Scientific Inquiry in Middle Schools by combining Computational Thinking, Wet Lab Experiments, and Liquid Handling Robots

Tamar Fuhrmann
Deeana I. Ahmed
tamarrf@gmail.com
Teachers College, Columbia University, USA

Len Erickson
Mike Wirth
Mark L. Miller
mlmiller@learningtech.org
Learningtech.org, USA

Ethan Li
Amy T. Lam
Department of Bioengineering, Stanford University, USA

Paulo Blikstein
paulob@tc.columbia.edu
Teachers College, Columbia University, USA

Ingmar H. Riedel-Kruse*
ingmar@arizona.edu
Department of Molecular and Cellular Biology, University of Arizona, USA

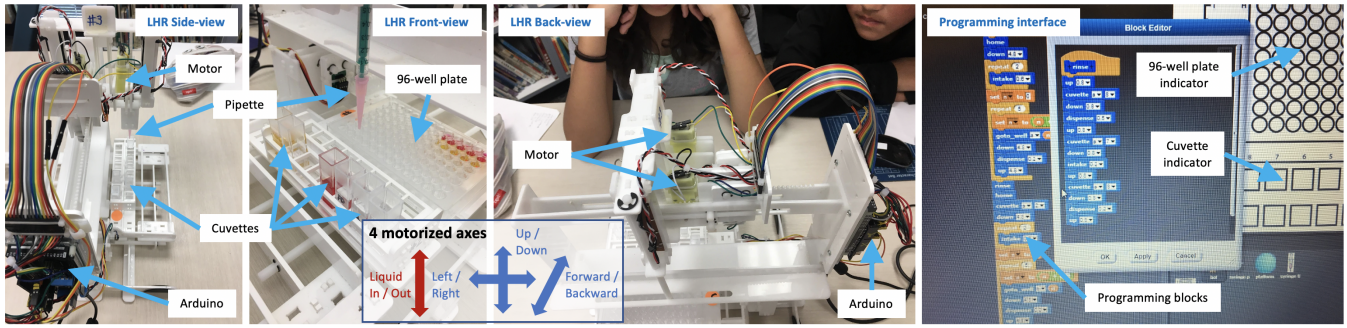

Figure 1: Liquid Handling Robot (LHR) hardware and programming environment.

ABSTRACT

Computational thinking (CT) is necessary for Science, Technology, Engineering, and Mathematics (STEM) literacy, but it can be difficult for many students to develop and it is challenging to integrate into science curricula. Here, we present a five-session curriculum where sixth-grade students programmed a Liquid Handling Robot (LHR) to conduct a science experiment while engaging in CT. We used a mixed-methods approach to assess how the curricular integration of robotics and science experimentation advances students' CT skills and perceptions of computation in science. We identified growth in CT skills, specifically regarding Algorithmic Thinking. Students identified as key advantages of this approach the increased precision in experimental procedures, time-efficiency, and easier debugging. This course provides a proof of concept curriculum on how the implications for teaching and learning of CT can be assessed, and how CT and robotics can be brought to science classrooms, especially for chemistry and biology.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics.

KEYWORDS

Computational thinking, computational literacy, hands-on experimentation, science labs, robotics

ACM Reference Format:
Tamar Fuhrmann, Deeana I. Ahmed, Len Erickson, Mike Wirth, Mark L. Miller, Ethan Li, Amy T. Lam, Paulo Blikstein, and Ingmar H. Riedel-Kruse. 2021. Scientific Inquiry in Middle Schools by combining Computational Thinking, Wet Lab Experiments, and Liquid Handling Robots. In *Interaction Design and Children (IDC '21), June 24–30, 2021, Athens, Greece.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3459990.3465180

1 INTRODUCTION

Computer science education has been implemented in K-12 education both as a stand-alone discipline and as a way to incorporate coding into existing disciplines such as science or mathematics

*Corresponding authors: tamarrf@gmail.com, paulob@tc.columbia.edu, ingmar@arizona.edu

IDC '21, June 24–30, 2021, Athens, Greece

ACM ISBN 978-1-4503-8452-0/21/06.
https://doi.org/10.1145/3459990.3465180

[2, 20, 23, 33] based on the argument that computational thinking (CT) is a fundamental skill relevant for all students and to many subject areas [39, 40]. Bringing CT approaches and skills into the classroom can deepen the learning of mathematics and science content, especially when applied in integrated curricula [4, 7, 12, 28, 29, 37]. Reciprocal enrichment of computation and science learning is powerful because science curricula can provide meaningful contexts within which CT can be applied [13, 17, 36]. CT concepts can be mapped onto a variety of conventional school subjects. For example, decomposition can be used in language art or as abstraction in a science classroom [3]. The integration of CT into science curricula in K-12 has not been widely adopted, and the deconstruction of CT's boundaries with other fields is in need of greater attention [2].

Students who engage in hands-on science display a more positive perception of science learning and a deeper understanding of scientific concepts when compared to ones engaging in traditional textbook science [9]. Problem-based active learning positively impact students' academic performance and attitudes towards their science courses [1]. Several frameworks for the integration of CT into K-12 science curricula have been proposed; however, these are often digital in nature and lack the hands-on or lab component of traditional science education [36].

Here we present a pilot study where CT was integrated with science education through a week-long curriculum designed for sixth-grade students. Students had to program a liquid handling robot (LHR) to perform complex, multi-step chemical experiments using a block-based programming language [25]. These experiments are motivated by (1) the current lack of automation and programming in 'wet science' education (biology, chemistry), especially when compared to robotics or physics [10, 38]; (2) increasing automation in the professional life-sciences, e.g., high-throughput pipetting robots [19]; and (3) the recent advancement of accessible, interactive biology technologies [11, 18, 22, 24, 27], biological cloud labs [14, 15, 35], biotic games [6, 21, 30] and low cost LHRs [8, 10, 16]. Our main questions are: Which of students' CT skills does this approach improve, and what are students' perceptions of the affordances of this approach?

2 METHODS

Participants and setting: Eleven 6th grade students (7 male, 4 female; 9 Asian, 2 ethnicity not stated) were recruited as part of an elective class in a public school setting in the San Francisco Bay Area (USA). Few students (2 or 3) had any coding experience; the others never had coded before, nor heard about the concepts of pH, robotics, or programming a robot. Students worked in groups of two (one group had three students), and each group used their own LHR.

The Liquid Handling Robot (LHR): Liquid handling robots (LHRs) are common in research labs [19], which allow for fast, high-throughput and reproducible results, therefore enabling scientists to investigate thousands of combinations of different reactions or conditions. Here we used an LHR that was intended to convey these features to middle and high-school students (Fig.1). The technical details of this LHR utilized will be published elsewhere [8], has equivalent functionality to our previously published Lego robot [10], and can perform equivalent biology, physics and chemistry experiments. Briefly, the LHR holds a pipette to load and dispense liquids (Fig.1). It is has four motors to drive the pipette plunger in and out and to move the pipette along three dimensions. Standard plasticware such as multi-well plates and cuvettes are held by the robot. The hardware is made from laser-cut acrylic, standard electronic parts, and an Arduino. The LHR can be programmed through a Snap4Arduino environment.

Lesson Plan and Study Design: We developed a five-session curriculum (50 minutes per session) to engage students in CT in the science classroom through a project-based and inquiry-driven environment while employing practices aligned with the Next Generation of Science Standards (NGSS) [5]. The LHR aimed to position students as scientists that could directly experience how technological advances can accelerate scientific research, furthermore to introduce the idea of encoding the procedures of a science experiment through a computer program, thereby integrating computing and scientific work. The content specific learning goals were to develop students' understanding of pH (relevant for chemistry and biology) and to teach students how to program a robot to execute a scientific experiment.

Session 1 - Introduction and Automation: After a short project outline, students were introduced to the concept of automation. Students gave examples from everyday life where they experienced automation, and then they discussed its advantages and limitations. We then introduced the LHR by explaining that commercial and academic labs use tools like the LHR to accelerate medical research. Then students conducted a manual experiment (Fig.2A,B): They were tasked to use a syringe to fill a 96-well plate with exactly two drops of water per well and to time themselves. Our aim was to guide students to reflect upon the advantages of automation in science and the limitations of manual work.

Session 2 - Learning about Computer Instructions: In order to understand algorithms and how to create them, students were instructed to create step by step instructions to make a peanut butter and jelly sandwich (Fig.2C). This assignment exposed students to the importance of sequential thinking and precision in task specification in programming. Following these student instructions literally, researchers attempted to make sandwiches which was not successful at first. For example, students would write instructions to put the peanut butter on the bread, which when taken literally, resulted in placing the jar of peanut butter on top of the bread. Through this experience, students learned that instructions need to be specific and broken out into ordered sub-steps. Students were also able to identify flaws in their instructions that could then be corrected, introducing them to the idea of iterative design and debugging.

Session 3 - Programming the LHR: This session shifted from exclusive CT instruction to the introduction of a scientific experiment (Fig.2D-I). Students were asked about processes for maintaining a swimming pool's cleanliness, and then taught about the function of chlorine, the concepts of pH, alkalinity, acidity, and the use of phenol red as an indicator to measure pH (Fig.2F). Next, students were introduced to writing block code using Snap4arduino in order to program the LHR (Fig.1,Fig.2G,H) to rinse the syringe with water

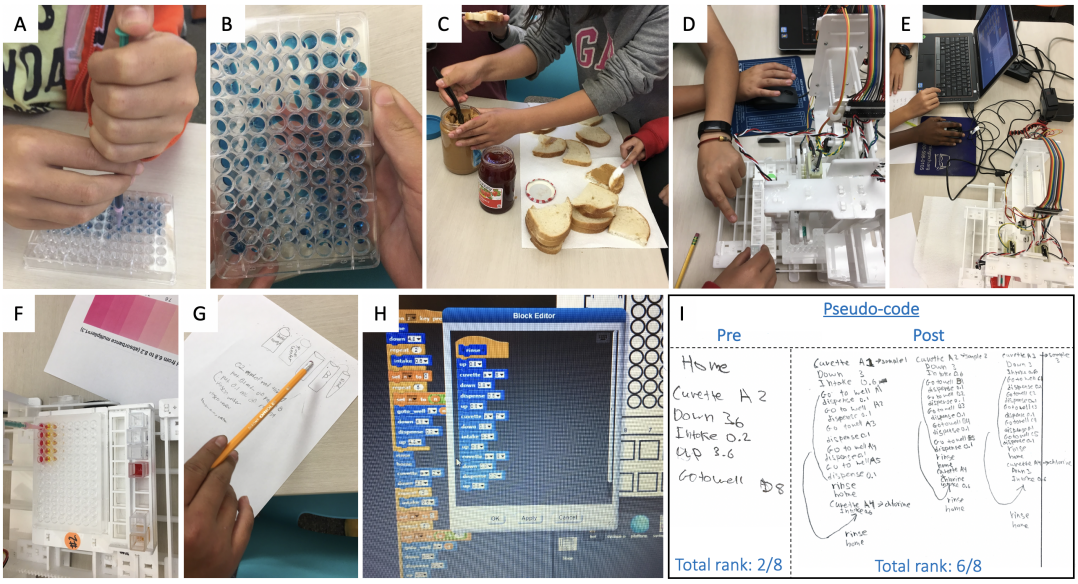

Figure 2: Illustration of student tasks: A,B) Manually filling a 96-well plate and timing this activirty. C) Procedural preparation of peanut-and-jelly sandwiches. D,E) Operating robot and programming interface. F) Measuring pH with phenol red and color chart. G) Developing pseudocode. H) Programming. I) Pre-and post-test pseudocode example (session 3 vs. 5).

Table 1: Rubric to evaluate students' pseudocode. Score 0, 1, 2 equals number of criteria achieved per CT skill (except for Algorithmic Thinking: 1 equals correct sequence was completed, 2 was given for additionally words like "chlorine indicator" or "red stuff").

CT Skills	Criteria
Abstraction: The skill to decide what information about an entity/object to keep and what to ignore [39].	Keywords: "intake", "go to", "down", "up", "dispense" (at least 3); Representation: "x=1", "A1 - sample1"
Decomposition: The skill to break a complex problem into smaller parts that are easier to understand and solve [4, 39].	Defines action/event; Uses definition
Algorithmic Thinking: The skill to devise a step-by-step set of operations/actions of how to go about solving a problem [32].	Correct sequence: Go to cuvette x; Intake from cuvette; Go to well y; Dispense into well; Rinse (optional); Sequence is shown for cuvettes 1,2,3 AND mention values e.g. 0.6mL
Generalization: The skill to formulate a solution in generic terms so that it can be applied to different problems [32].	Groups a chunk of code and indicates with arrow/"repeat"; States "repeat x amount of times"

Table 2: Results for 11 students on evaluating their pre- and post-test pseudocode (compare to Table 1 regarding the four CT skills). 1-tailed t-test on whether average increased from pre- to post-test: ** p<0.01; * p<0.05; (*) p=0.057; all others n.s..

	Pre-test												**Post-test**											
Student	1	2	3	4	5	6	7	8	9	10	11	Avg	1	2	3	4	5	6	7	8	9	10	11	Avg	Diff
Abstr	1	2	2	1	2	2	1	1	1	1	0	1.3	0	2	2	2	1	1	2	1	0	2	1	1.3	0.0
Decom	0	0	1	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	2	0	0	0	1	0	0.3	0.2
Algor	2	0	0	0	0	0	0	0	1	1	0	0.4	2	2	1	1	1	1	2	2	1	1	1	1.4	1.0**
Gener	2	1	0	1	0	0	0	0	0	2	0	0.5	2	2	0	2	0	2	1	2	0	2	0	1.2	0.7(*)
Sum	5	3	3	2	2	2	1	1	2	4	0	2.3	4	6	3	5	2	6	5	5	1	6	2	4.1	1.8*
Po-Pr													-1	3	0	3	0	4	4	4	-1	2	2	1.8*	

while moving between cuvettes. Students then executed corresponding programming and experimental tasks (Fig.2D,E). Students were then introduced to the problem that they would attempt to solve using both programming and the scientific method, i.e., to measure the pH in multiple wells by adding 0.1 mL of phenol red and where the resulting color of the liquid would then be visually compared to a color strip to determine the pH. Students were then tasked with writing the corresponding pseudocode on paper (Fig.2G,I 'Pre'), which then served as the pre-test for later analysis.

Sessions 4 and 5 - Automating an Experimenting with Code: Students were tasked with translating their pseudocode to real code to program the LHR and run corresponding experiments (Fig.2D,E). Students were given two cuvettes with unknown pool water and a cuvette with phenol red. Students wrote code so that the LHR would add 0.1 mL of phenol red to each well to test the pool water's pH (Fig.1, 2F). Students then executed and iteratively debugged their code. All groups were eventually successful in having the LHR execute this task. Students then also provided their final (working) pseudo-code after all experiments, which served as the post-test for later analysis (Fig.2I 'Post').

3 DATA SOURCE, PROCESSING, AND ANALYSIS

We employed a mixed-methods approach to evaluate the efficacy and plausibility of integrating CT into science curricula in the sixth grade. These data included student pre- and post-pseudocode tasks after session 3 and 5 (Fig.2I), and students' post-interviews. We also video recorded the students; an in-depth analysis is beyond the present publication, but we highlight some key observations: Experiment and programming were challenging for the students, eventually they all succeeded albeit with varying level of sophistication and speed. One group did a top-down paper design, then created a rinsing block, and then asked for just water in a cuvette so that they could debug their block from the "bottom-up." The other teams were considerably less advanced in their CS understanding. These activities also stimulated students to reflect on the scientific content, e.g., they discussed using a different indicator than phenol red, "if the scale is 6.2-8.6, what color will it show if we have lemon juice, with high acidity or strong acid," and one student responded that it would be yellow (same color as week acid) and it would not show the actual pH. Student engagement was high as indicated by our observations of their focused work, their eagerness to finish tasks beyond class time, and by the teacher.

We asked students what they learned from these activities, which revealed a combination of scientific and computational concepts, for example: Student #1 stated that she learned how to test pH, what phenol red is, about base and acidity (talking about the chemistry first), and she said "I also learned to code of course" (smiling) and "learn to re-run it after you fix your mistakes, make sure that when you are actually doing it, it works." and "In science, we are doing ocean health and the teacher always talks about pH, I never understood what ... a pH is, and now I do." Student 2 said "I learned programming and a tiny bit of Chemistry like the pH stuff ... how pH system work and difference between base and acid ... Chemistry is important also cool to know the condition of the pool." Student 3 said "I learned that the computer is a lot more precise than hands on and it can go much faster than hands on. When we wanted to make the robot automated we kept failing over and over ... I learned not to get mad when you fail on something."

Table 3: The three benefits of integrating coding and science consistently mentioned by students.

Student	1	2	3	4	5	6	7	8	9	10	% mentioned
Precision	1	1	1	1	2	2	0	2	3	1	90%
Time-Saving	2	1	2	1	4	0	1	1	3	0	80%
Debugging	0	2	2	0	2	1	0	3	2	4	70%

To examine students' understanding and adoption of CT skills, we assessed students' pseudocode (Fig.2I) through a rubric based on previous work by others [32, 39]. This rubric focused on four CT skills, i.e., Abstraction, Decomposition, Algorithmic Thinking, and Generalization (Table 1). Each skill was ranked from 0 to 2, enabling a highest possible for pseudocode of 8. This rubric was iteratively refined until two researchers reached 90% agreement in assessing the pseudocode of two groups. Then the pseudocode of all groups was assessed. Table 2 then illustrates students' pseudocode (pre- and post-test) rank according to these four CT skills. A t-test reveals that students' pseudocode total rank average significantly improved, i.e., 7/11 (64%) displayed a positive change, 2 no change (18%), and 2 (18%) a decrease. Averaging over all students, Algorithmic Thinking improved most significantly, and Generalization potentially improved as well (boarderline significant). Abstraction did not improve, but we note that it already showed a much higher average in the pre-test compared to the other skills. Finally, Decomposition was low in pre-test and did not significantly improve.

In addition, 10 of the 11 students were interviewed for 20-30 minutes after the last session. Students were asked to discuss and define their perspectives on the benefits of integrating coding and science and of using a robot in order to carry out an experiment. Interviews were video recorded, transcribed, and qualitatively analyzed using NVivo 12. Answers were iteratively analyzed and grouped into categories using qualitative content analysis and a data reduction process [31]. We decided to focus on students' perception of learning CT in the science classroom and specifically, the benefits of integrating coding and science. We created a list of themes (nodes) which then revealed that students perceived three distinct advantages of integrating CT with scientific experimentation when compared to traditional manual work (Table 3): 1. Precision: the ability to execute the experiment according to specification and with little variation between repeats; 2. Time-saving: the potential for requiring less human work-time to conduct an experiment; and 3. Debugging: the opportunity to refine and repeat the experiment to improve the results.

The following excerpts from student interviews further illustrate the advantages students perceived in using the LHR to carry out their experiment: 1. Precision - Interviewer: "Why do you think we use programming in science?" - Student 1: "Programming in science, basically allowing robots to do stuff for us. And also letting computers do more theoretical stuff. Since computers and robots are more accurate and can do more automatic and non-human error tasks, and they can also make more theoretical things rather than

when humans do science it's more experimental." 2. Time-Saving - Interviewer: "And why do you think we use programming in a science class?" - Student 9: "To help make things easier so that we don't have to do it manually and take a lot of time." 3. Debugging - Interviewer: "Did you do some testing? How did you do the testing?" - Student 8: "We first, did the first round and if there's something wrong with it, we fixed it and did another round of testing and then if it was correct, we continued."

4 CONCLUSIONS

We demonstrated that students improved their understanding of CT through using the LHR to conduct a science experiment. Our specific curriculum significantly improved Algorithmic Thinking. Students connected to the science content (pH), and they were able to identify benefits of integrating coding, robotics, and scientific experimentation, i.e., precision, time-saving benefits, and debugging. The latter suggests that errors are not hiding digitally, but are explicitly shown in the physical world. The robot executes a "physical computation" that can be watched and has an effect on the real world, so students can make changes and debug the errors [34]. Before changing a curriculum, it is also important to assess students' receptiveness in order to ensure acceptance of the changes. We found that the students were able to articulate the precision and time-saving benefits of using CT through coding when conducting a science experiment (Table 3). This is possibly due to students' initially executing a similar repetitive experiment themselves manually (Fig.2A,B). Debugging was also significantly mentioned (Table 3), indicating that most of them had to debug their code in order to improve the execution of their experiment.

Our work motivates future research: (1) How do students execute these experiments and how do they go about the programming and debugging tasks? (2) What is the permanence of CT skills taught through physical science experiments and the relationship to students' receptiveness? (3) How would more complex experiments with an LHR as demonstrated previously [10] facilitate CT and science learning? (4) Why did some CT skills improve while others did not (Table 2), what caused these improvements, and how does that inform design of hardware, science experiments, and curriculum? (5) What is the optimal synergy between digitally simulating such activities vs. their physical implementations?

Overall, we see versatile value in integrating CT in science experiments, in particular it provides additional motivation for students to engage in those topics, and where robotics provide a suitable modality for physical computation. This study (together with our previous ones [10, 35]) demonstrates that there are ample, feasible, and valuable opportunities for this approach in middle- and high-school classroom. Studies with larger student sizes and more in depth observation and interviews for these activities would be desirable. A low-cost commercially produced LHR similarly to the one presented here could make such activities and curricula widely accessible, and the recent integration of Snap with Lego Mindstorm [26] provides another dissemination route based on our previous Lego LHR design [10].

ACKNOWLEDGMENTS

Supported by NSF grant #1638070 (NRI: Liquid Handling Robots - A New Paradigm for STEM Education). We thank F. Mac for project feedback. Author contributions: TF, DIA, PB user study design, execution and data analysis; MM, EL, AL, MW, IHRK: LHR engineering; MLM, PB, IHRK: project leadership. IRB: TC Columbia (D-190466) and Stanford (18334).

REFERENCES

[1] Orhan Akınoğlu and Ruhan Özkardeş Tandoğan. 2007. The effects of problem-based active learning in science education on students' academic achievement, attitude and concept learning. *Eurasia journal of mathematics, science and technology education* 3, 1 (2007), 71–81.
[2] David Barr, John Harrison, and Leslie Conery. 2011. Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology* 38, 6 (2011), 20–23.
[3] Valerie Barr and Chris Stephenson. 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 (2011), 48–54.
[4] National Research Council et al. 2011. *Successful K-12 STEM education: Identifying effective approaches in science, technology, engineering, and mathematics*. National Academies Press.
[5] National Research Council et al. 2014. Developing assessments for the next generation science standards. (2014).
[6] Rhiju Das, Benjamin Keep, Peter Washington, and Ingmar H Riedel-Kruse. 2019. Scientific discovery games for biomedical research. *Annual Review of Biomedical Data Science* 2 (2019), 253–279.
[7] Michael B Eisenberg and Doug Johnson. 2002. Learning and Teaching Information Technology–Computer Skills in Context. ERIC Digest. (2002).
[8] Tamar Fuhrmann Len Erikson Mike Wirth Mark L. Miller Paulo Blikstein Ingmar H. Riedel-Kruse Ethan Li, Amy T. Lam. 2021. DIY Liquid Handling Robots for Integrated STEM Education and Life-Science Research. *Submitted* (2021).
[9] Brian J Foley and Cameron McPhee. 2008. Students' attitudes towards science in classes using hands-on or textbook based curriculum. *American Educational Research Association* (2008), 1–12.
[10] Lukas C Gerber, Agnes Calasanz-Kaiser, Luke Hyman, Kateryna Voitiuk, Uday Patil, and Ingmar H Riedel-Kruse. 2017. Liquid-handling Lego robots and experiments for STEM education and research. *PLoS biology* 15, 3 (2017), e2001413.
[11] Lukas C Gerber, Honesty Kim, and Ingmar H Riedel-Kruse. 2016. Interactive Biotechnology: Design Rules for Integrating Biological Matter into Digital Games. *DiGRA/FDG* 13, 1 (2016), 16.
[12] Mark Guzdial. 1994. Software-realized scaffolding to facilitate programming for science learning. *Interactive learning environments* 4, 1 (1994), 001–044.
[13] Susanne Hambrusch, Christoph Hoffmann, John T Korb, Mark Haugan, and Antony L Hosking. 2009. A multidisciplinary approach towards computational thinking for science majors. *ACM SIGCSE Bulletin* 41, 1 (2009), 183–187.
[14] Zahid Hossain, Engin Bumbacher, Alison Brauneis, Monica Diaz, Andy Saltarelli, Paulo Blikstein, and Ingmar H Riedel-Kruse. 2018. Design guidelines and empirical case study for scaling authentic inquiry-based science learning via open online courses and interactive biology cloud labs. *International Journal of Artificial Intelligence in Education* 28, 4 (2018), 478–507.
[15] Zahid Hossain, Engin W Bumbacher, Alice M Chung, Honesty Kim, Casey Litton, Ashley D Walter, Sachin N Pradhan, Kemi Jona, Paulo Blikstein, and Ingmar H Riedel-Kruse. 2016. Interactive and scalable biology cloud experimentation for scientific inquiry and education. *Nature biotechnology* 34, 12 (2016), 1293–1298.
[16] Zahid Hossain, Xiaofan Jin, Engin W Bumbacher, Alice M Chung, Stephen Koo, Jordan D Shapiro, Cynthia Y Truong, Sean Choi, Nathan D Orloff, Paulo Blikstein, et al. 2015. Interactive cloud experimentation for biology: An online education case study. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*. 3681–3690.
[17] Kemi Jona, Uri Wilensky, Laura Trouille, MS Horn, Kai Orton, David Weintrop, and Elham Beheshti. 2014. Embedding computational thinking in science, technology, engineering, and math (CT-STEM). In *future directions in computer science education summit meeting, Orlando, FL*.
[18] Honesty Kim, Lukas Cyrill Gerber, Daniel Chiu, Seung Ah Lee, Nate J Cira, Sherwin Yuyang Xia, and Ingmar H Riedel-Kruse. 2016. LudusScope: accessible interactive smartphone microscopy for life-science education. *PloS one* 11, 10 (2016), e0162602.
[19] Fanwei Kong, Liang Yuan, Yuan F Zheng, and Weidong Chen. 2012. Automatic liquid handling for life science: a critical review of the current state of the art. *Journal of laboratory automation* 17, 3 (2012), 169–185.
[20] Siu-Cheung Kong. 2016. A framework of curriculum design for computational thinking development in K-12 education. *Journal of Computers in Education* 3, 4 (2016), 377–394.

[21] Amy T Lam, Jonathan Griffin, Matthew Austin Loeun, Nate J Cira, Seung Ah Lee, and Ingmar H Riedel-Kruse. 2020. Pac-Euglena: A Living Cellular Pac-Man Meets Virtual Ghosts. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

[22] Amy T Lam, Joyce Ma, Cory Barr, Seung Ah Lee, Adam K White, Kristina Yu, and Ingmar H Riedel-Kruse. 2019. First-hand, immersive full-body experiences with living cells through interactive museum exhibits. *Nature biotechnology* 37, 10 (2019), 1238–1241.

[23] Irene Lee, Shuchi Grover, Fred Martin, Sarita Pillai, and Joyce Malyn-Smith. 2020. Computational thinking from a disciplinary perspective: Integrating computational thinking in K-12 science, technology, engineering, and mathematics education. *Journal of Science Education and Technology* 29, 1 (2020), 1–8.

[24] Seung Ah Lee, Engin Bumbacher, Alice M Chung, Nate Cira, Byron Walker, Ji Young Park, Barry Starr, Paulo Blikstein, and Ingmar H Riedel-Kruse. 2015. Trap it! A playful human-biology interaction for a museum installation. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. 2593–2602.

[25] John Maloney, Mitchel Resnick, Natalie Rusk, Brian Silverman, and Evelyn Eastmond. 2010. The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)* 10, 4 (2010), 1–15.

[26] MIT. accessed 2020. *LEGO MINDSTORMS EV3*. https://scratch.mit.edu/ev3

[27] Darius G Rackus, Ingmar H Riedel-Kruse, and Nicole Pamme. 2019. "Learning on a chip:" Microfluidics for formal and informal science education. *Biomicrofluidics* 13, 4 (2019), 041501.

[28] Edward F Redish and Jack M Wilson. 1993. Student programming in the introductory physics course: MUPPET. *American Journal of Physics* 61, 3 (1993), 222–232.

[29] Alexander Repenning, David C Webb, Kyu Han Koh, Hilarie Nickerson, Susan B Miller, Catharine Brand, Ian Her Many Horses, Ashok Basawapatna, Fred Gluck, Ryan Grover, et al. 2015. Scalable game design: A strategy to bring systemic computer science education to schools through game design and simulation creation. *ACM Transactions on Computing Education (TOCE)* 15, 2 (2015), 1–31.

[30] Ingmar H Riedel-Kruse, Alice M Chung, Burak Dura, Andrea L Hamilton, and Byung C Lee. 2011. Design, engineering and utility of biotic games. *Lab on a Chip* 11, 1 (2011), 14–22.

[31] Margrit Schreier. 2012. *Qualitative content analysis in practice*. Sage publications.

[32] Cynthia Selby and John Woollard. 2014. Refining an understanding of computational thinking. (2014).

[33] Pratim Sengupta, John S Kinnebrew, Satabdi Basu, Gautam Biswas, and Douglas Clark. 2013. Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies* 18, 2 (2013), 351–380.

[34] Arnan Sipitakiat and Nusarin Nusen. 2012. Robo-Blocks: designing debugging abilities in a tangible programming system for early primary school children. In *Proceedings of the 11th International Conference on Interaction Design and Children*. 98–105.

[35] Peter Washington, Karina G Samuel-Gama, Shirish Goyal, Ashwin Ramaswami, and Ingmar H Riedel-Kruse. 2019. Interactive programming paradigm for real-time experimentation with remote living matter. *Proceedings of the National Academy of Sciences* 116, 12 (2019), 5411–5419.

[36] David Weintrop, Elham Beheshti, Michael Horn, Kai Orton, Kemi Jona, Laura Trouille, and Uri Wilensky. 2016. Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology* 25, 1 (2016), 127–147.

[37] U. Wilensky and K. Reisman. 2006. Thinking like a wolf, a sheep, or a firefly: Learning biology through constructing and testing computational theories—an embodied modeling approach. *Cognition and instruction* 24, 2 (2006), 171–209.

[38] Keeshan Williams, Irina Igel, Ronald Poveda, Vikram Kapila, and Magued Iskander. 2012. Enriching K-12 Science and Mathematics Education Using LEGOs. *Advances in Engineering Education* 3, 2 (2012), n2.

[39] Jeannette Wing. 2011. Research notebook: Computational thinking—What and why. *The link magazine* 6 (2011).

[40] Jeannette M Wing. 2006. Computational thinking. *Commun. ACM* 49, 3 (2006), 33–35.