Designing Multimodal Composition Activities for Integrated K-5 Programming and Storytelling

Robert Whyte, School of Education, University of Nottingham, robert.whyte@nottingham.ac.uk
Shaaron E Ainsworth, School of Education, University of Nottingham, shaaron.ainsworth@nottingham.ac.uk
Jane Medwell, School of Education, University of Nottingham, jane.medwell@nottingham.ac.uk

Abstract: A broad interest in embedding computing into K-5 settings has resulted in a proliferation of research on integrated approaches to programming. Although much of these efforts focus on supporting learning in science, engineering, and mathematics (or STEM), less work in non-STEM (e.g. literacy) disciplines has been conducted. This paper examines the potential for integrating computing and literacy education through designing and implementing multimodal composition activities in a visual programming environment, Scratch. Ten primary students took part in an after-school intervention in an inner-city primary school in the midlands of England. Tasks focused on algorithm design, program execution and loops for programming, and representation choice, and structural and visual design for storytelling. Our results demonstrate how a non-STEM context supported the process of integrated programming and storytelling in meaningful ways. We detail our results and their implications for future work.

Introduction

Computing education has been part of the mandatory curriculum in primary schools (K-5) in England since 2013. A report from the Royal Society (2017) found the implementation of the subject in the time since to be "patchy and fragile" (p. 6) with a lack of interest and poor engagement cited as principal reasons for students not pursuing computing beyond secondary school (47% of respondents). With most curricula placing an emphasis on the development of computational thinking (CT) or the understanding of the fundamentals of computation, current theorists have argued that this runs the risk of making computing appear irrelevant to students (Tissenbaum, Sheldon, & Abelson, 2019). Moreover, it has been argued that, within the current 'crowded' curriculum reforms, the goals and values of computing education could be considered too narrowly defined with scant evidence of the use of computing or programming in applied contexts available (Larke, 2019).

Some introductory curricula have proposed multimodal composition activities, primarily in the form of digital storytelling (Burke & Kafai, 2010; Kelleher & Pausch, 2007) as a means to promote programming in K-5 settings and address this issue. Multimodal composition (MMC)—or the process of creating multimodal texts—has gained increased scholarly attention for its motivational (Smith, 2013), communicative (Hull & Katz, 2006), and cultural affordances in the literacy classroom (Mills, 2010). Kelleher and Pausch (2007), for example, noted that storytelling using an adapted version of *Alice* promoted student interest in learning how to program; in other words, students were motivated to program in order to tell stories. While this finding is consistent across the literature base (Burke & Kafai, 2010; Franklin et al., 2013) we note that less attention has been given to the role of MMC through digital storytelling in the contemporary literacy classroom.

Storytelling-based approaches to programming might produce good programs or at least an engaging context for programming yet their place in computing education is contested (Adams & Webster, 2012). Likewise, our understanding of its value as an opportunity to practice storytelling in a multimodal context is less clear. In this paper, we investigate the value of integrating storytelling and programming through MMC tasks by designing an intervention to support learning in both areas. We begin by detailing literature relevant to our intervention design, including both computing and literacy education.

Research context

Storytelling

In recent years, literacy scholars have long examined how our current schema for literacy could be expanded to include other forms of texts, skills, dispositions and practices (Mills, 2010). The advent of new technologies and communicative forms has prompted researchers to articulate and revisit theories on writing and composition, predominantly those that relate to multimodal forms of communication and representation. Though children are increasingly engaged in a wide range of literacy practices in out-of-school contexts, the focus of formal literacy experiences is principally monomodal (i.e. text) (Vincent, 2006). This arguably provides few opportunities for students to connect in-school learning with their interests, knowledge, and experiences from outside of school.

While a strong body of literature exists that suggests MMC promotes greater engagement among reluctant readers and writers (Alvermann, 2008; Smith, 2013), less is known about how MMC could support children's storytelling. Skains (2017), for instance, in her work on the impact of multimodal composition providing an innovative context for storytelling, argues that developing tacit knowledge of the nature of such digital texts and their affordances is necessary to fully realize multimodal composition for narrative construction. She suggests that MMC is situated within a rhetorical context, and that context alters the forms of texts that can be produced. Elsewhere, research on children's storytelling in programming contexts (Franklin et al., 2013; Kelleher & Pausch, 2007) has indicated that narrative-based approaches are suitable for producing good programs. Pantaleo (2013), for instance, noted that visual elements of multimodal design can provide structure and organization to narrative construction though this was limited to comic book design. As MMC encompasses a broad set of practices (e.g. digital storytelling, videomaking, comic design), it is necessary to define what MMC looks like in each context. In the section *Intervention design,* we define how MMC tasks could meaningfully integrate storytelling and programming practices and where they might support one another.

Programming

Given the criticisms of computing education—in particular, computational thinking (CT)—for being narrow and irrelevant for students, it is also necessary to define our approach to CT and how it differs from dominant conceptions. Since its inception, scholars have long debated the scope and nature of CT (National Research Council, 2011) with earlier conceptions proposing a form of 'algorithmic thinking' or representing and solving problems through processes of abstracting and modularizing, being incremental and iterative, and testing and debugging (Brennan and Resnick, 2012) in order to design systems, solve problems, and understand human behavior. More recently, research has considered a broader definition that considers the sociocultural, situated and collaborative dimensions of CT, as well as the diversity of perspectives brought to computing classrooms (Kafai, Proctor, & Lui, 2019). Alongside work by similar theorists in the field (Forte & Guzdial, 2004), we propose CT as a situated practice emphasizing communication over computation—or application over abstraction—as a means to more successfully embed CT in K-5 contexts. This view regards learning as a situated within a particular context, with a particular group, and coordinated by sets of tools and resources that embody particular design principles intended to structure participation in particular ways (Greeno & Engeström, 2014). In this sense, we subscribe to the notion of *computational action* (e.g. Tissenbaum, Sheldon, & Abelson, 2019), emphasizing a situated set of practices designed to encourage authentic and meaning expression through computational tools with an explicit focus on real-world and meaningful problems or learning situations. Instead of prioritizing the 'fundamentals' of programming, students should be encouraged to express themselves by creating personally-meaningful artifacts that draw upon computational concepts and practices. For our designed curriculum unit, we hoped to engage all participants in computational tasks by aligning with this perspective. Moreover, by adopting a situated approach to learning, we are required to define these practices in a given context.

Method

Intervention design

We drew upon Weintrop et al. (2018) and Benton et al. (2018) and other scholars in considering students' early experiences of computing by using relevant concepts in applied contexts. Our aim was to encourage students to use programming features to embed their digital stories with multimodal affordances found in the chosen visual programming tool, Scratch.

To explore how MMC could support programming and storytelling, we found it necessary to consult appropriate literature on the process of creating multimodal texts (Bearne & Wolstencroft, 2007). From there, we looked at tasks relevant to our context. We then considered the task of composing a story (or storytelling) and how this might be augmented through a multimodal context (i.e. visual programming). For instance, as well as deciding on a rhetorical (e.g. narrative) goal, students need to decide on which representations to use, how to structure their text, and to consider which technical features to use for effect. We elaborate on these practices in the *Task-oriented analysis* section below. We then examined what storytelling and programming practices would be most salient in the chosen MMC tasks and consulted literature appropriate to the context (e.g. Meerbaum-Salant, Armoni, & Ben-Ari, 2013; Weintrop, Hansen, Harlow, & Franklin, 2018).

In Table 1 below, we generated a task sequence to structure students' participation in the intervention. The overarching goal was to prepare them for an independent composition task. In total, 6 overarching tasks were planned—5 structured, 1 independent—with mutual learning goals in each strand (see Table 1). The structured tasks (#1-5) are short with specific learning goals demonstrating how programming features (or 'blocks') (e.g. *Initialise sprites and write sequences*) can be used to support children's multimodal stories (e.g. *Define narrative*

goal and decide on representations). The final independent task (#6) draws upon these ideas and asks students to create an original multimodal story by employing some of the computing concepts and practices explored during the structured tasks. In the table, column 1 indicates the task number, column 2 details a description of the task, and columns 3-4 details the task-related goals for both storytelling and programming. In Table 2, we further define subtasks relevant to each task detailed in Table 1 and use these as our analytical codes. Where Table 1 details the MMC tasks implemented across multiple sessions, we also generated subtasks for each MMC task; Table 2 presents the subtasks relevant to Task 6, *Program an original multimodal story*. We conjectured that students would use many of the programming concepts from previous tasks (#1-5) to achieve their narrative goals in the final task (#6).

Table 1: Instructional task sequence

#	MMC tasks	Storytelling	Programming
1	*Decide on representation and content for specific purposes*	Define narrative goal and decide on representations	Initialise sprites and write sequences
2	*Structure texts*	Maintain story cohesion through cohesive devices	Manage execution and coordination through event-based programming
3			
4	*Use technical features for effect*	Use technical features for specific effects (e.g. narrative tension, to engage the audience)	Use programming features to animate sprites and create motion
5			
6	*Program an original multimodal story*	Create a narrative text and employ multimodal features for effect	Use a variety of programming concepts and practices to create a multimodal story, using more sophisticated programming features for narrative effect

Research design

A design-based research approach (Cobb, Confrey, diSessa, Lehrer, & Schauble, 2003) was adopted for this study to design, test and evaluate our proposed intervention design. DBR researchers and theorists argue that domain-specific theories of learning are needed to benefit practitioners with theoretically-inspired designs and applications (Guzdial, 2017). We hoped to add to this research effort by proposing a theory-driven intervention focused on supporting students to create multimodal texts in a visual programming environment. The intervention consisted of six one-hour sessions taking place in an after-school computing club in an inner-city primary school in the midlands of England across a six-week period. The club was open to all elementary students (ages 8-11). 10 students (three girls and seven boys) with little to no formal programming experience volunteered to take part in the study.

Data collection

Throughout the six-week period, a rich set of data was collected including (i) students' project work, including individual plans, and Scratch projects; (ii) audio recordings of interactions between teacher and participants (including classroom discussions and individual interactions between teacher and students) and; (iii) observational field notes generated during and after the intervention. The lead researcher acted as the intervention facilitator, guiding students through the curriculum unit. Data on students' prior computing experience was gathered through interviews with the participating teacher and demonstrated a wide spectrum of experience ranging from daily programming at home to several informal sessions as part of the after-school provision.

Data analysis

We adopted a task-oriented analysis approach to evaluating our intervention design (Dierdorp, Bakker, Eijkelhof, & van Maanen, 2011). In keeping with a situated perspective on learning, this approach conjectures how learning will occur before searching for patterns across students' participation within a context. This approach is similar to creating hypothetical learning trajectories or conjecture mapping, though they are specifically designed to analyse interventions at the task level and how these tasks are supported by different contextual factors. We began by creating conjectures, or predictions, based on how we thought learning would occur and then compared these with the actual intervention data. This allowed us to consider where an intervention was working and what design

features might be supporting this. It also allowed us to address how to improve the task sequence and whether a follow-up study could begin to address this (i.e. through iteration).

To conduct a task-oriented analysis, we first researched programming practices appropriate to an introductory context (Meerbaum-Salant et al., 2013; Weintrop et al., 2018) and considered their potential for application within the context of multimodal storytelling. We then adapted frameworks for multimodal composition practices from previous research (Bearne and Wolstencroft, 2007) to analyze students' multimodal storytelling practices and how they were shaped by the task demands. From these, task conjectures for each overarching MMC task relative to each content area (1: storytelling and 2: programming) were generated. The score beside each conjecture refers to the number of participants who we observed demonstrating this practice in dataset (i.e. 10/10 = 10 out of 10 participants).

Our adaption of the task-oriented analytical approach of Dierdorp et al (2011) differs in two ways: (i) as our intervention is concerned with two strands of learning conjectures—both storytelling and programming respectively—two distinct sets of conjectures are tested (e.g. see Table 2), and; (ii) these are viewed alongside one another to consider the relationship between these two sets of discreet practices. By aligning two discreet task-oriented analyses, we are able to determine whether programming tasks (e.g. use in-program events) and storytelling tasks (e.g. maintain cohesion) mutually support one another or not. We adapted practices from existing research to help us generate observable conjectures for analyzing students' practices. We illustrate where certain findings give evidence (or counter-evidence) to our conjectures and generated cases which illustrate both learning strands. In keeping with a situated perspective on learning, a task was coded as successful to the extent that students independently applied a programming concept or adopted a practice for the production of a multimodal story. If this goal was met—such as using wait commands to correctly sequence narrative events—it was coded as achieved. If the concept or feature was present yet did not fulfil its narrative goal, it was not scored as achieved. We detail examples of students' storytelling and programming practices through the independent task below—and use these to assess how the tasks supported students to program multimodal stories.

Findings

Our findings (summarised below) focus on the independent composition task to illustrate how such MMC tasks engage students in both storytelling and programming practices. We present these findings in three sections. First, we detail two sections evaluating students' multimodal composition as instances of both storytelling and programming. These highlight the merits of the intervention design and where the task sequence support students across both sets of practices. After, we consider students' practices in both learning strands to determine whether tasks mutually support one another and how this is achieved.

Using the conjectures articulated below as an analytical framework, we present illustrative cases from the complete dataset; given the space constraints, we solely focus on the independent composition task in this analysis (see Task 6, *Program an original multimodal story*, in Table 1). Moreover, we consider how the design and implementation of these activities framed the observed practices and eventual trajectories for each participant.

Evaluating multimodal composition as *storytelling*

During the final task, all participants were asked to create an original multimodal text. Each participant generated an initial story plan and discussed with the facilitator the required multimodal elements to program their story. These plans were referred to when programming though modified their plans throughout the intervention. This included the sprites (characters and objects), backdrops (setting), and a brief description of the story genre and synopsis.

In the previous iteration of this intervention (Authors, 2019), we found that students were unable to complete their stories in the given time and we concluded that greater support for story generation was needed. To address this, we allowed greater time to discuss story structure, and demonstrate 'worked examples' or completed story structures. Alongside this increased focus during discussions, students were encouraged to consult favourite books, stories and narratives (i.e. from other media) for ideas. In the end, 6/10 students adapted or adopted elements of their favourite stories taken from popular media—*Harry Potter, Diary of a Wimpy Kid,* and *Fortnite*—and the remaining 4/10 students generated original stories. One result of this increased focus on idea generation and story development—as opposed to a discussion of programming features or strategies—led to most participants (8/10) completing their entire story plan and realising their project goals.

Telling stories multimodally

From the data presented below (see Table 2), we identified four overarching tasks that tasked students with telling a story multimodally. From initial planning, students were asked to decide not only *what* story to tell, but *how* it should be told. This additional demand required support and demonstration of example stories where appropriate

multimodal practices were embodied (e.g. adapting content to suit personal intentions or communicative goal). By the independent stage, our interest centred on students' considered use of multimodal features for communicative effect or to clarify meaning. Given the centrality of the medium (*Scratch*) in delivering the message (a multimodal story), this was observed in specific ways.

All students approached the composition task *multimodally*. In other words, they began by compositing visual elements (e.g. sprites and backdrops) before moving on to text-based elements (e.g. dialogue blocks). They wrote sequences of dialogue to drive their narratives and, once coherent, began to consider how different multimodal effects could enhance their stories. For instance, in Figure 1, one participant, Elijah, used a looping costume change to represent one character walking out of the scene. While this character initially moved out of scene with stationary legs, the author felt it was more visually engaging to animate his legs as well. This desire to 'show, not tell' by animating characters was frequently mentioned in peer discussions; this resulted in 10/10 students using some form of motion block to simulate movement. Moreover, 6/10 participants used loops, like Elijah, to simulate animation or indicate action in their stories.

Students used different structural devices to ensure cohesion in their stories. All 10 participants used a combination of wait blocks, event-based scripts, including broadcast messages, to ensure their project ran sequentially. This allowed students to synchronize different elements of their stories to appear, disappear and speak at the right moment. It also meant for students to organise their stories into discreet *scenes* and to work on each independently. The use of event-based scripts allowed students to coordinate and execute scripts simultaneously, meaning that multiple multimodal features could be combined and employed simultaneously. In all cases, this meant that characters could interact with one another and conversations and dialogue could be formed. However, where this was not correctly executed, and dialogue fell *out of sync,* it meant that stories could not flow cohesively and be comprehensible to others. In our analysis of students' projects, we only found 2/10 cases where this issue presented itself.

Table 2: Task-oriented analysis for Task #6 (Program an original multimodal story)

MMC subtasks	**Conjecture (storytelling)**	**#**	**Conjecture (programming)**	**#**
Decide on representation and content for specific purpose and audience	Define narrative goal	10	Define program goal	10
	Select appropriate representations to express story elements (e.g. images or words for characters or dialogue)	10	Employ one or more backdrops/sprites	10
	Adapt content to suit personal intentions or narrative goal	9	Execute two independent sprites concurrently	10
	Use multimodal features to engage and hold a 'reader's' attention	9	Manipulate elements to personalise characters/objects/setting	9
Structure texts	Integrate and balance representational resources for narrative purposes	10	Use an initialising block (e.g. green flag)	10
	Vary background detail to create changes in setting	10	Use wait block(s) to manage program execution	10
	Use structural devices to ensure cohesion (e.g. when blocks)	9	Define initial sprite state using show/hide block(s)	10
	Use structural devices to organise longer compositions (e.g. broadcast messages)	5	Use broadcast scripts to coordinate multiple processes	5
Use technical features for effect	Illustrate action/movement using multimodal features	10	Use motion-based blocks (e.g. glide) to simulate movement	10
	Use layout and sprite organisation for narrative effect	10	Employ loops to animate sprites	6
	Use technical features to enhance meaning (e.g. costume change)	8	Use switch costume/backdrop block(s)	10
Reflect	Check narrative cohesion	8	Ensure program output is consistent	10
	With support, redesigns text for clarity or cohesion	10	Debugs program errors	9

Other Scratch-based activities similarly served students' narrative goals. We found that 9/10 participants modified or adapted existing multimodal features (e.g. sprites, backdrops and objects) to suit particular communicative goals. A common pattern that emerged from students' final projects is the editing of existing sprites to illustrate additional facial expressions, costumes or catalyst objects. In one notable example, two students worked together to manipulate sprite size to indicate a character's age. As they increased the sprite size, the character grew up. As well as deciding on the particular representation, and its associated attributes, students were able to articulate both which attributes to configure, and why this was appropriate for their stories (i.e. to indicate the passing of time). Students' ability to adopt multimodal features was, unsurprisingly, predicated on their ability to program those features. For instance, loop functions to create animation were more common with older students' project work. This meant that students with less experience of programming relied more on text blocks for expository text than motion blocks or animated loops (i.e. one character says 'let's go over there' as opposed to illustrating it). In this sense, their ability to tell stories multimodally—to show, not tell—and to consider the design constraints and opportunities of different representations, was bound by their programming repertoire.

Evaluating multimodal composition as *programming*

Our analysis of students' activity via screen recordings, alongside their final project work, formed the basis for our assessment of MMC tasks as vehicles for introductory programming. We found that students employed many of the basic programming concepts introduced in the structured tasks (see Table 1) during the independent composition task. These include program execution, event-based programming, and loop functions to simulate movement.

Program execution

We conjectured that students—through iterative testing—would ensure their programs could be executed multiple times with the same outcome; in other words, that students would ensure each sprite would *reset* to its original position/size/attributes. This emerged from a discussion on the need to use motion blocks to *set* each story element at the beginning of the program. As students were encouraged to routinely 'test' their stories for coherence, they used motion blocks (e.g. 'go to X: Y') and looks blocks (e.g. set sprite size to 100%) to reset their projects for repeat execution. This was necessary as, through multiple program executions, students were able to see how an outsider would be unable to know the particulars of the original program state (i.e. where sprites should begin on screen). One unanticipated observation we noted concerned some students executing scripts using specific key inputs. An unintended consequence for an outsider meant their ability to execute the program correctly was limited to their understanding of the running order of the text.

Event-based programming

Given the temporal nature of students' stories, where certain multimodal features are required at a certain time, program coordination was crucial to telling their stories. To achieve this, students adopted event-based programming features to structure and coordinate multimodal features. These took two different forms: (i) 10/10 students used event blocks (e.g. in-program events) to coordinate their story structure and, less commonly; (ii) 5/10 students used broadcast features (e.g. in-program messages) to coordinate parallel scripts. Moreover, 10/10 students also used intra- and inter-script delays (wait blocks) to ensure all scripts were executed at the appropriate time. While most students' programs could be executed automatically, 2/10 featured scripts bound to specific key inputs (e.g. 'when x is pressed') requiring the eventual reader to know the exact key sequence to run the story. Though this became a general point of discussion for the participant group, it was not corrected in the final projects.

Loop functions

The multimodal dimensions of students' stories similarly framed *how* and *what* students programmed. During the independent composition task, students were asked to consider these dimensions particularly as they relate to which programming features they will adopt. When translating their initial story plans from paper to screen, we sought to encourage greater use of multimodal forms of communication as, in turn, this contributed to more sophisticated program structures. In one example, we found that 6/10 students used loops to illustrate motion in their stories. This was usually coupled with costume change blocks to further simulate motion or change within a character. For instance, one participant, Eric, used loops to automate characters 'shrinking' into the distance (see Figure 1). In another example, this effect was used by two participants to simulate a character growing up before being shrunk down by a wizard. In both cases, this led to a discussion on the benefits of *showing* a certain story

element over *telling* it in the same instance. We also recognised that some students (4/10) chose not to use such an effect (or were unable to do so) and instead opted for textual representations.

Figure 1. Example use of loops to illustrate motion/animation.

Discussion

The goal of this study has been to demonstrate how programming and storytelling could be meaningfully integrated for mutual benefit. In general, we found that most participants could achieve their narrative goals and how such activities could provide opportunities for students to reflect on specific programming or storytelling features in their texts. Like previous theorists (Tissenbaum et al., 2019), we argue that the goal of computing education should focus on the application of computational ideas in the development of authentic products. In creating multimodal texts, students were encouraged to consider how loops, parallelism and event-based scripts could be used to create multimodal effects and apply this to tell more engaging visual stories. Likewise, visual programming provided students with a rich set of representational resources to consider both what story to tell and how to do so. Integrating programming and storytelling also demonstrated where complementary learning goals support one another at different points. For instance, using event-based (e.g. initialising) blocks functioned as a means to structure their stories episodically. That is, students were able to create and revise each section of their story separately. Moreover, multimodal effects (e.g. motion) required students to consider the necessary programming features and affordances of such an effect to tell more visually engaging stories.

Beyond our empirical findings, we demonstrated a novel analytical approach to evaluate how learning is supported in design-based experiments. Using Dierdorp et al. (2011) as a framework, we adopted and extended the task-oriented analysis to give better insights into the affordances of integrated approaches to both programming and storytelling. By combining task-oriented analyses for both learning strands, we were able to consider when and how programming tasks (e.g. use in-program events) and storytelling tasks (e.g. maintain cohesion) mutually support one another. There are some caveats to these insights. While we suggest areas where programming and storytelling can be meaningfully integrated, we also note some limitations of adopting MMC to do this. In line with previous research (Adams & Webster, 2012), MMC tasks were not an appropriate context for more complex programming instruction (e.g. limited use of loop functions or variables) nor the explicit development of writing skills (e.g. lack of dedicated text editing tools). However, we are encouraged by our results as they suggest that by situating the design and programming of digital artifacts in interest-driven activities supported personal expression and engagement within a programming context.

Our next step in this research project involves working with teachers to adapt the curriculum unit to fit the needs of local contexts. Whereas our first two cycles of design research explored the design of learning activities to support integrating programming and storytelling at primary level, our focus shifts to the implementation of the curriculum unit by in-service teachers and how their perceptions on pedagogy, programming as a curriculum activity, and the role of multimodality in the literacy classroom could influence the co-design and implementation of these activities. It is hoped this direction will stimulate further research concerning the value of multimodal composition activities in developing storytelling for the literacy classroom and programming to the computing classroom.

References

Adams, J. C., & Webster, A. R. (2012). What do students learn about programming from game, music video, and storytelling projects? *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education - SIGCSE '12*, 643.

Alvermann, D. E. (2008). Why Bother Theorizing Adolescents' Online Literacies for Classroom Practice and Research? *Journal of Adolescent & Adult Literacy*, *52*, 8–19.

Bearne, E., & Wolstencroft, H. (2007). *Visual Approaches to Teaching Writing: Multimodal Literacy 5-11*.

Thousand Oaks, CA: Sage Publications.
Benton, L., Saunders, P., Kalas, I., Hoyles, C., & Noss, R. (2018). Designing for learning mathematics through programming: A case study of pupils engaging with place value. *International Journal of Child-Computer Interaction*, *16*, 68–76.
Burke, Q., & Kafai, Y. B. (2010). Programming & Storytelling: Opportunities for Learning About Coding & Composition. *Proceedings of the 9th International Conference on Interaction Design and Children - IDC '10*, 348. New York, New York, USA: ACM Press.
Cobb, P., Confrey, J., diSessa, A., Lehrer, R., & Schauble, L. (2003). Design Experiments in Educational Research. *Educational Researcher*, *32*, 9–13.
Dierdorp, A., Bakker, A., Eijkelhof, H., & van Maanen, J. (2011). Authentic practices as contexts for learning to draw inferences beyond correlated data. *Mathematical Thinking and Learning*, *13*, 132–151.
Forte, A., & Guzdial, M. (2004). Computers for Communication, Not Calculation: Media as a Motivation and Context for Learning Andrea Forte. *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04)*, *4*, 40096.1.
Franklin, D., Conrad, P., Boe, B., Nilsen, K., Hill, C., Len, M., … Waite, R. (2013). Assessment of computer science learning in a scratch-based outreach program. *SIGCSE 2013 - Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, 371–376. New York, New York, USA: ACM Press.
Greeno, J. G., & Engeström, Y. (2014). Learning in Activity. In R. K. Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences* (pp. 128–148). Cambridge: Cambridge University Press.
Guzdial, M. (2017). Using Learning Sciences Research To Improve Computing Teaching: Predictions, Subgoals, And Parsons. *Computing At School 9th Conference for Teachers*, 1–30. Birmingham, UK.
Hull, G. A., & Katz, M.-L. (2006). Crafting an Agentive Self: Case Studies of Digital Storytelling. *Source: Research in the Teaching of English*, *41*, 43–81.
Kafai, Y., Proctor, C., & Lui, D. (2019). From Theory Bias to Theory Dialogue. *Proceedings of the 2019 ACM Conference on International Computing Education Research - ICER '19*, 101–109. New York, New York, USA: ACM Press.
Kelleher, C., & Pausch, R. (2007). Using storytelling to motivate programming. *Communications of the ACM*, *50*, 58.
Larke, L. R. (2019). Agentic neglect: Teachers as gatekeepers of England's national computing curriculum. *British Journal of Educational Technology*, *50*, 1137–1150.
Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (Moti). (2013). Learning computer science concepts with Scratch. *Computer Science Education*, *23*, 239–264.
Mills, K. A. (2010). A Review of the "Digital Turn" in the New Literacy Studies. *Review of Educational Research*, *80*, 246–271.
National Research Council. (2011). Report of a Workshop on the Pedagogical Aspects of Computational Thinking. *Report of a Workshop on the Pedagogical Aspects of Computational Thinking*, 176.
Pantaleo, S. (2013). Paneling "Matters" in Elementary Students' Graphic Narratives. *Literacy Research and Instruction*, *52*, 150–171.
Skains, R. L. (2017). The Adaptive Process of Multimodal Composition: How Developing Tacit Knowledge of Digital Tools Affects Creative Writing. *Computers and Composition*, *43*, 106–117.
Smith, B. E. (2013). Beyond Words. In *Exploring Multimodal Composition and Digital Writing* (pp. 1–19).
The Royal Society. (2017). *After the reboot: computing education in UK schools*. London: The Royal Society.
Tissenbaum, M., Sheldon, J., & Abelson, H. (2019). From computational thinking to computational action. *Communications of the ACM*, *62*, 34–36.
Vincent, J. (2006). Children writing: Multimodality and assessment in the writing classroom. *Literacy*, *40*, 51–57.
Weintrop, D., Hansen, A. K., Harlow, D. B., & Franklin, D. (2018). Starting from Scratch: Outcomes of Early Computer Science Learning Experiences and Implications for What Comes Next. *Proceedings of the 2018 ACM Conference on International Computing Education Research - ICER '18*, 142–150. New York, New York, USA: ACM Press.