Jl. of Computers in Mathematics and Science Teaching (2016) **35**(4), 313-318

Education:=Coding+Aesthetics; Aesthetic Understanding, Computer Science Education, and Computational Thinking

JONATHON GOOD, SARAH F. KEENAN, AND PUNYA MISHRA
Michigan State University, USA
goodjona@msu.edu
keenans1@msu.edu
punya@msu.edu

The popular press is rife with examples of how students in the United States and around the globe are learning to program, make, and tinker. The Hour of Code, maker-education, and similar efforts are advocating that more students be exposed to principles found within computer science. We propose an expansion beyond simply teaching computational thinking skills, by including an aesthetic framework that highlights the beauty and elegance inherent within the craft of coding. This approach not only introduces students to authentic experiences of computational work, but can result in higher levels of retention and achievement. Delivering science content through an aesthetic lens has been successful in other areas of science education. Such an approach in programming extends the possibility of reaching students that previously may not have been interested in the field.

INTRODUCTION

Computer science education and related activities have been gaining interest both within and outside of schools in the past decade . The Hour of Code, a non-profit organization dedicated to offering introductory computer science (CS) tutorials online, claims to have reached 15 million users in one week of 2014 (“Every child deserves opportunity,” 2015). The White House took part in a national event for a *Week of Making* in 2014, encourag-

ing students, teachers, and community members to become involved in activities such as programming, fabrication, and tinkering with electronics ("A Nation of Makers," 2015). There are multiple reasons given for this push toward computer science and related disciplines, ranging from employment need to gaps in gender representation in CS. These efforts are supported by educators in computer science while simultaneously seeking new ways to attract entrants into the field (Blankenship, 2015). We are proposing that there is an additional way to gain this interest, and possibly sustain students' curiosity for more than an hour or a week. Using an aesthetic approach to the teaching of computer science, we can not only introduce students to the beauty of the craft of coding, but also present them with an accurate portrayal of the diverse styles of computer science work, and the many contexts in which these skills can be used. Refining the perceptions surrounding the work, environment, and emotional experience of programming can make the field more appealing to a larger population of students.

AESTHETICS AND EDUCATION

Having an aesthetic understanding of subject matter has long been recognized as an important motivating and inspirational feature of many professional scientists' experience (Root-Bernstein, 1997; Girod, 2007). By "aesthetics" we mean not just a superficial appreciation of the lovely things science can bring us, like rainbows and waterfalls (though those certainly are access points to more developed content knowledge) but a deep and transformative appreciation of how powerfully science can explain the world around us.

Although well recognized as a source of creativity and inspiration among many of the most innovative and successful scientists, aesthetic interaction with a subject has historically been given short shrift in the science instruction experience of our students (Pugh & Girod, 2007). Flannery (1991) made a compelling argument for integrating aesthetics in science education, identifying aesthetic qualities in science across the objects studied and the experience within inquiry – suggesting that drawing out these features might lead to a more authentic and personable engagement with the subject. While this did not bring an aesthetic revolution, ideas similar to Flannery's have taken hold of various researchers across the past few decades, and resulted in a variety of interventions and results (e.g., Girod & Wong, 2002; Girod, Rau, & Schepige, 2003). Common across all studies, students taught for aesthetic understanding have more positive attitudes toward science and maintain their conceptual understanding of subjects far longer than their non-aesthetically taught peers.

There already exist bridges between this aesthetic approach and the efforts of computer science educators. The recently developed AP Computer Science Principles curriculum includes discussions of the "Big Ideas" of computing, beginning with the concept of creativity (The College Board, 2014). Theory surrounding the maker movement already brushes up against aesthetics with discussions of how students' aesthetic experiences can provide opportunities for entry into the field (Kafai, Fields, & Searle, 2014). In this piece, we go deeper to focus specifically on the concept of *beautiful code*.

BEAUTIFUL PROGRAMS AND THE CODERS WHO LOVE THEM

Professional programmers will readily share their experiences with code that they might describe as elegant, beautiful, or clean. In *Beautiful Code* Oram and Wilson (2007) interviewed programmers from multiple fields on software that they found to be beautiful, with subjects focusing on various qualities such as efficiency, unorthodox solutions, and simplicity in design. To the uninitiated, these descriptions of beauty may seem at odds with the strange text (code) they see on the screen, until they either become programmers themselves or are given a chance to listen to the accounts of those that go before them. Kozbelt et al. (2012) touch upon this in their examination of types of code both novices and experts describe as *ugly* and *beautiful*. Both groups reported aesthetic experiences related to code they had observed, albeit with functionality being of higher importance. Both novices and experts were able to qualitatively describe code with affective descriptions, and both groups' judgements were highly correlated (p. 61). This bodes well for the idea that novice programmers are able to experience, recognize, and appreciate the aesthetics of programming. In *Geek Sublime* (2014), Chandra discusses how code can go "beyond the purely practical; like equations in physics or mathematics, code can aspire to elegance" (p. 5). Graham (2010) made the case for how the hacking tradition within computer science is similar to the act of creating art. He offered that both mediums include multiple styles of work, go through multiple iterations of the final product, and are chaotic at times (pg. 25). These are only a few of many examples found in both academic and popular literature, providing researchers and educators with a rich area to mine for insight into the beauty found within programming. With such a rich source of information from practitioners of computer science, and the research supporting the use of aesthetic understanding within education, this is an approach worth pursuing.

AESTHETICS OF CODE WITHIN K-12

A recent effort to engage students in computer science principles in both CS and non-CS classrooms is that of computational thinking (CT). Computational thinking is described as a collection of problem-solving skills that draw from the methods used by computer scientists (Barr & Stephenson, 2011; Lee et al., 2011; Wing, 2006, 2008; Yadav, Mayfield, Zhou, Hambrusch, & Korb, 2014). While we have found merit in the ways computational thinking can be used to engage students with computer science principles, we have found that the literature surrounding this construct is largely devoid of language relating to beauty, aesthetics, and style. A recent analysis we conducted of some of the more prominent CT literature revealed that the most common themes largely revolved around skills and contexts; only discussions of optimization possibly intersected with the concept of aesthetics (Good, Yadav, & Mishra, forthcoming). We propose that these two approaches, CT and aesthetic understanding of content, can be complementary, with a shared goal of increasing K-12 students' interest in computer science.

Computational thinking methods within the curriculum allow teachers to show students how skills such as abstraction and problem decomposition (Barr & Stephenson, 2011; Grover & Pea, 2013) can be found within context completely devoid of programming or even computers. Aesthetics can add to this experience by encouraging students to "do it with style" when they propose a solution to a problem. This is not a "dumbing down" of the material, but rather elevating attention to detail and craft as an important component of the practice of computer science. Students will need to consider the users and context when considering a solution to a problem, much as a programmer must try to anticipate how users will react to an interface, or how programmers construct programs (solutions) that allow the users to become architects of the final product. This approach is helpful not only in preparing students to learn the language of the field, but also in exposing them to the authentic practice of computer scientists, who must translate real-world problems into a framework that they can later be analyzed through computation.

FUTURE DIRECTIONS FOR RESEARCH AND PRACTICE

With these ideas of computational thinking and aesthetic understanding working in tandem, there are a few areas we see as being fruitful for researchers. First, analysis is needed of the characteristics of code that pro-

grammers describe as beautiful. This is not only to provide a more coherent definition of *beautiful code*, but also to guide teachers in providing examples for students to review. Second, similar to the efforts in science education with aesthetic understanding, empirical work is needed to determine whether this approach indeed generates additional interest in computer science (both short and long term), results in any increases in retention of the material, and whether it aids in the application of programming knowledge. Finally, teachers' and teacher educators' perspectives need to be gathered by researchers on whether this approach is a.) one that will receive interest from educators, b.) can be translated into concrete practices that are possible in the classroom, and c.) may already be occurring in some form.

CONCLUSION

While this discussion of aesthetics in computer science education is its initial stages, we are hopeful that it will invigorate the practices within the CS and non-CS classrooms. The interest and engagement we have found in discussing beautiful code with programmers is certainly a phenomenon we can see having a profound effect if successfully transferred to the classroom.

References

A Nation of Makers. (2015). Retrieved October 21, 2015, from https://www.whitehouse.gov/nation-of-makers/

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. http://doi.org/10.1145/1929887.1929905

Blankenship, L. (2015, October 20). Disrupting the Gender Gap in Computer Science | The CSTA Advocate Blog. Retrieved from http://blog.csta.acm.org/2015/10/20/disrupting-the-gender-gap-in-computer-science/

Chandra, V. (2014). *Geek Sublime: The Beauty of Code, the Code of Beauty*. Graywolf Press.

Every child deserves opportunity. (2015). Retrieved October 21, 2015, from https://code.org/

Flannery, M. C. (1991). Science and aesthetics: A partnership for science education. *Science Education*, *75*(5), 577–593.

Girod, M. (2007). A Conceptual Overview of the Role of Beauty and Aesthetics in Science and Science Education. *Studies in Science Education*, *43*(1), 38–61. http://doi.org/10.1080/03057260708560226

Girod, M., Rau, C., & Schepige, A. (2003). Appreciating the beauty of science ideas: Teaching for aesthetic understanding. *Science Education*, *87*(4), 574–587. http://doi.org/10.1002/sce.1054

Girod, M., & Wong, D. (2002). An Aesthetic (Deweyan) Perspective on Science Learning: Case Studies of Three Fourth Graders. *The Elementary School Journal*, 199–224.

Good, J., Yadav, A., Mishra, P. (in press). Computational Thinking in Computer Science Classrooms: Viewpoints from CS Educators. To be presented at the 2017 annual conference for the Society for Information Technology and Teacher Education, Austin, TX.

Graham, P. (2010). *Hackers and Painters: Big Ideas from the Computer Age* (1 edition). Sebastopol, CA: O'Reilly Media.

Grover, S., & Pea, R. (2013). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher*, *42*(1), 38–43. http://doi.org/10.3102/0013189X12463051

Kafai, Y., Fields, D., & Searle, K. (2014). Electronic Textiles as Disruptive Designs: Supporting and Challenging Maker Activities in Schools. *Harvard Educational Review*, *84*(4), 532–556.

Kozbelt, A., Dexter, S., Dolese, M., & Seidel, A. (2012). The Aesthetics of Software Code: A Quantitative Exploration. *Psychology of Aesthetics, Creativity, and the Arts*, *6*(1), 57–65. http://doi.org/10.1037/a0025426

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., … Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*, *2*(1), 32–37. http://doi.org/10.1145/1929887.1929902

Oram, A., & Wilson, G. (2007). *Beautiful Code: Leading Programmers Explain How They Think* (1 edition). Beijing ; Sebastapol, Calif: O'Reilly Media.

Pugh, K. J., & Girod, M. (2007). Science, Art, and Experience: Constructing a Science Pedagogy From Dewey's Aesthetics. *Journal of Science Teacher Education*, *18*(1), 9–27. http://doi.org/10.1007/s10972-006-9029-0

Root-Bernstein, R. S. (1997). The Sciences and Arts Share A Common Creative Aesthetic. In A. I. Tauber (Ed.), *The Elusive Synthesis: Aesthetics and Science* (1997 edition). Dordrecht: Springer.

The College Board. (2014, June). AP Computer Science Principles Draft Curriculum Framework. Retrieved from http://media.collegeboard.com/digitalServices/pdf/ap/comp-sci-principles-draft-cf-final.pdf

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, *49*(3), 33–35. http://doi.org/10.1145/1118178.1118215

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *366*(1881), 3717–3725. http://doi.org/10.1098/rsta.2008.0118

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational Thinking in Elementary and Secondary Teacher Education. *ACM Transactions on Computing Education*, *14*(1), 1–16. http://doi.org/10.1145/2576872