot x 1 y 1
  else if quiet sound detected then
    plot x 2 y 2
    pause (ms) 500
    unplot x 2 y 2
```

```
forever
  if is very close to then
    play tone High B for 1/2 beat until done
```

```
forever
  if is running then
    play tone Middle C for 1/2 beat until done
  else if is walking then
    play tone High G for 1/2 beat until done
  else
    rest for 1 beat
```

Figure 3: A participant's code with sound (left), proximity (center), and physical (right) custom code blocks on Makecode

at distances greater than 7 meters (our largest measurements), and consistently sensed other devices between 7 and 5 meters distant. This high variation is expected due to the demonstrated lack of precision on other cases of multipath wave-fading [13, 20].

Sound Blocks (*Center Column,* 1 *):* The micro:bit's sound sensor was used to sense when a player may be speaking or shouting. This was completed by comparing sound sensor readings to a threshold value. If the sound sensor exceeded this threshold, the custom Boolean micro:bit block would return true rather than false. We determined not to allow users to manually set numerical thresholds, due to the lack of clear intuition of how those thresholds correspond to real world action, and the additional cognitive load this would burden users with. Instead, we translated these thresholds into qualitative signifiers (i.e., quiet noise, medium noise, loud noise).

Gameplay Information Blocks (*Right Column,* 1): Finally, we created a set of buttons to record and communicate gameplay information. This includes a score variable stored in the Micro:bit which is initially set to zero, that can be retreived or modified durring the course of gameplay. The current score is displayed at all times using the micro:bit's LED display. We also created a internal win threshold variable which can be set and reset, with a *win?* Boolean block that players can use to check if that threshold is surpassed, and a *you win!* action block which plays an appropriate tune on the micro:bit's speaker. Finally, we created a countdown timer block with an input value in seconds. The micro:bit is programmed to beep during countdown, and for the last ten seconds display the appropriate numbers on the LED display, with a final countdown tune to play once completed.

4.2 Preliminary Functionality Test

We ran a basic functionality test of our custom code blocks with nine adult graduate-level design students at The Hong Kong Polytechnic University to see how our custom micro:bit code blocks held up to the inherent variation of environmental and physical data when used in different situations by different people, and to get outside opinions on the utility of our code blocks as well as recommendations for improvement. Tests took about an hour, and included two sections (1) a basic test of all custom buttons with help from the researcher, and (2) brief demographic (e.g., age, gender, experience with coding) and reflection questions ("Do you think the coding blocks you tested today would be helpful for physical play?", "How do you think they could be improved?", "Are there movement blocks we didn't make which you think could be useful?", "How do you feel about attaching the micro:bit to your body using the belt?"). Data collected included (1) incidence of false positives/negatives of different custom code blocks, (2) qualitative data from reflection questions, and (3) demographic information. Participants ranged in age from 24 to 37, seven females and two males. Three participants had no experience with coding, four identified as beginners, and two identified as advanced programmers.

Generally, the custom code blocks were accurate, except for "is standing". However, there were occasional false positives or false negatives, which would prove frustrating in highly structured or competitive game environments. Therefore, our code blocks are likely to be of more use in generative and unstructured physical play. Overall, participants felt positively about the combination of using block-based programming and physical activity, expressing enthusiasm for the "embodied" educational element of play with our code blocks, as well as the "accessible" nature of block-based programming. Participants found that one of the more valuable contributions of the buttons was being able to test code with body movements, proximity, and sound, which they felt were intuitive and engaging ways for beginners to learn programming (Figure 3). Additionally, all participants felt that the belts were comfortable and easy to move around in. One participant suggested including theming or ornamentations on belts if used with children (e.g., superheros).

It is important to note the limitations of our testing methodology, particularly the small sample size (n=9). While the group sampled (graduate level design students) was very helpful in receiving thoughtful feedback from design professionals, it is not the target audience of our micro:bit extension (children learning computer science). Thus, we do not yet have a full understanding of how that audience would respond to our physical play extension.

5 DISCUSSION AND FUTURE WORK

In this late-breaking work, we share our design exploration of block-based programming commands to accommodate physical play. Through conducting physical-play-based coding workshops with children and analyzing the mechanism of representative children's physical games, we summarize a set of coding blocks to accommodate physical play. These coding blocks for the first time systematically present these kind of physical play friendly programming commands, supplementing previous works on developing coding tools to combine physical play and coding for children [16, 24]. Although we prototyped and implemented a programming interface on micro:bit using these blocks, these blocks are not limited to micro:bit. Designers and developers of children's programming tools can adjust and include them in other children's programming platforms (e.g., the Circuit Playground kit) to support physical-play-based programming activities, or develop new coding kits tailored for physical play.

We see a future experimental study comparing our coding blocks with other programming educational methods and/or physical coding tools as an avenue to test how effective our coding blocks are at teaching conceptual knowledge and practical programming skill. Simultaneously, we see areas of design improvement for future work, some of which may require a device with hardware capacity outside of micro:bit to implement successfully. For instance, the ideal physical play device would be able to sense proximity to other devices more accurately, which would likely require GPS tracking. Additionally, capturing more complex physical movements would require more than one motion sensor on the body. However, while such a device would be more powerful than the micro:bit extension we have developed, a tradeoff would exist in cost of development and distribution, and the necessity of organically developing a teaching userbase, a common struggle for innovations across educational research [27]. For next steps, we will conduct coding workshops with young children to test our micro:bit coding interface extension for physical play, iterate on the design of relevant coding blocks and programming interface, and further study how physical play can effectively facilitate children's computational learning in K12 settings.. Finally, we intend to investigate further how a coding kit like the micro:bit could take on the role of a technological authority in ambiguous situations where kids would otherwise have to determine for themselves the course of gameplay. We note this as potentially making gameplay easier, but perhaps at the expense of children organically learning how to handle interpersonal collaboration and conflict.

ACKNOWLEDGMENTS

We thank the child participants for the coding workshops and the graduate students in the functionality test. This research was funded by Junnan Yu's Departmental Supporting Fund project (#: P0047716) in the School of Design at The Hong Kong Polytechnic University.

REFERENCES

[1] Agassi, A. *et al.* 2019. Scratch Nodes ML: A Playful System for Children to Create Gesture Recognition Classifiers. *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow Scotland Uk, May 2019), 1–6.

[2] Antle, A.N. *et al.* 2008. Playing with the sound maker: do embodied metaphors help children learn? *Proceedings of the 7th international conference on Interaction design and children* (New York, NY, USA, Jun. 2008), 178–185.

[3] Aranda, G. and Ferguson, J. 2018. Unplugged Programming: The future of teaching computational thinking? *Pedagogika.* 68, (Dec. 2018). DOI:https://doi.org/10.14712/23362189.2018.859.

[4] Barsalou, L.W. 2008. Grounded Cognition. *Annual Review of Psychology.* 59, 1 (2008), 617–645. DOI:https://doi.org/10.1146/annurev.psych.59.103006.093639.

[5] Bodon, H. *et al.* 2022. Youth Experiences with Authentically Embedded Computer Science in Sport. *Proceedings of the 21st Annual ACM Interaction Design and Children Conference* (New York, NY, USA, Jun. 2022), 504–509.

[6] Braun, D. and Huwer, J. 2022. Computational literacy in science education–A systematic review. *Frontiers in Education.* 7, (2022).

[7] Bruner, J.S. 1963. *The Process of Education.* Vintage Books.

[8] Cheryan, S. *et al.* 2013. The stereotypical computer scientist: Gendered media representations as a barrier to inclusion for women. *Sex Roles: A Journal of Research.* 69, 1–2 (2013), 58–71. DOI:https://doi.org/10.1007/s11199-013-0296-x.

[9] Dewey, J. 1938. Experience & Education. (1938), 40.

[10] Ellis, K. *et al.* 2023. "Piece it together": Insights from one year of engagement with electronics and programming for people with intellectual disabilities. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, Apr. 2023), 1–17.

[11] Fadjo, C.L. 2012. *Developing Computational Thinking Through Grounded Embodied Cognition.* Columbia University.

[12] Fincher, S.A. and Robins, A.V. eds. 2019. *The Cambridge Handbook of Computing Education Research.* Cambridge University Press.

[13] Ghassemzadeh, S.S. *et al.* 1994. On the statistics of multipath fading using a direct sequence CDMA signal at 2 GHz, in microcellular and indoor environment. *International Journal of Wireless Information Networks.* 1, 2 (Apr. 1994), 117–130. DOI:https://doi.org/10.1007/BF02106514.

[14] Guzdial, M. 2015. Learner-Centered Design of Computing Education: Research on Computing for Everyone. *Synthesis Lectures on Human-Centered Informatics.* 8, (Nov. 2015), 1–165. DOI:https://doi.org/10.2200/S00684ED1V01Y201511HCI033.

[15] Guzdial, M. and du Boulay, B. 2019. The History of Computing Education Research. *The Cambridge handbook of computing education research (2019).*

[16] Hitron, T. *et al.* 2017. Scratch Nodes: Coding Outdoor Play Experiences to enhance Social-Physical Interaction. *Proceedings of the 2017 Conference on Interaction Design and Children* (Stanford California USA, Jun. 2017), 601–607.

[17] Kosmas, P. *et al.* 2018. Implementing embodied learning in the classroom: effects on children's memory and language skills. *Educational Media International.* 56, (Nov. 2018), 59–74. DOI:https://doi.org/10.1080/09523987.2018.1547948.

[18] Lagemann, E.C. 1989. The Plural Worlds of Educational Research. *History of Education Quarterly.* 29, 2 (1989), 185. DOI:https://doi.org/10.2307/368309.

[19] Litts, B.K. *et al.* 2019. I'm Drowning in Squirrels! How Children Embody and Debug Computational Algorithms Through Designing Mixed Reality Games. *Proceedings of the 18th ACM International Conference on Interaction Design and Children* (New York, NY, USA, Jun. 2019), 267–273.

[20] Liu, B.H. *et al.* 2008. The impact of fading and shadowing on the network performance of wireless sensor networks. *International Journal of Sensor Networks.* 3, 4 (2008), 211. DOI:https://doi.org/10.1504/IJSNET.2008.019006.

[21] Lockwood, J. and Mooney, A. 2017. *Computational Thinking in Education: Where does it Fit? A systematic literary review.*

[22] Loeffler, J. *et al.* 2023. Let's do the time warp again – embodied learning of the concept of time in an applied school setting. *Interactive Learning Environments.* 31, 1 (Jan. 2023), 397–406. DOI:https://doi.org/10.1080/10494820.2020.1789669.

[23] Margulieux, L.E. *et al.* 2019. Learning Sciences for Computing Education. *The Cambridge Handbook of Computing Education Research.* S.A. Fincher and A.V. Robins, eds. Cambridge University Press. 208–230.

[24] Millner, A. *et al.* 2017. Promoting unruly programming with random blocks and physical play. *2017 IEEE Blocks and Beyond Workshop (B&B)* (Oct. 2017), 113–114.

[25] Rohrer, T. 2008. The body in space: Dimensions of embodiment. *The body in space: Dimensions of embodiment.* De Gruyter Mouton. 339–378.

[26] Sawyer, R.K. 2006. Introduction: The New Science of Learning. *The Cambridge handbook of: The learning sciences.* Cambridge University Press. 1–16.

[27] Sawyer, R.K. 2006. *The Cambridge handbook of the learning sciences.* Cambridge University Press.

[28] Skinner, B.F. 1954. The science of learning and the art of teaching. *Harvard Educational Review.* 24, (1954), 86–97.

[29] Solomon, A. *et al.* 2018. Applying a Gesture Taxonomy to Introductory Computing Concepts. *Proceedings of the 2018 ACM Conference on International Computing Education Research* (Espoo Finland, Aug. 2018), 250–257.

[30] The Micro:bit Educational Foundation: https://microbit.org/about/. Accessed: 2023-12-04.

[31] The Routledge Handbook of Embodied Cognition: https://www.routledge.com/The-Routledge-Handbook-of-Embodied-Cognition/Shapiro/p/book/9781138573970. Accessed: 2023-12-01.

[32] Thorndike, E.L. 1910. The contribution of psychology to education. *Journal of Educational Psychology.* 1, 1 (Jan. 1910), 5–12. DOI:https://doi.org/10.1037/h0070113.

[33] Unruly Splats - Coding For Kids | Active STEM Learning: https://www.unrulysplats.com/. Accessed: 2023-12-05.

[34] Using thematic analysis in psychology: Qualitative Research in Psychology: Vol 3, No 2: https://www.tandfonline.com/doi/abs/10.1191/1478088706qp063oa. Accessed: 2023-12-14.

[35] Vygotsky, L.S. 1967. Play and Its Role in the Mental Development of the Child. *Soviet Psychology*. 5, 3 (Apr. 1967), 6–18. DOI:https://doi.org/10.2753/RPO1061-040505036.

[36] Whitebread, D. *et al.* 2009. Play, cognition and self-regulation: What exactly are children learning when they learn through play? *Educational and Child Psychology*. 26, 2 (2009), 40–52.

[37] Whitebread, D. 2012. The Importance of Play: A Report on the Value of Children's Play with a Series of Policy Recommendations. Toy Industries of Europe.

[38] Worsley, M. 2022. PE++: Exploring Opportunities for Connecting Computer Science and Physical Education in Elementary School. *Proceedings of the 21st Annual ACM Interaction Design and Children Conference* (New York, NY, USA, Jun. 2022), 590–595.

[39] Yu, J. 2022. A Design Exploration of Leveraging Physical Play to Support Computational Learning for Young People - ProQuest. University of Colorado, Boulder.

[40] Zosh, J.M. *et al.* Learning through play: a review of the evidence.