Trade-offs in Using Mobile Tools to Promote Action with Socioscientific Issues

Susan A. Yoon, Jooeun Shim, and Noora Noushad,
yoonsa@upenn.edu, jshim@gse.upenn.edu, noora@gse.upenn.edu
University of Pennsylvania

Abstract: In this study, we build on an emerging trend in socioscientific issues (SSI) education to support action through the use of personally relevant digital tools. We investigate the design of curriculum that integrates SSIs with the programming and construction of mobile apps. Through a series of design iterations, we highlight important tradeoffs in design choices that can potentially impact the depth of students' learning of SSIs and how students take action. These considerations include the sequencing of programming versus SSI instruction, enabling or restricting student choice of SSI topics, mandating collaboration on app development, and emphases on packaged computational components versus computational concepts. We conclude with several design suggestions to maximize efforts to promote action through app construction and SSI education.

Introduction

When promoting scientific action among students, it is important to provide context for knowledge (Buxton, 2010), motivation to learn the science content (Skamp et al., 2004), and tools to transfer the knowledge into action (McNeill & Vaughn, 2012). However, it can be difficult to design learning environments that engage learners in authentic investigation of science content while creating motivating tasks that will encourage action without compromising learning goals (Alchin et al., 2014). In this study we tackle the challenge of providing a context, motivation for learning, and means to transfer knowledge into action among middle school students by creating an environment that integrates socioscientific issues (SSIs) with the construction of mobile apps. Here, we provide a lens to understand educational challenges by describing the tradeoffs that we encountered in designing such a learning environment. Previous research has suggested that designers need to discuss the affordances and constraints unique to design iterations so that educators and curriculum designers can be informed about important changes and rationales for these changes that take place (Peppler et al., 2016).

Conceptual framework

We draw on three areas of scholarship to support the study goals–the use of SSIs in science education (Barton & Tan, 2008), mobile learning (Price et al., 2014; Sharples & Pea, 2014), and the programming of digital learning tools (Resnik et al., 2009; Werner, Denner & Campe, 2012).

Socioscientific issues

Teaching science through SSIs represents what we know best about how people learn. It makes science personally relevant (Karahan & Roehrig, 2015); enables the use and practice of domain specific skills such as scientific reasoning and argumentation (Sadler, 2004); and enables awareness of complex scientific issues that impact social and environmental conditions from multiple perspectives (Burek & Zeidler, 2015). The notion of action has also been highlighted as an important goal for SSI instruction (Lee, 2015). This moves students out of the place of "arm chair critic" (Hodson, 2003) to a place where they can work to improve the communities they live in. However, action is challenging to enact as students are not exposed to ways in which they can apply their reasoning skills to real world activity and are given limited opportunities to demonstrate their content knowledge beyond capstone presentations and classroom debates (McNeill & Vaughn, 2012). A recent content analysis of 122 SSI-themed studies in the top 5 science education journals found that less than 2% were focused on engaging in citizenship work which can be arguably interpreted as an action orientation (Tekin et al., 2016). To address this issue, we build on an emerging trend in SSI education that enables students to create personally relevant tools (Karahan & Roehrig, 2015) to provide them with mechanisms through which action can be taken.

Mobile learning

The recent trends in mobile learning have also revealed mechanisms for students to take action. The unique affordances of mobile learning platforms such as the ability to move to different locations along with technology-enabled affordances like location-aware sensors have been leveraged to engage learners in contextualized learning activities (McQueen et al., 2012; Sharples & Pea, 2014). Consequently, research in using mobile apps to support learning has shown promising results in promoting student engagement and motivation in STEM (Grover & Pea,

2013; Ni et al., 2016; Price et al., 2014). However, many mobile learning initiatives have tended to put the learner at the user end of mobile app engagement rather than allowing learners to construct the apps themselves (Kearney et al., 2012). Under this learning scenario, only the designers have real agency to create functions for specific purposes whereas users have little agency over function or purpose (Burrell, 2016). A specific goal of this study was to engage students in designing mobile apps that could serve a chosen function for a chosen purpose with the hope of encouraging student agency, and in turn, student action.

Programming digital artifacts

Relatedly, designing and then developing the mobile app requires programming ability. Programming can help to develop confidence in learners to deal with complex phenomena, work through challenging problems, and promote the setting and achieving of goals (Barr & Stephenson, 2011). Furthermore, creating digital artifacts has been shown to develop learners' reasoning skills while simultaneously embedding knowledge in relevant cultural and personal activity (Resnik et al., 2009; Werner, et al., 2012). With the advent of blocks-based programming platforms such as Scratch (Resnick et al., 2009), novice programmers are supported with computational logic that is built into the programming environment thereby eliminating frustrating syntax errors. This has, in part, influenced the marked increase in interest in helping all students become programming literate, e.g., *Hour of Code*; *Girls Who Code*, over the last several years. Yet, a major challenge exists in terms of finding authentic ways to integrate computer science with more mainstream subjects such as Science and Math in order to contextualize and make computer programming more relevant (Sengupta et al., 2013).

Solving local socioscientific problems by constructing mobile apps

In this study, we aim to address the challenges in the aforementioned literature through the design and construction of mobile apps that investigate a local socioscientific issue that has import in the students' local environment. We use the mobile learning platform called App Inventor to support these local investigations. The App Inventor platform uses a graphical programming language similar to Scratch in which computational procedures are built into easily assembled blocks (http://appinventor.mit.edu/explore/front.html). Grover and Pea (2013) highlight several benefits of using tools like App Inventor to promote computational thinking, interest, and access. They write that such tools are underpinned by the principle of "low floor, high ceiling," which means that the environment has a low threshold for learning the initial programming language but embeds opportunities for more advanced computational investigations.

The broad goal of the project was to understand the extent to which building mobile apps with an SSI focus could motivate students toward scientific action with content specifically anchored in science. Through two design iterations, we found several important trade-offs to consider in the design of curricular activities that appeared to have an impact on student learning and participation outcomes. In this paper, we first describe the design of our curricular activities in the two design iterations that encompassed a spring and a fall elective class with 7th grade students. We discuss changes that were made in each class's design in order to improve student learning and participation outcomes and we describe the tradeoffs that similarly-minded designers should consider when developing learning programs with these educational goals.

Methods

Design and intervention

In this exploratory study, we use a design-based research (DBR) methodology. DBR studies require interventions to run through cycles of conceptualization, design, implementation, evaluation, and redesign until results show promising outcomes in learning measures (Puntambekar & Sandoval, 2009). Both elective courses ran twice a week for 45 minutes each day over a 12-week cycle. The curriculum was delivered in 3 blocks. In the first iteration, block 1 focused on helping students learn the App Inventor programming language. Students were given tours of code, asked to create mini-apps such as how to make sounds and how to create an action by shaking the mobile tablet, and introduced to app cards that taught students more nuanced programming functions. During block 2, students explored the meaning of socioscientific issues first by learning about global challenges such as climate change, hurricanes, and the overabundance of garbage. They were then asked to think about their local community and issues related to science that they could examine. They brainstormed issues that they wanted to solve or that they could relate to, and that could be amenable to integration in an app. Next, in teams of two, students built paper prototypes and constructed their apps. Finally, in block 3, students tested and revised their apps. Based on results from the first iteration, several design changes were made in the second iteration. First, the topic of SSIs was presented before any App Inventor programming instruction. We differentiated roles between teams of

students to focus either on the programming or on the science. And we limited the number of programming ideas that students could use in the construction of the app.

Participants

We recruited 25 seventh-grade students who chose to participate in our study as an elective course in the spring and fall of 2016 from a public school located in a large urban school district in the north east part of the United States. In the spring semester class, 13 students (6 female and 7 males) participated and in the fall semester class, 12 students (4 females and 8 males) participate. Across the two groups, 7 students identified as White, 7 students identified as mixed race, 5 students identified as African American, 3 students identified as Asian, and 3 students identified as Other.

Data sources and analyses

Data collected in the study included a pre- and post-intervention survey with questions that asked about students' knowledge of socioscientific issues and programming, and interests in the application of science and technology. The survey included 10 Likert-scale questions with open-ended questions added for students to explain their ratings. Questions included:

- *Do you think science is useful in your everyday life?*
- *Do you think learning science helps you to take action in solving problems in your community?*
- *Do you think learning how to make apps helps you to take action in the community?*
- *Do out think you will use the mobile app you have developed?*

Students were also given a 20-minute post-intervention interview that probed their ideas related to the research goals as well as students opinions about how the class was structured. There were 13 semi-structured questions with multiple subquestions. Interview questions included:

- *Tell me about your experience building your final app.*
  - *What does your app do? What motivated you to choose the topic of your final app?*
- *Can you tell me what socioscientific issues are?*
  - *Is the problem that your app solves a socioscientific issue?*
  - *Has building the app helped you see why these issues are important in your life? How?*
- *What would you have changed in the way the class was taught?*

In addition, the 14 apps that were constructed in teams over the two iterations were analyzed to understand what students had created, what they had learned from the programming activities, and how they applied it in their project. Observation field notes were also kept throughout the course. These observations were focused on understanding the extent to which students were engaged and experiencing challenges both in content and interpersonal interactions. Since the methodology was exploratory and design based in nature, all data sources for this study were mined qualitatively and discussed among the project team.

Results

In general, we found that students were engaged in both iterations. The majority of students were able to produce a working app but we found that what kind of app they produced and the apps ability to function in terms of our research goals varied between iterations. We also found that student interests and knowledge of SSIs and programming were different. In Table 1, we list the apps that students constructed with a brief description of what the apps were constructed to do. We follow the table with a detailed discussion of the design challenges and associated tradeoffs that are hypothesized to have impact on the study goals.

Table 1: App project descriptions (1-team from first iteration; 2- team from second iteration)

Team	App project description
1-1	To reduce the chance of being bullied, the app gives a diet plan along with links to workout videos.
1-2	This app allows users to view the breakfast and lunch menu served at school and displays the nutritional value of selected items.
1-3	To reduce distraction while doing homework, this app times how long the user has stayed focused.
1-4	To help reduce electricity consumption, the user can record the time spent on various appliances.
1-5	To help students with their imagination, the app allows users to summon various mythical creatures.

1-6	To increase awareness about air/water pollutants, the app provides various dangers of air/water pollution with animated videos.
1-7	This app is a game that sorts' random trash correctly into 'recycle', 'compost', and 'trash' bins.
2-1	This app shows where users can leave and take plastic bags and allows users to add their own locations.
2-2	To make recycling fun, the app provides a recycling game. Users can share scores and complete action goals (e.g., recycle 15 pieces of trash in a public place) to gain higher scores.
2-3	This app helps to identify high carbon footprint foods with the quiz.
2-4	This app educates users by presenting examples of ways to lower one's carbon footprint with a quiz.
2-5	This app provides access to a link that rates products based on their carbon footprint. They can play a game in different characters, and share the app with friends.
2-6	To help users' make better food choices at local restaurants, the app provides a quiz as well as records shareable food comments with photos.
2-7	This app gives a quiz to inform people around the world what foods have a low carbon footprint.

Programming or socioscientific issues first?

Introducing programming to students in the first block before instruction about SSIs proved to be very motivating to students, however, this happened at the expense of students' learning what SSIs were and what constituted local SSIs that they could help to improve. Despite continual redirection away from tinkering with the code, students remained disengaged from the science content in the second block where instruction of SSIs occurred. This resulted in student teams choosing to make apps that had little to no local SSI content. For example, one team created *The Distraction App* (1-3 Table 1) that monitored how much time they spent surfing the internet rather than doing their homework. If they managed to stay on task for a period of time, an emoji appeared to congratulate their effort. Another team created an app that would enhance imagination through an exploration of unicorns and griffins–*The Magical Creatures App* (1-5 Table 1). While there appeared to be some personal relevance embedded in these artifacts, there was little that could be said to encourage action in a local SSI. Furthermore, in response to the survey question, "what are socioscientific issues?" only four students out of 13 were able to provide a coherent and accurate definition that included knowledge of SSIs as social issues with scientific content and that SSIs could be found in local contexts and addressed through local activity. Nevertheless, in their interviews, the majority of students said that their interest in science and technology improved from participating in the course.

Based on these results, we wanted to see if we could improve on student understanding of SSIs by manipulating the design of the course. Rather than beginning with the App Inventor programming activities, in the second iteration, we started with a lengthy investigation of SSIs. This enabled a more in-depth investigation of the SSI content. We also limited the topic to investigating one's carbon foot print in the local environment so that we could scaffold instruction with essential scientific information rather than having to address content in idiosyncratic student-chosen topics. In the end, all artifacts addressed issues of environmental impact which were social, scientific in nature, and included some action that could have local impact. For example, one group designed an app that identified high or low carbon packaging used by local restaurants (2-6 Table 1). This information could be crowd sourced, shared, and used by community members to make choices about where to eat. Survey and interview responses from students in the second iteration also showed a deeper understanding of the science content and the relationship between science and human activity and the environment. On the topic of how foods exhibit a carbon impact, one student commented, "I didn't know cottage cheese had carbon dioxide, like just pure CO_2. I knew like fizzy drinks had [it] cause that is what they are…carbonated drinks, but I didn't know cottage cheese had [it]. I think it is a shelf-life thing. A lot of companies do that…they are more worried about the products than [the] environment."

However, with the delayed introduction to the App Inventor programming activities, students in the second iteration discussed challenges to this design change. In response to the question about what they would change about how the class was taught, one student said, "Listening to more about programming rather than listening to how climate change is affecting our world." Another student said that, "the beginning wasn't really that fun." This sentiment was captured as well in the observation notes during those classes focused on delivering the SSI content such as the following comment, "Greg asked me three times now, when they would start making and programming games (Day 4)". However, despite students' obvious interest in programming over learning about SSIs, they were able to reflect on how their actions and daily choices could make a difference for the environment. In his interview one student said:

> *I learned that throwing away plastic bags has an effect on the environment, because like when you throw away plastic bags they can go to a landfill and they all just sit there and it takes them*

> *years to go away so it's not good for the environment and if they are burned they can produce a lot of like you know toxic chemicals that can pollute the air. So like my family we have this bin where we keep all the plastic bags in and I have always wondered like why my mom doesn't just throw away the plastic bags and now I see why she does that.* (Post-interview, ID 7, Marco)

Interestingly, in the pre-intervention surveys, this student said that he couldn't readily see what he was learning in school science as all that applicable to his everyday life. The data reported here provide evidence of students' preferences for app programming over learning the science content. It is a useful tradeoff to consider especially if time is limited, or when considering what the content goals are.

Differentiating collaboration roles

Collaboration was emphasized throughout both courses. However, how teams organized their work differed across iterations. During the first iteration, students were given the freedom to decide how to distribute project tasks between themselves. We thought that by allowing students to direct their app design with little instructor monitoring, this would create a less intimidating environment for the teams so that they could choose where to apply their respective strengths and interests. However, this happened at the expense of both members equally engaging critically with programming or with SSI content. With little exception there was one member of the team who did all the programming. For example, the *PAS* app (1-2 Table 1) was mainly constructed by Pete. His partner Craig did not do any programming until the fifth week of the class and only did so because Pete was absent. In their respective interviews, Pete and Craig offered differing evaluations of their contributions. When asked what they saw were the benefits of working on a collaborative team, Pete said this exit interview:

> *I think there are pros and cons of both [working along or working together]. One of the cons [you face] is you could get a teammate who doesn't really do much but there is nothing you can really do about that. Pros is if you have someone that is working then it can be a lot less stressful as you both contribute to a big project as you split the work 50/50.*

Craig said that he preferred "working in teams, so that one person wouldn't have to do all the work and it would be equally divided." He described his contribution in this way, "I had figured out the nutritional value and [Pete] had programmed the screens linking it to the menu." From observation notes, we saw that while Pete was working hard each week, Craig spent most of the time socializing and distracting others in the class.

In the second iteration we decided to formalize the contributions in such a way that the work could be perceived as more equally distributed. With the added emphasis on SSIs, we instituted roles that different members of the team could take charge of. These roles were the science driver or the coding driver. We also introduced a pair-programming rule such that the team members switched every half hour between building the app and researching the science materials. Indeed, in all the exit interviews, students said that they felt there was equal participation among team members. However, despite continued effort in the instruction to insist that every member took a turn to program, for the most part, the team members remained in their respective roles throughout the course with little switching. This led to a clear disadvantage for those students who could become more adept at programming with a little urging. For example, in his exit interview, Emmet said, "Knowledge expert, yeah I'm clever, I wouldn't say I'm smart but I'm clever, I know how to get around things. Well, so I like knowledge more than I like [programming]...[if] I find something I know I'll never get the hang of, I'd rather someone else do it…trying to do a Scratch project once...woh". He also mentioned that he signed up for the course because all of his friends were talking about computer science and he wanted to improve his programming skills. However, as he admitted in his interview that he was more comfortable with science content, given the choice, he decided to remain in his comfort zone rather than challenging himself in an area of lesser strength and interest.

Similarly for those students with a clear interest in the programming side, deeper level exploration of the scientific content was given short-shrift. Mike said this about the science content in his exit interview:

> *Well there were things that I didn't really understand and honestly didn't go out of my way to understand. I just felt like I said I didn't enjoy it enough to use it at home. So there were some stuff I didn't understand, I probably could figure it out but I think there are more people more knowledgeable than me in that area.*

James, who was Mike's partner, thought that the delineation in roles was a good design choice because as he discussed, "one of us had to be working, we couldn't work on the same thing at the same time…um but we got twice as much done."

In these data, we see an instructional issue with no clear-cut rationale in terms of whether more or less control over how the collaboration tasks is better or will benefit participation and learning outcomes. In the first iteration there was a perception (with good reason) of an unequal distribution of work. Correcting for this in the second iteration where we defined roles with suggestions to switch, this led to, for some students, preemptively limiting the depth of understanding that could have been experienced in either of the two content areas of science and computer programming.

Computational components versus computational concepts

The graphical programming language in App Inventor was used as a catalyst for students to transform their scientific knowledge into actionable artifacts, i.e., mobile apps that helped to address local SSIs in both iterations of the course. However, how programming instruction was delivered differed across iterations. For clarity, we differentiate between computational components and computational concepts. In this paper we define computational components as the various features of the App Inventor interface such as embedding videos or programming data collection using sensors; and computational concepts as more universally established knowledge of computer programming such as variables, procedures, conditionals, and loops. Given that many components are prepackaged for App Inventor, in the first iteration, we made the design decision to teach about the different possible components that could be programmed into the app, for example the camera or voice recording components. We hoped that understanding App Inventor functionality would trigger ideas for app construction on an SSI topic. We found that the apps constructed in the first course varied in both functionality and the usage of App Inventor features. However, this happened at the expense of students critically engaging with the computational concepts themselves. The analysis of the programming showed that students only engaged with computational concepts at a superficial level. For example, the *PAS* app (1-2 Table 1) used eight types of components to create their mobile solution. In the four screens that they programmed, they used the components of *Activitystarter, Accelerometer, Sharing, Buttons, Table Arrangement, Label, Textbox,* and *Canvas*. However, we can see in Figure 1, that no significant computational concepts were used in the app. This figure shows that in each of the five event handler blocks, they did not use variable values, conditional operations, or procedures for code organization. The blocks were simply used to open other screens or to link to a web address.

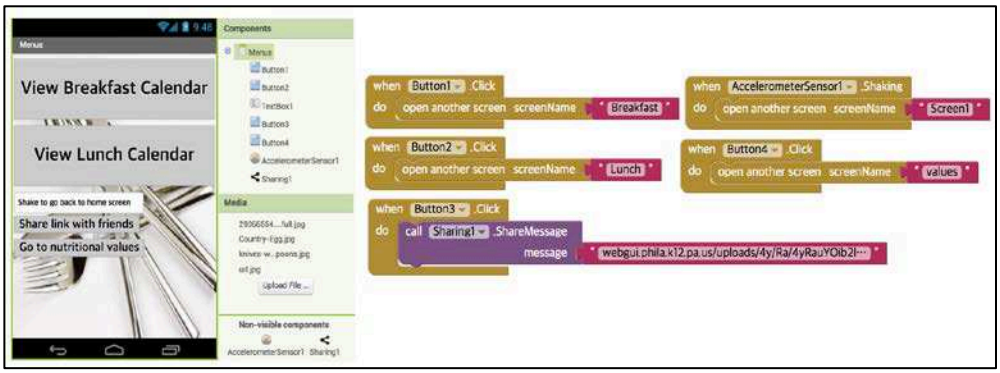

Figure 1. Sample programming logic from a first iteration mobile app.

To address this issue, in the second iteration, we constructed four mini-apps based on common types of app activities (i.e., a quiz app, a game app, a memo app, and a drawing app). Through these mini-apps, we modelled how computational concepts could be applied through various App Inventor features. For example, students were asked to practice making a different quiz on a blank screen by cutting and pasting code. Learning first principles of computer programming in addition to the affordance of remixing the mini-apps resulted in the majority of apps in the second iteration showing applications of computational concepts less complex notions (e.g., variables and lists) to more sophisticated ideas (e.g. conditionals, procedures, and databases). For example, Figure 2 shows the *Paper Toss Race* app (2-2 Table 1) in which the computational concepts of variables, lists, procedures, conditions, Boolean logic, data, and sharing created the various functionalities of the app such as the selection of game goals, a point system, and adding items to and picking items from a list in addition to 18 types of App Inventor components.

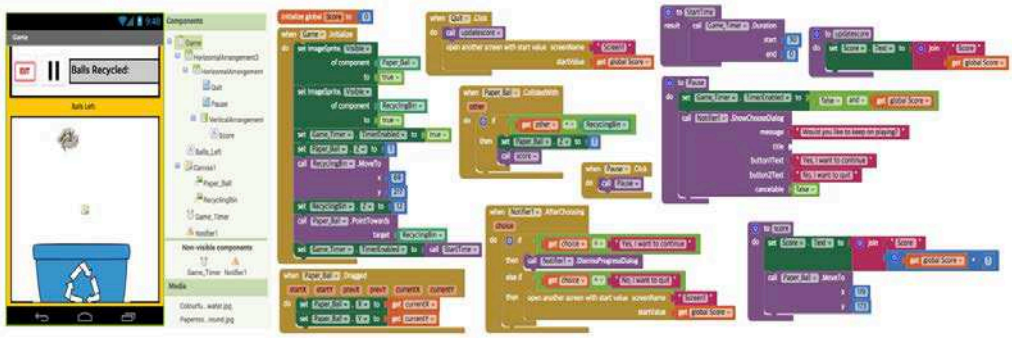

Figure 2. Sample programming logic from a second iteration mobile app.

One trade-off to note here however is a lack of diversity of the kinds of apps constructed. Most apps were modified versions of the apps that were modelled for them. For example, five out seven apps included remixed versions of the Low Carbon footprint quiz feature.

Discussion

Capitalizing on the growing trend in using artifacts to promote action in SSI education, our goal was to investigate design iterations of curricula that combined instruction in SSIs with constructing mobile apps to encourage local action. We analyzed student data from both iterations to look for positive and negative design features and we found that there were a number of trade-offs in each case.

In this paper, we outlined three primary trade-offs. First, we found that with a curricular model of programming first and SSI instruction second with complete student choice of SSI topic, this led to increased student engagement but decreased knowledge of the science. On the one hand, the programming first model establishes an environment where making is fun (Peppler et al., 2016) and allows for students to engage in topics that are most proximal to their interest. On the other hand, the apps that were created in the first iteration were arguably devoid of real local action in the community. Where the focus was on SSI instruction first to strengthen their science content, students exhibited less interest. This creates a conundrum for instruction as students with heightened awareness of SSI but with decreased interest are less likely to transfer the knowledge into action (Burek & Zielder, 2015).

Instituting collaboration rules also presented challenges (Werner et al., 2012). Allowing student collaborative choices to emerge in place of enforcing them through instruction resulted in unequal distribution and ownership of work among group members. In the second iteration, an environment for co-creativity emerged as students distributed the task of researching the science content versus programming the applications among the various members of the group (Lubatkin, 2001). However, while it was clear that there was more equal distribution of work, students more inclined to focus on one or the other of assigned tasks did so, which prematurely limited opportunities to learn new content and skills.

The last trade-off pertaining to how programming instruction was delivered demonstrated that when students were taught app components first without detailed description of the computational concepts embedded in the components, students showed relatively weak application of computational concepts which corroborates previous research findings (Grover & Basu, 2017). Conversely, when students were given pre-programmed model mobile mini-apps to remix, students demonstrated greater sophistication of computational concepts in their code. This might be an obvious win for the latter design choice but we also saw that the second set of apps showed much less diversity in functionality and purpose.

We highlight these design trade-offs to illustrate curricular features that will impact the desired goal of enacting action in the world through applied scientific content. We can see this information as potentially valuable for similarly minded learning scientists who may need to *a priori* establish which aspect of this action will take primacy. This is important because one consideration to note is that this instruction occurred in a course with a finite limit of about 18 hours of instruction. Overall, these trade-offs may fundamentally come down to how much choice students are given. In our study, we found that with more choice, there was greater interest but less content or skill development; and with less choice for the most part, the opposite was true.

References

Allchin, D., Andersen, H. M., & Nielsen, K. (2014). Complementary approaches to teaching nature of science: integrating student inquiry, historical cases, and contemporary cases in classroom practice. *Science Education, 98*(3), 461-486.

Barr, V., & Stephenson, C. (2011). Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community? *ACM Inroads, 2*(1), 48–54.
Barton, A. C., & Tan, E. (2008). Funds of knowledge and discourses and hybrid space. *Journal of Research in Science Teaching, 46*(1), 50–73.
Burek, K., & Zeidler, D. L. (2015). Seeing the forest for the trees! Conservation and activism through socioscientific issues. *In EcoJustice, citizen science and youth activism* (pp. 425-441). Springer International Publishing.
Burrell, J. (2016). Material Ecosystems: Theorizing (Digital) Technologies in Socioeconomic Development. *Information Technologies and International Development, 12*(1), 1–13.
Buxton, C. A. (2010). Social Problem Solving Through Science: An Approach to Critical, Place-Based, *Science Teaching and Learning. Equity & Excellence in Education, 43*(1), 120–135.
Grover, S., & Pea, R. (2013). Computational Thinking in K–12 A Review of the State of the Field. *Educational Researcher*, *42*(1), 38-43.
Hodson, D. (2003). Time for action: Science education for an alternative future. *International Journal of Science Education, 25*(6), 645-670.
Karahan, E., & Roehrig, G. (2015). Constructing media artifacts in a social constructivist environment to enhance students' environmental awareness and activism. *Journal of Science Education and Technology, 24(*1), 103-118
Kearney, M., Schuck, S., Burden, K., & Aubusson, P. (2012). Viewing mobile learning from a pedagogical perspective. *Research in Learning Technology, 20*(1).
Lee, Y. (2015). Learning activism, acting with phronesis. *Cultural Studies of Science Education*, 10, 1183–1188.
Lubatkin, M., Florin, J., & Lane, P. (2001). Learning together and apart: A model of reciprocal interfirm learning. *Human Relations, 54*(10), 1353-1382.
McNeill, K. L., & Vaughn, M. H. (2012). Urban high school students' critical science agency: Conceptual understandings and environmental actions around climate change. *Research in science education, 42(*2), 373-399.
McQueen, J., Wright, J. J., & Fox, J. A. (2012). Design and implementation of a genomics field trip program aimed at secondary school students. *PLoS Computational Biology, 8*(8), e1002636.
Ni, L., Sherman, M., Schilder, D., & Martin, F. (2016, February). Computing with a Community Focus: An App Inventor Summer Camp for Middle School Students. *In Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (pp. 690-690). ACM.
Peppler, K., Halverson, E., & Kafai, Y. B. (Eds.). (2016). *Makeology: Makerspaces as learning environments* (Vol. 1). Routledge.
Price, S., Davies, P., Farr, W., Jewitt, C., Roussos, G., & Sin, G. (2014). Fostering geospatial thinking in science education through a customisable smartphone application. *British Journal of Educational Technology, 45*(1), 160–170.
Puntambekar, S., & Sandoval, W. (2009). Design research: Moving forward. *Journal of the Learning Sciences, 18*, 323–326.
Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., Brennan,K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). *Scratch: Programming for all. Communications of the ACM*, *52*, 60-67.
Sadler, T. D. (2004). Informal reasoning regarding socioscientific issues: A critical review of research. *Journal of Research in Science Teaching, 41*(5), 513-536.
Sharples, M., & Pea, R. (2014). *25 Mobile Learning*. In R. Sawyer (Ed.), *The Cambridge Handbook of the Learning Sciences: Cambridge Handbooks in Psychology*, (pp. 501-521). Cambridge: Cambridge University Press.
Tekin, N., Aslan, O., & Yilmaz, S. (2016). Research trends on socioscientific issues: A content analysis of publications in selected science education journals. *Journal of Education and Training Studies*, 4(9), 16–24.
Werner, L., Denner, J., & Campe, S. (2012). The fairy performance assessment: Measuring computational thinking in Middle School. *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education - SIGCSE '12*, 7–12.

Acknowledgments

This work was funded by grants from the Gregory and E.J. Milken Strategic Faculty Support Fund and the University Research Fund at the University of Pennsylvania.