Design of an Expansively-Framed Board Game-Based Unit to Introduce Computer Programming to Upper Elementary Students

Victor R. Lee, Stanford University, vrlee@stanford.edu

Frederick Poole, Jody Clarke-Midura, and Mimi Recker
frederick.poole@aggiemail.usu.edu, jody.clarke@usu.edu, mimi.recker@usu.edu
Utah State University

Abstract: This paper presents an instructional design using expansive framing to introduce computer programming to upper elementary students. By using a tabletop board game as the context for learning, bridging connections between the learning in the board game and its digital instantiation, and privileging student authorship, we show how two students developed and transferred their understanding of several computational practices, including procedures and conditional logic, from the board game into their design of digital games in Scratch.

Introduction

Lessons and activities that do not require a digital computer to teach computational ideas and principles, or "unplugged" computer science (CS) instruction (e.g., Bell, Alexander, Freeman, & Grimley, 2009), have garnered a great deal of interest among CS educators. One of the challenges with unplugged instruction is how ideas encountered in an unplugged setting are mobilized into a "plugged" setting where computers are used. Our design (NSF Grant no. 1837224) involves starting unplugged instruction by playing a computer science-themed board game where students learn about basic algorithms, procedures, abstraction, and conditionals. We then have them play with and examine the code behind digital instantiations of the board game in the Scratch coding environment. Finally, students modify the Scratch code to create their own digital versions of the board game for their peers to play. Core questions we ask are: Do students succeed in moving through this progression such that they are able to successfully make their own digital versions of the board game? What features do they include in the design of their own digital versions? What programming-related content are students incorporating into their game designs?

Design of the unplugged-to-plugged unit

The instructional unit took place once per week over eight weeks for 20-minute sessions in both the classroom and the school library. Our design is situated in the expansive framing theoretical framework (Engle et al., 2012) and spanned across settings (library and classroom) and mediums (unplugged board games to digital instantiations in Scratch). Briefly, expansive framing is an approach for supporting transfer between learning contexts. It posits that by making connections between the context of learning and transfer, educators can help students create an encompassing context that aids in knowledge transfer and deeper learning. Here, the context of learning was the board game and the context of transfer was the digital instantiation of that board game in Scratch.

During weeks 1 and 2 of the unit, students played a published board game, *//CODE: On the Brink*, and completed several levels to learn the game mechanics and identify strategies for solving the puzzles, which involved navigating the board based on 'programmed' instructions associated with colored blocks. During these first two weeks, students created and called procedures, albeit not always knowingly, to solve the levels. In weeks 3 through 5 of the unit, students worked with Scratch shells of the same game and began to modify the existing code to produce the same behaviors as the board game. In this stage of the unit, we leveraged the relationship between the board game and Scratch shell to draw attention to and help students see the cards (movements) as procedures that would be called *if* the sprite (robot) landed on an associated color. In weeks 6 and 7, the students began creating their own game levels with added features. In this authorship stage, students needed to apply their knowledge of both the board game and the Scratch environment to create a functioning game. Finally, in week 8, the students shared their digital game boards and played each other's games during class and library time.

Data sources and analyses

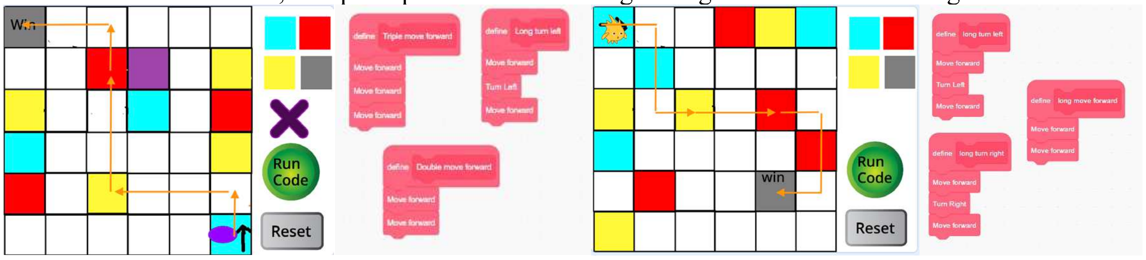
We collected several forms of data from three classes of fifth grade students (N=96) in a rural school in the United States. Class and library sessions were video recorded and students completed pre/post affective surveys that measured interest and confidence in programming. Ten students were interviewed after the unit.

Results

We present two abridged examples of levels where students creatively designed deception into their instantiations of the game to illustrate what student authorship of game levels looked like and discuss of what students learned.

We first spotlight a student-authored level created by Max. Survey results showed that Max started the project with high interest and self-confidence in programming. He reported enjoying this unit because "*[the unit] allowed us to be more creative than other projects in school.*" He also stated that he enjoyed being able to "*make our own boards*" and "*test our ability to program.*" Max designed a challenging level with a unique game mechanic (Figure 1). First, Max created a new if-then statement for a purple block that if landed on, would send the sprite off the board by moving it to the location of a new sprite he created. In addition, Max created three new procedures (See Figure 1). In creating these new procedures, Max called existing procedures and thereby built code with two layers of abstraction. Further, Max designed the board to have two false routes that involved colored blocks that would never be used. That was part of the deception he designed into his game. Further, there were blocks of each of the three pre-defined colors on the margins that were never used, but students thought they were necessary to solve the game. In class, several students tried to solve the level and determined it was unsolvable. They were thus surprised when Max revealed the solution publicly to the class during the last class session.

The second example was authored by Gina, who also started the project with high interest and confidence in programming. She said she enjoyed the unit because she learned about 'My Blocks', where new procedures can be defined and allowed her to do whatever she wanted. Leading up to creating her own level, Gina frequently helped her peers learn the game rules and solve some of the levels both in the board and Scratch versions of the game. Gina's level (see Figure 1, right) was similar to Max's board in that she created two false routes to confuse her classmates. To create this level, she needed to build three custom procedures. As with Max, these new procedures called existing procedures, thus displaying several levels of abstraction. The deception in Gina's board involved passing over several colored blocks but not executing their called procedures because the program would already be mid-procedure when those blocks were visited. She also had blocks that were not part of the winning path. As it had been for Max, deception proved to be a distinguishing feature in Gina's design.



```
define Triple move forward
Move forward
Move forward
Move forward

define Long turn left
Move forward
Turn Left
Move forward

define Double move forward
Move forward
Move forward
```

```
define long turn left
Move forward
Turn Left
Move forward

define long move forward
Move forward
Move forward

define long turn right
Move forward
Turn Right
Move forward
```

Figure 1. Max's (left) and Gina's (right) authored game board incorporating three custom procedures.

Discussion and conclusion

This project used expansive framing as a framework for designing an instructional approach in which computer programming moves from unplugged to plugged activities. By using board games as the context for learning and bridging connections between the board and digital versions of the game, we argue that students were able to transfer their understanding of CS concepts (e.g., building procedures) in creative ways. For instance, Max exhibited a key computational practice, abstraction, to combine movement procedures to traverse several white blocks. Similarly, in Gina's board, both a blue square and a red square are traversed without executing the procedures associated with them. This made the level challenging, but more importantly, it demonstrates Gina's knowledge of procedures, specifically that both called procedures must be completed before other procedures are executed. Both Max and Gina also created elements of deception into their game designs, a unique feature we had not anticipated at the onset of the unit. By starting with the unplugged activities and then designing digital versions of the game in Scratch, we see that students were able to transfer their understanding of the mechanics of game play in the board game to the Scratch version. They also transferred their understanding of the procedures and game mechanics while designing their games in Scratch. Thus, we believe this approach of playing board games as a pathway to programming appears to be a viable pedagogical approach for CS education.

References

Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer Science Unplugged: School students doing real computing without computers. *Journal of Applied Computing and Information Technology, 13*(1), 20-29.

Engle, R. A., Lam, D. P., Meyer, X. S., & Nix, S. E. (2012). How Does Expansive Framing Promote Transfer? Several Proposed Explanations and a Research Agenda for Investigating Them. *Educational Psychologist, 47*(3), 215-231. doi:10.1080/00461520.2012.695678