Journal of Science Education and Technology (2020) 29:83–100
https://doi.org/10.1007/s10956-019-09804-9

C2STEM: a System for Synergistic Learning of Physics and Computational Thinking

Nicole M. Hutchins[1] · Gautam Biswas[1] · Miklós Maróti[1] · Ákos Lédeczi[1] · Shuchi Grover[2] · Rachel Wolf[3] · Kristen Pilner Blair[3] · Doris Chin[3] · Luke Conlin[4] · Satabdi Basu[5] · Kevin McElhaney[5]

Published online: 3 December 2019

Abstract
Synergistic learning combining computational thinking (CT) and STEM has proven to be an effective method for advancing learning and understanding in a number of STEM domains and simultaneously helping students develop important CT concepts and practices. We adopt a design-based approach to develop, evaluate, and refine our Collaborative, Computational STEM (C2STEM) learning environment. The system adopts a novel paradigm that combines visual model building with a domain-specific modeling language (DSML) to scaffold learning of high school physics using a computational modeling approach. In this paper, we discuss the design principles that guided the development of our open-ended learning environment (OELE) using a learning-by-modeling and evidence-centered approach for curriculum and assessment design. Students learn by building models that describe the motion of objects, and their learning is supported by scaffolded tasks and embedded formative assessments that introduce them to physics and CT concepts. We have also developed preparation for future learning (PFL) assessments to study students' abilities to generalize and apply CT and science concepts and practices across problem solving tasks and domains. We use mixed quantitative and qualitative analysis methods to analyze student learning during a semester-long study run in a high school physics classroom. We document some of the lessons learned from this study and discuss directions for future work.

Keywords STEM+CT · Synergistic learning · Learning-by-modeling · Computational thinking · Evidence-centered design · Open-ended learning environment

Introduction

Computation is now considered to be the third pillar of science and engineering disciplines, alongside theory and experimentation (Wing 2016). Computing knowledge and skills provide the foundation for modern competency in STEM (Science, Technology, Engineering, and Math)-related fields, prompting research on how to best prepare students for the 21st century workforce and lifelong learning (Sengupta et al. 2013; Weintrop et al. 2016). Exploiting the synergies between science and computation can bring about a fundamental change in the way that science learning occurs. In addition, educators, researchers, and industry stakeholders now recognize that students need to learn computational thinking (CT) to become creators, and not just consumers of the next wave of computing innovations (Schnabel 2011; Wing 2006). This provides us with a unique and timely opportunity to develop computer-based learning environments that leverage the synergies between STEM and computing education and bring a *learning-by-modeling* and problem-solving approach to support learning with understanding that is active and engaging.

Conceptual learning of STEM and CT domains, individually and in integrated settings, is often difficult for learners (Román-González et al. 2017). While the introduction to foundational programming constructs in high school classrooms can be challenging, even in easy-to-use, visual programming environments (Grover and Basu 2017), a subsequent integration of these constructs into introductory STEM courses may further exacerbate difficulties that students have when working in coupled domains (Basu et al. 2016; Chi 2005). When building

✉ Nicole M. Hutchins
nicole.m.hutchins@vanderbilt.edu

[1] Vanderbilt University, Nashville, TN, USA
[2] Looking Glass Ventures, Palo Alto, CA, USA
[3] Stanford University, Stanford, CA, USA
[4] Salem State University, Salem, MA, USA
[5] SRI International, Menlo Park, CA, USA

computational models, students find the behaviors of individual objects intuitive but struggle to understand their relations with emergent phenomena, i.e., the aggregate behavior generated by a group of objects (Chi 2005; Wilensky and Resnick 1999). For example, students find it hard to extrapolate elastic collisions of gas molecules at the micro-level to macro-level properties of pressure and temperature. In biology, students understand how animals survive, grow, and reproduce in an environment, but find it hard to go from individual behaviors to explain systems level phenomena, such as evolution, natural selection, and population dynamics. Students also have difficulties in identifying relevant objects and their interactions, and in extending their intuitive understanding of domain concepts into a computational modeling framework (Basu et al. 2016).

On the other hand, Harel and Papert (1990) have argued that programming is reflexive with other domains, i.e., learning programming in concert with concepts from another domain can be easier than learning each separately. Others have built on this argument to show that STEM + CT have similar epistemic origins (Sengupta et al. 2013; Weintrop et al. 2016) and that programming and computational modeling serve as effective vehicles for learning challenging science and math concepts (diSessa 2001; Hambrusch et al. 2009; Jona et al. 2014; Repenning et al. 2010; Sengupta et al. 2015). Researchers contend that CT could be better inculcated if taught in the context of or in connection with other domains (Cooper and Cunningham 2010).

In our work, we apply a *learning-by-modeling* paradigm in science classrooms. In particular, we adopt a design-based research (DBR) approach (Barab and Squire 2004) to develop Collaborative, Computational STEM (C2STEM)—an open-ended learning environment (OELE) (Biswas et al. 2016) for integrated learning of STEM + CT in high school physics classes. We adopt an evidence-centered design (ECD) approach (Mislevy et al. 2017) to meet required curricular standards for the target domain (physics). While the explicit construction of computational models is central to helping students develop STEM + CT concepts and practices (Weintrop et al. 2016), we use a *step-by-step* modeling approach (in contrast to equation-based modeling) to help students better understand system dynamics (Redish and Wilson 1993; Sengupta et al. 2013). In addition, we leverage the affordances of visual programming languages along with domain-specific modeling languages (DSMLs) to facilitate and enhance synergistic STEM + CT learning. ECD-based formative assessments provide additional support for learning, and preparation for future learning (PFL) assessments (Bransford and Schwartz 1999) are designed to study how students generalize and apply problem-solving strategies in new situations.

This paper uses a mixed-methods approach to answer the following research questions:

1. How does a principled design approach to our *learning-by-modeling* framework implemented in the C2STEM environment lead to learning of both STEM and CT concepts? and
2. How does a qualitative analysis of students' model-building behaviors combined with embedded formative assessments and summative PFL assessments provide evidence of learning in physics and CT?

The remainder of this paper is organized as follows. Section "Synergistic Learning of STEM and CT" provides background on the role of CT and computational modeling in STEM learning. Section "Design Principles and the C2STEM System" discusses our design principles, process, and implementation of the C2STEM environment. Sections "Classroom Study" and "Results and Discussion" present initial results and case studies from a study with C2STEM in a high school physics classroom. Section "Conclusions and Future Work" discusses lessons learned from our analyses and future extensions to C2STEM.

Synergistic Learning of STEM and CT

In this section, we briefly review recent work on introducing computational approaches into STEM curricula, and then present a learning by modeling approach to learning STEM + CT.

STEM + CT in K–12 Classrooms

Wing (2006) spurred researchers, educators, and policymakers to introduce CT and CS as "a universally applicable attitude and skill set" oriented toward solving problems and designing solutions using computational methods (p. 33). These skills include logical and algorithmic thinking, abstraction, problem decomposition, pattern recognition and generalization, and debugging (error detection and resolution) (Grover and Pea 2018). The 2012 Science Framework (National Research Council, 2012) also acknowledged the multiple connections among domains—"more and more frequently, scientists work in interdisciplinary teams that blur traditional boundaries" (p. 31)—and "consider connections among science, technology, engineering, and mathematics" (p. 32). Mandates for an education that prepares learners for life and work—and specifically STEM + CT work—in the 21st century reflect this integrated STEM perspective (NGSS Lead States 2013).

Computing and STEM share a deeply symbiotic relationship (Grover and Pea 2018). There is evidence that specific aspects of science learning accrue benefits from integration with computing. For example, studying a phenomenon as a *step-by-step* discrete time process and a sequence of events is

easier for students to comprehend than continuous dynamics representations (diSessa 2001; Sengupta et al. 2013; Sherin 2001). In addition, having to specify how to *decompose* a model (e.g., separating upward and downward motions of an object) and making decisions—such as, "Should the velocity of an object stop at 0, or should It keep changing and go negative?"—make assumptions more explicit and student conceptions more visible. Last, visualizations through animation and plots afforded by simulating computational models help learners judge the legitimacy of model constructs (Sherin 2001).

Computational modeling environments, such as CTSiM (Basu et al. 2017), ViMap (Sengupta et al. 2015), and CT-STEM (Jona et al. 2014), support synergistic learning of domain and CT concepts. These environments extend NetLogo, a multi-agent programming language and authoring environment for building and simulating models of complex natural and social phenomena (Wilensky and Resnick 1999). CTSiM, ViMap, and CT-STEM provide a block-structured visual programming environment as an abstraction layer over NetLogo allowing students to focus on their modeling tasks, without being overwhelmed by the NetLogo programming language syntax. These systems have been successful in classroom studies (e.g., Basu et al. 2017; Weintrop et al. 2016).

Learning-by-Modeling as a Framework for Synergistic Learning

The leveraging of these key STEM + CT integration benefits has been actualized through the use of a *learning-by-modeling* pedagogical approach (e.g., Hambrusch et al. 2009). Integrating CT and scientific modeling can be synergistic, i.e., supportive of each other along multiple dimensions: (1) reorganizing science concepts around intuitive computational representations that introduce discrete and qualitative forms of fundamental laws, which are simpler to understand than equation-based continuous forms, lowers the learning threshold for these concepts (Redish and Wilson 1993); (2) computational modeling can represent core scientific practices, such as modeling, verification, and explanation (Soloway 1993); and (3) contextualized computational constructs make it easier to learn programming (Papert 1991). These benefits reflect the framing of proficiency in both science and CT (as defined by the Next Generation Science Standards [NGSS] and K–12 Computer Science Framework [2016], respectively), integrating knowledge and practice.

Learning-by-modeling in science can be looked upon as a constructive model-building paradigm (Mayer 1999), where students organize and convert their knowledge of science concepts into computational structures that can be executed to generate model behaviors. Additional features, such as *step-by-step* execution linked to animations of model behavior and plots of variable values as a function of time, provide scaffolds for interpreting and understanding the modeled phenomena. Therefore, *learning-by-modeling* environments support exploration by enabling learners to simulate and "play" with their models and study their behavior under various conditions.

Design Principles and the C2STEM System

This section outlines the design principles that govern the architecture and the implementation of the C2STEM system. This section also provides details of the curriculum modules and the embedded and PFL assessments we developed for the kinematics of motion domain.

Design Principles

Our design principles are directed toward developing an OELE that provides a low-entry threshold for the synergistic *learning-by-modeling* approach to developing STEM + CT knowledge. We outline a number of these design principles below.

Evidence-Centered Design

We use ECD (Mislevy et al. 2017) as an overarching framework for systematically integrating the STEM + CT disciplines and aligning the design of curricular activities and assessment tasks. ECD promotes design coherence by explicitly linking claims about student learning, evidence from student work products, and design features of instructional and assessment tasks that elicit the desired evidence. Our approach extends one used by Harris et al. (2016) to develop science assessments that integrate content knowledge with science practices along NGSS dimensions. We apply backward design methods (Wiggins and McTighe 2005) to develop instructional materials, where the design of learning experiences focuses on the achievement of learning outcomes. Figure 1 illustrates the process used to align the curriculum sequence design and embedded assessment tasks that integrate physics and CT. Like most design activities, the process is iterative; early prototypes of curricular activities and assessment tasks informed the refinement of the integrated learning goals. Beginning with the target domain, i.e., computational modeling of kinematics phenomena for high school, we conducted a domain analysis, which entailed a detailed unpacking of the physics disciplinary concepts, CT concepts, and computational modeling practices. Unpacking involves elaborating key concepts and aspects of practice, defining grade-band appropriate expectations for proficiency and learning boundaries, identifying the prerequisite knowledge and skills required to achieve proficiency, and articulating

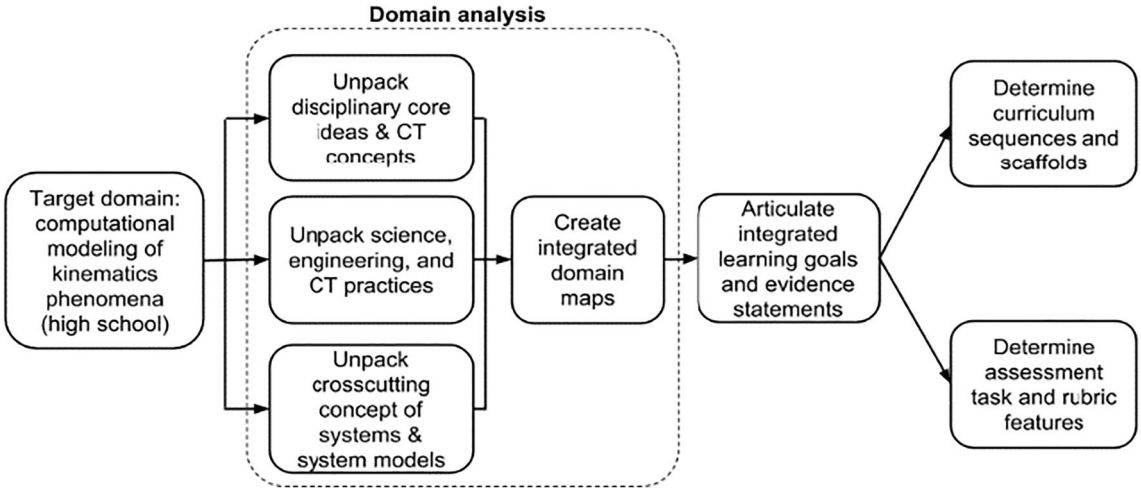


Fig. 1 Design process schematic for curriculum and assessment tasks that integrate science and CT learning

the evidence required to demonstrate the proficiencies associated with the practice.

Based on the mapping, we articulated a series of integrated learning goals that combine aspects of physics, CT, and computational modeling into assessable performance statements. Figure 2 shows an example of how learning goals arise from the unpacking elements. The integrated learning goals constitute the claims we make about students' synergistic learning of physics and CT by engaging in computational modeling. For each learning goal, we also articulated an evidence statement describing observable features of student performance that provide evidence of proficiency with the learning goal.

The integrated learning goals and associated evidence statements provide the basis for the design of the curriculum unit and its embedded assessments, ensuring alignment between the two. The integrated learning goals serve as anchors for a coherent sequence of instructional activities that integrate physics and CT and help ensure that each activity focuses on a small number of specific integrated learning goals. Additionally, the learning goals and evidence statements constitute the basis for assessment tasks that elicit the desired evidence, and scoring rubrics that guide teachers and researchers to attend to this evidence in students' responses.

Domain-Specific Modeling Language

Previous work has outlined a number of challenges that arise in implementing an integrative, *learning-by-modeling* curricula (Basu et al. 2016; Grover and Basu

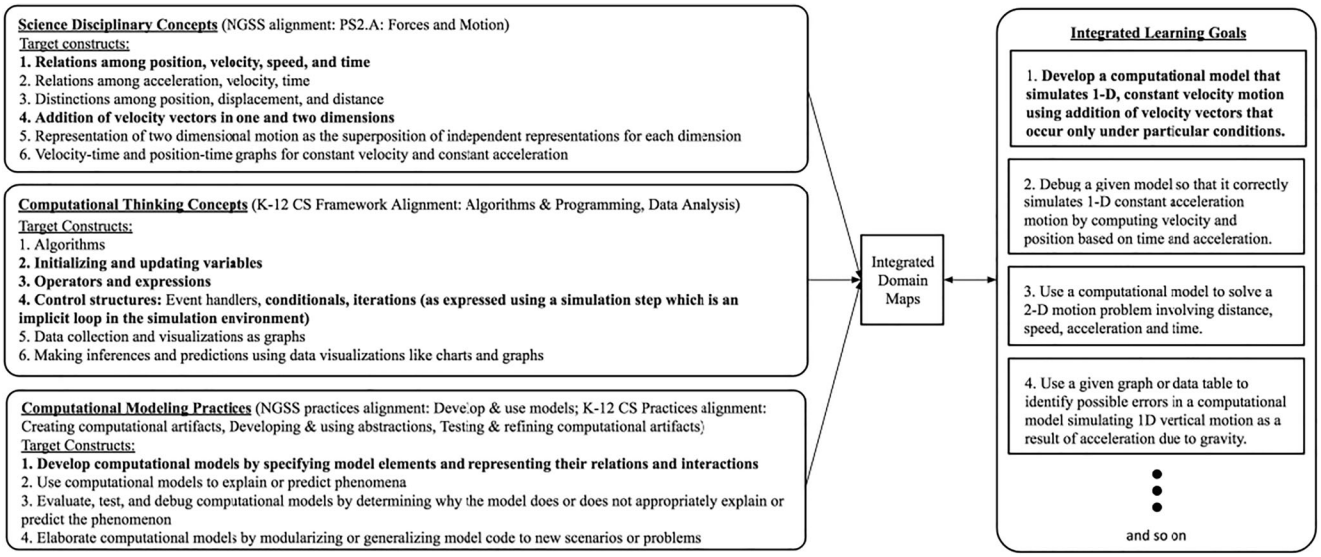


Fig. 2 Unpacking the physics and CT domains, identifying their relationships through integrated domain maps, and the articulation of integrated learning goals (bold text illustrates how a learning goal integrates concepts and practices across disciplines)

2017). To address these challenges, C2STEM adopts a block-based, physics DSML designed in coordination with the ECD processes discussed above. For high school physics, a DSML can play an important role in focusing learner attention to the relevant physics concepts and practices when building models and solving problems in the domain. In addition to the affordance of developing solutions at the level of abstraction of the target domain (e.g., using variables relevant to the specific STEM domain), a DSML allows for the building of programs that are concise and self-documenting by associating variables with properties of objects, and using templates that let learners set and update the variable values in discrete time steps.

Model verification is supported by linking the execution of the model blocks to the data generated, which can be represented as animations, plots, and tables. The DSML for kinematics helps students focus on the concepts of position, velocity, and acceleration, and their relations. It also helps students understand and interpret the computational constructs in a context-specific way.

Exploratory Learning of Dynamic Processes

Like previous work (diSessa 2001; Klopfer et al. 2005; Sengupta et al. 2013), our environment is designed for students to build their computational models so that the temporal evolution of a moving object's behavior can be represented using a *step-by-step* structure. In addition, behaviors can be decomposed into a set of fundamental processes; e.g., the motion of an accelerating object is modeled in two steps: (1) update the object's velocity using its acceleration and (2) update the object's position using its velocity. This approach facilitates *model building in parts*. In addition, particular attention is given to Δt, the simulation time step. Using different values for Δt students can study its effects on the observed behavior of an object, and this may help them understand the relation between the chosen Δt and the continuous behavior of objects.

The system supports exploratory learning (Hew and Brush 2007; NGSS Lead States 2013) because students can incrementally build, experiment, test, and refine their models. In addition, the curriculum implements a progression that enables students to learn new physics and computing concepts by building a series of models, where new models add concepts and practices to previous models. Students can explore with their models by varying parameters (e.g., the acceleration and initial velocity) to study how this affects their model behaviors. We hypothesize that these progressions facilitate *guided exploration*, which allows students to develop a deeper understanding of STEM + CT concepts and practices.

Preparation for Future Learning

C2STEM proposes that learning physics through computational modeling will not only support understanding of the content being taught, but also positively influence future learning of new STEM topics. This preparation for future learning (PFL) perspective (Bransford and Schwartz 1999) is a key design component of C2STEM curriculum and assessments. In general, PFL assessments are simultaneously *summative*: they occur near the end of a learning experience and measure outcomes; and *prospective*: learning resources are explicitly built into the assessment design and aim to measure what students are prepared to learn during the assessment. In this way, they capture benefits of student-driven activities that may not be captured by standard assessments, particularly when students do not all generate a canonical solution (Chin et al. 2010; Schwartz et al. 2011; Schwartz and Martin 2004).

PFL assessments adopt a double transfer paradigm (Schwartz and Martin 2004). Part 1 is identifying what students "transfer in" to help them learn from the resource. Part 2 is how well they can "transfer out" what they have learned from the resource to correctly solve a target problem. In terms of transfer in, there are two primary ways we might expect a learning experience to shape how individuals approach new problems and materials. One is by influencing the strategies and learning processes they may remember and deploy in the new context (Schwartz and Arena 2013). The other is that the learning experience shapes students' understanding of specific concepts, such that they may be better able to expand on them later (Grover et al. 2014).

C2STEM Learning Environment and Curriculum Modules

We adopted the design principles to develop C2STEM as a classroom learning environment for high school physics. Our first DBR iteration with C2STEM was piloted in a controlled environment with 15 students. This experiment pointed us to difficulties that students had in understanding and applying physics and CT concepts and practices when building and developing their models. Sometimes, these difficulties led to interesting discussions between students about their models and the generated behaviors. However, very few students in the pilot study used the data-capture tools to study the behaviors generated by the models. Instead, students focused on the animations generated when they simulated their models, and this often led to them using trial-and-error approaches to refining the models. Lessons we learned from this pilot study, including the difficulties that students faced, led us to better integrate the data-capture tools, providing more opportunities to perform more systematic exploratory analyses of model behaviors and to provide an anchored context to give students a context for each model-building task, and how the sequence

of tasks led to the solution of an overarching problem (Bransford et al. 2012).

Version 2 of the C2STEM system adopted a problem-based learning approach (Torp and Sage 1998). Students were first introduced to an overarching problem domain—transportation, in our case—and the overall challenge problem of *delivering medicines to a tribe in a remote Amazon jungle*. We used evidence-centered design to decompose the overall curriculum into four modules: three in kinematics: (1) 1D land transportation involving constant acceleration and deceleration, (2) 2D constant velocity motion for transportation across a river, and (3) 2D accelerated motion (with gravity as a factor) for package delivery in a remote area using a flying drone. The fourth module covered 1D and 2D motion with forces (including static and dynamic friction).

To support classroom instruction, each topical module was decomposed into sets of instructional, model-building, challenge, and embedded assessment tasks. Figure 3 illustrates the interfaces for the instructional, model-building, and challenge tasks. In each module, students worked through multiple tasks, starting with introductory (instructional) tasks targeting basic domain-specific (physics) and computational concepts. Combined with the classroom teacher's instruction, these tasks helped students prepare to solve more complex model-building tasks. In the classroom, this helped address potential discrepancies in students' prior physics knowledge and programming experience. Exploration during instruction was implemented as a form of *guided inquiry*. Students were asked to change parameters of simulation models (hidden from them) and evaluate physics relationships. For instance, in the instructional task for 1D constant acceleration, students were asked to document the results of running their models with different positive and negative initial values for acceleration and initial velocity. Students were encouraged to use the graphs and data tables to understand the relations between acceleration, velocity, and position of an object.

Following instructional tasks, students worked on model-building tasks, where they combined their learned physics knowledge with CT concepts and practices to build computational models of specified physics phenomena. For example, in the model-building task illustrated in Fig. 3, students applied concepts of 1D acceleration and deceleration to model the motion of a truck that starts from rest, speeds up to reach and cruise at the speed limit, and then slows down to stop at the STOP sign. In addition to the physics, students had to figure out how to use conditionals to model stopping conditions, e.g., when to switch from speeding up to cruise mode and when to switch from cruise to slowing down mode. For the latter, they had to model the "look ahead" distance (when to start decelerating to make the truck stop exactly at the STOP sign). Students were told that they would get extra credit if they created more general models, i.e., they used variables and expressions instead of hard-coding values into their models. Overall, the model-building tasks provided students with opportunities to translate their understanding of physics concepts to computational structures using variables and update functions to represent the system dynamics and to develop important practices of verifying their models using systematic debugging methods.

The challenge tasks that students worked on at the end of each module were comprehensive and tested students' abilities to put together multiple concepts and practices to build a computational model to solve difficult problems. For challenge problems, students had choice in how they went about designing, developing, and testing their solutions, but with sufficient constraints to meet the teacher's instructional goals.

Finally, formative assessments or "check ins" (elaborated further in section "Assessment Design") were designed to evaluate and make students aware of the physics and CT concepts and practices they would encounter in their modeling tasks.

C2STEM adopts a modular architecture, is web-based and deployed on a cloud-based server, allowing for ubiquitous and continuous access through web browsers and Internet resources. Our computational modeling environment and some of the instructional tasks are developed on top of NetsBlox

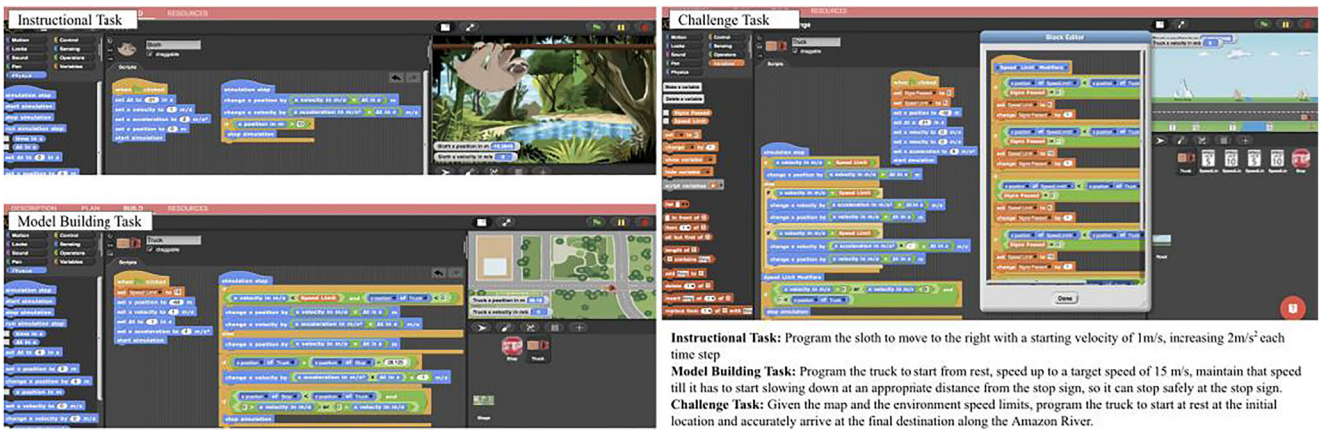


Fig. 3 Example instructional, model building, and challenge tasks with solutions

(Broll et al. 2016), an extension of the Snap! programming environment with added simulation and physics DSML blocks to help learners focus on physics modeling and simulation tasks. As discussed, we added graphing and data table tools to version 2 of C2STEM.

Assessment Design

We have developed three types of assessments for our physics curriculum: summative pre-post tests and PFL assessments, and formative embedded C2STEM check-ins.

Pre-Post Test Assessments

We designed this assessment to measure students' conceptual knowledge in physics and CT by adapting assessment items from other studies (Basu et al. 2017; Grover et al. 2017; Hestenes et al. 1992; McElhaney and Linn 2011). The instrument contained 10 physics items and 7 CT items in multiple choice and constructed response formats.

Embedded Assessments

We developed multiple assessment tasks to measure students' integrated proficiencies in kinematics and CT. We combined kinematics and CT concepts and computational modeling practices to reflect their synergistic nature. For example, calculating the velocity of an object based on its initial velocity, acceleration, and time closely relates to the CT concepts of initializing and updating variables (velocity, acceleration, and time are examples of variables), operators, and expressions. Additionally, combining the related physics and CT concepts with different aspects of CT practices, such as "Develop, Use, Test, Debug," helped create learning goals that guided task design specifications at different levels of complexity.

We developed 18 tasks of varying complexity aligned with our learning goals using multiple choice, open response, and programming formats. In some tasks, we provided most of the code and asked students to fill in a small part that targeted a specific concept, while in other tasks, we provided required blocks and asked students to focus only on arranging the blocks in a correct computational sequence. We created different versions of debugging tasks, such as asking students to correct a given buggy program, showing students a program snapshot and asking them to modify the block(s) to produce a correct model, and asking students to find modeling errors by studying data and graphs from a program not shown to them.

PFL Assessments

In PFL assessments, learning resources are explicitly embedded into the assessments, and we can examine how students approach new learning situations. In designing PFL assessments for C2STEM, we aimed to capture problem-solving strategies and conceptual understanding that a CT-based approach would help foster and that could be useful for learning in other STEM contexts. For example, within C2STEM, students take an iterative *step-by-step* approach to make sense of a continuous change. We coded for this *step-by-step* strategy when examining students' responses to assessments that include new learning resources students had not encountered before. We also analyzed how students may generalize conceptual understanding of the relationships between position, velocity, and acceleration to make sense of jerk, the rate of change of acceleration. Figure 4 presents an example PFL item developed to assess students' preparation to learn the

In class, you learned about **kinematics** – the ways to describe an object in motion.
We often describe an object's motion relative to its position:

- A *change* in an object's position, Δx, is called **displacement**.
- The *rate of change* in an object's position, $\Delta x/\Delta t$, is called **velocity**.
- The *rate of change* in an object's velocity, $\Delta v/\Delta t$, is called **acceleration**.

When we talk about these changes, we define Δ as the **difference** between two measurements. This might be the difference between our first and last measurements, or two sequential measurements.

We can take this one step further and talk about the *rate of change* of an object's **acceleration**, $\Delta a/\Delta t$, or something we call **jerk**. If you're riding in a car, jerk is what you feel when the driver hits the gas, or slams on the brakes, really quickly.

QUESTION: Imagine that a car is moving along a horizontal track. Its initial position is $3m$ along the track, its initial velocity is $1m/s$, and its initial acceleration is $4m/s^2$. For the entire ride, the jerk is $2m/s^3$.

After 3 seconds, what is the approximate position of the car? Keep your answer to one significant figure.

Fig. 4 PFL item providing introduction to jerk

concept of jerk. Jerk was chosen as a near transfer item, as it is a natural extension of a kinematics curriculum and rarely included in traditional physics instruction.

Classroom Study

We outline our study design and measures in this section.

Study Design

Following the initial usability study and the revisions to the system discussed in section "C2STEM Learning Environment and Curriculum Modules", our team conducted a semester-long classroom study in a high school physics classroom in Tennessee. The study included 174 students taking an Honors Physics course; 84 students participated in the experimental group and used our C2STEM system and curriculum. They had some traditional lectures, but most classroom activities combined lectures with the use of C2STEM. Students received homework grades (predominantly completion grades) for C2STEM tasks assigned as homework and for embedded assessments. The 90 students in the control group got traditional instruction through classroom lectures and labs. All sections were taught by the same teacher and followed the same general syllabus. The C2STEM instruction was molded to ensure that nearly equal time was allotted to the physics syllabus for both groups. In addition, all students' grades in the course were based on the same set of assessments designed by the teacher. These assessments were given to us prior to the study and impacted the design of our instructional and embedded assessment tasks.

In this paper, we evaluate student work on the three kinematics modules that primarily covered Newton's first and second laws of motion: 1D motion (with acceleration), 2D motion with constant velocity, and 2D motion with gravitational forces. Control and experimental groups completed all pre- and post-tests as well as the teacher-created conventional paper-and-pencil classroom assessments separate from the C2STEM environment.

Participants

We analyzed data for 34 of the 84 experimental group students who returned their student and parent permission forms. We formed a subsample of control group students (matched control group) to make robust comparisons between the control and experimental groups. The matched control group of 34 was drawn in a way that the distribution of kinematics pre-test scores (see section "Pre-Post Learning Gains") of the C2STEM students were about the same. For each student in the C2STEM group, a control group student was randomly selected (1) whose kinematics pre-test score was within a certain range (r) of the C2STEM group score, i.e., $control\_student_i\ score \sim [treatment\_score_i - r,\ treatment\_score_i + r]$, and (2) who had also completed the post-test and the PFL assessment. Twenty-seven of the 34 C2STEM students provided responses to the PFL item.

Data Sources and Analysis

We used multiple assessment methods (section "Assessment Design") to study students' learning performance and problem-solving approaches.

Pre-Post Test Assessments

Two graders initially scored 6 randomly selected pre- and post-tests together, using a pre-defined rubric to establish a baseline scoring scheme. The graders then scored 10 randomly selected tests separately and produced an inter-rater reliability of 89% across all of the test items. A single scorer graded the remaining tests and computed total scores for the physics and CT pre- and post-tests. The max score attainable was 40 for physics, and 37 for CT. We used a regular t test to determine if there were significant differences in the normalized learning gains for the two groups. The normalized learning gain for a student was computed as: $\frac{(post\_test\ score - pre\_test\ score)}{(max\_score - pre\_test\ score)}$. The normalized learning gain is number in the range [0, 1].

PFL Assessments

For the question discussed in section "PFL Assessments", students' answers were coded for problem-solving approach and correctness of response. Two researchers coded a subset of 40 responses independently. Discrepancies were discussed, and the responses were re-coded. Comparison of codes from the final iteration yielded 90.4% agreement. We found that students used two primary problem-solving strategies: (1) application of kinematics equations and (2) iteratively solving for acceleration, velocity, and position using the *step-by-step* approach. If the C2STEM students used more *step-by-step* strategies, this would provide evidence of transfer of learning processes. If they had less negative transfer of traditional kinematics equations, recognizing that these equations did *not* account for the change in acceleration from jerk, this could provide evidence that these students' conceptual understanding of rates of change enabled them to reason about and generalize to a new, more complex situation.

Embedded Assessments

Our analysis of students' responses to the embedded assessments was conducted as a case study using three specific assessment tasks from the curricular modules. The analysis

illustrates the extent to which students demonstrated evidence of synergistic learning between physics and CT, as well as how students may have struggled to integrate the two domains. Each of the three task rubrics scored students' final programming solutions on two aspects of integrated physics-CT proficiency: (1) the ability to express physics relations in a computational model and (2) the ability to use programming constructs to model a physics phenomenon.

Video Analysis

We recorded student work and conversations using OBS™ screen-capture software to capture key synergistic learning events. We analyzed the videos qualitatively to characterize students' model-building approaches and challenges. For each task, we noted key actions and conversations that highlighted synergistic learning events.

Results and Discussion

We discuss the results of our analyses in this section. In relation to our DBR approach (section "C2STEM Learning Environment and Curriculum ModulesS23"), we made two key study changes: (1) data tools were added to help students interpret how variable values changed over time and (2) instructional tasks and embedded assessments were modified to make the links between the physics equations and the computational modeling structures more explicit. In the context of a semester-long classroom study, the graphing tool was used by our teacher for classroom discussion. He often projected the results of model simulations on a SmartBoard© to explain model behaviors in relation to the laws of motion. This proved especially useful when the instructor found that the experimental group students were having trouble using equations to solve the traditional assessment problems in his classroom quizzes. Once students had developed proficiency in using and interpreting graphs, the teacher helped them connect the generated graphs with the traditional kinematics equations.

Pre-Post Learning Gains

The students' kinematics and CT pre-post test scores are shown in Fig. 5. For kinematics, the experimental group averaged a score of 21.12 ($sd = 6.16$) on the pre-test, improving to 29.82 ($sd = 4.67$) on the post-test. The corresponding average scores for the control group were 22.71 ($sd = 5.29$) and 27.79 ($sd = 5.79$). For CT, the experimental group averaged scores of 20.71 ($sd = 4.88$) and 25.82 ($sd = 7.58$), and the control group averaged 21.18 ($sd = 7.64$) and 23.29 ($sd = 6.25$), respectively. The t test analysis on learning gains showed significant improvements for both groups in kinematics (experimental: $p = 9.38 \times 10^{-9}$, control: $p = 0.0003$). In CT, the experimental group had significant learning gains ($p = 0.002$), but the control group did not ($p = 0.22$), which was not unexpected. A t test comparison of the normalized learning gains showed that the experimental group performed significantly better in kinematics ($p = 0.01$) and CT ($p = 0.008$).

A more detailed analysis of specific physics questions also indicates key learning gains for the experimental group as compared to the control group. In one question, students had to choose the plot that correctly depicted the free-falling motion of a ball subjected to gravity (constant acceleration—see Fig. 6). This tested students' ability to interpret the *step-by-step* motion of the free-falling ball. On the pre-test, 12 experimental group and 7 control students answered this question correctly. On the post-test, 28 experimental and 17 control students selected the correct answer. Therefore, *step-by-step* modeling seems to have helped the students in the experimental group gain a better understanding of constant acceleration motion.

In the CT domain, the experimental group's ability to debug their models improved significantly. Students were given the snippet of code depicted in Fig. 7 and asked to describe

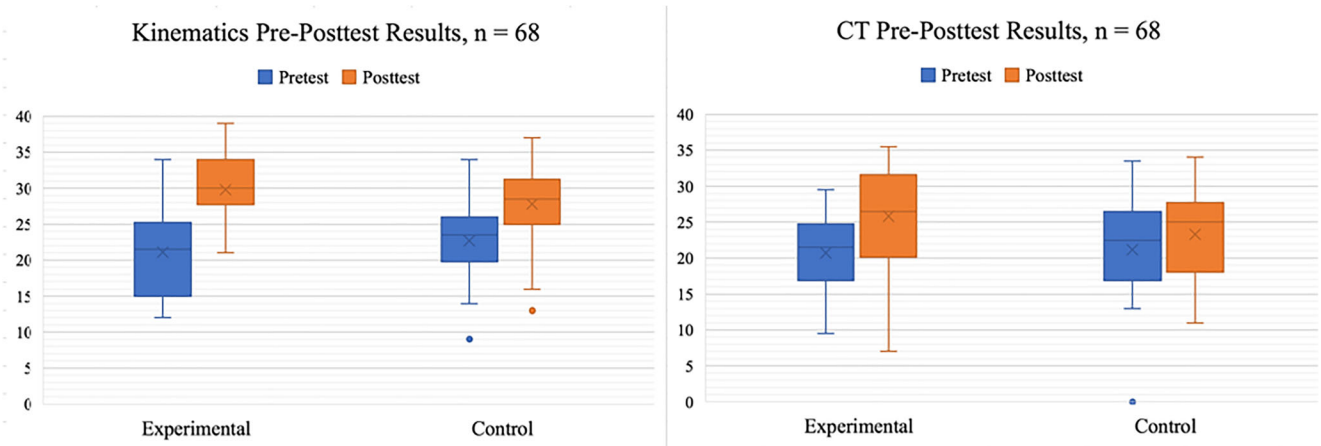


Fig. 5 Group results for kinematics and CT pre-posttests

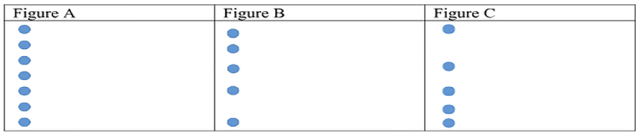


Fig. 6 Options for ball drop, kinematics pre-posttest question

how they would fix the code to accurately model a ball's horizontal motion to the right for 20 m at a constant velocity of 2 m/s. The correct solution required students to move the "set x position to 0 m" initialization block outside of the "repeat until" loop block. In the pre-test, only 2 experimental group students got the right answer, but 17 students had the correct answer on the post-test. In contrast, the number of correct answers for the control group increased from 6 to 8.

PFL

Open-ended student responses to the PFL question (Fig. 4) were coded for their problem-solving approach and final answer correctness. We found two primary problem-solving strategies: application of kinematics equations, and iteratively solving for acceleration, velocity, and position using the *step-by-step* approach. In the first case, students either applied the standard kinematics equations without accounting for changing acceleration (an example of negative transfer) or recognized the need for extending the formulae in some way to include jerk. The second case was evidence of students' *step-by-step* thought process, which we anticipated to be a more CT-inspired strategy that was supported within C2STEM.

Student answers were also coded for correctness. As the correct answer to this problem can only be achieved using integral calculus[1] or a small Δt, student responses were instead assessed on a range of "reasonableness." This range was defined as 25–52 m, using the minimum and maximum displacement values one could posit based on the acceleration value at 1-s intervals. The answer a student would obtain by applying standard kinematics formulae without accounting for jerk is outside this range. We found that students in the matched-control group were more likely to negatively transfer and apply the standard kinematics equations without accounting for jerk: 14 (41%) of the matched-control students made this error in solving the problem, while only 7 (26%) of the experimental students made this error. Very likely, these students did not recognize the limitations of the standard kinematics formulae for situations involving jerk, or they could not find an alternate strategy. In contrast, the experimental students implemented more flexible problem-solving strategies than their matched-control counterparts. Seven (26%) students in the experimental group used a *step-by-step* approach to solve the problem, while only 5 (15%) of the control student responses show evidence of the same strategy. Twelve (45%) of the experimental group students tried to adapt the traditional formula, compared to 8 (23%) of the control students. A comparison of student strategies is displayed in Fig. 8. We combined students who used either the *step-by-step* or the extension approach into a single "flexible strategy" category. When comparing the treatments and problem-solving strategies (formula vs. flexible), we find a non-significant trend that C2STEM students use the flexible strategies more frequently ($\chi^2 = 3.44$, $p = 0.064$). In terms of reasonableness of answer, the differences between groups were small and non-significant. Across both conditions, 6 students in each group (18% and 22% of the control and experimental groups, respectively) gave answers that met our criteria of being in the reasonable range.

[1] At the time of the study, no students in either condition had received integral calculus instruction.

Case Studies

We adopted an explanatory case study approach (Gomm et al. 2000) to examine how the environment, our design principles, and associated curriculum helped foster synergistic learning of physics and CT in practice, and how it supported students' transfer of modeling strategies. In this paper, we consider two illustrative cases from the experimental group: John and Amy (names changed to maintain anonymity). We chose John and Amy based on their performances on the pre- and post-tests, select embedded activities (one from each of the modules 1–3), and the PFL task.

Embedded Check-Ins

John started with relatively high pre-test scores in physics (29 out of 40) and CT (25.5 out of 37) and exhibited learning gains in both domains (physics post score = 35 and CT post score = 30.5). John performed consistently well on the embedded assessment tasks across modules 1, 2, and 3 and was also successful in applying the strategies he learned in C2STEM to the PFL task. Figure 9 provides a snapshot of John's submission for Check-In 1.1.1. John correctly arranged the given blocks to simulate the motion of an object traveling horizontally at a constant velocity of 0.5 m/s till it reached an *x-position* of 20 m. This demonstrates his understanding of and proficiency with the DSML blocks and the *step-by-step* modeling of object behaviors. He correctly separated out initialization actions from actions that needed to repeat at every simulation step to model the dynamic behavior, and correctly constructed the stopping condition.

Similarly, for Check-In 2.1.1, John correctly modified the given program to model the object's motion starting at Point B (Fig. 9). John's solution satisfied all six criteria across the rubric components. He showed a good understanding of relative velocity by modifying the procedure "set-Josh-resultant-velocity" to correctly model Josh's new velocity beyond Point

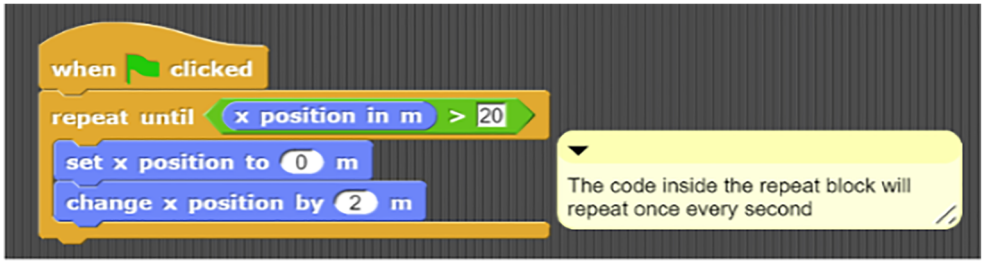


Fig. 7 Code snippet for debugging pre-posttest problem

B as the sum of the walkway speed and his walking speed. John correctly separated the initialization and update actions, and correctly model the update position actions in the code. His model did not include extraneous code, implying that he had a good understanding of the required CT concepts.

John continued to demonstrate proficiency in Check-In 3.1.2, which dealt with acceleration due to gravity, and the relation between velocity and acceleration. He was successful in debugging the erroneous model to correct the initial *y-velocity* of the ball dropped from a height of 20 m, and he correctly initialized the *y-acceleration* to − 9.8 m/s^2. He also had the correct expression for the *y-velocity* update at every simulation step. His final program submitted for this task (see Fig. 9), as well as his written explanation for how he debugged the program, provide evidence of his understanding of the relations between velocity, acceleration, position, and time, which he subsequently transferred and extended in the PFL task about jerk.

On the other hand, Amy represented a student who started out with low to medium physics and CT pre-test scores (25 out of 40, and 22 out of 37, respectively) and showed considerable learning gains in both domains (physics post-score of 34, and CT post-score of 30.5). Though Amy's post-scores were similar to John's, Amy, unlike John, did not perform consistently well on the embedded assessment tasks. She struggled with module 1, showed some improvement module 2, but again struggled with the physics concept of acceleration due to gravity and the CT practice of debugging in module 3, which was consistent with her response on the subsequent PFL task.

Figure 10 depicts Amy's solution for Check-In 1.1.1. Consistent with her low pre-test scores, she struggled with organizing the blocks provided into the correct model. Her initial model terminated the simulation after initializing position and velocity, making the rest of the code unreachable. Amy also struggled with other embedded assessment tasks in module 1 that involve code comprehension, development, or debugging. She showed some improvement in module 2 and generated the correct solution for Check-In 2.1.1, updating the object's speed on the walkway to 2.5 m/s instead of 0 m/s (see Fig. 10). This showed evidence of progress in her ability to develop a physics computational model. However, in Amy's response to Check-In 2.1.1, she hardcoded the resultant velocity value instead of expressing Josh's resultant velocity as the sum of the variables.

In Check-In 3.1.2, Amy was unable to express the physics relations between position, velocity, and acceleration. Also, she showed no proficiency in applying CT-related debugging practices. Figure 10 shows that she could not correct the initial

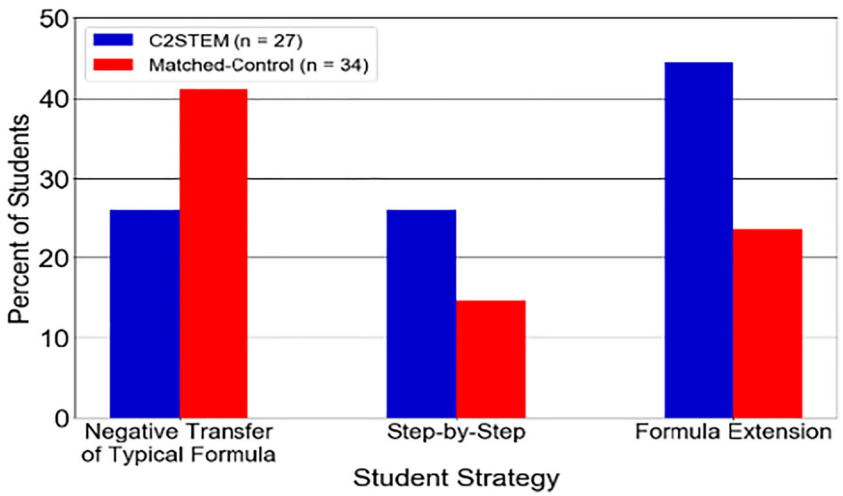
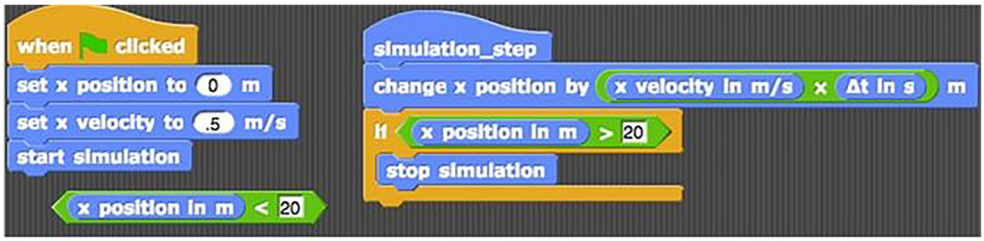
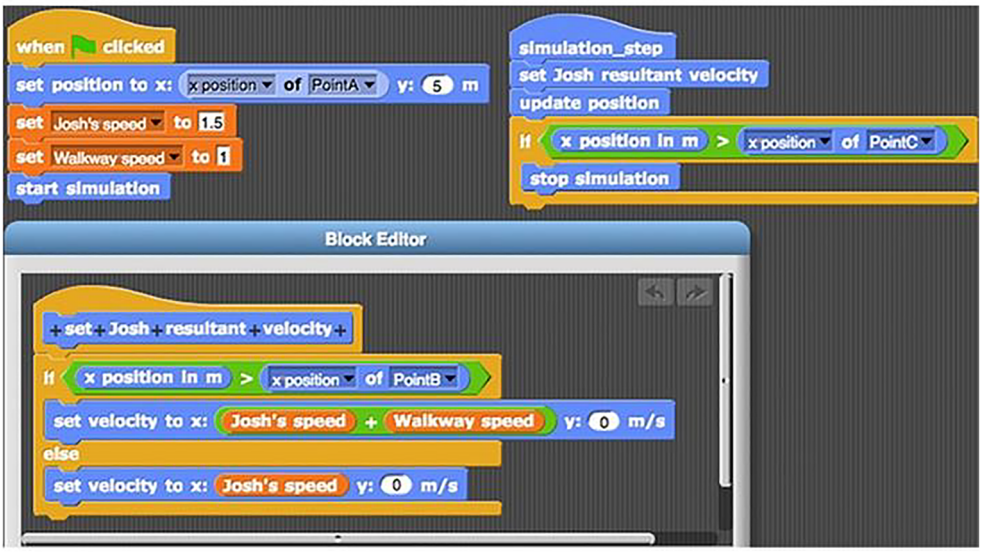


Fig. 8 Comparison of PFL response strategies

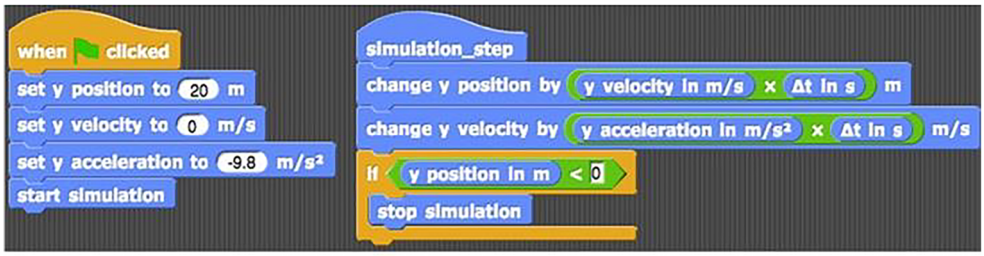
Fig. 9 John's checkin solutions



Check-in 1.1.1



Check-in 2.1.1

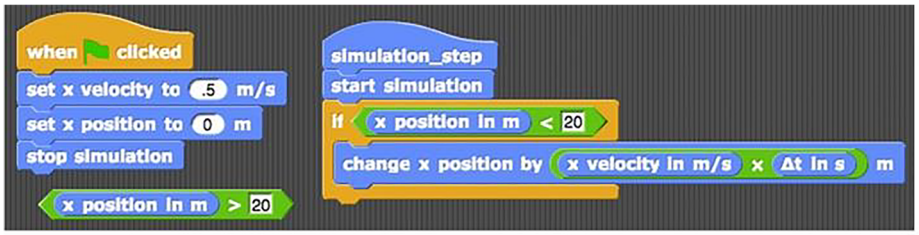


Check-in 3.1.2

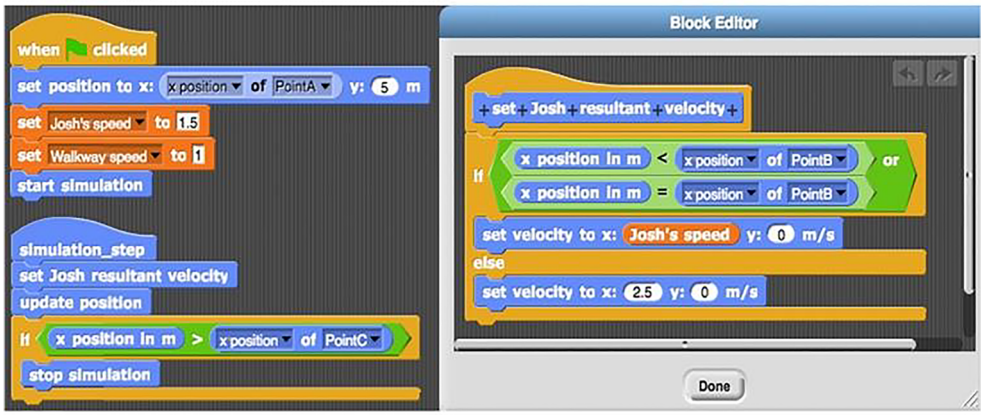
velocity and acceleration, or write down the expression for updating the *y-velocity* at every simulation step. However, she was able to make the correction to specify that the change in *y-position* should be in the negative direction by multiplying the given expression by −1. She made the same mistakes in this task and the PFL assessment, where she updated the position by the value of acceleration, implying that she did not understand the relation between velocity and acceleration. Her lack of knowledge of the physics relations may have added to her struggles with debugging. Amy consistently displayed lack of proficiency with the CT practice of debugging in various assessment tasks, and this was consistent with her low scores on the debugging task in the CT post-test.

In trying to reconcile Amy's high post-test scores with her struggles in the embedded check-ins, it is likely that Amy was unable to translate her physics knowledge to an understanding of the DSML constructs and their purpose to support model building. She had difficulties in understanding the simulation model structure (i.e., variable initialization followed by the repeated *step-by-step* update of variable values to model dynamic behaviors). On the CT summative assessment, students were required to correctly insert the "change" block in a given code segment to correctly update the behavior of the object. Amy was unable to do so on both the pre-test and post-test. In addition, she did not learn the debugging process. She did not attempt the task in the pre-test, and scored zero points on both sub-parts of the task (describing the buggy behaviors,

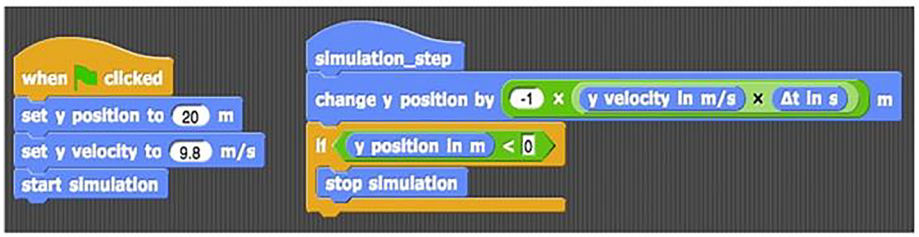
Fig. 10 Amy's checkin solutions



Check-in 1.1.1



Check-in 2.1.1



Check-in 3.1.2

and steps to debug the program) in the post-test. Her interactions with others and her responses indicate that she did not understand the DSML blocks, for example, that the *change x-position* block did not working correctly without the velocity variable block. It is clear that Amy needed additional scaffolding to overcome her difficulties and better learn the curricular physics and CT concepts and practices.

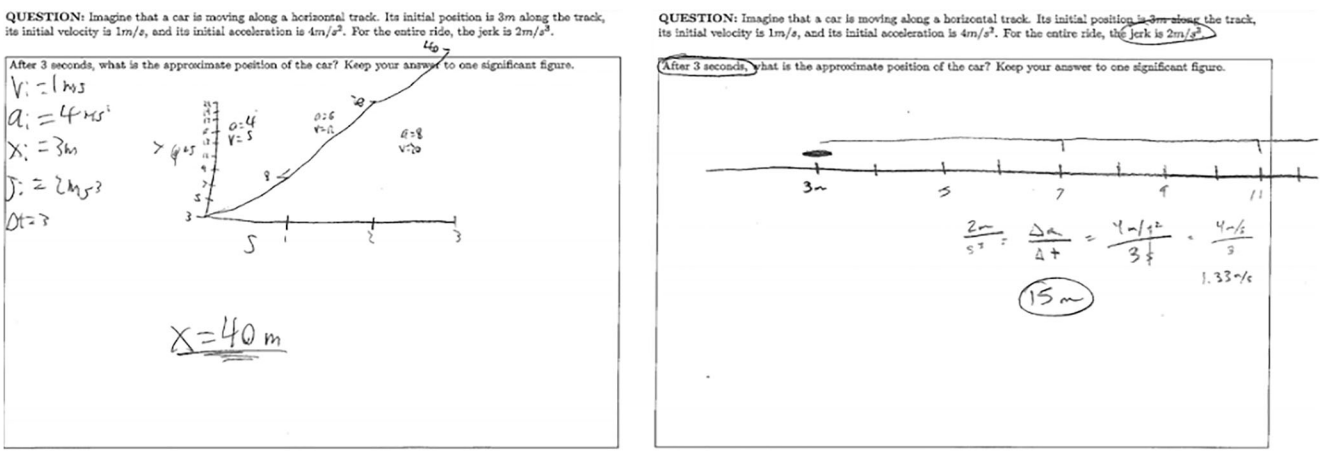


Fig. 11 John's (left) and Amy's (right) responses to the PFL item

PFL

John's response to the PFL item (Fig. 11, left) used the *step-by-step* strategy to achieve a reasonable final answer. He used the initial conditions provided in the problem statement to set up a graph of displacement as a function of time. In one-second time intervals, he found the acceleration, velocity, and displacement and then filled in the graph.

$$(1)\mathrm{a1} = \mathrm{a0} + \mathrm{j}; (2)\ \mathrm{v1} = \mathrm{v0} + \mathrm{a1}; \textit{and}\ (3)\ \mathrm{x1} = \mathrm{x0} + \mathrm{v1}$$

This clearly exhibits both parts of the double transfer paradigm: John had transferred in a strategy, and successfully applied it to solve the problem, finding a reasonable way to make sense of the new quantity introduced in the assessment.

In contrast, Amy's strategy (Fig. 11, right) was less clear, and the final answer was not within the reasonable range. Her drawn number line appears to represent the position of the vehicle on the track, as indicated by the starting value of 3 m. Demarcations along the line may indicate that she was attempting to analyze the problem in discrete time intervals, but the intervals were not explicitly identified. Amy tried to include jerk in the problem by first expressing the numerical value for jerk (2 m/s^3) as the change in acceleration over time. In this sense, she recognized that this situation presented a new problem context that went beyond what she had been taught. However, she could not successfully incorporate jerk into her final answer. Amy's next calculation conflated acceleration with the change in acceleration, and she used the initial value of acceleration and the total time interval to compute a new quantity. Her tentative hold on the physics concepts was also manifested in her final response. From the number line, she appeared to take the initial position and add to it the value of the initial acceleration three times, bypassing velocity, i.e.,

$$\mathrm{xf} = \mathrm{xi} + 3\mathrm{ai}.$$

Unlike John, Amy could not transfer in the strategies and physics understanding to extend her understanding of the relationships between position, velocity, and acceleration to make sense of this new, more complex scenario. However, she did bring in some of the concepts she had learned to facilitate her sense-making.

Video Analysis

Our assessments allowed us to establish whether synergistic learning occurred, but to gain a better understand how such learning occurs, it is important to examine students' moment-to-moment activities in C2STEM. When are students learning or applying concepts or practices of physics? When are students learning or applying CT concepts or practices? When is their learning of physics and CT mutually reinforcing?

To address these questions, we analyzed audio and video captures of a subset of computer screens in the classroom. Our findings show that moments of synergistic learning were supported by the C2STEM environment, often in unanticipated ways. We noticed several varieties of synergistic learning moments: (1) using a simulation to test a conceptual issue, (2) debating whether the model should capture the mechanism of the phenomenon, and (3) debugging a program that is not behaving in the expected or intended way. For all three varieties of synergistic learning, students encountered such moments while pursuing emergent goals as they engaged with model-building challenges within the C2STEM environment and curriculum.

Debugging was the most prevalent of the three types of identified moments of synergistic learning. Every student encountered the need to debug their models at various points, whenever the behaviors did not match their expectations. Such moments often invoked students' conceptual understanding of physics (e.g., object is not accelerating when it should) through engagement in the practices of physics (e.g., refining a model based on observational evidence), all while debugging the model code, a key CT skill.

As an illustrative example, we present the case of a debugging episode from Amy. This episode comes from module 3, where students were tasked with building a model of a drone flying with a constant velocity with two packages, and dropping one package onto a target on the ground, and the second package onto an elevated target. On day four of module 3, Amy started with a program she had already compiled in a previous session. In Amy's model, package 1 moved forward with the drone at a constant horizontal velocity and maintained that constant horizontal velocity after it was dropped while also accelerating in the *y-direction*. However, there were two main problems Amy noticed with her model: (1) package 1 did not quite reach the target and (2) package 2 did not move with the drone at all, but rather remained at rest in the air.

Amy articulated the problem several times that package 2 was not moving with the drone, but did not make progress on that problem immediately. Before she could work on these problems, other issues took priority. First, the instructor had all the students adjust the initial positions of their drones and packages. Then, the students did some paper-and-pencil work to determine the correct *x-position* for the drone to release each of the packages to hit their respective targets. After computing her drop points on paper, Amy returned to the C2STEM environment to adjust the release point of package 1. It hit the target on the first subsequent run of the simulation. However, there was still the problem of package 2 not moving at all with the drone. Amy articulated that this was a problem she was orienting to, and asked for help:

Amy: (looking at another student's code) What code is that? Is that the drone?
Student: Uh, yeah, and the packages.
Amy: This is what she—this is like what I did with Liz. Yeah, the drone moves, but it doesn't, like, I—This is the one that's not moving. It drops it here.
Student: Well, you gotta get all the packages to move.
Amy: I think something is cancelled in the y... (inspects student's code, then adds "change x-position of package 2" block to her own code)

With the help of some neighboring students, Amy engaged in a debugging process in which she realized she was missing a "change *x-position* of package 2" block of code. She added that code, but had inadvertently changed the velocity of the drone in the process (demonstrating that novices often make new bugs during the debugging process, as reported in McCauley et al. 2008).

On subsequent tests, package 2 started moving, but not at the same velocity as the drone. Amy again asked for help. As neighboring students offered suggestions, Amy honed in on the drone's velocity block as the problem and adjusted the velocity initialization to match that of the drone. This led to a successful landing of both package 1 and package 2 on the subsequent "Run."

Overall, the video analysis complements Amy's performance on the pre-and post-tests, on which Amy exhibited overall CT and physics pre-post learning gains. Specific areas of improvement in CT include variable initialization (determining correct values and purpose of variable) as well as interpreting sequences and conditions to predict results. These results coincide with the video results. Amy was able to initialize needed variables (CT) and demonstrated an understanding of what the simulation should look like (physics). However, as the CT application became more difficult, errors in debugging occurred. Her struggles in debugging and over-reliance on her peers could explain her low pre-post gain on the debugging task. The video analysis also reveals a potential misunderstanding regarding either the DSML blocks or the *step-by-step* dynamic motion. For instance, Amy considers that "something is cancelled in the y," regarding an issue with the motion of the object in the *x*-direction. The video data corroborate her difficulties interpreting DSML blocks in the embedded check-in assessments.

This case study of Amy's in-the-moment engagement with C2STEM highlights how it can support students' synergistic learning of physics and CT. In attempting to solve the challenge task, Amy was afforded authentic opportunities to engage in the practices of physics (revising a model based on observational evidence) as well as the practices of computation (debugging and problem decomposition). In resolving the unexpected problems with her model, Amy used her understanding of relative velocity between the drone and the package to identify the problematic code and make corrections. One interesting finding is that Amy's debugging was supported in-the-moment not only by C2STEM, but also by her peers who collaborated with her in debugging her code. Ultimately, she may have relied so heavily on her peers that she did not learn how to debug code on her own. We plan to examine a collection of video case studies to characterize the varieties of synergistic learning moments, to analyze how C2STEM supports such moments, and to inform future iterations of C2STEM's support of such moments.

Conclusions and Future Work

This paper presents a comprehensive approach to integrating STEM and CT using the affordance of a computer-based learning environment. Building on prior approaches (diSessa 2001; Klopfer et al. 2005; Lee et al. 2014; Sengupta et al. 2013; Sherina et al. 1993; Yoon et al. 2018), we designed this version of C2STEM specifically to engender exploratory experiences, where students build computational models of science phenomena; analyze, verify, and refine their models; and then apply their models to problem-solving tasks within the framework of a classroom instructional environment. This paper presents the theoretical framing for synergistic STEM + CT learning, articulates a set of design principles that facilitate synergistic learning of STEM and CT, and presents the results of a study conducted in conjunction with a high school physics teacher in which we applied these principles to implement C2STEM to support high school physics learning.

Our initial results are promising: students working with C2STEM developed a better understanding of concepts and practices of physics and CT than students who learned through a traditional curriculum. In particular, our analyses show that students using a *step-by-step* model-building approach did better in understanding and applying the primary kinematics concepts of position, velocity, and acceleration, while also developing model decomposition, refinement, and debugging skills—all important practices for modeling and problem solving in STEM domains and developing CT expertise. Students also demonstrated abilities to transfer and extend concepts and problem-solving skills to new problem-solving situations.

Overall, students seemed to be more motivated and worked harder on their physics tasks, even when they had difficulties. For example, John, whose work we described earlier, wrote to his teacher ". . . The most memorable time from this past year were [sic] during the programming segment. While a number of students absolutely despised this teaching method, it honestly helped me gain a deeper understanding of the subject matter, and this understanding helped me assist other students in learning the environment as well. I spent time in and out of class helping out and explaining the concepts to my peers, and

spending this time deepen [sic] our understanding of physics and personal bonds . . .” Other students remarked that the ability to execute their models and see immediate results made them work harder to construct correct physics models.

But a number of students had difficulties, as John remarks above, and as Amy exemplified. Amy relied heavily on her classmates to help her interpret her physics concepts in a computational modeling framework, and then to debug her models. Analysis of the data is pointing us to a number of conceptual difficulties that students encounter in using the C2STEM system, and in learning the physics and CT. Continuing with our DBR approach, we will use this information in the next step of iterative redesign and reimplementation of the C2STEM interfaces and curriculum. We will also design adaptive scaffolding mechanisms to help students with the specific difficulties they encounter during their model-building tasks.

Applications of our design principles provided useful information to support system improvements. Our approach for using ECD was novel in that it supported the coordinated design of system development, curriculum, and embedded assessments that proved useful in identifying misunderstandings related to domain and modeling knowledge. The exploratory learning of dynamic behaviors principle proved effective compared to the control group in post-test performance (including the ball-drop question from section “Pre-Post Learning Gains”) and with regard to our PFL design principle and student performance on the PFL assessment. Anecdotally, students submitted a feedback form at the end of the semester in which they were asked to describe what they liked or did not like about the *learning-by-modeling* approach. Responses included, “It helps explain why what happens, happens,” “It’s nicer than memorizing equations,” “Being given a scenario instead of just numbers was easy for me to see what is happening and how it needs to work,” and “I understand how to use physics in a more real-world level instead of just in a classroom.” This clearly illustrates that our approach goes beyond typical textbook approaches, and allows students to apply and practice domain principles, solve complex problems, explore and inspect their solution processes, and rethink and revise their solutions based on feedback provided by the system.

However, not all feedback was positive. One student noted that “I think I would have learned better with [the teacher] just talking to us and us taking notes, although it wouldn’t have been as much fun,” adding that we should “actually teach how to code—like the basics and what all the blocks and variables mean.” This response relates to potential issues regarding our DSML design principle. Amy’s struggle with the DSML indicates that we may not have done enough in terms of teaching to or providing resources on the physics DSML. As a result, we aim to evaluate and redesign our introduction to the system’s blocks, connecting descriptions to the visual simulation to address barriers that students with lower initial CT performance may have.

Limitations of Our Work

There is a tension between the learning goals of STEM and CT that will be present in any attempt to integrate the two. Moreover, there is a tension between the goals for students to learn content and to engage in authentic practices of the STEM discipline through exploration. C2STEM is an attempt to find ways to navigate these tensions such that the learning of content and practices of both STEM and CT mutually reinforce each other. This requires finding a balance between allowing students the freedom to explore multiple solution paths and scaffolding students in learning particular concepts that other concepts depend on. How to best navigate these tensions depends on the instructional context and requires constant vigilance.

Other attempts to integrate STEM and CT into a computer-based learning environment have involved more free exploration when compared with C2STEM (e.g., diSessa, 2001; Sherina et al. 1993). This is largely due to a different instructional context, including different pedagogical goals. For example, diSessa (2001) worked with elementary school students who were given the option to play with Boxer during their free period, taking turns on a small set of computers. In that context, the students independently generated an impressive diversity of explorations and designs. In contrast, C2STEM is an attempt to integrate CT into intact high school physics classes, thereby emphasizing curricular content goals. We had to find a balance that involved providing more scaffolding for all students to learn specific physics concepts.

In our high school study, we found evidence that our balance was not always ideal. For instance, after the C2STEM students had engaged in several kinematics modeling tasks, an in-class quiz revealed that they were falling behind their counterparts in their ability to use the kinematics equations to solve problems on paper. In response, the instructor provided more explicit connections between the students’ code, graphs, and the kinematics equations, and the post-test results suggest that this adjustment was successful in getting those concepts across. However, these sorts of adjustments may have tipped the balance too far away from student-led exploration as evidenced in the video case study (e.g., Amy’s reluctance to independently explore and debug her code). In future work, we will explore the dynamics of how different models of providing instructional scaffolding influences students’ content learning as well as their spirit of exploration.

Implications and Future Work

C2STEM demonstrates an effective, *learning-by-modeling* approach for the simultaneous learning of physics and CT in

existing physics classrooms. Our semester-long study illustrates benefits, limitations, and potential improvements for those interested in integrating modeling, and specifically CT concepts and practices, at the high school level while still ensuring learning of the STEM domain. As part of our DBR process, we hope to expand our system by advancing inquiry and exploratory learning opportunities, allowing for more customization of the DSML blocks, and enhancing collaboration opportunities. In our next DBR iteration, we will incorporate these lessons learned to develop a conceptual modeling phase in the environment that will provide students with the opportunity to perform more exploration using inquiry and experimentation tasks that direct the students to think more about the conceptual structure of their models (both from a domain and computational viewpoint) before they embark on their computational model-building tasks.

Acknowledgments We thank Dan Schwartz, Brian Broll, Justin Montenegro, Christopher Harris, Naveed Mohammed, Asif Hasan, Carol Tate, Shannon Campe, and Jill Denner for their assistance on this project. This project is supported under National Science Foundation Award DRL-1640199.

Compliance with Ethical Standards

Conflict of Interest The authors declare that they have no conflict of interest.

Ethical Approval All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki declaration and its later amendments or comparable ethical standards.

Informed Consent Informed consent was obtained from all individual participants included in the study.

References

Barab, S., & Squire, K. (2004). Design-based research: Putting a stake in the ground. *Journal of the Learning Sciences, 13*(1), 1–14.

Basu, S., Biswas, G., & Kinnebrew, J. S. (2017). Learner modeling for adaptive scaffolding in a computational thinking-based science learning environment. *User Modeling and User-Adapted Interaction, 27*(1), 5–53.

Basu, S., Biswas, G., Sengupta, P., Dickes, A., Kinnebrew, J. S., & Clark, D. (2016). Identifying middle school students' challenges in computational thinking-based science learning. *Research and Practice in Technology Enhanced Learning, 11*(1), 1–35.

Biswas, G., Segedy, J. R., & Bunchongchit, K. (2016). From design to implementation to practice a learning by teaching system: Betty's Brain. *International Journal of Artificial Intelligence in Education, 26*(1), 350–364.

Bransford, J. D., Sherwood, R. D., Hasselbring, T. S., Kinzer, C. K., & Williams, S. M. (2012). Anchored instruction: Why we need it and how technology can help. In *Cognition, education, and multimedia* (pp. 129-156). Routledge.

Bransford, J. D., & Schwartz, D. L. (1999). Chapter 3: Rethinking transfer: A simple proposal with multiple implications. *Review of Research in Education, 24*(1), 61–101. Retrieved from. https://doi.org/10.3102/0091732X024001061.

Broll, B., Volgyesi, P., Sallai, J., & Ledeczi, A. (2016). NetsBlox: a visual language and web-based environment for teaching distributed programming (technical report). http://netsblox.org/NetsBloxWhitePaper.pdf.

Chi, M. T. H. (2005). Common sense conceptions of emergent processes: Why some misconceptions are robust. *Journal of the Learning Sciences, 14*, 161–199.

Chin, D. B., Dohmen, I. M., Cheng, B. H., Oppezzo, M. A., Chase, C. C., & Schwartz, D. L. (2010). Preparing students for future learning with Teachable Agents. *Educational Technology Research and Development, 58*, 649–669.

Cooper, S., & Cunningham, S. (2010). Teaching computer science in context. *ACM Inroads, 1*(1), 5–8.

diSessa, A. A. (2001). *Changing minds: Computers, learning, and literacy*. Cambridge, MA: MIT Press.

Gomm, R., Hammersley, M., & Foster, P. (Eds.). (2000). *Case study method: Key issues, key texts*. London: Sage Publications.

Grover, S., & Basu, S. (2017). Measuring student learning in introductory block-based programming: Examining misconceptions of loops, variables, and Boolean logic. *In Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 267-272). ACM.

Grover, S., Jackiw, N., & Lundh, P. (2017). *Integrating Dynamic Mathematics to Advance Learning of Computing Concepts for Diverse Student Populations. Presented at the Annual Meeting of the American Educational Research Association*. TX: San Antonio.

Grover, S., & Pea, R. (2018). *Computational thinking: A competency whose time has come* (p. 19). *Computer Science Education: Perspectives on Teaching and Learning in School*.

Grover, S., Pea, R., & Cooper, S. (2014). Expansive Framing and Preparation for Future Learning in Middle-School Computer Science. In *In Proceedings of the 11th International Conference of the Learning Sciences*. Boulder: CO.

Hambrusch, S., Hoffmann, C., Korb, J. T., Haugan, M., & Hosking, A. L. (2009). A multidisciplinary approach towards computational thinking for science majors. *ACM SIGCSE Bulletin, 41*(1), 183–187.

Harel, I., & Papert, S. (1990). Software design as a learning environment. *Interactive Learning Environments, 1*(1), 1–32.

Harris, C. J., Krajcik, J. S., Pellegrino, J. W., & McElhaney, K. W. (2016). *Constructing assessment tasks that blend disciplinary core ideas, crosscutting concepts, and science practices for classroom formative applications*. Menlo Park, CA: SRI International.

Hestenes, D., Wells, M., & Swackhamer, G. (1992). Force concept inventory. *The Physics Teacher, 30*(3), 141–158.

Hew, K. F., & Brush, T. (2007). Integrating technology into K–12 teaching and learning: Current knowledge gaps and recommendations for future research. *Educational Technology Research and Development, 55*(3), 223–252.

Jona, K., Wilensky, U., Trouille, L., Horn, M. S., Orton, K., Weintrop, D., & Beheshti, E. (2014). Embedding computational thinking in science, technology, engineering, and math. *Presented at Future Directions in Computing Education Summit*, Orlando, FL.

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

Klopfer, E., Yoon, S., & Um, T. (2005). Teaching complex dynamic systems to young students with StarLogo. *Journal of Computers in Mathematics and Science Teaching, 24*(2), 157–178.

Lee, I., Martin, F., & Apone, K. (2014). Integrating computational thinking across the K–8 curriculum. *ACM Inroads, 5*(4), 64–71.

Mayer, R. E. (1999). Designing instruction for constructivist learning. *Instructional-Design Theories and Models: A New Paradigm of Instructional Theory, 2*, 141–159.

McCauley, R., Fitzgerald, S., Lewandowski, G., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: A review of the

literature from an educational perspective. *Computer Science Education, 18*(2), 67–92.

McElhaney, K. W., & Linn, M. C. (2011). Investigations of a complex, realistic task: Intentional, unsystematic, and exhaustive experimenters. *Journal of Research in Science Teaching, 48*(7), 745–931.

Mislevy, R. J., Haertel, G., Riconscente, M., Rutstein, D. W., & Ziker, C. (2017). Evidence-Centered Assessment Design. *In Assessing Model-Based Reasoning using Evidence-Centered Design* (pp. 19–24). SpringerBriefs in Statistics. New York: Springer International Publishing.

National Research Council. (2012). *A Framework for K–12 Science Education: Practices, Crosscutting Concepts, and Core Ideas*. Washington, DC: The National Academies Press.

Lead States, N. G. S. S. (2013). *Next Generation Science Standards: For states, by states*. Washington, DC: The National Academies Press.

Papert, S. (1991). Situating constructionism. In I. Harel & S. Papert (Eds.), *Constructionism* (pp. 193–206). Westport, CT: Ablex Publishing.

Redish, E. F., & Wilson, J. M. (1993). Student programming in the introductory physics course: MUPPET. *American Journal of Physics, 61*(3), 222–232.

Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (pp. 265–269). New York: ACM.

Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior, 72*, 678–691.

Schnabel, R. B. (2011). Educating computing's next generation. *Communications of the ACM, 54*(4), 5–5.

Schwartz, D. L., & Arena, D. (2013). *Measuring what matters most: Choice- based assessments for the digital age*. Cambridge, MA: MIT Press.

Schwartz, D. L., Chase, C. C., Oppezzo, M. A., & Chin, D. B. (2011). Practicing versus inventing with contrasting cases: The effects of telling first on learning and transfer. *Journal of Educational Psychology, 103*(4), 759–775.

Schwartz, D. L., & Martin, T. (2004). Inventing to prepare for future learning: The hidden efficiency of encouraging original student production in statistics instruction. *Cognition and Instruction, 22*(2), 129–184.

Sengupta, P., Dickes, A., Farris, A. V., Karan, A., Martin, D., & Wright, M. (2015). Programming in K-12 science classrooms. *Communications of the ACM, 58*(11), 33–35.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with k-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies, 18*(2), 351–380.

Sherin, B. L. (2001). A comparison of programming languages and algebraic notation as expressive languages for physics. *International Journal of Computers for Mathematical Learning, 6*(1), 1–61.

Sherina, B., diSessa, A. A., Hammer, D., & Sherin, B. L. (1993). Dynaturtle revisited: Learning physics through the collaborative design of a computer model. *Interactive Learning Environments, 3*, 91–118.

Soloway, E. (1993). Should we teach students to program? *Communications of the ACM, 36*(10), 21–25.

Torp, L., & Sage, S. (1998). Problems as possibilities: Problem-based learning for K-12 education. ASCD.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127–147.

Wiggins, G. T., & McTighe, J. (2005). *Understanding by design*. Alexandria, VA: Association for Supervision and Curriculum Development.

Wilensky, U., & Resnick, M. (1999). Thinking in Levels: A Dynamic Systems Perspective to Making Sense of the World. *Journal of Science Education and Technology, 8*(1), 3–19.

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

Wing, J. (2016, March). Computational thinking, 10 years later. Retrieved from https://www.microsoft.com/en-us/research/blog/computational-thinking-10-years-later/

Yoon, S. A., Goh, S. E., & Park, M. (2018). Teaching and learning about complex systems in K-12 science education: A review of empirical studies 1995–2015. *Review of Educational Research, 88*(2), 285–325.